A FRAMEWORK FOR EMPLOYEE APPRAISALS BASED ON INDUCTIVE LOGIC PROGRAMMING AND DATA MINING METHODS

DARAH MOHAMMAD AQEL

SCHOOL OF COMPUTING, SCIENCE AND ENGINEERING, INFORMATICS RESEARCH CENTRE, COLLEGE OF SCIENCE AND TECHNOLOGY, UNIVERSITY OF SALFORD

Submitted in Fulfilment of the Requirements of the Degree of Doctor of Philosophy, November 2013

Bismillah al Rahman al Rahim

Contents

List of Figures

List of Tables

Acknowledgments

First of all, I would like to thank ALLAH ALMIGHTY who gave me the ability, patience and strength to conduct and complete this thesis. I am very thankful to my supervisor Professor Sunil Vadera for his assistance, support and feedback during this research, as well as his help in pointing me in the right direction.

Special thanks to my family, in particular my father, my mother, my brother, my sisters, and my husband for all their assistance, guidance and encouragement.

Last but not least, I would like to acknowledge and thank everyone who helped me with good advice during my research work.

The following papers have been published as part of this research, where these papers include some of the content of this thesis:

- Learning a Grammar for SMART Objectives Using Inductive Logic Programming. In Proceeding of the 5th European Conference in Intelligent Management Systems in Operations (IMSIO2013). The Operational Research Society, 3-4 July 2013, pp. 66-76.
- A Framework for Employee Appraisals based on Inductive Logic Programming and Data Mining Methods. In Proceeding of the 18th International Conference on Applications of Natural Language to Information Systems (NLDB2013). Springer, Berlin Heidelberg, 19-21 June 2013, pp. 404-407.
- A Framework for Employee Appraisals based on Sentiment Analysis. In Proceedings of the 1st International Conference on Intelligent Semantic Web – Services and Applications (ISWSA2010). ACM, 14-16 June 2010, Article 8, pp. 62-67.

Chapters 4, 5 and 6 are the developed version of these papers which presented interim results.

Abstract

Employee performance appraisal systems are widely regarded as fundamental for evaluating employees' performance and enhancing organisations' success. Yet, there is evidence that employees doubt their benefits and fairness, organisations find them difficult to implement and their value is questioned. Although commercial systems that support appraisals have been developed, their focus remains on recording and tracking information, thereby not providing the kind of meaningful and deeper support for appraisals and the goal setting process such as ensuring that the objectives are SMART (specific, measurable, achievable, realistic, time-related) and providing feedback on these objectives.

Developing a supportive employee appraisal system for setting objectives represents a major challenge for computer science since the objectives are unstructured, assessing objectives is a subjective process, and there is no known system for writing effective objectives, providing feedback and supporting the decision making process.

Thus, helping employees to write SMART objectives requires finding the rules of writing these objectives. As the objective sentences are expressed in natural language, natural language processing (NLP) techniques and machine learning may have the potential for supporting the process of setting objectives by first analysing the objectives text and then identifying the rules for writing SMART objectives. More specifically, machine learning methods such as inductive logic programming (ILP), which investigates the inductive acquisition of first-order theories from background knowledge and examples, could be applied to automatically learn the rules.

The process of setting objectives also requires assessing whether the objectives can be met given the available resources and time. Data mining techniques may have the potential to be used for assessing the objectives.

This thesis develops a new framework for employee appraisals. The developed framework supports the process of setting SMART objectives and providing feedback. The framework utilises ILP to learn the grammar rules for writing SMART objectives and applies data mining techniques for assessing the objectives. The framework has been implemented using the ILP system ALEPH as well as prediction and classification rule induction algorithms in the WEKA data mining software. A novel system has been developed based on the proposed framework to show the feasibility of the framework. An empirical evaluation of the developed system has been conducted using a corpus of 300 objective examples and achieved promising results with an overall accuracy of 83%. The thesis also includes the limitations of the developed framework and proposes the potential for further research.

Chapter 1: Introduction

Organisations aim to improve their performance, increase profitability and achieve set goals. This requires high-quality management targeted at maximising employee skills, managing utilisation of resources and increasing employee motivation. Employee appraisal systems are extensively required for evaluating employee performance, enhancing company productivity, and endorsing employee loyalty and motivation to achieve organisational goals (Waldman et al., 1987; Murphy and Cleveland, 1991; Boice and Kleiner, 1997; Roberts, 2003; Kline and Sulsky, 2009).

Appraising employee behaviour is a challenge that faces organisations in the current economic crisis where these organisations are struggling in the recession. Even though employee appraisal systems have numerous benefits for organisations and employees, several authors have suggested that the level of employees' trust for management is decreasing (Farnham 1989), especially if there is a misuse in the performance appraisal systems (Mayer and Davis, 1999). Evidence shows that some employees fail to find performance appraisal systems beneficial and they doubt their effectiveness (Smither et al., 2005). Furthermore, others question their fairness because they regard them as based on incomplete information, opinions and subjective bias of appraisers, personal preferences, inaccurate rating and lenient in evaluations (Murphy and Cleveland, 1995; Arvey and Murphy, 1998; Rowland and Hall, 2012). The element of subjectivity represents a weakness in the process of appraising performance, where this element may cause unfairness for employees and it is unavoidable in performance appraisal (Boachie-mensah and Seidu, 2012; Choon and Embi, 2012). This leads to weaker incentives and may cause harm for workers and employees, where unfair performance appraisals may actually reduce employee motivation and performance (Latham and Mann, 2006). In addition, some say that the process of appraising employees is complicated and is a very time consuming process (e.g. Finlay and McLaren, 2009).

Although, there are commercial systems for supporting performance appraisals, they focus on recording and tracking the process and provide little or no support for deeper aspects of the appraisal process or the goal setting process (Locke, Shaw, Saari and

Latham, 1981; Locke and Latham 1990, 2002) such as ensuring that objectives are SMART (specific, measurable, achievable, realistic, time-related) (Doran, 1981; Hurd et al., 2008; Desmond, 2011) and providing feedback for the quality of the objectives set. Therefore, poor management and the lack of support for the goal setting process reduce intrinsic motivation and make the process of goal setting less meaningful and less effective for the employee, manager and organisation (Locke and Latham, 2009; Ordonez et al., 2009). Furthermore, studies on goal setting programs have concluded that the process of goal setting may be affected by different variables and factors which may influence the whole process of performance appraisal. For instance, the employees' perceptions of goal-setting process influence their performance, job satisfaction, productivity and goal commitment (Barrick et al., 1993; Antoni, 2005; Bipp and Kleingeld, 2011).

Yet, developing a supportive performance appraisal system for setting objectives represents a major challenge since the objectives are unstructured, assessing objectives is a subjective process and the process of setting and assessing objectives requires effort and time. Moreover, there is no known system for setting, writing and assessing the objectives as well as providing feedback and supporting the decision making process.

Thus, helping employees to write SMART objectives is a critical pre-requisite for appraisal systems to achieve their aims. To support the process of setting SMART objectives, this requires first analysing the written objectives and then finding the rules of writing SMART objectives. By analysing some examples of written SMART objectives, it is evident that these objectives are expressed in a certain way in natural language. Accordingly, natural language processing (NLP) (Manning and Schütze, 1999; Jackson and Moulinier, 2007; Collobert et al. 2011) techniques may have some potential for analysing the text of the written objectives linguistically. In general, NLP is a field in artificial intelligence (AI) (Russell and Norvig, 2010) which is based on processing natural language, both in written and spoken form, by using automatic methods.

The main idea is then to discover the rules of writing SMART objectives based on the performed linguistic analysis of the objectives. But how can we find these rules? One option is to interview specialists and attempt to hand-craft the rules. The process of obtaining the rules can be time consuming and is dependent on the availability of experts and can lead to the Fiegenbaum (1977) bottleneck for knowledge acquisition. Moreover, systems based on hand-crafted rules can only be made more precise by making the rules more complex; however, this will make the process more difficult. An alternative, which is explored in this thesis, is to attempt to use machine learning methods (Mitchell, 1997; Alpaydin, 2010) to automatically learn the rules.

Machine learning methods are utilised for learning useful patterns and rules from observations, processing unstructured text to extract structured information and deriving significant rules that analyse linguistic patterns in a meaningful way. These methods have been applied effectively to different domains such as bioinformatics (e.g. Larrañaga et al. 2006), medicine (e.g. Savage, 2012), finance (e.g. Khandani, Kim and Lo, 2010), business (Samanovic, Cukusic, and Jadric, 2011), and NLP (e.g. Ratnaparkhi, 1999; Saito and Hagiwara, 2010).

There are many machine learning methods, such as neural networks (McCulloch and Pitts, 1943), decision tree induction (Quinlan, 1993), and inductive logic programming (Muggleton and De Raedt, 1994). Methods such as neural networks have been successfully used for classification, however tend to be black box methods in that they are not good for presenting the rationale for their classification and not particularly suitable for representing the contents of textual data.

Inductive logic programming (ILP) (Muggleton and De Raedt, 1994; Muggleton, 1999; De Raedt, 2008) is an approach to machine learning that integrates machine learning and logic programming to induce rules (theories) in a form of logic programs from background knowledge and examples. In particular, it uses a knowledge representation language to represent complex data in first order logic and to describe relations between a set of entities in order to infer general rules that describe the logical representations of these entities. ILP investigates induction rather than deduction as the core operator of inference. The main advantages of ILP are that it is suitable for knowledge representation and is also able to represent linguistic knowledge. It has been widely proposed for NLP (e.g. Mooney, 1997; Cussens, 1997; Lindberg and Eineborg, 1998), information extraction (IE) (e.g. Ramakrishnan et al. 2008; Dědek, 2010), learning grammar rules and semantics (e.g. Claveau et al., 2003; Eisenstein et al., 2009), bioinformatics (e.g. Santos et al., 2012) and engineering (e.g. Dolsak, Bratko and Jezernik, 1994).

As ILP machine learning has been successfully applied to learn useful rules in different domains and applications, it may have the potential to be applied in a domain such as performance appraisal in order to discover novel linguistic rules that define a grammar for formulating SMART objectives.

As well as having rules that help in structuring objectives, objectives need to be "achievable" and "realistic". This implies being able to assess if a stated objective can be accomplished within the available resources and time. Since the objective statements express what needs to be achieved in the future, then the task of objectives assessment could be addressed by utilising data mining techniques (Han, Kamber, and Pei, 2011). In particular, classification and prediction techniques in data mining, which have been previously utilised for making precise decisions (Tanner et al. 2008; Lee, 2010) and forecasting future events (Bianco, Manca and Nardini, 2009) based on past experience, may have the potential for assessing whether sufficient resources and time are available to achieve an objective.

Hence, this thesis explores the potential for using semantic technologies of AI based on natural language processing, machine learning and data mining techniques for developing a novel system which supports employee performance appraisals. The thesis develops a new semantic framework for employee performance appraisals. The developed framework facilitates the setting of SMART objectives and providing constructive feedback that enables users to improve their objectives. The framework uses ILP to learn the rules that define a grammar for writing SMART objectives, which can be exploited to ensure that the objectives are "specific", "measurable" and "time-related". The framework also utilises data mining techniques for assessing whether the objectives are "achievable" and "realistic". A system has been developed based on the proposed framework to show the feasibility of the framework and an empirical evaluation of the developed system is presented in this research.

1.1 Motivation and Challenges

Supporting employee appraisal systems to be more effective and improving the process of setting goals and objectives represent a major challenge for computer science, since at present; there are no systems that can (Beer, 1982; Waldman et al., 1987; Cleveland et al., 1989; Locke and Latham 1990; Murphy and Cleveland, 1995; Mayer and Davis, 1999; Lam and Schaubroeck, 1999; Fletcher, 2001; Locke and Latham, 2002; Hurd et al., 2008; Kline and Sulsky, 2009):

(a) Help employees to define effective objectives.

(b) Assess whether the written objectives are SMART.

(c) Provide feedback on the objectives.

(d) Predict the required resources based on some historical data in order to make accurate decisions for allocating the important resources in the future.

(e) Support organisations in planning business strategy and evaluating the future risk factors of a business based on the suggested objectives.

(f) Measure the progress of a company and its employees by evaluating the progress of achieving the organisational goals.

(g) Reduce the effort required by managers and save the time required in appraising the objectives.

(h) Facilitate the communications between employees and managers.

(i) Increase employee motivation, job satisfaction and goal commitment.

To the best of the author’s knowledge, there is no known system for writing good objectives, providing feedback and supporting the process of decision making. This thesis focuses on exploring the extent to which it is possible to achieve the above mentioned points and to develop a system that can perform the above capabilities. The focus in this thesis will be on a, b, c, and d, while the others (e, f, g, h and i) are also important but not the subject of this thesis, given the time limitations.

1.2 Research Objectives and Methodology

Given the above motivation, this section presents the research objectives, hypothesis, and methodology.

1.2.1 Objectives and Main Contributions

The thesis objectives are:

1. To develop a novel framework for supporting employee performance appraisal systems and the goal setting process so that employees are able to write SMART objectives.

2. To explore and assess the use of ILP and data mining techniques for developing a novel system and framework that support performance appraisals and goal setting to assess whether the objectives are SMART and provide feedback on the objectives.

3. To evaluate the framework by applying it to a particular domain and reviewing the outcomes in terms of system accuracy.

The primary research hypothesis is that the use of ILP and data mining techniques can help to develop a system for assessing whether the written objectives are SMART and providing feedback on these objectives. This will support performance appraisals to provide a better basis for employee performance appraisal and goal setting systems.

1.2.2 Research Methodology

Research is a scientific investigation and systematic search to find new knowledge and information (Kothari, 2004).

The main types of research include (Kothari, 2004):

- Descriptive research and analytical research
- Applied research and fundamental research
- Qualitative research and quantitative research
- Conceptual research and empirical research

Descriptive research describes characteristics and data about phenomenon. In descriptive research, the researchers can not control the required variables. The methods used in descriptive research include survey investigation and fact finding, while in analytical research, the researchers use the available variables and information in order to analyse and evaluate the study (Kothari, 2004).

Applied research is based on finding solutions for social, industrial or business problems (e.g. marketing research or evaluation research), whereas fundamental research is based on generalising theories and finding information about applications. The research related to natural phenomenon is an example of fundamental research (Kothari, 2004).

Quantitative research investigates empirically a phenomena or hypothesis which is based on quantity measurement and on the use of statistical techniques (e.g. experimental approach), while qualitative research is concerned with analysing human behaviour and opinions to produce information on particular case studies (e.g. opinion research) (Kothari, 2004).

Conceptual research is used to develop a new theory and concepts or add a new explanation to existing ones. Empirical research is data-based research that uses facts, data and variables in order to prove research hypotheses and construct results by investigating questions and analysing observations and experiences (Kothari, 2004).

Each type of research uses different kinds of research methods and techniques to solve the research problem scientifically (Franklin and Osborne, 1971; Kothari, 2004). For example, library research uses analysis of historical records method (e.g.

Tape and Film listening) or analysis of documents method (e.g. reference and abstract guides). The relevant research methods to be used in library research involve non participant direct observation, participant observation, mail questionnaire, personal interview, telephone survey and case study (Kothari, 2004).

The scientific method is one of the most commonly used research methods. This method involves all the techniques that rely on empirical research. The scientific method is used for investigating phenomena and analysing observations by using statistical methods and experimental studies in order to test the research hypotheses, and then determine the validity of outcomes (Franklin and Osborne, 1971; Kothari, 2004).

Hence, given that the objectives in this research (Section 1.2.1) involve assessing accuracy and the use of machine learning methods, the research methodology adopted is based on empirical research and the scientific method. The following steps describe the research methodology adopted to address the research hypothesis and objectives identified in the foregoing section:

1. Identifying the research problem by stating the factors related to the area of performance appraisals and specifying the ways of writing effective objectives.

2. Performing a comprehensive literature review on studies of setting goals and objectives and studying what constitutes well written SMART objectives.

3. Reviewing some examples for effective SMART objectives and specifying the key characteristics of the SMART criteria.

4. Constructing a corpus of objectives that includes the reviewed examples of SMART objectives and some other objective examples.

5. Survey of relevant AI methods which can be used to learn the required rules for writing SMART objectives and to assess these objectives.

6. Developing the framework based on AI methods identified in step 5.

7. Testing the outcomes of the research hypothesis which is based on using ILP and data mining techniques to develop a novel system that helps to assess whether the written objectives are SMART and aims to provide feedback on the written objectives.

8. Evaluating the system, which has been developed based on the proposed framework, on the sales domain by using a corpus of objectives related to the sales area in order to estimate and evaluate the performance of application for the utilised AI methods empirically. The evaluation will use a standard statistical method for estimating accuracy, such as cross-validation (Geisser, 1975; Kohavi, 1995). The rationale for choosing the sales domain is presented later in Chapter 4.

1.3 Thesis Outline

The thesis is organised as follows. Chapter 2 presents an overview of approaches to employee performance appraisal, goal setting and SMART objectives including a survey of what constitutes well written SMART objectives. Chapter 2 also includes a survey of relevant NLP, machine learning and data mining methods. The survey of NLP includes a review of text pre-processing tasks and IE, while the survey of machine learning focuses on ILP. The survey presented for data mining concentrates on the classification and prediction methods used in data mining. A discussion about the methods that have been selected to develop the framework is also included in Chapter 2. Chapter 3 proposes a new framework for setting SMART objectives and providing feedback based on the use of ILP and data mining methods and presents a typical scenario for showing the potential capabilities for a system that supports employee appraisals. Chapter 4 describes the development of a corpus of objectives and presents the linguistic analysis performed on this corpus. Chapter 5 describes the main components of the framework that are utilised to develop a system for setting SMART objectives, where ILP is used to learn a grammar for writing SMART objectives and data mining techniques are utilised to assess the objectives. Chapter 6 summarises the system implementation and presents a snapshot of the system interface with some illustrative examples of objective sentences. Chapter 6 also presents an empirical evaluation of the developed system and compares it with related systems as well as illustrates the generality of the framework. Chapter 7 concludes the thesis and includes a summary of the achievements, main contributions and directions for future work.

1.4 Summary

In this chapter, an introductory description of performance appraisal and the goal setting process as well as their challenges was given. Furthermore, a brief description of machine learning (in particular ILP) and data mining was presented in the introduction section in terms of the background to the research work presented in this thesis. The motivation of supporting employee performance appraisal systems and the process of setting goals and objectives followed by the objectives and hypothesis of this PhD research were presented. Finally, the methodology adopted to address the specified research objectives was given.

In the next chapter, a relevant literature review of the areas of performance appraisals, goal setting, the SMART approach, NLP, IE, ILP, data mining is presented. A discussion about the techniques that have been chosen to develop the framework proposed for this research will be given.

Chapter 2: Literature Review

As outlined in the introduction, this research explores the use of natural language processing, machine learning and data mining techniques for developing a new semantic framework for employee performance appraisals that supports the writing of SMART objectives and providing feedback on these objectives. This chapter presents the relevant literature review of these areas. Section 2.1 reviews performance appraisal, goal setting and the SMART approach. Section 2.2 presents a survey of natural language processing and information extraction. Section 2.3 gives an overview of machine learning and data mining methods. Section 2.4 presents an overview of inductive logic programming. Section 2.5 describes the functions that are normally used for evaluating the performance of machine learning and data mining methods. Section 2.6 presents a discussion and summary about the reviewed studies in this chapter. Moreover, it describes the techniques that have been selected to develop the framework for this research and illustrates the motivational reasons behind choosing such techniques. Section 2.6 also includes a set of tables that brings together the reviewed research in the areas of natural language processing, information extraction, machine learning and data mining.

2.1 Performance Appraisal, Goals Setting and the SMART Approach

This section begins with an overview of performance appraisal methods in Section 2.1.1, and then provides a detailed review on the process of goal setting and the SMART approach in Section 2.1.2.

2.1.1 Overview of Performance Appraisal Methods

The success of every organisation depends on the quality and availability of the well-motivated human resources (Judge and Ferris, 1993). Organisations use subjective performance appraisals to evaluate an employee’s job performance, provide annual reviews about the employee’s attitude, enhance work quality and improve a company's profitability, growth and success (Waldman et al., 1987; Murphy and Cleveland, 1991, Boice and Kleiner, 1997; Roberts, 2003; Kline and Sulsky, 2009). In general, performance appraisal is also useful for making important decisions regarding staff promotions, salary, retention and layoffs. Moreover, it is considered a supportive technique for identifying the training required for employees, evaluating the achievement of organisational goals and providing performance feedback (Cleveland et al., 1989).

There are many types of performance appraisal systems, which are based on different performance appraisal methods such as management by objectives (MBO) (Drucker, 1954), 360 degree feedback (London and Beatty, 1993), behaviourally anchored rating scales (Schwab, Heneman, and Decotiis, 1975) and forced distribution methods (Scullen et al., 2005).

The MBO performance appraisal method was first suggested by Peter Drucker (1954) as a goal setting technique for defining objectives in an organisation. In MBO, employees are involved in the process of goal setting to set strategic business objectives and ensure that the organisational objectives are achieved. In 360 degree feedback (multi-resource feedback), different reviewers e.g. subordinates, peers, supervisors, collaborators and the employees themselves are involved in appraising an employee's performance and providing feedback. The behaviourally anchored rating method is used to illustrate a performance rating for specific behaviours which indicate a good or poor performance. The forced distribution method evaluates the employee performance where managers are responsible of distributing the rates for employees in order to specify the most and least talented members of the work team.

Several researchers have investigated different approaches for appraising an employee's performance and described the importance of using these performance appraisal methods for building effective appraisal systems. For instance, De Andrés et al. (2010) have presented a framework for appraising employee performance by using the 360 degree appraisal method. In their approach, different groups of reviewers are involved in the evaluation process, where these reviewers may have different degrees of knowledge about employees. The methodology focuses on aggregating judgements, opinions and evaluations about employees in order to assess their performance. The 360 degree feedback method has been also used in the study proposed by Espinilla et al. (2012) to evaluate employees’ performance. The utilised method can manage heterogeneous information from several sets of reviewers by applying an effective aggregation process for the reviewers’ opinions and judgements.

The study suggested by Antoni (2005) develops a model for group goal setting to investigate the effects of MBO mechanism at team level. This study is based on examining the effects of the group goal commitment and group goals moderated by task interdependence on group processes. Results show that group goals and goal commitment affect group productivity and job satisfaction, while group processes mediate only job satisfaction, since the moderating effect of task interdependence on group processes is not supported in this study. Another study that is based on the use of MBO and goal setting, is the one introduced by Bipp and Kleingeld (2011), where a goal setting program (process) has been utilised to evaluate an employee's performance. The study has investigated how employees' perceptions of the goal-setting process influence job satisfaction and goal commitment. The results show that conscientiousness in goal commitment does not affect the employees' perceptions of the process of goal-setting. However, neuroticism may influence the job satisfaction through the perceptions of the goal content.

Berger, Harbring, and Sliwka, (2013) suggest an approach for evaluating the performance of employees based on bonus payments by using a forced distribution system. They have demonstrated that the productivity and incentive effects are higher under a forced distribution when workers work independently. However, their study showed that supervisors' social preferences have a great impact on rating employee behaviour which affects the appraisal process.

Even though the above studies of performance appraisal show its benefits and effectiveness, the appraisal approaches may suffer from biased ratings, lenient assessments, personal preferences and unfair judgments (Murphy and Cleveland, 1995; Arvey and Murphy, 1998; Rowland and Hall, 2012). Therefore, performance appraisals may affect employees' perception of the appraisal process because the process is mainly based on the supervisors' judgments and it is still affected by subjectivity (Choon and Embi, 2012; Boachie-mensah and Seidu, 2012). This may cause perception of injustice and lack of trust in management, where most employee ratings are obtained from employees' managers. Furthermore, the goal setting process also may be affected by different factors which may influence the employee's performance, job satisfaction, productivity and goal commitment (Barrick et al., 1993; Antoni, 2005; Bipp and Kleingeld, 2011).

2.1.2 Overview of Goal Setting and the SMART Approach

The importance of making effective decisions which support the organisational needs makes organisations to recognise the significance of performance appraisal systems and their impact on motivation, cognition and success. The first step of developing an efficient performance appraisal system is to specify the organisational goals. Setting business goals and writing meaningful objectives are essential aspects for performance appraisals that every organisation must take it into account, in order to build strategic plans (Locke and Latham, 1990).

The goals setting process helps to achieve organisational goals by motivating employees and staff to meet the objectives set (Locke, Shaw, Saari, and Latham, 1981; Locke and Latham, 2002). Furthermore, goal setting can affect the performance of employees and their productivity (Locke and Latham 1990; Locke and Latham, 2002; Locke and Latham, 2006). The process of goal setting becomes more effective and may improve the performance and satisfaction of employees if it includes the feedback on the quality of the goals and objectives (Locke and Latham 1990; Locke and Latham, 2002).

There is a difference between the terms goals and objectives. Goals are broad statements of intended actions that are set by organisations to guide actions and accomplish the desired mission (Hurd at el., 2008). Objectives are specific outcomes that should be achieved successfully within an expected timeframe (Locke, Shaw, Saari, and Latham, 1981). Goals can be either short-term, which could be achieved in a year or less; medium- term, that could be accomplished in 1 to 5 years; or long-term, that could be attained in 5 to 7 years (Hurd et al.,2008, Ryan, 2011).

As described before, the MBO method (Drucker, 1954) is normally used for setting business objectives where employees are responsible to their managers in formulating the objectives, evaluating the results and determining how the objectives will be achieved under the organisational strategy. The strategic and effective objectives should be clear and possible to be achieved (Jaques, 2005). In MBO, the objectives need to be set by using the SMART approach (specific, measurable, achievable, realistic and time-related) (Doran, 1981; Hurd et al., 2008; Desmond 2011).

The SMART approach is a common technique for setting good and effective objectives. This method is based on involving employees in setting objectives in order to appraise the quality of the business objectives and measure the achievement of the objectives (Doran, 1981; Hurd et al., 2008; Desmond, 2011).

Several researchers have studied what constitutes well written objectives and they have applied the SMART approach for setting objectives, where this approach has been described by these researchers in different ways. The following studies summarise the process of setting objectives using the SMART approach and provide some examples of SMART objectives.

For example, Doran (1981) has proposed the SMART approach for setting goals and objectives in company management, where the acronym SMART refers to “specific”, “measurable”, “assignable”, “realistic” and “time-related”. He has illustrated that an objective is considered “specific” if it declares a specific area of improvement. Likewise, the objective is “measurable” if it specifies an indicator of improvement and progress. The objective is “assignable” if it identifies who will achieve the objective. The objective is “realistic” if it is possible to be achieved given the available resources. The objective is “time-related” if it states the deadline to achieve the objective.

Desmond (2011) is another author who has proposed a different definition of the SMART approach for setting business and project objectives. He has declared that the SMART acronym refers to “specific”, “measurable”, “achievable”, “realistic” and “time-bound”. In his definition, an objective is “specific” when it focused, detailed and well defined. It should also state clearly what needs to be done. The objective is “measurable” when its results can be measured. The objective is “achievable”, if it can be attained within the required time. The objective is “realistic” when the resources are available to achieve the objective. The objective is “time-bound” if the deadline for achieving it is stated in the objective. Desmond has mentioned that the deadlines need to be both achievable and realistic. He has also presented some examples for good business objectives:

- “Design and implement a new Internet service to generate over $15M profit over the next five years”.

- “Implement new processes for selection of positive new business opportunities that will produce three net positive services over the next two years”.

Hurd et al. (2008) proposed a definition for the SMART approach, where the acronym SMART refers to “specific”, “measurable”, “achievable”, “realistic” and “time-related”. They present the following example to illustrate the idea of SMART objectives:

> “the sales and marketing staff is responsible for bringing three new conferences to the Binghamton Convention Center by the end of 2009”.

In Hurd et al's. (2008) definition, the acronym “S” is for “specific” and means that the objective is focused, detailed and well-defined. It should also specify what is to be achieved. In the above example, the objective states clearly what is to be attained, such as the sales and marketing staff need to find three new conferences in the Convention Centre. The acronym “M” is for “measurable” and means that it is possible to measure the achievement of the objective. In the above example, the objective specifies a numeric indicator to measure the achievement of the objective, such as the stating of the three conferences. The acronym “A” is for “achievable” and means that an objective can be achieved within a timeframe. In the above example, it may be possible to bring three new conferences within 2009. If the centre did not find any time to achieve the three conferences until year 2012, the objective would not be “achievable”. The acronym “R” is for “realistic” and means that sufficient resources are available to achieve the objective, where these resources can include staff, budget, items, skills, equipments etc. For the above example, suppose that the sales and marketing department consists of 10 employees, who are capable of accomplishing three conferences per year. If the number of conferences was changed to eight conferences for the same year, 10 people may not be able to accomplish the eight conferences, and in this case, the objective will not be “realistic”. The last acronym “T” is for “time-related” and means that the deadline (e.g. year, month) for achieving the objective is stated in the objective. In the above example, the deadline “2009” is specified clearly in the objective.

The paper by Holmes (2005) proposed a study for connecting some programs to the organisational performance. He has utilised the SMART approach for setting effective objectives for organisational programs, where the SMART acronym refers to the five characteristics: “specific”, “measurable”, “achievable”, “relevant” and “time-bound”. The following are some examples of well-written SMART objectives suggested by Holmes (2005):

- “By June 30, 2006, the company will increase sales to multicultural markets, as demonstrated by a 15% increase in sales revenue from select ethnic target groups”.
- “By December 31, 2005, the quality of employees within the manufacturing division will improve, as demonstrated by a 10% increase in the quality hire metric used to assess overall performance”.
- “By September 30, 2005, the company will improve service to multicultural markets, as demonstrated by a 25% decrease in customer complaints from stores in ethnically diverse areas”.

Based on these examples, Holmes suggests the following model for writing effective SMART objectives:

“By the end of ______, we will ______, as demonstrated by ______.”,

where the first blank states the timeframe for achieving the objective (“time-bound”) (e.g. “June 30, 2006”), the second blank describes the action to be taken in order to achieve the objective (“specific”) (e.g. “increase sales”) and the final blank specifies a measure for assessing the achievement of the objective (“measurable”) (e.g. “15 %”). Moreover, for ensuring that an objective is “achievable”, the selected measure in the objective should be achieved within the timeframe. To ensure that an objective is “relevant”, it must be related to the organisation’s strategic objectives (Holmes, 2005).

Carliner (1998) presents a study that explores how to set effective business objectives for a project. The study also describes how technical communication products can contribute to improving the performance of an organisation. His study specifies the essential elements for writing SMART (specific, measurable, attainable, realistic and trackable) objectives, where these elements are: the observable goal, the conditions for achieving the business goal and the level of performance. The observable goal describes explicitly what is to be achieved, such as specifying whether the goal pertains to increasing sales, generating revenue, or reducing expenses. In case that the goal pertains to revenue, the level of performance must be specified in the goal by stating the amount of revenue (e.g. “$4562”) or its percentage (e.g. “11%”). Moreover, the required conditions for achieving the goal (deadlines) (e.g. “2002”) should be declared in the goal (or objective). For assessing the business objective, the objective should be achieved within deadlines and specific measures of business performance. The assessment will depend on tracking existing measures of business performance. The study presents the following examples to illustrate the way of writing effective business objectives by using the SMART approach.

- “Increase sales 5% by June 30 through sales representatives’ use of “The Calculator Program.””.
- “Generate $25 million in sales between January and June”.
- “Obtain ISO 9000 registration by June of next year”.

In the first objective example, Carliner has tracked the sales records in order to check whether the sales amount has really increased by 5%. He argues that the required elements for writing SMART objectives are specified correctly in this objective. For example, the level of performance (“measurable”) is stated (“5%”) in the objective. The deadline for achieving the objective (“Trackable”) is specified clearly in the objective (“June 30”). Furthermore, the observable criterion or action is indicated in the objective, such as this objective pertains to an increasing of the sales (“specific”).

Carliner also gives examples of some ineffective business objectives (non SMART objectives) such as:

- “Improve goodwill”.
- “Contain support costs”.

He argues that the first objective is not an effective business objective (not SMART objective), because it does not specify the observable criterion and does not describe to what it applies to (e.g. sales, revenue, costs, expenses…). In addition, the other elements that are needed for writing an effective business objective are missing, such as the level of performance and deadlines.

Rouillard (2003) describes the important elements for writing well-defined objectives by using the SMART approach. For example, he argues that an objective is "specific" if it can express the action to be achieved, since the objective must contain an action verb (e.g. increase, gain...) to express the accomplishment of the outcomes. An objective is "measurable" if the outcomes of the objective can be measured ("quantifiable") (e.g. 5%). An objective is considered "time- and resource-constrained" if it specifies the deadline to achieve the outcomes (e.g. August 15). An objective is "action-oriented" if it can produce results within the given deadline. In order to be "realistic", sufficient resources should be available to achieve the outcomes of the objective. The following examples are proposed by Rouillard to illustrate the required elements for writing SMART objectives:

- "Increase productivity in our division 5% by August 15, without adding any personnel".
- "Gain five new customers and increase gross sales to $20,000 by July 1 within an expense budget of $1,000".
- "Expand market share to 5% by December 31, without increasing advertising expense beyond current levels".

Williams and Curtis (2007) study the use of SMART objectives in the marketing domain. They state that the setting of effective objectives may affect positively on the marketing plan strategy, while the long term objectives are difficult to be achieved and they will not be SMART. The following examples represent some of the types of SMART objectives suggested by Williams and Curtis:

- "Profitability objectives – To achieve a 25 percent return on capital employed by August 2010".

- "Market share objectives – To gain 30 percent of the market for umbrellas by September 2009".
- "Objectives for growth – To increase the size of our operation from $200,000 in 2006 to $400,000 in 2009".

The study addressed by Rampersad (2001) introduces a visionary management model based on the development of the organisational mission, vision, objectives, and strategies which may improve an organisation's performance. His study proposes some examples of the organisational objectives based on the use of the SMART (specific, measurable, achievable, realistic, and time-specific) approach:

- "An increase of 6 percent in the market share of product B in South America within two years".
- "Achieve a market growth of 6-7 percent for the next two years".
- "Improve delivery reliability by 50 percent within six months".
- "Reduce the waste percentage from 3 percent to 0.5 percent within one year".

The above studies (e.g. Doran, 1981; Carliner, 1998; Rampersad, 2001; Rouillard, 2003; Holmes, 2005; Williams and Curtis, 2007; Hurd et al., 2008; Desmond, 2011) suggest the important elements for writing SMART objectives and provide some examples for showing the potential structure of the SMART objectives. In general, the objective should be "specific", where it must declare the action to be achieved, specify the accomplishment of the outcomes by using an action verb, and state the domain of the objective. The objective should be "measurable", and this requires stating an indicator of progress (i.e. numeric indicator of quality/quantity or measurement source) to measure the achievement of the objective. The objective should be "achievable", which means that the objective could be achieved within a given timeframe. The objective should be "realistic", which means that sufficient resources are available to achieve the objective. The objective should be "time-related", and this includes stating the deadline for achieving the objective. Later, in Chapters 3, 4, 5 and 6 of the thesis, these insights are used with a view to automatically check the presence of these features in written objectives.

2.2 Natural Language Processing and Information Extraction

This section describes natural language processing, its tasks and applications as well as the approaches that are used for natural language processing. Moreover, it provides a detailed overview of information extraction and its systems.

2.2.1 Natural Language Processing

Natural language can be ambiguous, such that many common words may have the same writing but indicate different meanings and multiple interpretations depending on the context in which these words occur. The fast growth of information and technologies on the Internet has increased the amount of unstructured data where this data is expressed in natural language. In addition, many knowledge resources available online such as blogs, surveys, articles, web pages, documents and corpora (collection of documents) are expressed in free (unstructured) texts written in natural language. This has increased the demand for software that analyses text of all forms to solve the ambiguity problem.

Automatic methods in natural language processing (NLP) (Manning and Schütze, 1999; Jackson and Moulinier, 2007; Jurafsky and Martin, 2009; Collobert et al. 2011) are normally applied to process the ambiguity in the natural language and free textual data in order to extract knowledge, summarise the available content and transform the unstructured textual data into structured textual information. In general, NLP is a field in artificial intelligence (AI) (Russell and Norvig, 2010) which is concerned with the interactions between computers and human natural language. It uses computer software and the state-of-the-art systems for analysing linguistic patterns and generating useful rules from texts in a meaningful way (e.g. learning grammar rules which can indicate whether a sentence is well formed). Modern NLP systems are based on the use of machine learning (Mitchell, 1997; Alpaydin, 2010) and statistical techniques for processing the language (more details about machine learning and its methods are surveyed in Section 2.3).

In the domain of NLP, machine learning is utilised to derive significant rules by analysing large text corpora of typical real-world data. In general, most of the machine learning methods in NLP utilise external knowledge resources to provide useful information which help in processing the textual data. These knowledge resources include the following:

- Machine-readable dictionaries, which are common to use in NLP for providing useful information (e.g. meanings of words, grammatical categories of words, relations etc) (Manning and Schütze, 1999). The WordNet lexicon (Miller et al. 1990; Fellbaum 1998) (described in more detail in Section 2.2.3) and Longman Dictionary of Contemporary English (LDOCE) (Proctor, 1978) are some examples of machine readable dictionaries.
- Corpora, which consist of sets of texts, used for learning concepts and language models. The text provided in the corpora can be raw text or annotated with word senses (i.e. semantics) and parts of speech (POS) tags. Both kinds of corpora represent a knowledge resource and could be used by the NLP approaches (Manning and Schütze, 1999). The Brown corpus (Kucera and Francis 1967), Wall Street Journal corpus (Charniak et al. 2000) and American National corpus (Ide and Suderman, 2006) are some examples of the raw corpora. The SemCor corpus (Miller et al. 1993) (i.e. a subset of the Brawn corpus that is manually tagged with senses from WordNet) and the interest corpus (Bruce and Wiebe, 1994) are some examples of corpora annotated with word senses and POS tags.
- Thesaurus, which provides information about relationships between words (e.g. synonym (words with the same meanings) and antonym (words that have the opposite meanings) relations) or word meanings (Kilgarriff and Yallop 2000). The Roget's International Thesaurus (Roget, 1911) is an example of a well-known thesaurus which has been extensively used in the NLP field.
- Ontologies, which are used in AI and NLP as knowledge resources, include definition of objects and semantic relations for a set of concepts within a specific domain (Gruber, 1995).

NLP includes tasks which are usually performed on unstructured text in order to transform it into a structured format. These tasks represent the pre-processing steps which are considered the key procedures in NLP for linguistic analysis. The pre-processing tasks extract relevant features from text, where these features can be used to describe syntactic information (e.g. grammatical structure, parts of speech etc) or semantic information (e.g. meanings). In other words, pre-processing tasks can be applied to perform the linguistic analysis of a given text including the syntactic and semantic analysis of this text (Manning and Schütze, 1999; Jackson and Moulinier, 2007; Jurafsky and Martin, 2009; Collobert et al. 2011). Some of the NLP tasks are grouped into subfields of NLP like information extraction (IE) (Appelt, 1999; Moens, 2006), or they may have direct and separate real-world applications such as information retrieval (IR) (Goker and Davies, 2009) (IE and IR are discussed in Section 2.2.2). The main NLP tasks include the following pre-processing steps: tokenization, part of speech (POS) tagging, lemmatization, named entity recognition (NER), chunking, parsing, co-reference, semantic class disambiguation (SCD) and word sense disambiguation (WSD) (Manning and Schütze, 1999; Jackson and Moulinier, 2007; Jurafsky and Martin, 2009; Collobert et al. 2011). In the following points, a brief description is presented for each of the above mentioned NLP tasks (the tasks of SCD and WSD are illustrated later in Sections 2.2.3 and 2.2.4 respectively):

- Tokenization, which splits the text into a set of tokens. For example, documents are broken into paragraphs, paragraphs into sentences, and sentences into individual words.
- Part of speech (POS) tagging, which provides a grammatical category for each word in the text (e.g. 'eat': verb, 'boy': noun, 'happy': adjective, 'widely': adverb, 'in': preposition etc). It has been widely proposed by many authors (e.g. Brill, 1992 and Hepple, 2000) as a main task for analysing the text syntactically at the word level. This task is useful for linguistic analysis and is normally performed on the tokenized text.
- Lemmatization (morphological analysis), which groups together different forms of a given word into a single word to represent the lexical root of this word (e.g. is → be, ran → run).
- Text chunking (shallow parsing), whose aim is to label segments of a sentence with syntactic constituents which called chunks. These chucks can be either verb phrase (e.g. have eaten) or noun phrase (e.g. the girl).
- Parsing, representing the process of generating the parsing tree in order to determine the syntactic structure of a sentence. This process specifies the rules

for POS that can generate a well-formed phrase.

- Named entity recognition (NER), a main task in IE, which classifies entity names in text into predefined categories. This task is able to identify expressions that refer to people (e.g. 'John'), locations (e.g. 'France'), organisations (e.g. 'Microsoft') etc. It uses heuristic rules that rely on the syntactic structures of the surrounding context. For example, the NER task has classified the word "John" into the "person" category, because it is a proper noun started with a capital letter and it has been expressed in a way that indicates a name of a person.
- Co-reference, an IE task, which links multiple expressions in text that refer to a given entity. By considering this sentence "John went to the school yesterday, he was in a holiday for three days", the task of co-reference states that the word "he" in the given sentence example refers to "John".

2.2.2 Applications of NLP

There are many applications for NLP such as IR, sentiment analysis, machine translation and IE. The following describes IR, sentiment analysis, and machine translation briefly in Sections 2.2.2.1, 2.2.2.2 and 2.2.2.3 respectively, while IE, its tasks and systems are discussed in more detail in Section 2.2.2.4.

2.2.2.1 Information Retrieval

Information retrieval (IR) (Goker and Davies, 2009) is based on searching information and documents in order to provide a user with the information requested in response to a query that includes the search terms. The task of IR is to compare the query terms with the index terms that appear in the documents. IR uses some NLP tasks such as stemming (reduce a word to its root form) and it could be applied as an NLP task for automatic document retrieval, which is mainly based on free text indexing (e.g. Lewis and Jones, 1996). The web search engines (e.g. Google) which use stemming algorithms are the most well-known IR applications for retrieving the needed data.

2.2.2.2 Sentiment Analysis

Sentiment analysis (opinion mining) (Pang and Lee, 2008) is the process of applying NLP techniques to extract opinions and subjective expressions from text by classifying the polarity of this text at the document, sentence or word level in order to determine whether the expressed opinions have a positive, negative or neutral sentiment. It has been extensively proposed for market analysis, product recommendation, and government intelligence applications.

2.2.2.3 Machine Translation

Machine translation (Weaver, 1955) is the process of using computer software to automatically translate one natural language into another by transforming source language text into target language. It applies different approaches to find linguistic rules that interpret all words in the text in a linguistic way by analysing all the linguistic features of the text (e.g. grammar, syntax, semantics etc). There are two main approaches of machine translation: rule-based and statistical machine translation. The rule-based machine translation approaches use grammar rules and dictionaries to find linguistic information (syntactic, morphological and semantic information) that maps the grammatical structure of the source language into the target language. On the other hand, statistical machine translation generates translations by applying statistical methods to previously translated large text corpora in order to translate similar or new texts (Weaver, 1955; Hutchins, 2007).

2.2.2.4 Information Extraction

2.2.2.4.1 Information Extraction and its Tasks

The need for processing large amounts of text, documents and databases, makes information extraction (IE) (Appelt, 1999; Moens, 2006) a significant subarea of NLP. IE is based on processing unstructured textual documents to extract structured, relevant and semantic information by creating machine readable text. The IE systems are useful and effective in many domains and applications. The Message Understanding Conference (MUC) (Lehnert and Sundheim, 1991) provides significant reference sources for understanding the evolution of IE systems and illustrates many domains and tasks of IE. Some researchers have used IE for identifying protein names in biological documents (Fukuda et al., 1998), extracting medical information from reports (Hahn, Romacker and Schulz, 2002) and finding business information from web pages (Sung and Chang, 2004). The main generic tasks for IE are named entity recognition (NER), template element task (i.e. extracting entities with their attributes) and co-reference identification (Appelt, 1999; Moens, 2006). Any IE system is focused on a set of extraction patterns that are used for pattern matching in order to match regular expressions in text and extract relevant information. There are different IE systems that are concerned with learning linguistic patterns or extracting rules automatically from training examples such as AutoSlog (Riloff, 1996), RAPIER (Califf and Mooney, 1999) and CRYSTAL (Soderland et al., 1995).

2.2.2.4.2 Information Extraction Systems

Most of IE systems are used to perform NLP tasks and provide an appropriate and annotated text which is then given as an input to a machine learning method for learning concepts and linguistic rules. A lot of free open source tools and systems for IE and NLP have been designed and implemented such as NLTK (Loper and Bird, 2002), OpenNLP (Baldridge, 2005), and GATE (Cunningham et al., 2013). These IE systems and tools are explained in more detail in the following paragraphs.

The natural language toolkit (NLTK) (Loper and Bird, 2002) is a free open source package implemented in Python. It provides interfaces for text processing, analysing linguistic structure and accessing a large collection of corpora. NLTK includes libraries and programs of NLP components such as tokenization, POS tagging, parsing, chunking, semantic analysis, classification and clustering. Different studies have employed the NLTK toolkit for processing natural language and extracting knowledge. For instance, McKenzie et al. (2010) present a study that is related to the domain of engineering for extracting data from helicopter maintenance records by using IE methods. Their study has used the NLTK toolkit for partial parsing of text. Moreover, the study addressed by Stoyanchev et al. (2008), which develops a question answering system, employs the NLTK toolkit to analyse questions linguistically. Particularly, the NLTK tool is utilised to extract noun, verb and prepositional phrases (phrase chunking) from two datasets: the AQUAINT corpus (Graff, 2002) (a collection of news documents) and the web.

The OpenNLP system (Baldridge, 2005) is another free toolkit for IE and linguistic analysis of texts. This open source tool is concerned with creating linguistic annotations of texts automatically based on the use of maximum entropy (Berger et al., 1996) statistical models. It involves many NLP tasks including tokenization, sentences segmentation, POS tagging, NER, chunking and co-reference. The OpenNLP toolkit has been used in different studies and domains. For instance, the study by Kang et al. (2011) applied the OpenNLP system on a biomedical corpus for performing noun phrase chunking (NP) and some other tasks including tokenization, sentence splitting and POS tagging. They compared the OpenNLP system with different IE systems for the task of phrase chunking. They demonstrated that it has achieved the best score of performance in producing the annotations of phrase chunking. Ek et al. (2011) describe an approach for named entity detection in an SMS corpus written in Swedish. They have used the OpenNLP toolkit to annotate the SMS corpus by applying the tokenization and POS tagging tasks on the corpus.

The GATE (General Architecture for Text Engineering) (Bontcheva et al., 2004; Cunningham et al., 2013) architecture is a publicly available system, implemented in Java (Gosling, Joy, Steele, and Bracha, 2005) and developed at the University of Sheffield for processing natural language. It is platform independent, provides facilities for text processing, annotating, defining ontologies and using them for semantic annotation. The GATE system has been employed in many domains such as biological domain (Khelif, Dieng-Kuntz, and Barbry, 2007), tourism domain (Feilmayr et al., 2009) and business analysis domain (Saggion et al., 2007). In IE applications, GATE is run over a corpus of texts to analyse text and generate text annotation automatically. The annotations can be also edited or created manually in the graphical user interface (GUI) of GATE. Each annotation has a type and a set of features and values, as well as it refers to an annotation set. The GUI of GATE provides facilities for configuring pipelines by adding and removing components.

The GATE architecture is developed in the IE component set called ANNIE (A Nearly-New Information Extraction). ANNIE IE system contains a set of processing resources that implement algorithms for extracting information from unstructured text. These processing resources can be applied individually or combined together with new plugins to create annotations automatically. Some of the main processing resources provided by the ANNIE plugin include: ANNIE English Tokenizer, ANNIE Gazetteer, ANNIE sentence splitter, ANNIE part-of-speech (POS) tagger, ANNIE named entity (NE) transducer, Java Annotations Pattern Engine (JAPE) transducer, and Orthographic co-reference (ANNIE Orthomatcher) (Cunningham et al., 2013).

The ANNIE English Tokenizer splits the text into a set of tokens in order to illustrate the annotation features such as token kind (e.g. words, symbols, numbers, punctuation and space token), token category (e.g. verb in the base form (VB), singular noun (NN), plural noun (NNS), preposition (IN) etc), length of token, the string that is covered by the token, and orthography (orth) (e.g. lowercase/uppercase), as well as the feature values. The ANNIE Gazetteer finds occurrences of entity names in text by using a set of lists (e.g. units of currency, days of the week, organisations, cities etc) to create Lookup annotations. Basically, the ANNIE Gazetteer creates a Lookup annotation if an entry from a Gazetteer list matches some text in a corpus. The Lookup annotation has two features, majorType and minorType. The majorType feature must be specified in the Lookup annotation for describing essential information about patterns, whereas the minorType feature specifies only optional information. The ANNIE sentence splitter divides the text into sentences. The ANNIE POS tagger is the Hepple part of speech tagger (Hepple, 2000) (a modified version of the Brill tagger (Brill, 1992)), which performs a word level syntactic analysis to provide a grammatical category (POS tag) for each word/symbol in text. The ANNIE NE transducer classifies entity names in text into predefined types (e.g. organisation, person, location). The JAPE transducer executes handcrafted rules in order to match textual patterns and regular expressions in text annotations. The ANNIE Orthomatcher identifies the words in text that refer to the same entity by finding relationships between text entities. GATE also includes various plugins such as the noun phrase (NP) Chunker and machine learning.

Different studies have showed the usefulness of using the GATE architecture. Feilmayr et al. (2009) present an approach that implements a rule/ontology-based IE system for analysing tourism websites and extracting structured data from accommodation web pages based on the use of the GATE system. In particular, GATE is used to perform text annotation of the web pages in the utilised corpus (a set of accommodation web pages) in which the processing resources such as ANNIE English Tokenizer, ANNIE Gazetteer, ANNIE sentence splitter and JAPE transducer are arranged in a pipeline then run over the corpus. The study by Saggion, et al. (2007) develops an ontology-based IE system for the business domain based on the use of the standard and adapted processing resources in GATE. The developed IE system has been applied to a number of information sources including company profiles, country or region profiles, company websites and newspaper articles. They have used the standard GATE components for text annotation and also developed some new Gazetteer lists and NER processors for identifying the text types and mapping concepts identified by their application into ontological classes. A recent study that utilises GATE for IE is the one presented by Joshi et al. (2012), which shows the benefits of using social networking sites like twitter on the marketing domain. Their study develops a domain specific NER for classifying named entities in the posts of twitter which are written by buyers and sellers. Accordingly, these posts (a collection of tweets) are analysed and processed by using the GATE components (e.g. ANNIE English Tokenizer, ANNIE sentence splitter, ANNIE POS-tagger, ANNIE Gazetteer and ANNIE NE transducer) in order to extract information that is suitable for generating useful suggestions for farmers and merchants.

2.2.3 Semantic Class Disambiguation

This section illustrates the task of semantic class disambiguation (SCD) in more detail and presents some studies that apply this task.

Classification of lexical semantic information is an important task in NLP and IR systems. Many NLP systems, that perform semantic analysis of texts, use knowledge resources like machine readable dictionaries to retrieve useful information such as meanings of words, semantic classes of words, and relations between word forms or word meanings. Semantic class disambiguation (SCD) (Manning and Schütze, 1999) is a partial task for WSD, so by using this task, words in texts can be classified into semantic classes in terms of their syntactic and semantic features. Moreover, the task of SCD extends the task of NER, since the classification process in NER is restricted to the entity names of proper nouns only (e.g. location, person, organisation etc). On the other hand, some machine readable dictionaries have the ability to classify both proper and common nouns as well as verbs, adjectives and adverbs into semantic classes (e.g. animal, possession, event, body etc) (Ciaramita and Altun, 2006).

The lexical semantic information available in machine readable dictionaries is used to label the words in text by using a set of semantic class labels specified by these dictionaries. For example, the WordNet lexicon, which is the most used knowledge resource for NLP, is utilised to disambiguate the words at the semantic class level by retrieving the semantic categories of words.

WordNet (Fellbaum, 1998; Miller, et al. 1990) is a large machine-readable lexicon for English part of speech, developed at Princeton University[1]. It is widely used in NLP and IR applications for retrieving the meanings (senses) of words from the WordNet database. The WordNet lexical semantic resource represents the sense inventory, as it has been employed in many WSD systems to provide the possible sense for each word in text. The WordNet database consists of four categories: nouns, verbs, adjectives and adverbs. Words in WordNet are organised into sets of synonyms called synsets. Each synset represents a concept and consists of a group of synonymous words together with a definition of the word sense and some example sentences. The definitions of words in WordNet called glosses. Several relationships between synsets or words are encoded in WordNet including POS relations, lexical relations and semantic relations. Lexical relations represent the relations between the word forms (e.g. synonyms and antonyms), while the semantic relations represent the relations between the word meanings (synsets). Some of the semantic relations are hypernymy/hyponymy relations (is a kind of relation, e.g. ‘vehicle’ is a hypernym of ‘train’, and ‘train’ is a hyponym of ‘vehicle’) and meronymy/holonym relations (is a part of relation, e.g. ‘wheel’ is a meronym of ‘car’, and ‘car’ is a holonym of ‘wheel’). Nouns and verbs are organised into a hieratical structure in WordNet in terms of the hypernymy/hyponymy relations between synsets.

Lexicographers (lexicographer files) in WordNet organise synsets into many semantic classes depending on their syntactic categories (noun, verb, adjective, and adverb) and logical groupings. WordNet has 26 lexicographer files for nouns (e.g. possession, food, animal, feeling etc), 15 for verbs (e.g. change, emotion, consumption, perception, possession etc), 3 for adjectives and 1 for adverbs. The WordNet lexicographers group together many synsets, where some nouns or verbs can share the same semantic class label. For example, the first senses of the nouns “cat” and “bird” are classified semantically into the same noun semantic class label called “animal”. Furthermore, the first senses of the verbs “feel” and “hate” are classified semantically

[1] http://wordnet.princeton.edu

into the same verb semantic class called “emotion”.

A word that belongs to several meanings is ambiguous. Classifying the correct sense or semantic class of a word is based on the context in which it occurs in a sentence. For instance, the word “book” has many senses and semantic class labels in WordNet. This word (“book”) can have the semantic class label: “communication”, “artefact”, “possession”, or “group” if it appears as a noun in a sentence, while it can have the semantic class label “social” or “cognition” if it appears as a verb in a sentence.

Several studies have utilised the WordNet lexical database for classifying the semantic classes of synsets. For example, Ciaramita and Johnson (2003) present an approach for supersense tagging by using WordNet. Their approach aims to automatically determine the semantic classes of common nouns based on the lexicographer class labels (supersenses) available in the WordNet lexical database. The Bllip corpus (Charniak, et al., 2000) is used in their experiments, where it has been provided as an input into a multiclass averaged perceptron classifier (Crammer and Singer, 2003) for class label classification. All occurrences of collocation (words often occur together), spelling/morphological and syntactic context features for common nouns are extracted and mapped into one of the class labels available in WordNet. The study by Ciaramita and Altun (2006) describes an approach that uses a supersense tagger for annotating the text with the tagset of lexicographer classes existing in WordNet. This approach is based on sequence labelling of words in text by using the perceptron trained Hidden Markov Models (HMMs) algorithm (Collins, 2002). The classification task is built on analysing the morphological and contextual features of word senses. The experiments of the applied supersense tagger have been evaluated on Semcor (Miller et al., 1993) and Senseval 3 (Snyder and Palmer, 2004) datasets and achieved F-score rate[2] of 77.2% and 70.5 % for each dataset respectively. Kohomban and Lee (2005) apply a task for learning coarse-grained semantic classes from a sense-tagged corpus (i.e. Senseval 2 (Edmonds and Cotton, 2001) and Senseval 3 datasets) based on the lexicographer files available in WordNet. The applied coarse-grained classifiers are mapped into fine-grained senses by using a heuristic mapping. The empirical findings showed that the performance of their

[2] F-score (F-measure) is a measure for testing the accuracy of a system. See Section 2.5.

system was promising. The research proposed by Curran (2005) presents an approach for supersense tagging of unknown nouns. The proposed approach uses vector-space semantic similarity in order to return synonyms for each unknown common noun based on their supersenses in WordNet. The supersenses of these synonymous words of common nouns are grouped to specify the relevant supersense by comparing the contexts in which each word appears. Korhonen (2002) applies a method for semi-automatic semantic classification of verbs into Levin classes (Levin, 1993)[3] using the semantic classes of WordNet. The applied method categorises the WordNet senses into semantic classes and classifies verbs semantically in terms of their most frequent sense in WordNet. Korhonen has applied a verb classification algorithm to three WordNet sub hierarchies (contact, motion and possession verbs). The algorithm has classified 181 verbs correctly, while 43 verbs have been classified incorrectly. The overall accuracy of the applied algorithm is 81%.

2.2.4 Word Sense Disambiguation

This section describes the word sense disambiguation task and its approaches as well as the methods that are used to compare the performance of word sense disambiguation approaches.

2.2.4.1 Description of Word Sense Disambiguation

Text disambiguation can be utilised to representations of word contexts. It is based on semantic analysis which can provide better results in retrieving the required information from large amounts of data formulated with different wordings more than the syntactic analysis of text. Word sense disambiguation (WSD) (Agirre and Edmonds, 2006; Navigli, 2009) is one of the main problems in NLP used for disambiguating text and resolving semantic ambiguity. It plays a critical role as a classification task for automated translation of text, since in the late 1940s; it was considered as a major task for machine translation (Weaver, 1955). WSD has been addressed by many researchers (e.g. Mooney, 1996; Towell and Voorhees, 1998; Escudero et al. 2000) who have used the state-of-the-art techniques to identify the meanings of words in text. It has also been applied in some potential real applications such as machine translation (e.g. Carpuat and Wu, 2007), IR (e.g. Schütze and

[3] Levin's classification of verbs is the largest verb classification in English which groups verbs into sets of classes based on syntactic properties.

Pedersen, 1995) and IE (e.g. Ciaramita and Altun, 2006). The task of WSD involves examining contextual information to find the correct sense of a target word in a sentence by assigning the most related one to this word depending on the context in which it occurs. The context is determined by the other words in the neighbourhood in the same sentence, so that every sense of the target word to be disambiguated is compared to the senses of the surrounding words (Agirre and Edmonds, 2006; Navigli, 2009).

By applying NLP tasks, a set of significant features could be extracted and used for WSD in order to describe the context of a given text. These features include the following (Agirre and Edmonds, 2006; Navigli, 2009):

- Local features, which represent the features of words surrounding the target word (e.g. POS tags, lemmas (word forms) and positions of words that surround the target word).
- Topical features, which specify more general contexts such as identifying the general topic of a text (bag of words).
- Syntactic features, which illustrate the grammatical relations between the target word and the surrounding words in text.
- Semantic features, which identify semantically the senses of words in a context for a given text.

Based on the extracted features, the occurrence for each word can be used as a feature vector to determine the context of text and disambiguate the ambiguous words. Different sizes for a sequence of words could be used to determine the size of context including the target word. This process symbolizes the n-gram language models in which words in a sentence are assumed to occur independently. For instance, unigrams (one word) represent the occurrence of the target word in text. Bigrams (sequence of two words) represent the occurrence of the target word and one of the surrounding words (either to the left or right of the target word) in text. Trigrams (sequence of three words) represent the occurrence of the target word and two of the surrounding words in text. The context size may increase up to a full paragraph including the target word (Agirre and Edmonds, 2006; Navigli, 2009).

The state-of-the-art systems of WSD use knowledge resources (e.g. thesaurus, machine-readable dictionaries, ontologies etc) to provide the required data for assigning senses to words in text.

There are two main types of WSD approaches that can be applied to solve the ambiguity of words in text: the supervised WSD and unsupervised WSD approaches. The supervised WSD approaches disambiguate the senses of words from labelled corpora, while the unsupervised WSD approaches identify the meanings of words from unlabeled corpora. These two approaches (i.e. supervised and unsupervised WSD approaches) are discussed in Sections 2.2.4.2 and 2.2.4.3 respectively.

Methods in WSD can be also categorised into knowledge-based approaches or corpus-based approaches, where this type of categorisation depends on the basis of using knowledge resources or not when disambiguating the senses of words in texts. For example, the knowledge-based approaches (Mihalcea, 2006) disambiguate the senses of words by using information from machine readable dictionaries, thesaurus, ontologies etc. On the other hand, the corpus-based approaches do not use any knowledge resources in the disambiguation process, since these approaches disambiguate the senses of words by using information from a training corpus (Manning and Schütze, 1999). An overview of the knowledge-based approaches is presented in Section 2.2.4.4.

2.2.4.2 Supervised WSD Approaches

The common used approaches in NLP for WSD are the supervised approaches which classify the senses of words in context from sense-annotated corpora. These approaches provide better results in disambiguating the senses of words more than the unsupervised WSD approaches (Agirre and Edmonds, 2006; Navigli, 2009). Some of the highest-performing approaches of supervised WSD are Naïve Bayes (e.g. Mooney, 1996), neural networks (e.g. Towell and Voorhees, 1998) and *k*-nearest neighbor (e.g. Ng, 1997). These approaches are discussed in more detail below.

2.2.4.2.1 Naïve Bayes

The Naïve Bayes (Domingos and Pazzani, 1997; Mitchell, 1997) classifier has been widely utilised for disambiguating the senses of ambiguous words in text. It is based on calculating the conditional probabilities for each sense of a word given a set of features (POS tags, neighbouring words, positions of words etc) in context. Naïve Bayes assumes that all features are conditionally independent given the word sense. The frequent sense that has the most features occurrence is normally chosen to be the appropriate sense in context for an ambiguous word.

Many researchers have utilised Naive Bayes classifier for disambiguating the meanings of ambiguous words in text. For instance, Mooney (1996) employs Naive Bayes and some other learning methods for the problem of disambiguating the senses of words in context. The performance of the applied models is evaluated on the line corpus (Leacock et al., 1993). Results showed that the Naive Bayes classifier has achieved better accuracy than the alternative methods such as decision trees (Quinlan, 1993) and rule based techniques (Michalski, 1983) for all training set sizes. A study addressed by Pedersen (2000) utilises Naive Bayes classifiers for WSD. The approach used in this study combines a number of Naive Bayes classifiers into an ensemble, such that each classifier is based on the co–occurrence of lexical features in a variety of context window sizes. The specified lexical features determine if a given word appears within some number of surrounding words. Pedersen's approach has been evaluated by using 5 fold cross validation[4] on the line data and the interest data (Bruce and Wiebe, 1994). It has achieved a high accuracy in disambiguating the senses which is around 88% and 89% for each data respectively.

2.2.4.2.2 Neural Networks

The neural networks (NNs) (McCulloch and Pitts 1943; Mitchell, 1997) model is constructed from a set of connected input/output units (artificial neurons) that exploits a computational model for information processing. The classification task in neural networks for WSD is based on providing the features as an input to the learning model. Then, these features are used for splitting the training contexts into non overlapping sets related to the desired responses. By processing the data, the weights of units are adjusted to produce the desired response by the output unit that has a greater activation than any other output unit.

Several studies have applied neural networks for the WSD problem. For instance, the

[4] Cross validation is a statistical technique for evaluating the accuracy of a system. See Section 2.5.

study presented by Towell and Voorhees (1998) develops a neural network classifier for using the semantic information available in lexicons such as WordNet to improve the performance of IE systems. The developed classifier learns independently the topical and local context features of a given target word from a set of sense-annotated example sentences. Then, it combines these features into contextual representations for WSD. The classifier has been evaluated on three data: the noun line data, the verb serve data and the adjective hard data. The experiments showed that it has achieved a high accuracy which is around 87%, 90%, and 81% for each data respectively. V´eronis and Ide's (1990) approach constructs a neural network model from the definition texts available in a machine readable dictionary. This approach assigns each word to its correct sense and the senses are connected to the words appearing in their textual definitions.

2.2.4.2.3 *K*-Nearest Neighbor

The *k*-nearest neighbor (*k*-NN) (Mitchell, 1997) algorithm is one of the most used methods in WSD. It is built from examples, where each example has a set of feature values. The classification of new example is based on estimating the distance between the new example and the stored examples. The set of the closest examples is selected. Then, the new example is attached to the class (i.e. sense) that assigned to the most examples within the set.

The study presented by Escudero et al. (2000) compares two supervised learning methods for WSD: Naïve Bayes and exemplar–based classifiers. In their approach, the implementation of the exemplar-based classifier is based on the *k*-nearest neighbor algorithm. They tested the applied learning methods on the DSO corpus (Ng and Lee, 1996), which includes 192,800 sense-annotated tokens of 191 words (i.e. 121 nouns and 70 verbs). The experiments showed that the performance of the exemplar–based approach outperforms the performance of the Naïve Bayes classifier. Ng (1997) suggested an exemplar-based learning approach called PEBLS for WSD. The PEBLS algorithm is an implementation of the *k*-nearest neighbor algorithm. It has been used for determining the best *k* (number of nearest neighbors) to use for disambiguating a word in a specific training data. Ng compares the exemplar-based classifier to the Naïve Bayes algorithm, where both algorithms have been tested on a large sense-annotated corpus (Ng and Lee, 1996). By using 10 fold cross validation, the experimental evaluation of the algorithms performance showed that the exemplar-based algorithm has achieved higher disambiguation accuracy than the Naïve Bayes algorithm.

2.2.4.3 Unsupervised WSD Approaches

As described before, the unsupervised WSD approaches classify the senses of words from untagged corpora and they do not use any machine readable dictionaries when classifying the senses of words. The disambiguation process of word senses using such approaches is based on grouping the word occurrences in a given text into clusters in order to classify new occurrences into the learned clusters (Manning and Schütze, 1999; Pedersen, 2006). The common approaches of unsupervised WSD are the context clustering (e.g. Schütze, 1998) and the word clustering methods (e.g. Lin, 1998). These two approaches are illustrated below in Sections 2.2.4.3.1 and 2.2.4.3.2.

2.2.4.3.1 Context Clustering Methods

In the context clustering methods, the occurrences of a target word in a text are represented as context vectors. These vectors are then classified into clusters, each representing a sense of the target word (Manning and Schütze, 1999; Pedersen, 2006). The study presented by (Schütze, 1998) describes an unsupervised WSD approach based on the context clustering. Schütze's approach uses a clustering algorithm called context-group discrimination to gather the target word occurrences into a set of clusters for the word senses depending on the contextual similarity between these occurrences. Hence, the context vectors presented by Schütze are second order context vectors that represent an instance by estimating the average of context vectors of word features which occur frequently in the context of the target word. Another study proposed by Pedersen and Bruce (1997) applies unsupervised learning algorithms for WSD. The study uses agglomerative clustering algorithms in which the observations of features are grouped into clusters in order to induce clusters that minimise the distances between the features of each cluster. The most similar pair of clusters is combined together to form a new cluster, and the processes is repeated until a few number of clusters remain. Purandare and Pedersen (2004) employ an approach of unsupervised WSD that augments the occurrence feature vectors of content words (surrounding words) that appear in the context of a target word to be disambiguated with the content words which occur in the glosses (definitions) of its senses (meanings) found in a machine readable dictionary (WordNet). The average of these feature vectors is then estimated to create a single vector that represents the context of the target word.

2.2.4.3.2 Word Clustering Methods

Word clustering techniques are based on grouping and clustering the words that are semantically close and similar. An approach described by Lin (1998) utilises a word clustering algorithm to discriminate between word senses by determining the similarity between the words. The applied algorithm identifies the words that are similar to a target word depending on the information content of the words' features in terms of the syntactic dependencies between words. Pantel and Lin (2002) present another word clustering method called the clustering by committee (CBC) algorithm to automatically determine the senses of words in context. Their approach calculates the similarity between a set of words and a target word by representing each word as a feature vector, where each feature expresses the syntactic context of the word. By using the CBC algorithm, the centroid of a cluster is created by averaging the feature vectors of a subset of the cluster elements. Then, the words are assigned again to their most similar clusters. After assigning a word into a cluster, the overlapping features are removed from the word representation; where this allows the CBC algorithm to identify less frequent features of the word.

2.2.4.4 Knowledge-Based Approaches

The main knowledge-based methods which utilise knowledge resources when disambiguating the senses of words in text include: the methods of overlap of sense definitions and the structural approaches (Manning and Schütze, 1999; Mihalcea, 2006). These approaches are described in more detail in Sections 2.2.4.4.1 and 2.2.4.4.2 respectively.

2.2.4.4.1 Overlap of Sense Definitions

The most well-known knowledge-based approach for WSD is the Lesk approach (Lesk, 1986), which is also used as a good baseline method in the evaluation of performance of the other WSD systems. This approach is based on the overlap strategy which finds the shared vocabulary between the words' definitions (bag of words). It computes the overlaps (i.e. number of words in common) between the sense definitions of a target word and the other sense definitions of words in the neighbourhood (context). The sense of the target word whose definition has the largest number of words in common (overlaps) is chosen to be the right sense of that target word.

The approach introduced by Banerjee and Pedersen (2003) presents a measure for semantic relatedness between concepts based on the use of the Lesk algorithm which has been adapted to WordNet. The idea is to extend the comparison between the sense definitions (glosses) of words to include the definitions of concepts (relations) that are related according to the hierarchies of concepts in WordNet (e.g. hypernyms, hyponyms, holonym etc). The authors have evaluated their approach on the Senseval-2 data and the algorithm achieved an overall accuracy of 34.6%. They demonstrated that the extended Lesk algorithm has achieved better disambiguation accuracy than the original Lesk algorithm (the achieved accuracy by the original Lesk algorithm was 18.3%). Vasilescu et al. (2004) implement different variations of the Lesk algorithm which have been adapted to the WordNet for WSD. They demonstrated that the strategy of an algorithm variation depends on counting the overlaps between the bag of words (BOW) of a target word to be disambiguated and the BOW of its context, where the selected sense of the target word is the one that has the highest number of overlaps. The variation approaches have been evaluated on the Senseval-2 data and the results show that these variants outperform the original Lesk algorithm. The best precision rate[5] that has been achieved by the implemented disambiguation approaches is 58%. A new study presented by Zouaghi, Merhbene and Zrigui (2012) develops a system that combines some information retrieval measures with the Lesk algorithm for Arabic WSD. The experimental evaluations of their system have shown an accuracy of 78% on collected corpora from the web (e.g. Wikipedia, Quran Arabic Corpus etc).

There are some open source systems which have been implemented to solve the ambiguity of words in text by specifying the correct sense of each target word in context. One of these systems is the free available Perl package called WordNet::SenseRelate::AllWords (SR-AW) (Pedersen and Kolhatkar, 2009). This system utilises different semantic relatedness measures including the extended '*lesk*'

[5] Precision is a statistical classification function for evaluating the performance of a system. See Section 2.5.

measure (extended gloss overlaps) (Banerjee and Pedersen 2003) which is one of the main knowledge-based approaches for WSD. The SR-AW software assigns senses to all words in text by using information from a sense inventory provided by the WordNet lexical database.

The web interface of SR-AW enables the user to enter a sentence or an input file containing a text to be processed and disambiguated. The text in the input file can be untagged (raw) or tagged either with WordNet tags or with the Penn Treebank POS tags (Santorini, 1990). The SR-AW software applies ten measures to estimate the similarity and relatedness between words. The similarity measures use the hierarchical information available in WordNet to compare two words (either noun to noun or verb to verb). Some of these similarity measures include the '*lch*' measure (Leacock and Chodorow, 1998) and '*jcn*' measure (Jiang and Conrath, 1997). The '*lch*' measure is based on finding the shortest path between two concepts based on the hierarchies of concepts in WordNet, while the '*jcn*' measure combines the information content values of two concepts with the information content of individual concepts to estimate values that specify the semantic distance between words.

On the other hand, the semantic relatedness measures include the adapted '*lesk*' measure (Banerjee and Pedersen, 2003), context vectors (Patwardhan, 2003), and the '*hso'* (Hirst and St-Onge) measure (Hirst and St-Onge, 1998). The context vectors compare sense definitions of words to create a vector space model for measuring relatedness between words, while the '*lesk*' measure uses gloss overlaps for string matching. The 'hso' measure is based on estimating lexical chains and finding paths between concepts in order to distinguish the concepts that are semantically related (see structural approaches in Section 2.2.4.4.2).

It is possible for the user of the SR-AW software to provide a stoplist file or use the default one. This file contains regular expressions, where any word in the text that matches one of these regular expressions is removed. Moreover, the SR-AW algorithm allows the user to fix the senses and choose a value for the disambiguation scheme. The user has also the option to control the size of the context window. The context window specifies which words are involved in estimating the relatedness between words. Generally, a context window of size N consists of the target word which is the word in the centre of the context window (centre word) and then expands out to the left and right of the target word for N/2 content words. By utilising the SR-AW disambiguation algorithm, all possible senses of the target word to be disambiguated are calculated for relatedness to the possible senses of the surrounding words (context words) in the context window. Then, the sense of the target word that has the highest score for relatedness to the senses of the context words is considered to be the correct sense of that target word (Michelizzi, 2005). The score of a sense s_i is computed as follows:

$$Score_{s_i} = \sum_{j,k} \max_k relatedness(s_i, s_{jk}) \tag{2.1}$$

where s_{jk} is the k-th sense of j-th word around the target word. The result of SR-AW depends on the selected relatedness measure and the size of context window.

After the SR-AW algorithm disambiguates the senses of words in context, it generates the output. The output displays the words that appear in the given text with the WordNet sense tags assigned. The WordNet sense tags include the word, the POS tag, the sense number and the sense definition for each of the assigned sense.

The performance of the SR-AW algorithm has been evaluated by Pedersen and Kolhatkar (2009). Their experiments were performed using three corpora (SemCor, SENSEVAL-2 and SENSEVAL-3) which have been manually tagged with senses from WordNet. They have used three sizes of the context window (2, 5 and 15) and three relatedness measures (lch, jcn and lesk) for the three corpora in the experiments. The results show that the disambiguation process using the '*lesk*' measure provides the highest precision and recall[6] for the three corpora. The estimated f-measure rate is 61% for SemCor, 59% for SENSEVAL-2 and 54% for SENSEVAL-3.

The SR-AW software has been utilised successfully in different problems and studies for the task of WSD. For example, the study proposed by Klyuev and Oleshchuk (2010) demonstrates a task for semantic query expansions. Their study utilised the SR-AW system and the WordNet lexicon to detect the meanings of the keywords of

[6] Recall is a statistical measure for evaluating the performance of a system. See Section 2.5.

the user query and check the semantic similarity between the user queries and the retrieved sentences of documents. In the approach presented by Shabanzadeh et al. (2010) for query expansion, the SR-AW software is also employed for disambiguating the query terms entered by a user. They evaluated their approach by computing the precision and recall rates, and then comparing them with keyword based information retrieval. Their experimental evaluations have achieved promising results.

2.2.4.4.2 Structural Approaches

Another kind of the knowledge-based methods is the structural approaches. These approaches exploit the structure of concepts available in computational lexicons such as WordNet to classify data based on the structural relations of features. Some approaches of this type are the similarity-based and graph-based techniques (Agirre and Edmonds, 2006; Navigli, 2009).

The similarity-based approaches apply several semantic similarity measures to estimate the similarity between concepts (or word senses) based on information available in a semantic lexicon such as WordNet. As WordNet organises the words into hierarchal relations of concepts, the measures of semantic similarity are used to compare the similarity between pair of concepts (or word senses) by estimating the shortest semantic distance (path) between them in the WordNet hierarchical structure. Many of the introduced similarity measures (e.g. the '*lch*' measure (Leacock and Chodorow, 1998) and the '*jcn*' measure (Jiang and Conrath, 1997)) have been implemented in the freely available Perl package called WordNet::Similarity (Pedersen et al. 2004) to compute the semantic similarity between words. The implemented modules of these measures take two concepts as input, and retrieve a numeric value which indicates the degree of similarity between these concepts. The WordNet::Similarity package utilises the WordNet::QueryData (Rennie, 2000) to access the WordNet database for creating objects.

The WordNet::QueryData (Rennie, 2000) module is a Perl interface that provides a direct access to the WordNet database files. It has the ability to retrieve semantic (e.g. hypernyms, hyponyms, domain categories etc) and lexical (e.g. antonym, verb group etc) relations from WordNet by using a set of functions (discussed later in more detail in Chapter 4). This Perl module has been applied successfully in many studies. For instance, Wubben (2010) suggests an approach for ranking a set of verbs paraphrasing noun compounds by using features from WordNet and the Google N-gram corpus. In particular, his study applies WordNet::QueryData to determine the kind of semantic relations between noun compounds. Furthermore, the developed question answering system by Whidden (2005) uses WordNet::QueryData to access the WordNet database for processing relations and entities in order to retrieve semantic relations (e.g. hypernyms) that help in assigning a category for each question (expression).

With regards to the graph-based techniques, there are many approaches that are built on the use of the graph-based strategy for disambiguating the senses of words in context. The approaches of this type are mainly inspired by the notion of lexical chain (Morris and Hirst, 1991). The lexical chain is used to determine the context and the semantic relations (e.g. is-part-of relation, is-a kind of relation) between semantically related words (e.g. vehicle → car). Some researchers have applied different algorithms for computing the lexical chains between words in order to determine the semantic relations between them (e.g. Hirst and St-Onge, 1998).

2.2.4.5 Methods Used to Compare the Performance of WSD Systems

The functions that are used to estimate the performance of the WSD systems or any machine learning algorithms are presented in Section 2.5. The performance of the state-of-the-art WSD approaches is usually compared with the well-known system called baseline (Gale et al., 1992). The main two approaches for the baseline performance are the random baseline and the first sense baseline. The random baseline computes the average of the random choices for a sense from the senses available for each word in a corpus. On the other hand, the first sense baseline system assigns the first sense to each ambiguous word based on the ranking of each word in a corpus. For example, the senses of a word in WordNet are ranked based on the occurrences for each sense in the SemCor corpus, where the most common sense in this corpus is associated to each word.

2.3 Machine Learning and Data Mining Methods

This section presents an overview of machine learning and data mining and also summarises some of the open source systems that support various data mining and machine learning processes.

2.3.1 Machine Learning

Machine learning (ML) (Mitchell, 1997; Alpaydin, 2010) is the core of AI which is concerned with computer programs that can learn to optimise performance using training examples and past experience. It employs computational and statistical techniques to construct mathematical models that find and exploit regularities and patterns in the training examples.

In general, there are different types of machine learning methods, including: inductive learning, deductive learning and learning by analogy (Xue and Zhu, 2009). Inductive learning is the process of transforming specific examples in some concept into general concept description (Xue and Zhu, 2009). In particular, it uses computational and statistical methods to extract rules and patterns from datasets. On the other hand, deductive learning is based on transforming the general concepts into logically specific examples by analysing the existing knowledge in order to find the most useful information (Xue and Zhu, 2009). Learning by analogy is an inference method in machine learning that can demonstrate the similarity between objects by transforming the information from the source to the target (Xue and Zhu, 2009). Most methods in machine learning are based on inductive learning or learning by analogy methods.

Machine learning methods include supervised learning, unsupervised learning and semi-supervised learning. Supervised learning is able to learn a classifier from training examples annotated by human experts, whilst unsupervised learning learns from unlabeled training examples. Semi-supervised learning uses labelled and unlabeled training data to learn a classifier. Some examples of machine learning methods include: inductive logic programming (Muggleton and De Raedt, 1994) (discussed in Section 2.4), Naïve Bayes (Domingos and Pazzani, 1997; Mitchell, 1997) and neural networks (McCulloch and Pitts, 1943; Mitchell, 1997). Several machine learning approaches have been applied successfully to different fields and problems such as NLP (e.g. Ratnaparkhi, 1999; Saito and Hagiwara, 2010), bioinformatics (e.g. Larrañaga et al. 2006) and sentiment analysis (e.g. Ye, Zhang and

Law, 2009).

2.3.2 Data Mining

The huge amount of data in the world is increasing by time and the necessity to turn this data into useful information is highly desirable. An important subfield of computer science is knowledge discovery in databases (KDD) (Fayyad, Piatetsky-Shapiro and Smyth, 1996a; Fayyad, Piatetsky-Shapiro and Smyth, 1996b; Han, Kamber, and Pei, 2011; Witten, Frank, and Hall, 2011), which uses techniques for extracting useful and understandable information from the rapidly growing volumes of data. The process of KDD includes many steps such as data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation and knowledge presentation.

Data mining (Fayyad, Piatetsky-Shapiro and Smyth, 1996a; Fayyad, Piatetsky-Shapiro and Smyth, 1996b; Han, Kamber, and Pei, 2011) is an essential step in KDD that applies algorithms to discover interesting data patterns hidden in the large amounts of raw data, where this data can be stored in databases, data warehouses or any other information repositories. It combines methods from AI, machine learning, statistics and databases to analyse and summarise data into structured information. The information and knowledge discovered by data mining methods can be used in different applications such as business management, market analysis, science and health care.

2.3.3 Common Methods in Machine Learning and Data Mining

The above two sections described machine learning and data mining. In general, there are a few differences between machine learning and data mining, where machine learning includes data mining, statistical and theoretical computer science methods, while data mining is a subfield of machine learning. Moreover, the output of some machine learning methods is more like complex rules represented in a logical form, while the output of data mining is represented in a form of decision trees, simple rules (if-then rules) or mathematical formula (Mitchell, 1997; Alpaydin, 2010; Langley and Simon, 1995; Mannila, 1996; Fayyad, Piatetsky-Shapiro and Smyth, 1996a; Fayyad, Piatetsky-Shapiro and Smyth, 1996b; Han, Kamber and Pei, 2011).

Some of the common data mining and machine learning tasks are classification, prediction, clustering and association. The most popular methods in data mining and machine learning include: rule induction learners, decision tree induction, neural networks, *k*-nearest neighbor, Naïve Bayes, association rule mining (ARM) and support vector machine (SVM) (Mitchell, 1997; Alpaydin, 2010; Han, Kamber, and Pei, 2011). The classification and prediction methods in data mining and machine learning are discussed in more detail in Section 2.3.3.1. The open source systems of data mining and machine learning are presented in Section 2.3.3.2.

2.3.3.1 Machine Learning and Data Mining Methods Used for Classification and Prediction

Machine learning and data mining apply classification methods to extract models that describe useful data classes or concepts. Moreover, these methods can be used for predicting future data trends by analysing observations, past experience and training data. The induced model by these methods is represented in a form of decision trees, classification rules or mathematical formula (Mitchell, 1997; Alpaydin, 2010; Han, Kamber, and Pei, 2011). The following describes some of the common classification and prediction methods applied in machine learning and data mining.

2.3.3.1.1 Naïve Bayes

Naïve Bayes (Domingos and Pazzani, 1997; Mitchell, 1997; Han, Kamber, and Pei, 2011) is a statistical classifier that is based on the Bayes' theorem and can be trained successfully in the supervised learning setting. This classifier calculates the conditional probabilities for each class from a training set, given a set of features in order to predict that a given feature belongs to a particular class. It uses the maximum likelihood formula for estimating the probabilities. The most frequent class which has the highest probability that maximises the score of the formula is selected to be the appropriate class for the given feature. The Naïve Bayes classifier assumes that the occurrence of each feature in a given class is conditionally independent of the occurrence of any other feature.

2.3.3.1.2 *K*-Nearest Neighbor

The *k*-nearest neighbor (*k*-NN) (Mitchell, 1997; Han, Kamber, and Pei, 2011) classifier is built from examples, where all of the training examples are stored in a pattern space. In *k*-NN, the classification of a new example is based on searching the pattern space and calculating the distance between the new example and the stored examples in the pattern space. The new example is assigned to the class most common among the *k* nearest training examples to that new example in the pattern space.

2.3.3.1.3 Artificial Neural Networks

The artificial neural networks (ANNs) (McCulloch and Pitts, 1943; Mitchell, 1997; Han, Kamber, and Pei, 2011) model is composed of a set of interconnected input/output units that exploits a computational model for processing information based on the connectionist learning approach. In general, ANNs can be used as a decision making tool to estimate nonlinear relationships which represent a complex data model. In ANNs, the training dataset is split into non overlapping sets. Moreover, each unit in the network has a weight associated with it. During the learning process, the weights are adjusted, so that the output unit with the desired response will have greater activation than any other output unit in the training set. The patterns of output units can be transformed into numeric predictions or discrete decisions about the input data.

2.3.3.1.4 Decision Tree Induction

Decision trees induction (Han, Kamber, and Pei, 2011) is a very popular classification approach for supporting decision making process. It is used to represent classification rules in a structure of tree such that the induction of a decision tree is based on the training examples. Decision tree algorithms are able to handle both nominal and categorical data. The main aim of decision trees is to develop a set of rules that will correctly classify the examples into known classes. Each node of a decision tree corresponds to an attribute (feature), each branch represents the attribute value (feature value), and each leaf represents a class. A path is traced from the top node (root) to the terminal node (leaf) such that the prediction of a class is made when the terminal node is reached.

The best known decision tree induction algorithm is the C4.5 algorithm (Quinlan, 1993) which is an extension of the ID3 algorithm (Quinlan, 1986). This algorithm builds decision trees in a top-down recursive divide-and-conquer manner. It examines each attribute and uses a greedy heuristic search for selecting the attribute. It computes the information gain for selecting the best attribute to be the root node which will split the examples into subsets. The process is applied repeatedly to each subset until all examples are correctly classified.

Decision tree induction algorithms have been used in a number of applications. For example, the study suggested by Lee (2010) in the marketing field applies the decision tree algorithm called C4.5 on a source of customer data from the retail business in Taiwan to develop a recommender system for supporting commodity recommendation of retail business according to customer preferences. In the medical domain, a study presented by Tanner et al. (2008) employs the C4.5 decision tree algorithm on a patient enrolment, clinical and epidemiological data collection for detecting the diagnosis of dengue disease in the early phase of illness. Another study has been proposed in the medical domain by Tsumoto and Hirano (2010) which applies the C4.5 decision tree algorithm on a nurses' incident data and a clinical data for patients with blood stream infection in order to generate decision trees. The if-then-rules are extracted from the induced decision trees, where these rules can then be used for detecting the risk factors in clinical environments. The approach described by Siltepavet, Sinthupinyo and Chongstitvatana (2012) utilises the C4.5 algorithm for classifying the important parameters that can improve the quality of products in hard drive manufacturing in order to reduce the number of defected products.

2.3.3.1.5 Rule Induction Learners

Rule induction learners are used to handle large datasets and induce formal rules based on training examples, where these rules can be utilised for classifying data and making precise decisions. The induced rules by these learners are represented in a form of if-then classification rules, which are easy to understand (Witten, Frank, and Hall, 2011). The rule induction algorithms employ condition action rules and perform a heuristic search to find the optimal rules set whose conditions match the instances in the training examples. Some of the most popular rule induction algorithms are CN2 (Clark and Boswell, 1991) and RIPPER (Cohen, 1995).

The rule induction algorithm CN2 (Clark and Boswell, 1991) is based on the concepts of the AQ (Michalski et al., 1986) and ID3 (Quinlan, 1986) algorithms. It performs a general-to-specific search to learn from a given set of training examples for generating an unordered list of optimal rules. CN2 starts to search for rules and apply a set of conditions to a set of examples in order to test rules and find the best rules that cover the examples. Then, it repeats the search until no more optimal rules can be found.

The CN2 rule induction algorithm has been applied to a variety of domains for inducing classification rules. Džeroski and Lavrac (1996) employ the CN2 rule induction algorithm in the medical diagnosis domain. The algorithm is applied on patient records with corresponding diagnosis for inducing rules that can be used to diagnose new cases in early diagnosis of rheumatic diseases. The paper by Džeroski, et al. (1997) uses a rule induction approach for biological classification in the ecological domain. In their approach, the CN2 rule induction method is applied on a data of biological samples for classifying several problems related to the river water quality. Furthermore, in the study presented by Samanovic, Cukusic, and Jadric (2011), a rule induction approach has been used in the business domain. Their study applies the CN2 algorithm on publicly available online databases of business web sites for inducing rules that predict the amount of foreign direct investments in a country, based on various business indicators.

The well-known rule induction algorithm called RIPPER (Repeated Incremental Pruning to Produce Error Reduction) (Cohen, 1995) is suggested by William Cohen as an extended version of the IREP algorithm (Furnkranz and Widmer, 1994) for generating classification rules in the form of if-then rules. It applies a propositional rule learner to create and test rules until it could find a list of optimal rules. The RIPPER algorithm generates a rule set by adding rules repeatedly into an empty list (i.e. rule set) until there are no more positive examples to be covered. It starts by splitting the training data into two sets. The first set is for growing the rules, while the second set is for pruning the weak rules. In the growing phase, the RIPPER algorithm starts to grow a rule by adding conditions greedily to the rule until it becomes optimal and covers no negative examples. At this time, the algorithm also starts to check the attribute values in the dataset for choosing the condition that has the highest information gain. After that, the pruning phase starts by incrementally pruning rules and weak conditions in order to reduce the errors in the created rule set. The algorithm will stop repeating the growing and pruning phases if one of the following cases occur: the error rate of the last rule is greater than or equal to 50%, the description length (number of bits needed to perform rules) of the rule set is 64 bits greater than the smallest description length produced so far, or there are no more positive examples to be covered. Then it generates an initial rule set. Afterwards, the optimisation stage begins to randomise data by growing and pruning two rules from the initial rule set until it can find a rule with the smallest description length to be added to the last representation of the rule set (Cohen, 1995).

The RIPPER rule induction algorithm has been applied to real world problems and applications. For instance, Jovic and Bogunovic (2009) have proposed a study for heart rate analysis based on using some classification algorithms, e.g. RIPPER, C4.5 decision tree, Bayesian network (Friedman, Geiger, and Goldszmidt, 1997), and random forest (Breiman, 2001). Thus, the RIPPER algorithm is applied into a collection of databases of patient records obtained from the PhysioBank website for analysing features of heart rate variability in order to find the rules that classify the patient records. Ulutaşdemir and Dağlı (2010) present a study in the medical diagnosis domain for predicting death risk in hepatitis. They have employed different algorithms (RIPPER, PART [a mixture of C4.5 and RIPPER algorithms] (Witten et al., 1999), and J48 [an implementation of C4.5 algorithm]) including JRIP (i.e. a WEKA (Hall et al., 2009) implementation of RIPPER) on clinical hepatitis datasets obtained from the UCI Machine Learning Repository (Frank and Asuncion, 2010) for finding rules that can be made use of to evaluate the risk of death in hepatitis. In the study introduced by Peng et al. (2011), a wide range of classification algorithms (Bayesian network, Naïve Bayes, *k*-nearest neighbor, C4.5, RIPPER, support vector machine (SVM) (Platt, 1999), linear logistic regression (Le Cessie and Van Houwelingen, 1992) and radial basis function (RBF) network (Bishop, 1995)) have been utilised for discovering any financial risk that might have taken place in the most recent years. In their study, the RIPPER rule induction algorithm is applied on real credit risk and fraud risk datasets from six countries in order to find the rules that could be used for predicting any financial risk.

2.3.3.1.6 Regression Analysis

Nowadays, many fields including business, education, health and industry are focusing on the importance of predicting accurate decisions, evaluating the future risk factors and specifying the individual needs. Therefore, regression analysis models (Han, Kamber, and Pei, 2011) play a vital role in making future decisions and predicting results in many significant domains such as energy economics domain (e.g. Bianco, Manca and Nardini 2009) and the business domain including sales and marketing (e.g. Chu and Zhang, 2003).

Regression analysis is a statistical technique which has been applied widely in the machine learning and data mining fields for predicting continuous variables. This technique, which handles numeric data, is based on analysing and modelling the best relationship between one dependent variable and one or more independent variables.

There are different methods (forms) of regression analysis which include: linear regression, multiple regression and nonlinear regression (Han, Kamber, and Pei, 2011). These methods are described in greater detail in the next paragraphs.

Linear regression is the simplest form of regression analysis. It models the best line relationship between two variables. In particular, this approach is based on identifying an independent variable "*X*" and past historical values (*x1, x2, x3,,.. x_n*) in order to predict a dependent variable "*Y*". Equation (2.2) represents the formula of linear regression (Anderson, Sweeney, and Williams, 2005; Han, Kamber, and Pei, 2011):

$$Y = \alpha + \beta X \tag{2.2}$$

where *α* and *β* are regression coefficients, α represents the Y-intercept of regression line and *β* is the slop of the estimated regression line. Regression models use the least squares fit approach, which was published by Legendre in 1805 and Gauss in 1809, to estimate the regression coefficients (*α*, *β*). This approach is applied to fit an equation into a dataset, such that it minimises the difference (sum of squared errors) between the observed data and the estimated one. The values of *α* (intercept) and *β* (slope) could be calculated by using the equations (2.3) and (2.4) (Anderson, Sweeney, and Williams, 2005; Han, Kamber, and Pei, 2011) which are defined as,

$$\beta = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \tag{2.3}$$

$$\alpha = \bar{y} - \beta\bar{x} \tag{2.4}$$

where:

- x_i represents the value of the independent variable
- y_i represents the value of the dependent variable
- $\bar{x}$ is the mean of the independent variable "*X*"
- $\bar{y}$ is the mean of the dependent variable "*Y*"
- α is the y intercept of the regression line
- β is the slope of the estimated regression line

Multiple regression is another type of regression analysis for fitting complex data models. The difference between the linear and multiple regression is that the linear regression uses one independent variable in order to predict the dependent one, while the multiple regression uses two or more independent variables in order to predict the dependent variable. In multiple regression, the dependent variable "*Y*" is modelled as a linear function of a multidimensional feature vector. Equation (2.5) is an example of a multiple regression model based on two variables (X_1 and X_2), where the method of least squares can also be used to estimate α, β_1, and β_2 (Anderson, Sweeney, Williams, 2005; Han, Kamber, and Pei, 2011):

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 \tag{2.5}$$

Non-linear regression is a form of regression analysis that uses a non-linear function to specify the relationship between a set of non-linear variables, where this function depends on one or more independent variables (Han, Kamber, and Pei, 2011).

The importance of using regression analysis for predicting useful data and forecasting future events is noticeable in many fields such as business (Dalrymple, 1975; Waddell and Sohal, 1994; Anderson, Sweeney, and Williams, 2005). Forecasting the future demands in the business field is essential to business and management success, where it plays a vital role to reduce uncertainty and make accurate, effective and strategic decisions in organisations. Most of these decisions are related to significant subareas of business including sales and marketing. The sales and marketing forecasting helps companies in planning the business and in identifying the sales demand in order to build an effective investment strategy and allocate the required resources to achieve the expected sales (Dalrymple, 1975; Waddell and Sohal, 1994).

In general, there are two types of forecasting methods that are used for predicting future demands: quantitative methods and qualitative methods (Waddell and Sohal, 1994; Anderson, Sweeney, and Williams, 2005). The idea of the quantitative methods is based on using mathematical models with past historical data about a variable in order to estimate the future data patterns about the same variable. The forecasting in the quantitative methods could be employed using the causal methods (e.g. regression analysis) or the time series methods (e.g. exponential smoothing, trend projection and moving average). The qualitative techniques utilise judgments, observations and opinions of experts or consumers to develop forecasts. These methods can be applied when the historical data and past information are unavailable. Delphi, market survey and life cycle analogy are some examples of the qualitative techniques (Waddell and Sohal, 1994; Anderson, Sweeney, and Williams, 2005).

In forecasting business demands, people must apply some criteria in selecting the most appropriate method for making decisions and predictions, since they must know in which period of time the selected forecasting method can perform the best predictions. In addition, the applied forecasting technique should be suitable to the company’s needs and the availability of historical data needed to perform forecasts (Reid and Bojanic, 2009). For instance, the regression models, which are common to use in sales and marketing forecasting, are more effective for medium-term (up to two years) forecasts (Waddell and Sohal, 1994; Reid and Bojanic, 2009), while the time series methods are suitable for business forecasts and they are more appropriate for short-term (one to three months) forecasts. Finally, the qualitative methods are more relevant for long-term (more than two years) forecasts (Reid and Bojanic, 2009).

Over the last decades, a range of studies have utilised regression analysis models for forecasting demands in economics, finance and business (e.g. sales and marketing domains) applications. Huss (1985) for example, applies linear regression and some time series techniques for forecasting sales using annual electric utility energy sales data. Furthermore, the study introduced by Desai and Bharati, (1998) compares linear regression with nonlinear regression techniques for predicting excess returns on large stocks. This study shows that the forecasts of nonlinear regression models are conditionally efficient with the respect to the forecasts produced by the linear regression models. Bianco, Manca and Nardini (2009) present a study that utilises linear regression models in the energy economics domain for forecasting electricity consumption in Italy. The linear regression model is applied on a historical electricity consumption data (from 1970 to 2007) and the study shows an increase in the electricity consumption during the coming years.

Moreover, regression analysis has been used in the study proposed by Morphet (1991), where a multiple regression model is applied to forecast grocery store sales in the north-east of England. Also, the study addressed by Forst (1992) exploits multiple regression and another statistical model for forecasting restaurant sales. By testing all of the applied models on a weekly sales data (year 1 and year 2) of a small restaurant, results show that the multiple regression model has achieved the best forecasts of the restaurant sales. More recently, Ramanathan (2012) presents an approach for forecasting promotional sales of a UK manufacturing company based on studying various demand factors. His approach applies multiple regression models on an actual sales data of soft drink products (from 2005 to 2006) from two main retailers. The study concludes that understanding the significance of product specific demand factors will aid managers to specify the essential information for creating accurate demand forecasts.

Furthermore, different studies in the business domain have applied artificial neural networks (ANNs) to forecast demand. For example, the study addressed by Alon et al. (2001) explores the use of ANNs for forecasting aggregate retail sales. This study also uses some other traditional statistical techniques including the multiple regression model to generate the forecasts. All of the applied techniques are tested on a monthly retail sales data (from January 1978 to December 1985 and from January 1986 to April 1995). The study concludes that the ANNs model outperforms the other traditional statistical techniques, since it has provided the most accurate forecasts. Chu and Zhang (2003) have examined the use of linear and non-linear regression models for forecasting aggregate retail sales. All of these models are applied on a monthly retail sales data (from January 1985 to December 1999). Their experiments showed that the neural networks model outperforms the linear regression models in forecasting the retail sales movements. Recently, a study proposed by Kumar and Mittal (2012) describes an approach that exploits ANNs to forecast sales in an Indian automobile manufacturing company. After they applied the ANNs model to ten years aggregate monthly sales data of motorcycles (from January 2001 to December 2010), they have evaluated the performance of the model by comparing it with traditional time series techniques. The experiments showed that the ANNs model has achieved better forecasts than the traditional time series models.

2.3.3.2 Open Source Software of Data Mining and Machine Learning

There are a lot of open-source systems that apply data mining and machine learning algorithms for solving real world problems such as RapidMiner (Mierswa et al. 2006) and WEKA (Hall et al., 2009; Witten, Frank, and Hall, 2011) which are summarised in the following paragraphs.

RapidMiner (formerly YALE) (Mierswa et al. 2006) is a Java-based software that provides a simple and friendly GUI to support different data mining and machine learning processes such as data analysis, pre-processing, modelling, visualization and evaluation. It exploits internal XML (eXtensible Markup Language) representations for defining the analytical processes that are needed to perform data mining experiments. The RapidMiner software has been employed in different domains such as bioinformatics (Han, Rodriguez and Beheshti, 2008) and education (Montalvo et al., 2010).

The WEKA (Waikato Environment for Knowledge Analysis) (Hall et al., 2009; Witten, Frank, and Hall, 2011) toolkit is a machine learning work bench, implemented in Java and developed by researchers at the University of Waikato. It provides a unified package which enables users to access the state-of-the-art technologies in the data mining and machine learning environment for pre-processing, classification, clustering, association and visualization. The WEKA software is available for free under the GNU General Public License and it could be applied easily due to its simple GUI. The tools supported by the WEKA workbench are based on statistical evaluations of the models (algorithms). Consequently, the WEKA user can easily compare the results and accuracies of the applied machine learning algorithms for a given dataset in flexible ways in order to choose the most appropriate algorithm for the given dataset (Hall et al., 2009; Witten, Frank, and Hall, 2011).

The WEKA workbench uses dataset file of a relational database table that contains a set of examples implemented by class instances. Special kinds of formats (e.g. ARFF, CSV, and C4.5) are used by the WEKA tool. The attribute relationship file format (ARFF) is the mostly used format by WEKA to create dataset files. It describes a number of instances sharing a set of attributes which are associated with some variables. The ARFF file consists of particular tags: '@relation', '@attribute' and '@data'. The '@relation' tag defines the relation name for the ARRF file, while the '@attribute' tag states the name, type (e.g. nominal, numeric, string, date) and the values of each attribute. The '@data' tag specifies the data values (Hall et al., 2009; Witten, Frank, and Hall, 2011). The WEKA toolkit has been applied successfully in many studies such as bioinformatics research (Frank et al., 2004), psychological studies (Plarre et al., 2011) and financial research (Wang and Ma, 2012).

2.4 Overview of Inductive Logic Programming (ILP)

Several successful machine learning and data mining methods have been illustrated above. One of the key aspects of machine learning which distinguishes it from data mining is its ability to analyse complex data and handle generalisations represented in a logical form based on using logical and relational learning (De Raedt, 2008). In contrast, data mining methods can only handle data in a limited representation language.

The logical and relational learning methods (De Raedt, 2008) have a knowledge representation language that is able to represent complex data structures and learn knowledge from examples by finding an unknown relationship in terms of relations already known. This type of learning employs a kind of reasoning called inductive inference (inductive learning) for generating rules from a set of entities and their relationships which are represented using first order logic. This section describes inductive logic programming and its systems. Then, it presents an overview of some studies that have applied inductive logic programming for learning grammar and semantic rules, natural language processing and information extraction.

2.4.1 Inductive Logic Programming

Inductive logic programming (ILP) (Muggleton and De Raedt, 1994; Muggleton, 1999; De Raedt, 2008) is a subfield of machine learning that utilises logic programming and machine learning for inducing hypotheses (i.e. theories) from background knowledge *B* and examples *E*. The ILP theory describes the inductive inference (the inverse of deduction) of logic programs from background knowledge and examples. The examples are divided into a positive examples subset *P* and a negative examples subset *N*. In ILP, a hypothesis *H* which has to be derived by an ILP system should satisfy the following conditions (Muggleton and De Raedt, 1994; Muggleton, 1999):

- Completeness, such that *H* is considered complete if it covers all positive examples:

$$\forall p \in P : H \cup B \vDash p$$

  where the logical symbol $\forall$ is a universal quantifier that denotes "for all", *p* is a positive example, the logical symbol $\in$ indicates membership in a set, the logical symbol $\cup$ denotes a union and the logical symbol $\vDash$ means an implication (entailment).

- Consistency, such that *H* is considered consistent if it does not cover any of the negative examples:

$$\forall n \in N : H \cup B \nvDash n$$

where *n* is a negative example and the logical symbol $\nvDash$ indicates a negated implication.

More specifically, the derived hypothesis by ILP should entail all positive examples and none of the negative examples. It should also meet the language constraints.

The following brief introduction to logic programming is based on a description by De Raedt (2008). Logic programming is based on the use of the PROLOG logic programming language (Bratko, 1990), which is applied to describe relations between entities using clauses. In PROLOG, a term is a number, a constant, a variable, an atom or a compound term. The compound term consists of an atom (i.e. predicate) called a functor followed by a number of terms (i.e. arguments). The syntax of a compound term is: $f(t_1,\dots,t_n)$, where *f* is the functor (function) symbol that is used to construct a relation and $t_1,\dots,t_n$ are some terms of the predicate *f*. For example, the compound term called "specific(A,B)" is a predicate (relation) that consists of the functor "specific" and the two variables (terms) "A" and "B". The number of arguments is called arity, e.g. specific/2, where the arity "/2" means that the predicate "specific" consists of two arguments.

Another key aspect in logic programming is the concept of horn clauses (Kowalski, 1974). A horn clause (i.e. a disjunction of literals) is a PROLOG clause which consists of logical predicates. The horn clauses are represented in the form of a rule (e.g. '$h \leftarrow b1,\dots, bm$'), where *h* is the head of the rule and *b* is the body of the rule. The symbol ',' indicates a conjunction, while '←' indicates an implication. This means that the head *h* is implied (←) by the body which consists of one atom (or one predicate) *b* or the conjunction of more than one atom (or more than one predicate) '*b1, b2…bm'*. The condition of the head predicate for a rule is satisfied if the conjunction of the body predicates is also satisfied.

ILP is different from other machine learning methods due to its expressive representation concept language, since it is able to learn information and theories from a set of relevant facts from any domain represented in a form that the ILP learner can understand. The most important characteristic of ILP is its ability to generate meaningful and well-formed hypotheses, where this depends on the ability of providing declarative background knowledge to the inductive learner and getting the background knowledge right. In general, a set of positive and negative examples are provided as ground facts to the ILP learner together with background knowledge for generating theories. The background knowledge consists of horn clauses which represent information about the predicates that appear in the induced theory later. In particular, the background knowledge predicates (body predicates) describe the literals of the body for the constructed clauses. The target predicates (head predicates) should not appear in the body of the constructed clauses. A first order representation is defined as a set of constant symbols, predicate symbols and functor symbols. ILP can only learn from first-order horn clauses (Muggleton and De Raedt, 1994; Muggleton, 1999; De Raedt, 2008).

The process of an ILP learner is based on searching the space of possible hypotheses which satisfy some quality criteria. Particularly, the hypothesis should satisfy some syntactic (form of the constructed clause) and semantic (variables type) restrictions called the language bias. The language bias is used to limit the search space by reducing the number of potential solutions, preventing overfitting and learning well-formed hypotheses. The ILP learner begins by constructing a clause based on the provided examples and background knowledge. Then, it starts searching the space for the best hypotheses which cover more positive examples. To structure the search space, typical ILP systems utilise θ-subsumption as generalisation or specialisation operator (i.e. refinement operator) for partial ordering of clauses in order to determine which examples a clause covers (Muggleton and De Raedt, 1994; Muggleton, 1999; De Raedt, 2008). A clause C θ-subsumes a clause G if and only if there is a substitution θ, such that $C\theta \subseteq G$, where $\subseteq$ indicates a subset. The substitution θ is a function that turns a set of variables into a set of terms. For illustration, consider the following example of the clauses C and G:

C: mother(X, Y) ← parent(X, Y), female(X).

G: mother(ann, james) ← parent(ann, james), parent(ann, dave), female(ann), male(james).

. C θ-subsumes G, with $\theta = \{X / ann, Y / james\}$

In the above example, all the literals of the clause *C* are included in the set of the literals of the clause *G* via the θ-subsumption. In particular, the θ-subsumption identifies the notion of generality, where *C* is called a generalisation of *G* and *G* is a specialisation of *C* under θ-subsumption (Plotkin, 1970).

In general, there are two main ILP strategies of information processing and knowledge ordering: top-down and bottom-up. The top-down strategy is based on searching the hypothesis space from general to specific, while the bottom-up strategy searches the hypothesis space from specific to general. It has been noticed that much of the research has applied the top-down approaches to induce useful hypotheses, since these approaches use short clauses in the search and this helps to reduce the size of the search space. In contrast, the bottom-up approaches start with long clauses and this increases the size of the search space as well as the cost of subsumption tests. In addition, the search performed by the bottom-up approaches may suffer from the problem of overfitting when using small datasets with large examples (Arias and Khardon, 2004; Flach, 1998). The common top-down ILP methods for learning rules and inducing hypotheses include ALEPH (Srinivasan, 1999), FOIL (Quinlan, 1990) and PROGOL (Muggleton, 1995). On the other hand, the system called GOLEM (Muggleton and Feng, 1992) represents one of the well-known ILP techniques that is based on the bottom-up search strategy. The following subsections give an overview of the top-down ILP methods ALEPH, PROGOL and FOIL.

2.4.2 Overview of Inductive Logic Programming Systems

An ILP system such as ALEPH (A Learning Engine for Proposing Hypotheses) (Srinivasan, 1999), which is written in PROLOG, can be run by using a PROLOG compiler (e.g. YAP compiler (Santos Costa et al., 2000)) to induce hypotheses from background knowledge and examples. The ALEPH learner performs a top-down (general-to-specific) search by utilising θ-subsumption as refinement operator. It requires three files to construct a theory, the background knowledge file (file.b), the positive examples file (file.f), and the negative examples file (file.n). The following steps summarise the learning process utilised by ALEPH (Srinivasan, 1999):

- Select an initial example to be generalised.

- Build the most specific clause (MSC) based on the restrictions language, where the constructed MSC must entail the selected example; this step is called "saturation".
- Begin to search for a clause which could be more general than the MSC and has the best score in terms of covering the examples, in order to add it to the theory (i.e. this represents the generalisation process, where the literal of predicates appearing in the constructed clause are utilised to search for the best clause that covers all the positive examples and none of the negative examples). This step is called "reduction". The best score is defined as the difference between the number of positive and negative examples covered by the constructed clause. The search strategy applied by ALEPH in the reduction step is based on a restricted breadth first branch-and-bound search (Papadimitriou and Steiglitz, 1982). This type of search uses a tree structure which consists of a set of nodes for finding the optimal solutions to problems by selecting the node with the minimum cost to be the goal of the performed search (it considers and tries all possible paths from the starting node to the end one). In the ILP system ALEPH, each node in the tree includes a clause. The choice of the best clause is based on comparing a score value assigned to each clause by an evaluation function. By using the ALEPH's default evaluation function (i.e. coverage) which calculates the difference between the positive and negative examples covered by the clause, the clause that has the minimum value of the evaluation function "coverage" (minimum cost) will be selected first to be the best clause with the best score. If the score at a clause is no better than the value of score of the best clause found so far, this clause will not be selected during the attempt to search for the best clause.
- Remove all redundant examples that have been covered by the MSC. The generalisation process is repeated until ALEPH can construct a theory that covers all positive examples and none of the negative examples.

The PROGOL system (Muggleton, 1995) is based on combining the "Inverse Entailment" with the "general-to-specific" (top-down) search by using the refinement graph search. It uses "Inverse Entailment" and mode declaration (described later in Chapter 5) defined by the users to constrain the search for a good hypothesis. The

PROGOL learner starts by finding a consistent clause that entails a given example. After that, it searches for a clause that is better than the constructed one. All redundant examples that have been covered by the constructed clause are removed. Then, the generalisation process is repeated until the learner can derive a hypothesis which covers all the positive examples. The search strategy performed by PROGOL is an A*- like algorithm (Hart, Nilsson, and Raphael, 1968) with an approximate compression measure for finding an optimal clause that maximises the compression measure and covers all the positive examples and none of the negative ones. The A* search strategy is based on summing up the values of the path cost (distance from the starting node) and the evaluation function (distance to the goal) in order to find a measure which is then used to search for the optimal node. The compression measure is based on maximising the number of positive examples covered and minimising the number of negative examples covered.

FOIL (First Order Inductive Learner) (Quinlan, 1990) is based on learning first-order clausal hypotheses from relational background knowledge, positive examples and negative examples. The given examples and the background knowledge predicates are the input to the FOIL learner, while a rule (or a set of rules) in first-order predicate logic is (are) the output of the learner. FOIL constructs a clause that covers a specified example. Then, it starts adding literals to the current constructed clause until it excludes all negative examples. After that, it learns a rule that satisfies all positive examples and none of the negative examples. To find an optimal clause, FOIL performs top-down (general-to-specific) greedy hill-climbing search (Russell and Norvig, 2010) which is based on the depth first algorithm. This search strategy starts with an initial node (clause), and then it seeks to find a better node by repeatedly improving the current node (it considers one path during the search for the best node and therefore this search strategy does not guarantee to find the optimal solution).

2.4.3 Using ILP for Learning Grammar and Semantic Rules, NLP and IE

Many studies utilise different ILP systems for learning grammar and semantic rules, processing natural language and deriving information extraction rules. In this section, some of these studies are summarised to provide the context for this research and to learn any key lessons from these studies. Section 2.6 includes a discussion that summarises the key aspects from these studies.

In the problem of learning grammar and semantic rules, the study introduced by Claveau et al. (2003) utilises ILP to build semantic lexicons from a POS and semantically tagged corpus called MATRA-CCR[7]. This study applies ALEPH on the annotated noun and verb corpus to extract relevant noun-verb pairs as well as semantic rules. The learned rules can be used for distinguishing the relevant pairs from the irrelevant ones based on the surrounding syntactic (including POS tags) and semantic context. The performance of ALEPH has been evaluated using 10 fold cross validation on 3,099 positive examples and 3,176 negative examples produced from the MATRA-CCR corpus. The estimated precision rate of ALEPH is 81% and the recall rate is 89%. The study introduced by Eisenstein et al. (2009) applies an ILP learner to perform semantic abstraction of documents based on syntactic, lexical and semantic analysis of the text. This study employs the ALEPH learner on a POS and semantically analysed text of instruction that consists of natural language sentences of a game data in order to induce rules which can be applied for learning how to make legal moves in the game of Freecell solitaire. The study addressed by Quinlan (1994) utilises the FOIL machine learning to learn rules for transforming English verbs from present to past tense. A corpus of 1391 verbs is used in the experiments, where a random selection has been conducted to select 10 trails of samples, each containing 500 verbs. The theories which have been learned by FOIL from this data are then tested on a sample of 500 different verbs. The average accuracy of the theories learned by FOIL is 83.7%. Event thought the achieved results are encouraging, the study shows that FOIL failed to ensure that the learned rules are generative and was also unable to reorder the learned clauses due to its search strategy.

In the problem of IE, Aitken (2002) presents a study that exploits ILP to induce information extraction rules which can be utilised to identify the meanings of sentences in text. In specific, the FOIL learner has been applied on natural language text for deriving relations that are defined in an existing ontology in order to characterise the content of the natural language text. In the study addressed by Tanabe and Wilbur (2004), a large lexicon of gene and protein names has been generated

[7] MATRA-CCR is a French handbook (i.e. corpus) of helicopter maintenance, owned by Aerospatiale Matra CCR. This corpus includes more than 104,000 word occurrences.

from MEDLINE database by using ILP, where the developed lexicon can be utilised for identifying gene-related semantic categories. Particularly, the ALEPH system is applied for generating rules that are based on examining morphological features of gene/protein names in order to identify gene/protein names in natural language of biomedical text.

In the issue of NLP, Lindberg and Eineborg (1998) propose a study that utilises the PROGOL system to learn the rules of constraint grammar for POS tagging from a POS tagged corpus of Swedish text called "Stockholm Umeå". The applied system induced several disambiguation rules which can then be used for disambiguating the POS tags of words. A lexicon was created from the used corpus and by testing the induced rules on unseen data, results showed that 98% of words have been disambiguated with the correct POS tags. The study presented by Cussens et al. (1997) utilises ILP for NLP, where the PROLOG system has been applied for learning English grammar rules from sentences of a series of children's books. In addition, Cussens (1997) presents a study which also uses ILP for NLP task, where the PROGOL machine learner has been trained on the Wall Street Journal corpus (3 million words) for English to induce rules for specifying the possible POS tags of words in the sentences from this corpus. Cussens has split this corpus into 2/3 training set and 1/3 test set. The PROGOL system has achieved 96.4% per-word tagging accuracy on the test data. Zelle and Mooney (1993) employ an ILP system called CHILL which is inspired by bottom-up (e.g. GOLEM) and top-down (e.g. FOIL) techniques to learn a natural language parser that integrates syntactic and semantic constraints for inducing semantic grammar rules which can be used to support the parsing process. In their study, two experiments of the system have been conducted. In the first experiment, an accuracy of 92% is achieved with 150 examples from the M & K corpus (McClelland and Kawamoto, 1986), while in the second experiment an accuracy of 93% is achieved with 50 examples from the tourist domain.

2.5 Performance Evaluation Functions for Machine Learning Methods

A key part of any study in this field is how development is evaluated. This section summarises the methods used to evaluate the performance of machine learning systems.

The confusion matrix represents the contingency table that allows analysing and visualizing the performance of the machine learning algorithms. It consists of instances for the actual and predicted data (Provost and Kohavi, 1998; Sokolova, Japkowicz and Szpakowicz, 2006). Table 1 presents a confusion matrix,

	Predicted		
Actual		+	-
	+	TP	FN
	-	FP	TN

Table 1 Confusion Matrix

where (Buckland and Gey, 1994; Sokolova, Japkowicz and Szpakowicz, 2006):

- TP (true positive) represents the examples that are correctly predicted and marked as positive,
- FP (false positive) indicates the examples that are incorrectly classified as positive,
- TN (true negative) denotes the examples that are correctly predicted as negative,
- FN (false negative) represents the examples that are incorrectly marked as negative.

There are different techniques used to evaluate the performance of the systems/classifiers. These techniques are described in the following paragraphs.

The precision and recall rates (Buckland and Gey, 1994; Sokolova, Japkowicz and Szpakowicz, 2006), which originated from IR, are widely utilised in the empirical studies of AI to measure experimentally the effectiveness of machine learning methods.

Precision is a statistical classification function that measures the probability of retrieving relevant examples divided by the total number of the retrieved examples. Equation (2.6) represents the formula of *precision P* (Buckland and Gey, 1994; Sokolova, Japkowicz and Szpakowicz, 2006):

$$P = \frac{\#\text{relevant examples retrieved}}{\#\text{retrieved examples}} = \frac{TP}{TP + FP} \tag{2.6}$$

Recall rate is a statistical classification function that measures the probability of retrieving relevant examples divided by the total number of the existing examples that are expected to be retrieved. Equation (2.7) represents the formula of *recall R* (Buckland and Gey, 1994; Sokolova, Japkowicz and Szpakowicz, 2006):

$$R = \frac{\#\text{relevant examples retrieved}}{\#\text{total examples to retrieve}} = \frac{TP}{TP + FN} \tag{2.7}$$

The statistical function called *F-score (F-measure)* estimates the harmonic mean of *precision* and *recall* to evaluate the accuracy. Equation (2.8) represents the formula of *F-score F* (Sokolova, Japkowicz and Szpakowicz, 2006):

$$F = \frac{2 \cdot precision \cdot recall}{(precision + recall)} \tag{2.8}$$

The accuracy is the proportion of examples that are correctly classified. Equation (2.9) represents the *accuracy A* formula (Sokolova, Japkowicz & Szpakowicz, 2006):

$$A = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{2.9}$$

The best known technique for evaluating the accuracy of a system or a classifier model is the cross validation method. Cross validation (Geisser, 1975; Kohavi, 1995) is a statistical analysis procedure that splits the data into two sets: the first set is the training dataset which is used to train the classifier, whilst the other set is the testing dataset which is utilised for examining the model. The common form of the cross validation is the *k*-fold cross validation. In the *k*-fold cross validation, the data is divided into *k* subsets called folds which are equal in size. One of these folds is used for testing the model (*k* subset), while the other folds (*k*-1 subsets) are used for learning. The learning process is repeated *k* times, such that in each time different fold is tested. At the end, the average error is calculated for all the *k* trials (Kohavi, 1995).

2.6 Discussion and Summary

The above sections present the relevant literature review for this research. In this section, a set of tables that summarises the reviewed research is presented in Section 2.6.1. Then, a discussion on the SMART approach and surveyed techniques of NLP, IE, machine learning and data mining is given in Section 2.6.2. The presented discussion clarifies the methods that have been selected to develop a framework for a system to support employee appraisals and also describes the motivational reasons of selecting these methods. A summary of the chapter is presented in Section 2.6.3.

2.6.1 Literature Review Summary Tables

Based on the surveyed studies, some literature review summary tables have been created to bring together the reviewed research in the areas of NLP, IE, machine learning (ML) and data mining. Table 2 summarises the open source IE systems for NLP, IE and text annotation.

Information Extraction (IE) Systems for Text Annotation			
Author	**System /ML**	**Aim of Using the System/ML**	**Data Source**
McKenzie et al. (2010)	NLTK	Perform partial parsing of text	Corpus of helicopter maintenance records
Stoyanchev et al. (2008)	NLTK	Perform phrase chunking	AQUAINT corpus (a collection of news documents) and the web.
Kang et al. (2011)	OpenNLP	Perform noun phrase chunking	Biomedical corpus
Ek et al. (2011)	OpenNLP	Perform tokenization and POS tagging	Swedish SMS corpus
Feilmayr et al. (2009)	GATE	Utilise Tokenizer, Gazetteer, sentence splitter and JAPE for annotating text	Tourism and accommodation web pages
Joshi et al. (2012)	GATE	Utilise Tokenizer, sentence splitter, POS tagger, Gazetteer and NE transducer	A collection of tweet posts related to the marketing domain

Table 2 Survey Summary of Information Extraction Systems for Text Annotation

Table 3 presents the studies that employ ML and NLP methods for semantic class disambiguation (SCD) based on WordNet.

Semantic Class Disambiguation (SCD)			
Author	**System/ML**	**Aim of Using the System /ML**	**Data Source**
Ciaramita & Johnson (2003)	Multiclass Averaged Perceptron Classifier	Supersense tagging of common nouns based on the WordNet lexicographers	Bllip training set
Ciaramita & Altun (2006)	Perceptron Trained Hidden Markov Models	Semantic class labelling of words based on the WordNet lexicographers	Senseval 3 & Semcor datasets
Kohomban & Lee (2005)	Coarse-Grained Classifiers	Learn semantic classes based on the WordNet lexicographers	Senseval 2 & Senseval 3 datasets

Table 3 Literature Review Summary on Semantic Class Disambiguation

Table 4 summarises the studies that use ML and NLP approaches for WSD.

Word Sense Disambiguation (WSD)				
	Author	**System/ML**	**Aim of Using the System/ML**	**Data Source**
Supervised WSD	Mooney (1996)	Naive Bayes (NB)	Disambiguate the meaning of words	Line corpus
	Pedersen (2000)	Naive Bayes (NB)	Combine NB classifiers into an ensemble to specify the lexical features for WSD	Line corpus & Interest corpus
	Towell & Voorhees (1998)	Neural Networks	Learn topical and local features of target words from a corpus based on WordNet	Noun line, verb serve, adjective data
	V´eronis & Ide (1990)	Neural Networks	Learn neural networks from definitions in a machine readable dictionary	
	Escudero et al. (2000)	*k*-Nearest Neighbor	Disambiguate word senses	DSO corpus
	Ng (1997)	*k*-Nearest Neighbor	Disambiguate word senses	DSO corpus
Unsupervised WSD	Schütze (1998)	Context-Group discrimination	Group occurrences of target words into sense clusters	
	Pedersen & Bruce (1997)	Agglomerative context clustering	Features of target words are grouped into clusters	
	Lin (1998)	Word clustering	Discriminate between word senses based on syntactic dependencies.	
Knowledge - based WSD	Banerjee & Pedersen (2003)	Extended Lesk algorithm	Measure semantic relatedness between concepts based on WordNet	Senseval-2 data
	Vasilescu et al. (2004)	Variations of the Lesk algorithm	Count overlaps between definitions of words and its context based on WordNet	Senseval-2 data
	Zouaghi et al. (2012)	Lesk algorithm + IR measures	Arabic word sense disambiguation	Corpora from the web

Table 4 Literature Review Summary on Word Sense Disambiguation

Table 5 demonstrates the studies that apply classification data mining algorithms for making useful decisions and learning simple classification rules.

Classification Methods for Inducing Decision Rules				
	Author	**System /ML**	**Aim of Using the System/ML**	**Data Source**
Decision Trees	Tanner et al. (2008)	C4.5	Detect diagnosis of dengue disease in the early phase of illness	A collection of patient enrolment, clinical and epidemiological data
	Lee (2010)	C4.5	Support commodity recommendation of retail business	A source of customer data from retail business in Taiwan
	Tsumoto & Hirano (2010)	C4.5	Detect the risk factors in clinical environments	A nurses' incident data and a clinical data of blood infection
Rule Induction	Džeroski & Lavrac (1996)	CN2	Diagnose new cases in early diagnosis of rheumatic diseases	Patient records with corresponding diagnosis
	Samanovic et al. (2011)	CN2	Predict the amount of foreign direct investments in a country	Online databases of business web sites
	Jovic & Bogunovic (2009)	RIPPER	Classify features of heart rate variability	Databases of patient records from the PhysioBank website
	Ulutaşdemir & Dağlı (2010)	RIPPER	Predict death risk in hepatitis	A clinical hepatitis datasets obtained from UCI
	Peng et al. (2011)	RIPPER	Predict any future financial risk	Credit risk and fraud risk datasets

Table 5 Survey Summary of Classification Methods for Inducing Decision Rules

Table 6 illustrates the studies that exploit prediction data mining algorithms for forecasting demands and predicting future data trends.

Prediction Methods for Forecasting Demands and Predicting Future Data Trends				
	Author	**System/ML**	**Aim of Using the System/ML**	**Data Source**
Linear / Multiple Regression Models	Huss (1985)	Linear Regression	Foresting sales of electric utility energy	Annual sales data of electric utility energy
	Bianco et al. (2009)	Linear Regression	Forecasting electricity consumption	Electricity consumption data (1970-2007)
	Forst (1992)	Multiple regression	Forecasting restaurant sales	Restaurant sales data
	Ramanathan (2012)	Multiple regression	Forecasting promotional sales of UK manufacturing company	Sales data of soft drink products (2005-2006)
Neural Networks	Alon et al. (2001)	Neural Networks	Forecast aggregate retail sales	Monthly retail sales data
	Chu & Zhang (2003)	Neural Networks	Predict aggregate retail sales	Monthly retail sales data
	Kumar & Mittal (2012)	Neural Networks	Forecast sales in automobile manufacturing company	Aggregate monthly sales data of motorcycles

Table 6 Survey Summary of Prediction Methods for Predicting Demands and Future Data

Finally, Table 7 presents the studies that apply ILP for learning grammar and semantic rules, NLP, and IE.

Inductive Logic Programming (ILP) for Learning Grammar and Semantic Rules, NLP and IE			
Author	**System /ML**	**Aim of Using the System/ML**	**Data Source**
Claveau et al. (2003)	ALEPH	Extract relevant noun-verb pairs and semantic rules from a POS and semantically tagged corpus	MATRA-CCR corpus
Tanabe & Wilbur (2004)	ALEPH	Generate rules of morphological features in order to identify gene/protein names in biomedical text	MEDLINE database
Eisenstein et al. (2009)	ALEPH	Perform semantic abstraction of documents and induce rules for the game of Freecell solitaire	Documents of instruction text of game data
Quinlan (1994)	FOIL	Learn rules for transforming English verbs from present to past tense	A corpus of 1391 verbs
Aitken (2002)	FOIL	Learn information extraction rules for identifying the meanings of sentences in text	Natural language text
Lindberg & Eineborg (1998)	PROGOL	Learn the rules of constraint grammar for POS tagging from a POS tagged corpus	A corpus of Swedish text (Stockholm Umeå)
Cussens et al. (1997)	PROGOL	Learn English grammar rules from sentences of children books	A series of children books
Cussens (1997)	PROGOL	Induce rules for specifying the possible POS tags of words in the sentences	Wall Street Journal corpus

Table 7 Survey Summary of Inductive Logic Programming for Learning Grammar and Semantic Rules, NLP and IE

2.6.2 Discussion

As presented in the introduction, this research explores the potential for using natural language processing, machine learning and data mining methods to develop a novel framework for employee performance appraisals that facilitates the setting of SMART objectives and providing feedback on these objectives. The developed framework applies ILP to learn grammar rules for writing SMART objectives and employs data mining techniques for assessing the objectives. Moreover, it uses NLP systems for analysing the objectives text. Thus, the literature survey presented in this chapter has focused on these areas.

The survey first investigated performance appraisal, goal setting and the SMART approach for setting objectives. With regards to the SMART approach, the literature survey, presented in Section 2.1, has shown that the SMART criteria have a lot of definitions which have few differences. Given that Hurd et al's. (2008) definition of SMART objectives clarifies the difference between “achievable” and “realistic” more clearly; this definition is adopted for this research.

The literature survey identified several systems based on ILP for learning grammar and semantic rules which include: ALEPH (Srinivasan, 1999), FOIL (Quinlan, 1990) and PROGOL (Muggleton, 1995). In principle, any of these could be adopted but following the survey of literature, this study utilised the ALEPH system to learn a grammar for writing SMART objectives because there was some evidence from the study by Claveau et al. (2003), that it produced good results when learning grammar and semantic rules. Even though they have used FOIL as well, their experiments showed that ALEPH is better than FOIL, because some of the induced rules using FOIL did not match the defined requirements in their study. The reason identified for this is the greedy search strategy used by FOIL. In addition, Quinlan (1994), who applies FOIL to learn rules for transforming English verbs from present to past tense, has demonstrated that FOIL was not able to produce general rules that cover all the examples and was unable to reorder the learned clauses due to its search strategy. In contrast, the ALEPH system has been applied successfully by many authors for learning grammar and semantic rules (e.g. Eisenstein et al., 2009, Claveau et al., 2003), where it was able to infer general and accurate rules from the given background knowledge and examples.

For creating classification and decision rules that could be used for making accurate decisions, there are several classification data mining algorithms which can be applied to perform this task such as decision trees (e.g. C4.5 (Quinlan, 1993)) and rule induction learners (e.g. CN2 (Clark and Boswell, 1991) and RIPPER (Cohen, 1995)). Thus, to construct classification and decision rules which can be utilised for assessing whether the objectives can be achieved within the available resources, any good and appropriate rule induction or decision tree algorithm could be used, however in this research, the RIPPER rule induction algorithm has been chosen for illustration purposes. Moreover, this algorithm has been applied efficiently by many researchers for making important decisions and classifying useful information (e.g. Jovic and Bogunovic, 2009; Ulutaşdemir and Dağlı, 2010; Peng et al., 2011).

There are many different methods that could be employed for forecasting demands and predicting future data such as: linear regression, multiple regression and artificial neutral networks (Han, Kamber, and Pei, 2011). In this research, linear regression has been used to assess whether an objective could be achieved within a given timescale. The linear regression has been applied in this study for illustration purposes, but more complex prediction data mining methods could also be easily utilised where necessary.

Furthermore, the literature review suggested various open source tools that support machine learning and data mining processes such as the WEKA (Hall et al., 2009; Witten, Frank, and Hall, 2011) and RapidMiner (Mierswa et al. 2006) toolkits. In this research, the WEKA data mining toolkit is used to build classification and predication models and to evaluate the performance of the constructed models on some datasets. As the WEKA software has been utilised successfully in different applications for making important decisions, learning classification rules, and predicting future events, it may have the potential to be used in a domain such as performance appraisals to support the process of setting SMART objectives by building the models which can be used for assessing the objectives.

For analysing, processing, and annotating the text of the objectives, the literature review has shown that there are a lot of open source NLP and IE systems which could be used such as NLTK (Loper and Bird, 2002), GATE (Cunningham et al., 2013), and the OpenNLP system (Baldridge, 2005). In this study, the GATE system has been utilised, where it has been applied effectively and extensively for processing texts, extracting useful information, and generating text annotations (e.g., Saggion, et al., 2007; Feilmayr et al., 2009; Joshi et al. 2012). The presented literature survey has demonstrated that the architecture of GATE is well-designed, since it offers a GUI for many NLP tasks. Moreover, among the available state-of-the-art technologies of NLP and IE, GATE is one of the most important software architectures due to its possibilities to integrate several modules written in different languages as well as its ability to deal with the web ontology language (OWL). In addition, it is platform independent software that supports Unicode and different data formats (e.g. HTML, XML, RTF, Doc and PDF) (Wahl, Winiwarter, and Quirchmayr, 2010).

The literature survey revealed different NLP techniques for the semantic analysis of text. These techniques can be applied either to perform the semantic class disambiguation (SCD) task or word sense disambiguation (WSD) task. For WSD, different approaches have been presented and reviewed including supervised WSD, unsupervised WSD and knowledge-based approaches. The surveyed approaches of supervised WSD use Naïve Bayes (e.g. Mooney, 1996; Pedersen, 2000), neural networks (e.g. Towell and Voorhees, 1998; V´eronis and Ide, 1990), and k-nearest neighbour (e.g. Escudero et al., 2000; Ng, 1997). All these approaches depend on the availability of sense-labelled training data. The studies that apply the supervised WSD approaches have achieved different experimental results, because of the different testing datasets, sense inventories (machine readable dictionaries), and the external knowledge resources adopted. With regards to unsupervised WSD, the surveyed approaches of this type include context clustering (Schütze, 1998; Pedersen and Bruce, 1997) and word clustering (Lin, 1998; Pantel and Lin, 2002), where these approaches do not rely on any sense-labelled training data. The surveyed knowledge-based approaches use knowledge resources and machine readable dictionaries for WSD and include the structural approaches (e.g. Leacock and Chodorow, 1998; Hirst and St-Onge, 1998) and the methods of overlap of sense definitions such as the Lesk approach (e.g. Banerjee and Pedersen, 2003; Vasilescu et al., 2004; Zouaghi, Merhbene and Zrigui, 2012).

In this research, the WordNet machine readable dictionary, which defines the meanings of words and encodes many relationships (semantic/lexical) between synsets or words, has been used as a knowledge resource for performing the semantic analysis of the objectives. The semantic analysis of the objectives is performed to identify semantically the target words that are commonly used in writing SMART objectives. There are several open source tools that can be used to access the WordNet database for retrieving the semantic information required to perform the semantic analysis of the objectives. For example, the open source software called WordNet::SenseRelate::AllWords (SR-AW) disambiguates the senses of words in context by using information available in WordNet. This software is easy to use and it has been employed effectively by many researchers (e.g., Klyuev and Oleshchuk, 2010; Shabanzadeh et al., 2010) for WSD. Additionally, the SR-AW software includes the ‘lesk’ (extended gloss overlaps) semantic relatedness measure which is based on the Lesk approach that represents one of the well-known knowledge-based approaches utilised for WSD.

Thus, the SR-AW system has been applied in this study to process the ambiguity of words in the objectives text and retrieve the senses of words in context from WordNet, where the experiments have been conducted using the ‘lesk’ measure in the SR-AW algorithm. Furthermore, the literature review has suggested various approaches for SCD such as perceptron trained Hidden Markov Models and coarse-grained classifiers as well as the open source package called WordNet QueryData. Given that the WordNet QueryData module is free and able to access the WordNet database easily to return some useful semantic information, this module is also adopted in this research to complement the task of the semantic analysis of objectives text. In more detail, the WordNet QueryData module has been used to retrieve the semantic class labels (categories) of the target words in the objectives text in order to semantically annotate the target words in the objectives with the retrieved semantic classes, and therefore distinguish the words that are commonly utilised in formulating SMART objectives.

2.6.3 Summary

In this chapter, a detailed overview of performance appraisal and goal setting was presented for illustrating their benefits and challenges. The presented overview of goal setting also included an extensive survey of the SMART approach for studying what constitutes well-written SMART objectives. A comprehensive survey of NLP, its tasks and applications including IE was given. In the given survey of NLP, several studies that apply various IE systems for text annotation were reviewed. Furthermore, many studies that employ different NLP techniques for processing natural language text semantically were also surveyed. A detailed explanation of data mining and machine learning including classification and prediction methods was proposed. The survey illustrated and described a number of studies that utilise a range of classification methods for creating useful classification and decision rules. In addition, it presented some studies that use a variety of prediction methods for forecasting demands and predicting future data. A thorough overview of ILP and its methods was presented. The presented overview for ILP also described several studies that apply different ILP systems for learning grammar and semantic rules, NLP and IE. A description of the functions that are used for evaluating the performance of machine learning and data mining methods was given. The given survey in this chapter also included a set of tables that brings together the reviewed research in the areas of NLP, IE, machine learning and data mining. A discussion of the surveyed studies was presented for summarising the key aspects from these studies, clarifying the techniques that have been selected for developing a framework for a system to support employee appraisals, and describing the motivational reasons behind choosing such techniques for developing the framework for this research.

In the next chapter, a framework for employee performance appraisal will be proposed. A description of the proposed framework which helps employees to set SMART objectives and provides feedback regarding the written objectives based on the use of ILP and data mining methods will also be presented. Moreover, a typical scenario of the system that supports performance appraisals will be given.

Chapter 3: A New Framework for Employee Appraisals

The above chapters describe the motivation, research objectives, and research methodology as well as present a literature review for appraisals, NLP, machine learning and data mining methods. This chapter develops and presents a new framework for supporting employee performance appraisals based on the use of inductive logic programming and data mining techniques. Section 3.1 presents a typical scenario to illustrate the potential capabilities of the system that supports performance appraisals. Section 3.2 proposes the framework for this research. Section 3.3 presents a brief summary of this chapter.

3.1 System Scenario

This section presents a scenario to illustrate some of the potential features that can be expected in a system that supports employee performance appraisals.

An employee logs into the system to participate in the process of setting objectives. The system provides an interface where the employee has the option to enter one or more objective sentences to be assessed by the system. The employee starts writing the objectives on the system interface. Then, the system views (parses) the objective sentences, assesses whether they are SMART and gives guidance on formulating these objectives. In particular, the system will be able to give feedback to the employee regarding the written objectives. In case that the employee did not enter the required data in an objective sentence (data is missing), then a feedback message will appear to prompt the employee to enter the missing data. This feedback message will also demonstrate the reason for the objective not being SMART. In case the employee has entered incorrect data in the objective sentence, a feedback message will appear to suggest which part of the objective sentence must be corrected to proceed. The system will store the entered objective sentences by the employee with the results of appraising these objectives in a form. Once the form has been completed to the satisfaction of the employee, then it is saved and submitted to be reviewed by the appraiser. Figure 1 presents a use case diagram of the proposed system to support employee performance appraisals.

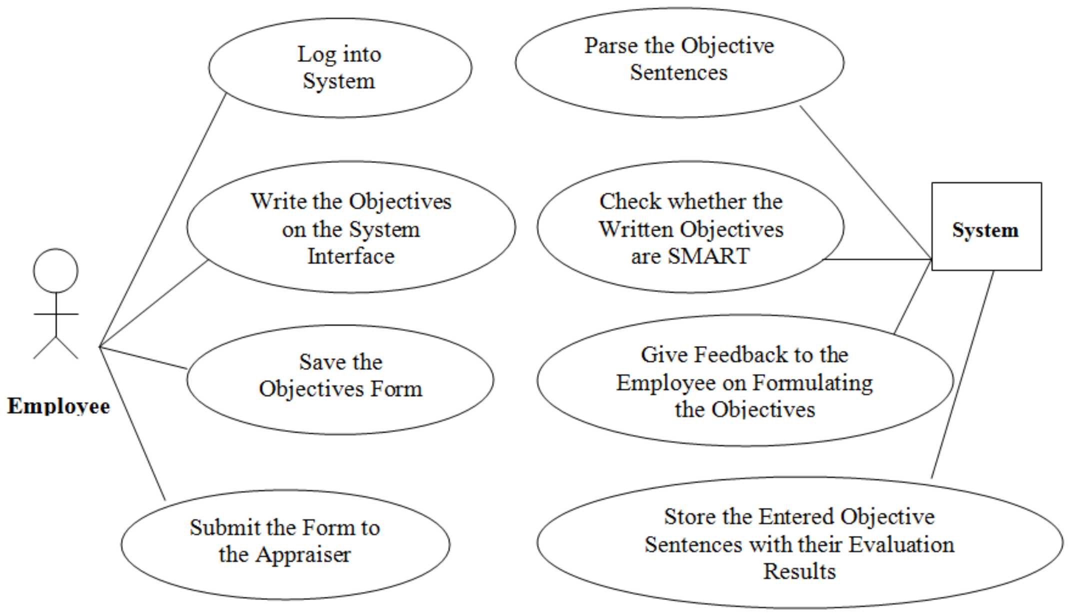

Log into System
Parse the Objective Sentences
Write the Objectives on the System Interface
Check whether the Written Objectives are SMART
System
Employee
Save the Objectives Form
Give Feedback to the Employee on Formulating the Objectives
Submit the Form to the Appraiser
Store the Entered Objective Sentences with their Evaluation Results

Figure 1 Use Case Diagram of the Potential Capabilities of the System that Supports Appraisals

3.2 A Framework for Appraisals Based on ILP and Data Mining Methods

As described in Section 2.1 of the literature review, the starting point for carrying out appraisals is for an employee to write SMART (specific, measurable, achievable, realistic, and time-related) objectives.

This research develops a new framework for supporting employee performance appraisals, where the developed framework aims to help employees to set effective business objectives and provide them with constructive feedback about their objectives. Thus, this framework has to include components that support the process of setting SMART objectives and providing feedback, where these components can be used for checking whether the objectives written are “specific”, “measurable”, “achievable”, “realistic”, and “time-related” and, at a minimum requires capabilities for:

- Analysing the written objectives and assessing whether these objectives include the features expected for “specific”, “measurable”, and “time-related” criteria such as specifying the action taken to achieve an objective, setting a clear end date to attain the objective and determining a quality/quantity measure (numeric indicator) that can be used for evaluating the achievement of the objective.

- Assessing whether the objectives are “achievable” and “realistic”, where this involves checking whether they can be achieved given the available resources and time.
- Providing feedback for employees so they are able to write SMART objectives.

The challenge, of course is, how can this be achieved? The proposal in this thesis is to develop a framework that has two main phases:

- Firstly, to learn grammar rules which can be used for checking whether a written objective is SMART and ensuring that an objective is “specific”, “measurable”, and “time-related”.
- Secondly, to assess if the objectives are “achievable” and “realistic”.

Given that sentence analysis is necessary to assess whether objectives are written appropriately, NLP methods seem appropriate. As mentioned in the survey, there are no known algorithms for checking whether objectives are SMART, so the rules for writing SMART objectives need to be discovered.

In this research, machine learning is used to induce a grammar for writing SMART objectives, while data mining methods are applied for assessing the objectives. First, a set of objective sentences is created, and since the text of these objective sentences is unstructured, NLP and IE techniques are applied to perform the linguistic analysis of the text and process the objective sentences in order to be an appropriate input to a machine learning method. A system has been developed based on the grammar rules learned by the applied machine learning method as well as the rules and equations generated by the utilised data mining methods in order to assess whether an objective is SMART. A user (i.e. employee) enters the objectives into the system for processing and assessing. Then, the developed system assesses whether these objectives are SMART and provides the employee with a constructive feedback on the written objectives. Figure 2 presents a general view for the framework.

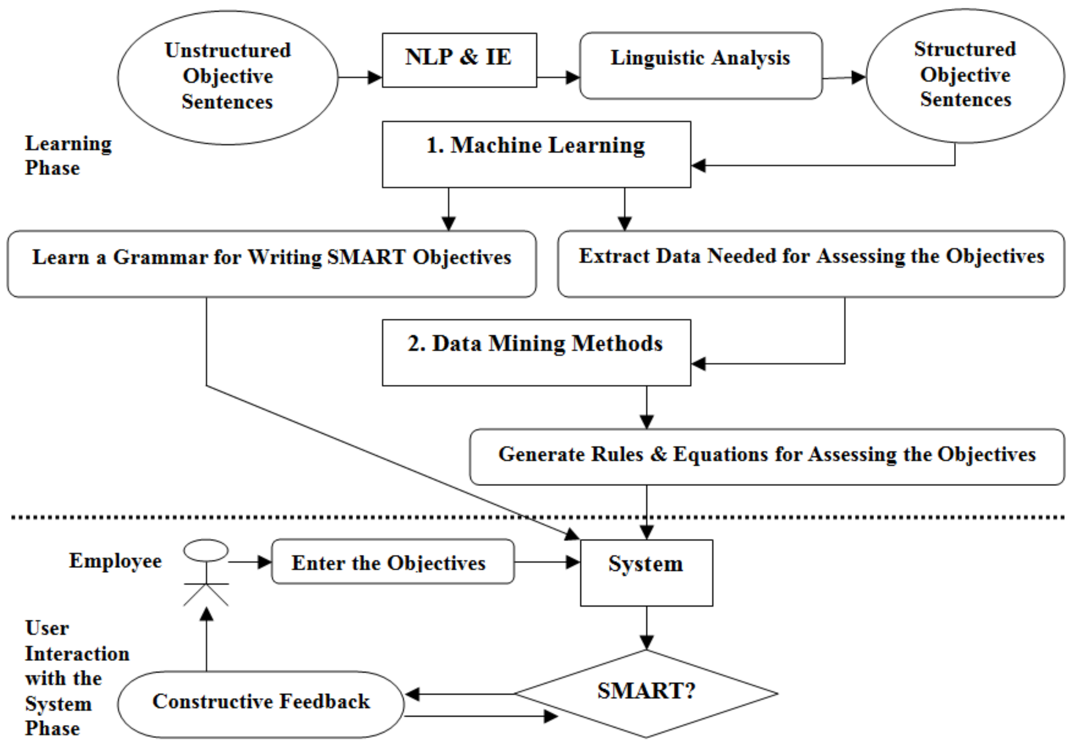

Unstructured Objective Sentences
NLP & IE
Linguistic Analysis
Structured Objective Sentences
Learning Phase
1. Machine Learning
Learn a Grammar for Writing SMART Objectives
Extract Data Needed for Assessing the Objectives
2. Data Mining Methods
Generate Rules & Equations for Assessing the Objectives
Employee
Enter the Objectives
System
User Interaction with the System Phase
Constructive Feedback
SMART?

Figure 2 General Framework

In more detail, the developed framework applies inductive logic programming (ILP) to learn grammar rules which can be used for helping employees to write SMART objectives and ensuring that objectives are “specific”, “measurable”, and “time-related”.

Before ILP can be used, there is a need to develop a corpus of objective sentences, process its text, annotate it and select the suitable training examples of objective sentences. As mentioned in Chapter 2, various tools are available for NLP, IE and text annotation including NLTK (Loper and Bird, 2002), GATE (Cunningham et al., 2013), and the OpenNLP system (Baldridge, 2005). In this research, text annotation of the corpus is done by using the GATE architecture (Cunningham et al., 2013). As described before, there are some features that must be present in the written objectives in order to be SMART. Thus, the GATE system is applied to process the objectives text in order to extract some useful information which can then be used to check the presence of these features in the objectives. Basically, GATE is exploited to perform a syntactic analysis of the text at the word level by identifying a part of speech (POS) tag (e.g. verb, noun, adjective, adverb etc) for each word in the objectives. Moreover, it has been utilised to provide some semantic information such as recognising entity names (i.e. proper nouns) in the objectives text. This includes identifying the numerical expressions (i.e. numeric indicators) and temporal expressions (i.e. dates) which can then be used to ensure that an objective is “measurable” and “time-related”. To be able to assess if an objective includes the features expected for the “specific” criterion, this requires checking if the given objective specifies clearly the action to be taken to achieve the objective outcomes by using a special type of verbs such as action verbs (e.g. ‘increase’, ‘reduce’, ‘maximise’, ‘generate’ etc) (Rouillard, 2003). The use of action verbs in writing objectives makes the objectives to be more action-oriented, specific and focused. Consequently, we must be precise in choosing the right verbs when setting useful objectives, where these verbs should describe specifically the observable action which needs to be made in order to achieve the desired results (i.e. as long as the verbs are specific, they will be much easier to measure). On the other hand, we must avoid using verbs that may have unclear/general meanings to describe the intended results and they are difficult to be measured (e.g. ‘understand’, ‘recognise’, ‘know’, ‘realise’, ‘appreciate’, ‘acknowledge’). Moreover, we must avoid utilising verbs that describe emotions and feelings (‘feel’, ‘enjoy’, ‘desire’, ‘want’, ‘love’) or verbs that describe perceptions (e.g. ‘see’, ‘notice’, ‘observe’, ‘hear’).

Some of the studies presented in the literature review which apply the SMART approach for setting good objectives have shown that there is another criterion that must be taken in to account when assessing if an objective is “specific”, since the objective must state precisely the observable action by specifying also the objective domain (e.g. the objective pertains to revenue/sales/costs/profits etc) (Carliner, 1998).

Based on these aspects, the words which indicate the objective domain and the action verbs that are used commonly in writing SMART objectives must be studied semantically in order to specify the words that help to formulate explicit and well-written SMART objectives. In particular, this involves performing more semantic processing of the words (e.g. verbs and common nouns) in the objectives in order to specify some semantic features which can be utilised to ensure that an objective is “specific”. Therefore, the WordNet (Fellbaum, 1998; Miller et al., 1990) lexicon which defines the words and provides semantic information of words and meanings is used in this study as a knowledge resource to derive a semantic interpretation of text by semantically annotating the target words in the objectives. This will facilitate identifying the target words that are commonly utilised in setting SMART objectives. Here the target words represent the action verbs and the words that indicate the objective domain, where these words are recommended to be used for writing quality objectives.

To assess whether an objective is “achievable” requires assessing whether it can be achieved within a given timeframe. This requires comparing the measure set in the objective sentence with the suitable timeframe to achieve this measure. To be able to carry out this assessment, some variables from the objective sentences such as evaluation measures/indicators (e.g. amount of sales) and timescales (e.g. dates) should be first extracted. If these can be extracted, the tasks can be summarised as follows:

- Estimate the expected value of some target value for a specified timeframe, for example sales in the case of sales and marketing.
- Compare this with the value (measure) set in the objective, and check whether the value in the objective could be achieved within the proposed timescale.

How can we carry out the estimation and comparison with respect to the target value and proposed timescale? In general, this can be a subjective process but in some fields data may enable a more quantitative approach. A significant amount of research has been carried out in both subjective assessments (e.g. Yeh and Chang, 2009; Mallinson, 2002) and quantitative assessments (e.g. Chor, 2010; Blind, 2012). In this study, data mining methods will be utilised for a particular domain to ensure that the objectives are “achievable” by forecasting the objective outcomes based on past historical data, but this may vary and the intension of this framework is for the most appropriate methods to be selected given the domain of application. Of course, there will be domains where estimation and even subjective prediction is difficult in which case this part of the framework will not be appropriate.

To assess whether an objective is “realistic”, this means that sufficient resources are available to achieve the objective. The assessment task here necessitates classifying the required resources that should be allocated within a given timeframe in order to attain the objective outcomes. Therefore, the classification methods in data mining, which support the decision making process by creating decision and classification rules, could be utilised for ensuring that an objective is "realistic".

These considerations lead to the framework presented in Figure 3, which is described in more detail in the paragraphs below the figure. The dotted line in the middle of the following figure separates the development of the framework into two main phases. The first phase is the learning phase, while the second phase is the user interaction with the system phase. The boxes (rectangles) that appear in the figure represent the processes (approaches), rounded rectangles represent the actions made by the processes, diamonds indicate decisions, ovals represent the results and output, the parallelogram notation indicates the input data, and the octagon notation represents the final assessment. The diamond notation has two arrows coming out of it; one of them is a solid line arrow which indicates the positive decision (e.g. yes/true), while the other is a dashed line arrow which denotes the negative decision (e.g. no/false).

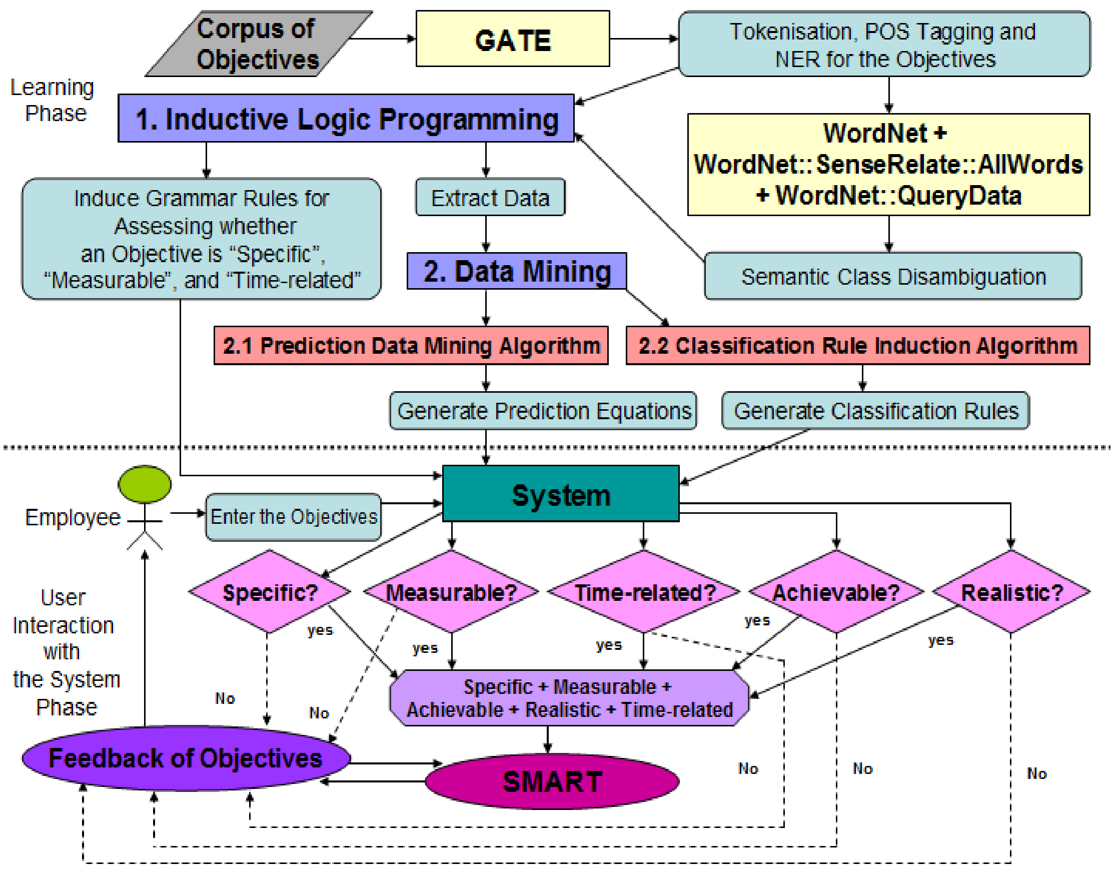

Corpus of Objectives
GATE
Tokenisation, POS Tagging and NER for the Objectives
Learning Phase
1. Inductive Logic Programming
WordNet + WordNet::SenseRelate::AllWords + WordNet::QueryData
Induce Grammar Rules for Assessing whether an Objective is "Specific", "Measurable", and "Time-related"
Extract Data
2. Data Mining
Semantic Class Disambiguation
2.1 Prediction Data Mining Algorithm
2.2 Classification Rule Induction Algorithm
Generate Prediction Equations
Generate Classification Rules
System
Employee
Enter the Objectives
User Interaction with the System Phase
Specific?
Measurable?
Time-related?
Achievable?
Realistic?
yes
No
Specific + Measurable + Achievable + Realistic + Time-related
Feedback of Objectives
SMART

Figure 3 Detailed Framework

In the above detailed framework, a corpus is created containing positive and negative examples of objective sentences, where the positive example sentences indicate SMART objectives, while the negative example sentences indicate non SMART objectives. The learning phase starts first by processing and annotating these objective sentences using the GATE system. The annotation process includes performing tokenisation, POS tagging of the objectives text, identifying entity names in text (i.e. by carrying out the named entity recognition (NER) task), and specifying some other semantic annotations. To illustrate the annotation process performed by GATE, consider the following example of a SMART objective.

> *Example 1*: "To improve PC sales by 4% at the end of the first quarter of fiscal year 2008."

By applying GATE for text processing, the above sentence of *Example 1* is tokenised (segmented) into a set of words as follows:

> To, improve, PC, sales, by, 4, %, at, the, end, of, the, first, quarter, of, fiscal, year, 2008, .,

Then, the POS tags are specified for each word in the above sentence as follows:

> To/TO improve/VB PC/NN sales/NNS by/IN 4/CD %/NN at/IN the/DT end/NN of/IN the/DT first/JJ quarter/NN of/IN fiscal/JJ year/NN 2008/CD ./.,

where VB means a verb in the base form, NN indicates a singular noun, NNS means a plural noun, IN is a preposition, CD means a cardinal number, DT represents a determiner, and JJ indicates an adjective. Moreover, the entity names that represent proper nouns are identified (extracted) in the sentence (e.g. '4%' which represents a percentage (numeric indicator/measure) and 'first quarter of fiscal year 2008' which indicates a date).

After utilising the GATE system, the WordNet lexicon is applied to provide the meanings of words in the objective sentences as well as to retrieve the semantic classes of the word meanings. The semantic analysis of the objectives will help identifying the words that are commonly used in writing SMART objectives.

To be able to access the WordNet database for retrieving the needed semantic information, there are some open source packages that can be utilised to perform this task such as the WordNet::SenseRelate::AllWords software (Pedersen and Kolhatkar, 2009), and the WordNet QueryData module (Rennie, 2000). In this research, the WordNet::SenseRelate::AllWords system is applied on the POS tagged corpus of objective sentences in order to automatically retrieve the meanings of all words in the sentences from WordNet based on the context in which they occur. Then, the WordNet QueryData package is used to disambiguate the words in the objectives text at the semantic class level in order to semantically annotate the target words that appear in the objective sentences with the appropriate semantic class labels available in WordNet. To clarify how WordNet is used to perform the semantic analysis for an objective sentence such as the one presented in *Example 1*, first this objective sentence is given to the WordNet::SenseRelate::AllWords algorithm to be processed, where the POS tags are specified for each word in this sentence. The algorithm will disambiguate the meanings of words in the sentence and assign each word with its possible WordNet sense based on the context in which it appears. For example, the meaning that has been retrieved from WordNet by the algorithm for the word '*improve*' in *Example 1* is: "*to make better; "The editor improved the manuscript with his changes"".* Furthermore, the disambiguated meaning by the algorithm for the word '*sales*' is: "*income (at invoice values) received for goods and services over some given period of time*".

By reviewing the above objective sentence of *Example1*, it is clear that the retrieved meanings of the words "*improve*" and "*sales*" are correct in term of the context in which they appear. The algorithm has also provided a meaning for each of the other words in the objective sentence of *Example 1*.

Then, the defined meanings of the target words (e.g. 'improve' and 'sales') in *Example 1* are given to the WordNet QueryData module for specifying a semantic class for each meaning of these words based on the WordNet semantic classes (lexicographers). For instance, the retrieved semantic class by WordNet QueryData for the target word ‘improve’ is ‘verb. change’, where the change verbs include: verbs of size, temperature change, intensifying etc. It is clear that any verb classified under these types (verbs of size, temperature change, intensifying etc) indicates an action. Moreover, the retrieved semantic class by WordNet QueryData for the word ‘sales’ is ‘noun. possession’, where the nouns of this type denote possession/ownership or transfer of possession. Therefore, the retrieved semantic classes by WordNet QueryData are used for annotating the target words in the objective sentence semantically (e.g. ‘improve $_{\textbf{verb. change}}$’, ‘sales $_{\textbf{noun. possession}}$’).

As described in the literature review, lexicographers in WordNet include many semantic classes for all word categories including nouns and verbs (e.g. verb. perception, verb. consumption, verb. emotion, verb. change, verb. creation, noun. feeling, noun. food, noun. possession, noun. animal etc), where these semantic classes group together many synsets. Thus, the semantic analysis of objectives text clarifies which kinds of verbs and nouns that should be used when setting SMART objectives. For example, an employee must write a “specific” objective by using an action verb classified under the verb semantic class called “change” (e.g. verbs of size, temperature change, intensifying etc), and he/she must avoid using any verb classified under the verb semantic classes “emotion” (e.g. verbs of feeling) and “perception” (e.g. verbs of seeing, hearing, feeling).

After utilising GATE and WordNet for linguistic analysis, the POS and semantically tagged objective sentences are provided to the ILP learner to learn a grammar for writing SMART objectives. This grammar can then be used to check whether an objective is “specific”, “measurable”, and “time-related”. In addition, some of the extracted information by ILP is used again in developing the other parts of the framework (e.g. “achievable” and “realistic”). In the part related to check whether an objective is “achievable”, a prediction data mining algorithm is applied for assessing if the objective outcomes can be met within the proposed timeframe in the objective sentence. For assessing whether an objective is “realistic”, a classification rule induction algorithm is utilised to find a set of classification and decision rules that can be used for specifying the required resources to achieve the objective outcomes in a given timeframe.

The second phase of the framework (i.e. the user interaction with the system phase) is based on developing a system that uses all the grammar rules learned by the applied ILP method as well as the classification rules and the prediction equations generated by the utilised data mining techniques for assessing whether an objective is SMART. In particular, the developed system will utilise this information to check whether all five requirements (e.g. specific, measurable, achievable, realistic, and time-related) are met for an objective which has been entered and submitted by an employee for processing and assessing. In case they are satisfied for the objective, then the objective will be SMART. Otherwise, guidance and feedback messages are given to the employee for helping him/her to write SMART objectives.

3.3 Summary

In this chapter, a typical scenario for showing the potential capabilities of the system that supports performance appraisals was presented. A new framework for employee performance appraisal was proposed. A brief description of the proposed framework which facilitates the setting of SMART objectives and providing feedback based on the use of ILP and data mining methods was also presented. An illustrative example was given to demonstrate who an objective sentence is linguistically analysed using some components mentioned in the framework such as GATE, WordNet, WordNet::SenseRelate::AllWords and WordNet QueryData.

In the next chapter, a description of the development of a corpus of objectives for evaluating the research will be presented. A linguistic analysis of the developed corpus will be given.

Chapter 4: The Development and the Linguistic Analysis of the Corpus

The previous chapter outlined the framework and Figure 3 demonstrated the main parts of the framework. This chapter develops a novel system based on the proposed framework and presents the development of a corpus of objectives and the linguistic analysis performed on this corpus which will then be used by a machine learning technique for learning a grammar for SMART objectives. The chapter is organised as follows:

- Section 4.1 describes the developed corpus of objectives.
- Section 4.2 illustrates the linguistic analysis and the semantic classification of the corpus of objectives.
- Section 4.3 presents a summary of this chapter.

4.1 The Development of the Corpus

To the best of the author's knowledge there is no publicly available benchmark data that could have been used at the time of this study. Hence, this research aimed to develop a corpus that could be used for this kind of research.

For developing a relevant corpus for this research, different methodologies, adopted by some authors for building corpora, have been reviewed in order to identify the general concepts for constructing a well-designed corpus that can be used easily by the other researchers and users. For example, the methodologies presented by McEnery and Wilson (2001) and Meyer (2002) describe the standards for developing a corpus of text for use in linguistic analysis and language studies (e.g. lexicography, grammar, semantics and NLP). These standards include designing a corpus that is suitable for a variety of uses, specifying the type of the language for the corpus text and then gathering the appropriate texts for inclusion in the corpus. At this stage, the corpus builders have to do some research to discover the suitable textual data in order to include in the corpus. The standards for constructing a well-formed corpus also include specifying the mode of text (e.g. if the data in the corpus is a collection of written texts or transcribed speech), the domain of text, the type of textual data to be sampled (e.g. an article, a journal, a book etc), and the size of data samples. Furthermore, the contents of a corpus should represent the language from which they are chosen and the text samples must be balanced (McEnery and Wilson, 2001; Meyer, 2002). By applying these criteria to construct a corpus, the corpus is then ready for the manual annotation by experts and for text pre-processing.

The research carried out in this study is different from the kind of corpus developed in studies mentioned above, but the key criteria of developing a representative corpus apply. Thus, for this research, an English corpus has been constructed consisting of a set of objectives which have been created by studying what constituted good practice in writing well-written objectives and by reviewing the SMART objective examples in the studies surveyed in the literature review (e.g. Carliner, 1998; Rouillard, 2003; Hurd et al., 2008; Desmond, 2011). In particular, the suggested examples of SMART objectives in these studies have influenced the development of this corpus and the training examples that are used to learn a grammar for SMART objectives.

The process of developing this corpus has been conducted in two phases. Firstly, a corpus of 150 objective sentences has been developed, where each objective sentence in this corpus has been manually annotated as SMART or non SMART. More specifically, this corpus consisted of 100 examples of SMART objectives and 50 examples of non SMART objectives. As the research progressed and following comments from reviewers of the paper submitted to NLDB'2013, the use of only 150 objective sentences was not considered large. Therefore, this corpus was extended to include a total of 300 examples of objective sentences, where 130 objective examples have been manually tagged as SMART objectives, while the other 170 objective examples have been manually labelled as non SMART objectives (see Appendix C5).

The objective sentences in the created corpus for this study are related to a business application, particularly the sales domain, which is a significant part of many organisations and has sufficiently well-defined characteristics for a research trial. However, the business application also includes other important subareas (e.g. profitability, productivity, market share, costs and expenses), in which the SMART approach can be applied for setting good business objectives.

As mentioned before, the objectives in the applied corpus were developed by selecting examples from the literature (and also by adding some other new objective examples), such that there are some variations were carried out on the numerical measures and dates in these examples based on the selected sales domain. To ensure that all of these examples were tagged correctly, the author consulted experts from the human resource and the sales departments of a company[8] in Jordan.

The following steps summarise the methodology used for developing a corpus of objectives:

1. Review some methodologies of corpus development adopted by some authors.
2. Survey some of the key studies that have applied the SMART approach for setting objectives.
3. Review some examples of SMART objectives suggested in the surveyed studies.
4. Create a set of objectives that includes some real SMART objective examples that have been suggested in the reviewed studies, some other new SMART objectives, and some ineffective objectives.
5. Annotate each objective sentence in the developed corpus as SMART or non SMART by experts.

The following section describes how the developed corpus of objectives has been processed, analysed and annotated linguistically.

4.2 The Linguistic Analysis and the Semantic Classification of the Corpus

This section describes how the corpus of objectives has been syntactically and semantically annotated to provide some contextual information for the learning method which is then utilised to learn a grammar for writing SMART objectives.

As mentioned in Chapter 3, this research develops a new framework that supports the setting of SMART objectives and providing feedback. A novel system has been developed based on this framework to assess whether the written objectives are

[8] Jordan Techniques for Bio Detergent, Amman, Jordan.

SMART, where it uses machine learning to check whether an objective is “specific”, “measurable”, and “time-related” and applies data mining techniques to assess whether an objective is “achievable” and “realistic”. To perform these assessments and to support the process of setting SMART objectives, the rules of writing SMART objectives must be discovered.

To find the rules of formulating SMART objectives, the text of objectives must be analysed first in order to extract some useful features that can be then used in learning the rules of writing SMART objectives. In particular, the analysis of the objectives will support finding the expected features for an objective to be “measurable”, “time-related” and “specific”, where these features include numeric measures, dates and target words which are needed to write SMART objectives.

As the objective sentences in the developed corpus represent unstructured text, NLP and IE techniques are utilised to process the objectives text linguistically. Processing the text of objectives will help to transform this text into a structured form that can be used as an input to a machine learning technique. It also supports identifying useful linguistic patterns for the SMART criteria in the objective sentences. Moreover, the objectives must be also semantically analysed to identify the target words that are used commonly in writing SMART objectives.

The rest of this section is organised as follows. Section 4.2.1 describes how the corpus of objectives has been processed and annotated linguistically by using GATE. Section 4.2.2 illustrates the semantic analysis and classification of the objective sentences in the created corpus by using the WordNet lexicon.

4.2.1 Using GATE for Text Annotation

This section describes how the GATE architecture is used to automatically annotate the corpus of objectives and extract some useful information that can then be utilised by the applied machine learning method for learning a grammar for SMART objectives in order to ensure that the objectives are “specific”, “measurable”, and “time-related”. Moreover, it presents a brief description of the JAPE grammar and illustrates the annotations that have been created using the JAPE transducer processing resource in GATE. Finally, it displays a snapshot of the annotations which have been extracted by GATE.

4.2.1.1 Using ANNIE in GATE

The literature review presented some of the open source software for IE and NLP. In this research, the GATE architecture (Cunningham et al., 2013), which provides various tools for NLP and IE, is used for analysing, processing and annotating the objectives that appear in the developed corpus.

In particular, the ANNIE IE system, whose components are included in the GATE architecture, is utilised to discover some important linguistic patterns from the objectives text. These linguistic patterns are then employed by the applied machine learning method for learning a grammar for SMART objectives which can be used to check whether the written objectives are “specific”, “measurable”, and “time-related”.

Therefore, the corpus of objectives is provided as an input to the GATE developer, which is an integrated development environment (IDE) to run the ANNIE components and plugins. Then, some of the main ANNIE processing resources (i.e. components) such as ANNIE English Tokenizer, ANNIE gazetteer, ANNIE sentence splitter, ANNIE POS tagger, ANNIE NE transducer and JAPE transducer, are applied for processing the objectives text. All these processing resources are combined and arranged in a GATE pipeline with a noun phrase (NP) Chunker plugin. After that, this pipeline is run over the corpus of objectives to automatically generate text annotations. The following points describe how these processing resources and the NP Chunker plugin are employed to process and annotate the objective sentences in the corpus:

- ANNIE English Tokenizer is used to tokenise the objective sentences in the corpus and generate the ‘Token’ annotations. Each of the extracted tokens identifies the annotation features for the processed textual patterns (e.g. the annotation features specified for the token that extracts the word ‘sales’ are: “string: ‘sales’, category: NNS (plural noun), kind: word, length: 5 and orth (orthography): lowercase”).

- ANNIE Gazetteer, which includes lists of entity names (proper nouns), is used to create the ‘Lookup’ annotations by finding entities (e.g. %, 2008, year, March etc) in the corpus text that match some entries in the gazetteer lists (e.g. percent list [%, percent], year list [2008, 2009 etc], date unit list [year, years, months etc], months list: date [January, February, March etc] and number list [one, two, three etc]).
- ANNIE sentence splitter is used to annotate each sentence in the corpus with a ‘Sentence’ annotation and tag each sentence break (e.g. full stops) with a ‘Split’ annotation.
- ANNIE POS tagger, which performs a syntactic analysis at the word level, is utilised to automatically provide a part of speech tag for each word in the corpus text (e.g. ‘increase’: VB, ‘gross-sales’: NNS, ‘by’: IN).
- ANNIE NE transducer, which supports a main task in IE like NER, is applied to automatically identify the semantic annotations in text by classifying entity names (proper nouns) in the given objectives into predefined types (e.g. ‘Date’, ‘Percent’, ‘Money’). As shown in the examples of the SMART objectives which are presented in the literature review, numeric measures (e.g. ‘5%’, ‘3 percent’, ‘$20,000’) must be specified in the objective sentences in order to be “measurable” and more focused (“specific”). Furthermore, dates (e.g. ‘2008’, ‘June of next year’) must be stated in the objective sentences in order to be “time related”. By processing the corpus of objective sentences in GATE, the ANNIE NE transducer extracts some semantic annotations of entity names (e.g. ‘Date’, ‘Percent’, and ‘Money’) from the given objective sentences. Each of the generalised annotations indicates patterns that represent temporal expressions (e.g. dates) or numerical expressions (e.g. amounts of money and percentages). The applied ANNIE NE transducer annotates the patterns that represent dates, years, temporal expressions that include date-unit expressions, months of the year, or any month followed by a year with a ‘Date’ annotation. In addition, it annotates each number followed by a percentage sign or the string ‘percent’ with a ‘Percent’ annotation, and any currency sign followed by a number with a ‘Money’ annotation. For example, ‘2%’ is annotated as ‘Percent’, ‘£900’ is annotated as ‘Money’, and

‘13/11/2009’, ‘2009’, ‘November 2008’, and ‘next year’ are annotated as ‘Date’.

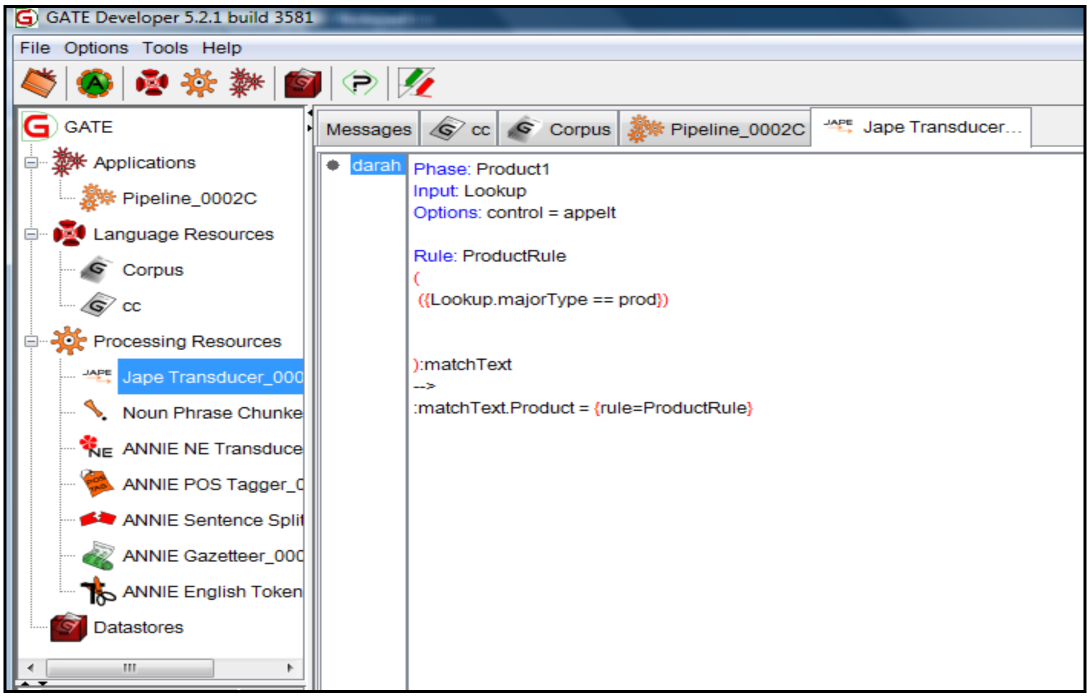
- JAPE transducer, which is a processing resource in ANNIE for finding regular expressions/patterns in text, is used to generate some annotations (e.g. ‘Product’ and ‘PrepositionalPhrase’) by matching textual patterns in the objectives text with created JAPE grammar rules (this step is described in more detail in Section 4.2.1.2).
- NP Chunker plugin, which segments each sentence into chunks (noun phrases), is employed to identify noun phrase chunks in the objective sentences by annotating each noun phrase with the ‘NounChunk’ annotation (e.g. ‘the volume’: NounChunk, ‘the average’: NounChunk).

4.2.1.2 Using JAPE Grammars

In addition to the POS tags and the named entity (NE) annotations of words that appear in the objective sentences, some other important textual patterns, which occur in these objective sentences, must be specified as well. To extract these textual patterns from the objectives text, a set of rules written in JAPE grammar has been created to generate the annotations that match these patterns in text. In GATE, JAPE grammar is processed by a transducer called the JAPE transducer to perform tasks like chunking, named entity recognition, and matching a simple text string. Therefore, the JAPE transducer processing resource has been utilised to create some useful annotations that find textual patterns in the corpus of objectives. Before illustrating these annotations, a brief description of the JAPE grammar is presented below.

The JAPE transducer (Cunningham et al., 2013) processing resource is used to run handcrafted rules of information extraction grammar for identifying patterns/regular expressions in the text processed by GATE. A JAPE grammar consists of a set of phases, where each phase is composed of a set of pattern rules. The JAPE rule has two sides, left hand side (LHS) and right hand side (RHS). The LHS of the rule includes an annotation pattern to be matched to the text annotated by GATE, while the RHS of the rule includes annotation manipulation statements that identify the action to be taken on the matched pattern/sequence of text. Accordingly, a rule first matches a pattern/sequence of text annotated with the annotation pattern specified in the LHS of the rule. Then, the matched part of text is allocated a label by the rule in the LHS. The part of text matched on the LHS of the rule is referred on the RHS by using the label specified in the LHS.

For instance, consider the following example for the grammar of the annotation called ‘Product’ which has been generalised by applying the JAPE transducer processing resource on the text of objective sentences, where this annotation describes the type of the product (e.g. PC or mobile) that appears in the provided objective sentences.



```
Phase: Product1
Input: Lookup
Options: control = appelt

Rule: ProductRule
(
 ({Lookup.majorType == prod})


):matchText
-->
:matchText.Product = {rule=ProductRule}
```

Figure 4 The Created JAPE Grammar for the ‘Product’ Annotation

As shown in Figure 4, the LHS of the rule is the part before the “-->”, whilst the RHS of the rule is the subsequent part. In this grammar example, there is only one rule for matching the patterns, and the name of this rule is ‘ProductRule’. This rule matches the text annotated with the LHS annotation pattern, which is a ‘Lookup’ annotation with a ‘majorType’ feature (discussed below in the next paragraph) of ‘prod’ (i.e. here the major type feature entitled ‘prod’). A list of product types called ‘product’ has been created and added to the gazetteer lists for finding occurrences of specific strings in the corpus text. This list includes specific keywords (e.g. ‘PC’, ‘mobile’, ‘PCs’, ‘mobiles’, ‘personal computer’, ‘personal computers’). The rule extracts the ‘Product’ annotation by matching the strings in text with the keywords defined in the created gazetteer list of product types. The matched text is assigned to a label called ‘matchText’ which is specified in the LHS of the rule. Then, the matched text on the LHS can be referred to on the RHS by utilising the label given in the LHS (‘matchText’). As a result, the matched text is annotated with the annotation called ‘Product’ which is specified in the RHS of the created rule.

In general, each JAPE grammar starts by specifying a phase name to a grammar. In the above example, the phase name is ‘Product1’ (i.e. Phase: ‘Product1’). In the ‘Input’ line, the grammar sets the annotation types (e.g. Token, Lookup, SpaceToken) which will be used when attempting a match for patterns. For instance, the annotation type for the grammar presented in the above figure is a ‘Lookup’ annotation (i.e. ‘Input: Lookup’). Generally, the major type and minor type features are specified when the Lookup annotation is used, where these features give access to all items stored in particular gazetteer lists or combinations of lists. The major type feature is specified in the JAPE grammar to match major information about patterns, while minor type feature is specified in the grammar to identify only optional information about specific patterns.

Moreover, there are different options that can be set at the start of each grammar for matching patterns in text, such as the control and debug options. One of these options must be set in the ‘Option’ line of the JAPE grammar. The control option specifies the method of rule matching and it has five different styles: ‘Brill’, ‘All’, ‘First’, ‘Once’ and ‘Appelt’. Only one of these control styles must be specified at the beginning of each JAPE grammar. If no control style is assigned to the grammar, the default is Brill.

- The ‘Brill’ style indicates that if there is more than one rule that matches the same part of the text, they are all executed without any need for a priority ordering of the rules during the execution. This style will execute all matching rules starting from a specified position in the text. The process of rule matching in this style will continue from a position in text where the longest match of a sequence of text ends.

- The ‘All’ style executes all matching rules that match the same segment of text. However, in this style the matching will advance and carry on from the next offset to the current one.
- In the ‘First’ style, a rule executes for the first match that is detected.
- In the ‘Once’ style, the whole JAPE phase of the grammar finishes after the first match, when a rule has fired.
- The ‘Applet’ style means that if there is more than one rule matching the same segment of text, only one rule can be executed for this part of text, depending on a set of priority rules.

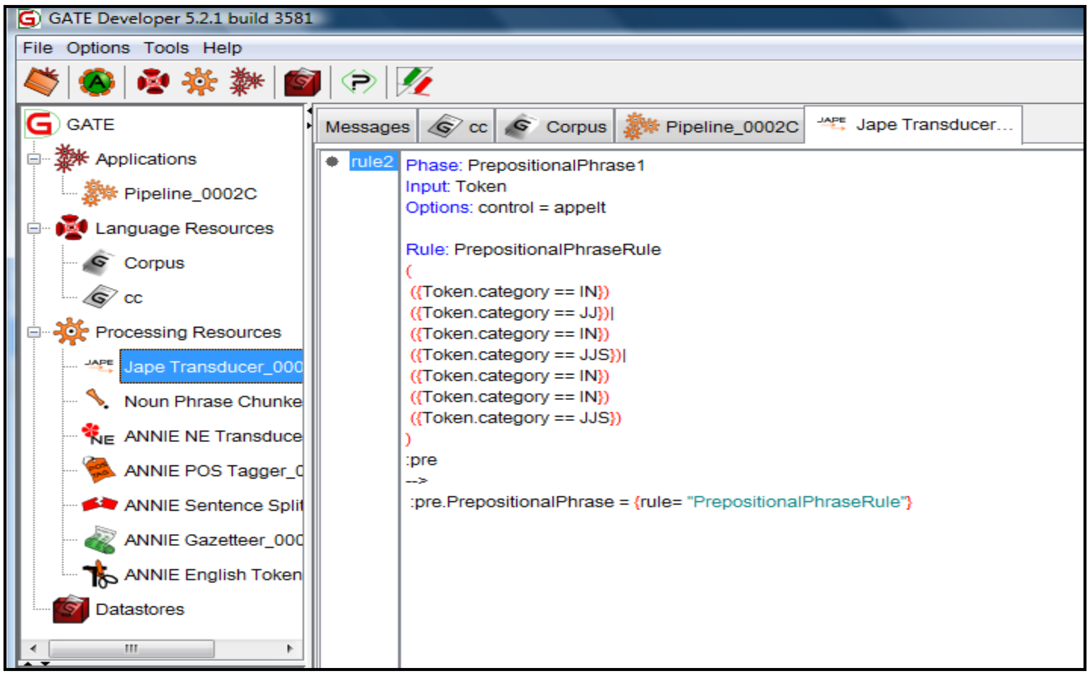
In the ‘Applet’ style, the rule priority works in the following manner:

1. Length of the matching rule. A rule that matches the longest segment of text is executed.
2. Associating optional priority declaration. If a priority declaration is assigned to each of the matching rules, then the rule that has the highest number of priority is executed.
3. Rules ordering. If there is more than one rule has the same value of priority, the one stated first in the grammar is executed.

In the above grammar example, the option of matching patterns in text is set to ‘control’ (i.e. Options: control), while the method of rule matching is set to the ‘Appelt’ mode (i.e. Options: control = applet).

The JAPE grammar provides some regular expression operators (e.g. *, +, ?, |) which appear in the LHS of the rule. The ‘*’ operator is specified in the JAPE grammar for matching zero or more patterns, while the ‘+’ operator is used in the grammar for matching at least one or more patterns. However, the ‘?’ operator indicates that the matching of a pattern will be optional, while the ‘|’ (OR) operator is used to indicate alternatives.

The JAPE transducer has also been used to generalise the annotation called ‘PrepositionalPhrase’ from the corpus of objectives text. Figure 5 presents the created JAPE grammar for detecting the ‘PrepositionalPhrase’ annotation.



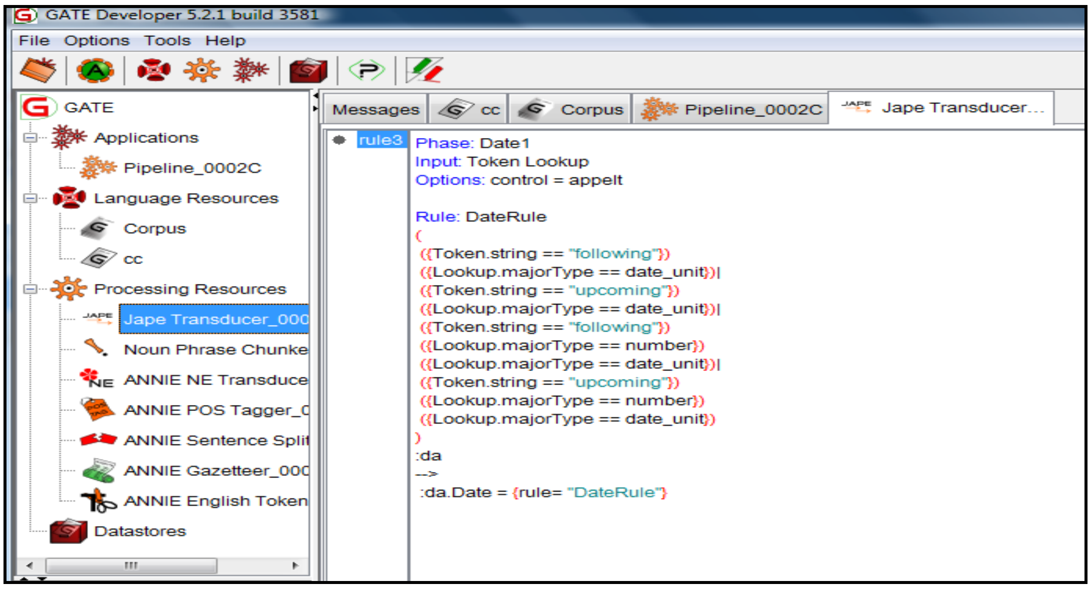
```
Phase: PrepositionalPhrase1
Input: Token
Options: control = appelt

Rule: PrepositionalPhraseRule
(
 ({Token.category == IN})
 ({Token.category == JJ})|
 ({Token.category == IN})
 ({Token.category == JJS})|
 ({Token.category == IN})
 ({Token.category == IN})
 ({Token.category == JJS})
)
:pre
-->
 :pre.PrepositionalPhrase = {rule= "PrepositionalPhraseRule"}
```

Figure 5 The Created JAPE Grammar for the 'PrepositionalPhrase' Annotation

In the grammar presented in the above figure, the rule called 'ProportionalPhraseRule' is created to match a sequence of text that consists of a set of tokens based on their category features (POS tag features). More specifically, the created rule extracts propositional phrases from the text in the corpus such as an 'IN' (preposition) token followed by a 'JJ' (adjective) token (e.g. as minimum), an 'IN' (preposition) token followed by a 'JJS' (superlative adjective) token (e.g. at least), or an 'IN' (preposition) token followed by another 'IN' (preposition) token and a 'JJS' (superlative adjective) token (e.g. by at least).

As mentioned before, the ANNIE NE transducer is applied on the corpus of objectives text and generates the 'Date' annotation which matches any textual pattern that represents a temporal expression (e.g. '2008', '9/8/2009', 'July 2009', 'June of next year', 'October next year', 'coming year', 'coming two years', 'next year', 'next two years', 'second quarter of fiscal year 2009' etc). However, there are some textual patterns of temporal expressions (e.g. 'following year', 'upcoming year', 'following two years', 'upcoming two years') which also appear in the objective sentences and indicate dates but have not been extracted by the 'Date' annotation. Thus, a JAPE grammar rule has been created to extract these patterns from the objectives.

Figure 6 presents the created grammar for matching some of the textual patterns of temporal expressions (e.g. ‘following year’, ‘upcoming year’, ‘following two years’, ‘upcoming two years’) that represent dates. The grammar matches a string token in the corpus text followed by one or two ‘Lookup’ annotations and annotates them with a ‘Date’ annotation, where the matched sequence of text should indicate a date.



```
Phase: Date1
Input: Token Lookup
Options: control = appelt

Rule: DateRule
(
 ({Token.string == "following"})
 ({Lookup.majorType == date_unit})|
 ({Token.string == "upcoming"})
 ({Lookup.majorType == date_unit})|
 ({Token.string == "following"})
 ({Lookup.majorType == number})
 ({Lookup.majorType == date_unit})|
 ({Token.string == "upcoming"})
 ({Lookup.majorType == number})
 ({Lookup.majorType == date_unit})
)
:da
-->
 :da.Date = {rule= "DateRule"}
```

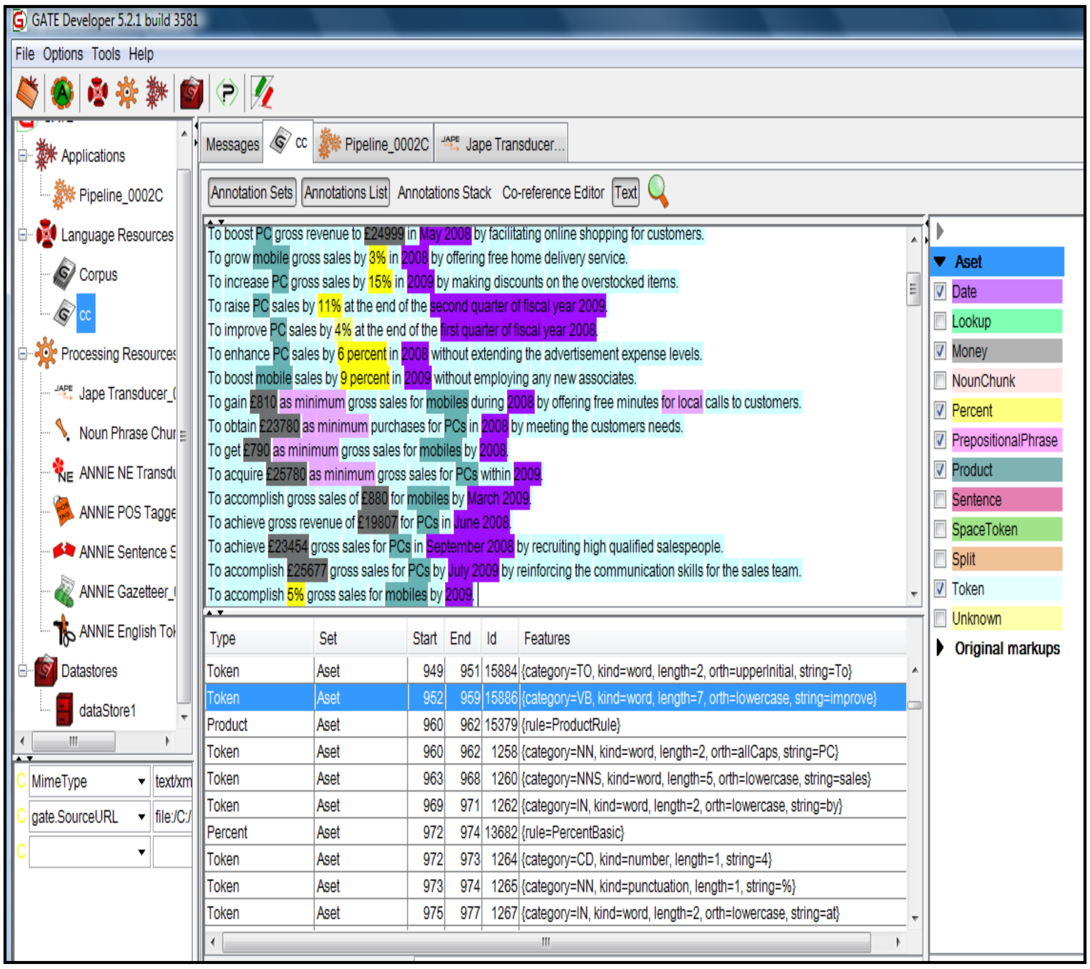
Figure 6 The Created JAPE Grammar for Matching Some Patterns of Temporal Expressions in the Objectives

As illustrated in the above figure, the rule called ‘DateRule’ is created to match a ‘Token’ annotation that covers the string ‘following’ followed by a ‘Lookup’ annotation with a ‘majorType’ feature of ‘date_unit’, a ‘Token’ annotation that matches the string ‘upcoming’ followed by a ‘Lookup’ annotation with a ‘majorType’ feature of ‘date_unit’, a ‘Token’ annotation that covers the string ‘following’ followed by a ‘Lookup’ annotation with a ‘majorType’ feature of ‘number’ and a ‘Lookup’ annotation with a ‘majorType’ feature of ‘date_unit’, or a ‘Token’ annotation that covers the string ‘upcoming’ followed by a ‘Lookup’ annotation with a ‘majorType’ feature of ‘number’ and a ‘Lookup’ annotation with a ‘majorType’ feature of ‘date_unit’.

4.2.1.3 The Annotations Produced by GATE

By applying ANNIE processing resources described earlier and the NP Chunker plugin on the corpus of objectives, GATE is able to generate a set of annotations which are grouped into an annotation set called ‘Aset’. After running the pipeline of the plugged processing resources and the NP Chunker plugin over the corpus of objectives, different annotations have been generalised, e.g. ‘Token’ annotation, ‘SpaceToken’ annotation, ‘Sentence’ annotation, ‘Split’ annotation, ‘Lookup’ annotation, ‘Percent’ annotation, ‘Money’ annotation, ‘Date’ annotation, ‘Product’ annotation, ‘PrepositionalPhrase’ annotation and ‘NounChunk’ annotation.

Figure 7 shows a snapshot which presents some typical annotations extracted from the corpus using the ANNIE processing resources and the NP Chunker plugin in GATE. The entities that match the generated annotations are highlighted in the text.



To boost PC gross revenue to £24999 in May 2008 by facilitating online shopping for customers.
To grow mobile gross sales by 3% in 2008 by offering free home delivery service.
To increase PC gross sales by 15% in 2009 by making discounts on the overstocked items.
To raise PC sales by 11% at the end of the second quarter of fiscal year 2009.
To improve PC sales by 4% at the end of the first quarter of fiscal year 2008.
To enhance PC sales by 6 percent in 2008 without extending the advertisement expense levels.
To boost mobile sales by 9 percent in 2009 without employing any new associates.
To gain £810 as minimum gross sales for mobiles during 2008 by offering free minutes for local calls to customers.
To obtain £23780 as minimum purchases for PCs in 2008 by meeting the customers needs.
To get £790 as minimum gross sales for mobiles by 2008.
To acquire £25780 as minimum gross sales for PCs within 2009.
To accomplish gross sales of £880 for mobiles by March 2009.
To achieve gross revenue of £19807 for PCs in June 2008.
To achieve £23454 gross sales for PCs in September 2008 by recruiting high qualified salespeople.
To accomplish £25677 gross sales for PCs by July 2009 by reinforcing the communication skills for the sales team.
To accomplish 5% gross sales for mobiles by 2009.

Type	Set	Start	End	Id	Features
Token	Aset	949	951	15884	{category=TO, kind=word, length=2, orth=upperInitial, string=To}
Token	Aset	952	959	15886	{category=VB, kind=word, length=7, orth=lowercase, string=improve}
Product	Aset	960	962	15379	{rule=ProductRule}
Token	Aset	960	962	1258	{category=NN, kind=word, length=2, orth=allCaps, string=PC}
Token	Aset	963	968	1260	{category=NNS, kind=word, length=5, orth=lowercase, string=sales}
Token	Aset	969	971	1262	{category=IN, kind=word, length=2, orth=lowercase, string=by}
Percent	Aset	972	974	13682	{rule=PercentBasic}
Token	Aset	972	973	1264	{category=CD, kind=number, length=1, string=4}
Token	Aset	973	974	1265	{category=NN, kind=punctuation, length=1, string=%}
Token	Aset	975	977	1267	{category=IN, kind=word, length=2, orth=lowercase, string=at}

Figure 7 Typical Annotations of the Corpus of Objectives Using GATE

By reviewing the above figure of the extracted annotations, GATE processed the objectives text and discovered useful annotations which identify numeric measures and dates in the objectives as well as specify the POS tags of words in the sentences and some other important textual patterns.

The extracted information from the corpus text by GATE (e.g. POS tags of words, NE annotations, and the other semantic tags) is one of the elements utilised by the applied machine learning technique in this study to learn a grammar for formulating SMART objectives and assessing whether the objectives are “specific”, “measurable”, and “time-related”. The next section illustrates how the corpus of objective sentences has been semantically analysed.

4.2.2 Using WordNet for Semantic Tagging

As illustrated in Chapter 3, the analysis of the “specific” criterion in the SMART approach has shown that there are some semantic features that must be present in an objective in order to be “specific”. This section demonstrates the use of the WordNet lexicon to add extra semantic information to the corpus of objectives and build a semantic classification for these objectives. This semantic classification will then help in ensuring that an objective is “specific” by classifying the kinds of words that are utilised to formulate well-written SMART objectives and to write focused, action-oriented, and well-defined objectives.

Section 4.2.2.1 describes the main WordNet semantic classes that are used for the semantic tagging of the target words in the corpus of objectives. Section 4.2.2.2 illustrates the use of the WordNet::SenseRelate::AllWords system for disambiguating the senses of all words in the given objectives by using semantic information from WordNet. Section 4.2.2.3 demonstrates how the WordNet::QueryData module is employed to provide a semantic class tag for each target word in the objective sentences based on WordNet in order to identify the appropriate words that are used commonly in writing SMART objectives.

4.2.2.1 Using Semantic Classes in WordNet

As described before in the literature review, the task of classifying words semantically based on their semantic classes supports NLP. Primarily, it extends the task of NER and at the same time, it provides a partial disambiguation step for WSD. A knowledge resource like the WordNet lexicon has been extensively utilised for retrieving useful information such as meanings of words, lexical relations between words or word forms and semantic relations between word meanings. The studies highlighted in the literature survey depict the importance of using the WordNet lexicographers (e.g. body, communication, possession, animal, change etc) in automatic semantic classification of words in text by assigning words in text into WordNet semantic classes. Consequently, lexicographers in WordNet hold the potential to be useful in this study for the semantic analysis of objectives in order to semantically classify the target words that are commonly used in creating SMART objectives by identifying an appropriate semantic class for each of these target words. This will support the applied machine learning method in learning the rules that ensure that the objectives are “specific”.

Another reason which also encourages performing a semantic classification of the corpus in this study before applying a machine learning method for learning the grammar rules, is the study addressed by Bouillon et al. (2001) which utilises ILP to extract the related noun-verb pairs from the MATRA-corpus. Their study shows that the semantic tagging (classification) of the corpus improves the quality of the learning for the rules generalised by the ILP method. In particular, they have compared the use of an ILP method (i.e. PROGOL) on a POS tagged version of the MATRA-corpus and on a semantically tagged version of the same corpus. The experimental results show that the rules learned by the ILP method from the semantically tagged corpus are better than those learned from the POS tagged one. They argue that the semantic tagging has increased the level of generalisation for the learned rules and illustrated some semantic properties of the words surrounded a semantically related noun-verb pair in the corpus. In contrast, the learning applied on the POS tagged corpus using the ILP method has led to poorer contextual information and less generalised rules containing more linguistic features to extract the relevant noun-verb pairs.

The following paragraphs describe how the WordNet lexicographer class labels (e.g. possession, change, social etc) are used for determining the semantic class, to which each target word in the objectives belongs.

In this research, the text of the objective sentences presented in the developed corpus has been syntactically and semantically analysed. Results showed that there are certain categories of words used in writing SMART objectives. With regards to the definition of the SMART approach, an objective is considered “specific” if it is well-defined and specifies clearly what is to be achieved (Hurd et al., 2008). The definition of the “specific” factor in the SMART approach also implies that the objective should describe the action taken for achieving it by using an action verb (e.g. increase, achieve, boost, reduce etc) (Rouillard, 2003). Furthermore, the objective should state what it applies to, by specifying the domain of application in the objective (e.g. the objective pertains to sales, costs, profits etc) (Carliner, 1998). Consider the following example of a SMART objective which appears in the created corpus:

“By the end of next year, PC sales will increase by 8%.”

The word “increase” in the above objective sentence is an action verb that describes the action taken for achieving this objective, whilst the word “sales” represents the domain of application for the above objective. These two features (action verb and the domain of application for the objective) are required in an objective sentence in order to be “specific”. The POS tagger in GATE provides the grammatical categories for each word in the above sentence and tags the word “increase” as a verb and the word “sales” as a noun.

Based on WordNet, a semantic classification has been built for the semantic tagging of the corpus of objectives, where the most generic lexicographer semantic classes have been chosen to semantically classify the target words (action verbs, common nouns that represent the objectives’ domain) which are used commonly to write SMART objectives. Therefore, a tagset has been defined for the semantic classification of the main POS categories (nouns and verbs) of the target words in the given corpus of objectives. In particular, all possible WordNet semantic class labels have been assigned to the target nouns and verbs in the objective sentences, where irrelevant semantic categories for the given corpus have been discarded. As a result, 5 verb semantic classes have been chosen to semantically categorise the target verbs that appear in the given objective sentences, while only 1 noun semantic class has been selected to classify the target common nouns in the objectives. Figure 8 presents the defined tagset of the most generic WordNet semantic classes for the target common nouns and verbs which occur in corpus of objective sentences, since this tagset will be used later for the semantic tagging of the target words in this corpus.

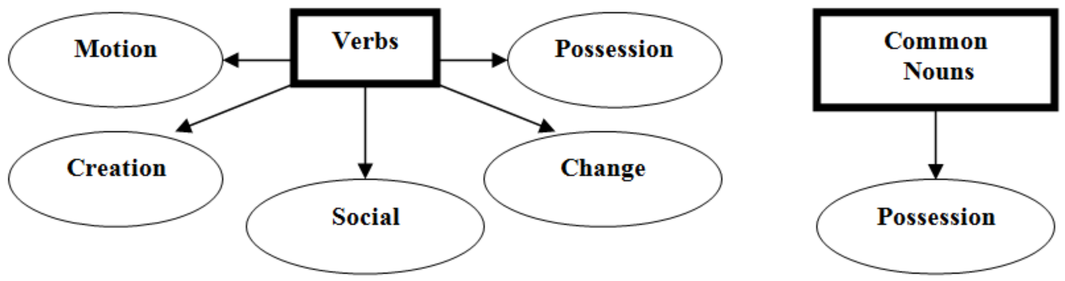


Figure 8 The Main WordNet Semantic Classes of the Target Nouns and Verbs in the Objectives

To automatically identify the semantic classes of the target words in the objective sentences, automatic methods in NLP could be used to access the WordNet database and then retrieve the most appropriate semantic class for each target word in a given context.

The idea is then to compare the defined tagset of the semantic classes for the target nouns and verbs in the objective sentences with the results obtained by applied automatic NLP methods to check the performance of these methods in disambiguating the target words in the objectives semantically.

The following two sections describe two open source systems which have been applied to access the WordNet database for retrieving the required semantic information for the target words in the objectives. The first system is used to identify the senses of all words in the objectives based on the context in which they occur. Since the first system does not provide the semantic classes of the words in text, a second system has been applied to perform this task based on WordNet and annotate the target words in the objectives with the defined tagset of the semantic classes of verbs and nouns by using the disambiguation results obtained from the first system.

4.2.2.2 Using WordNet::SenseRelate::AllWords for WSD

As described before, the semantic analysis of the objectives will support the applied machine learning technique in ensuring that an objective is "specific" by identifying the target words that are commonly used in structuring SMART objectives and in formulating specific and well-defined objectives.

To perform the semantic analysis of the objectives, the open-source software called WordNet::SenseRelate::AllWords (SR-AW) (Pedersen and Kolhatkar, 2009) is applied on the already POS-tagged corpus of objective sentences to automatically disambiguate the senses of all words in text based on the context in which they appear. The following sections illustrate how the SR-AW disambiguating algorithm is used to find the most accurate sense for each word in a given context based on WordNet as well as present the experimental evaluation of this algorithm on the corpus of objectives.

4.2.2.2.1 The Methodology of Applying WordNet::SenseRelate::AllWords on the Corpus

As mentioned in Section 4.2.1, the text of the objective sentences in the developed corpus has been POS tagged by the GATE POS tagger (i.e. Hepple POS tagger). This POS tagged text is then provided as an input to the SR-AW disambiguating system for WSD by determining the senses of all words in objectives using semantic information available in WordNet. One of the reasons for applying the SR-AW disambiguation algorithm on a POS tagged text is that several studies have shown that this algorithm provides more accurate results when disambiguating tagged text comparing to the raw (untagged) one.

Before starting the disambiguation process using the SR-AW system, there are different choices including some disambiguation options have been specified in the system interface. For example, the format of the input text of the objective sentences is set to ‘tagged’, the option of fixing the senses of words in the sentences is set to ‘normal’, and the selected stoplist is the default one.

The software starts processing the entered text sentence by sentence. Then, the specified POS tags (e.g., VB, NNS, IN, JJ etc) in the provided objective sentences are mapped to the main WordNet POS tags (e.g. verbs ‘*v*’, nouns ‘*n*’, adjectives ‘*a*’, adverbs ‘*r*’) in order to be processed by the SR-AW software which mainly depends on the WordNet lexical database. During the processing step, the software applies a morphological analysis, which is provided by WordNet, in order to determine the root form of a word in the input text. At the end, the SR-AW system assigns each word in the objective sentences with its possible meaning in WordNet such that the output of the system is the text of the objective sentences with WordNet sense tags assigned. The following paragraphs present some examples of (SMART) objective sentences and the output of analysing and processing these sentences using the SR-AW disambiguating algorithm.

As described before, the disambiguation process of the objective sentences is performed on a POS tagged text. Therefore, by using the GATE POS tagger to annotate an objective sentence such as "*To increase PC gross-sales by 15% in 2009 by making discounts on the overstocked items*.", the result of tagging this sentence with the POS tags is as follows:

To/TO increase/VB PC/NN gross-sales/NNS by/IN 15/CD %/NN in/IN 2009/CD by/IN making/VBG discounts/NNS on/IN the/DT overstocked/JJ items/NNS./.

Here VB means a verb in the base form, NN means a singular noun, NNS means a plural noun, IN means a preposition, CD means a cardinal number, JJ indicates an adjective, DT indicates a determiner, and VBG means a gerund verb. The disambiguation operation using the SR-AW algorithm gives the following result for the above POS-tagged sentence:

To increase#v#2 PC#n#1 gross_sales#n#1 by#r#2 15#a#1 in#r 2009#a by#r#2 make#v#3 discount#n#1 on#r#3 the overstocked#a item#n#1 .

The above disambiguation result shows that the algorithm has assigned a sense for each word in the given objective sentence. Some words in this objective sentence such as "*To*" and "*the*" are ignored in the disambiguation process, since these kinds of words (e.g. determiners, pronouns, conjunctions etc) are not in the WordNet database.

With regards to the output of the meanings defined by the algorithm for the words in the above objective sentence, the meaning of the target word "*increase*" is defined correctly:

increase#v#2 : make bigger or more; "The boss finally increased her salary"; "The university increased the number of students it admitted".

Here *v* means a verb and *2* means the second sense of the word. This means that the verb "*increase*" has been assigned with the second sense of the word in the WordNet lexical database. Basically, the numbers of senses locate nodes (meanings) of words on the semantic tree hierarchies in WordNet. As shown in the above meaning of the word "increase", the selected definition (gloss) of the word is expressed with some example sentences (in quotation marks) for illustrating the meaning of the disambiguated word.

The meaning of the target word "*gross-sales*" is also chosen precisely:

gross_sales#n#1 : income (at invoice values) received for goods and services over some given period of time.

Here *n* means a noun. The word "*gross-sales*", which is a compound word, has been assigned with the first sense of the noun "*gross-sales*" in WordNet.

The output of the disambiguation operation of the above objective sentence showed that not only the target verb (e.g. "*increase*") and noun (e.g. "*gross-sales*") have been disambiguated correctly, but also all the other words in the given objective sentence have been assigned with their correct WordNet senses.

The following presents another objective sentence from those sentences which are provided to the SR-AW algorithm to be processed and disambiguated.

"*To maximise PC sales by 5% for the upcoming year.*"

The GATE POS tagger has tagged the text in the above objective sentence as follows:

To/TO maximise/VB PC/NN sales/NNS by/IN 5/CD %/NN for/IN the/DT upcoming/VBG year/NN ./.

The result of the disambiguation analysis for the above POS-tagged sentence is as follows:

"To maximise#v#1 PC#n#1 sales#n#1 by#r#2 5#a#1 for#r the upcoming#v year#n#3 ."

The disambiguation algorithm has assigned a meaning for each word in the above objective sentence. The selected meanings of the target words "*maximise*" and "*sales*" are correct:

maximise#v#1 : make as big or large as possible; "Maximize your profits!" .

sales#n#1 : income (at invoice values) received for goods and services over some given period of time.

The target word "*maximise*" has been assigned with the first sense of the verb "*maximise*" in WordNet. Furthermore, the target word "*sales*" has been assigned with the first sense of the noun "*sales*" in WordNet.

For an objective sentence such as "*To attain at least £888 sales for mobiles by June of next year.*", the POS-tagged text is as follows:

To/TO attain/VB at/IN least/JJS #/# 888/CD sales/NNS for/IN mobiles/NNS by/IN June/NNP of/IN next/JJ year/NN ./.

The disambiguation gives:

To attain#v#1 at#r least#a#1 888#a sale#n#3 for#r mobile#n#2 by#r#2 June#n#1 of#r next#a#1 year#n#1 .

The meaning of target word "*attain*" is disambiguated precisely:

attain#v#1 : to gain with effort; "she achieved her goal despite setbacks".

The target word "*attain*" has been assigned with the first sense of the verb "*attain*" in WordNet. On the other hand, the selected meaning of the target word "*sales*" is not correct.

sales#n#3 : an occasion (usually brief) for buying at specially reduced prices; "they held a sale to reduce their inventory"; "I got some great bargains at their annual sale" .

Here, the word "*sales*" has been assigned with the third sense of the noun "*sales*" in WordNet. The retrieved meaning of the word "*sales*" here indicates a special event for purchasing at reduced prices. By reviewing the above objective sentence, it is clear that the retrieved meaning of the word "*sales*" by the algorithm does not reflect the correct meaning of this word in term of the context in which it appears, since the word "*sales*" in the objective sentence indicates the income gained by a company for products or services during a time period.

4.2.2.2.2 Experiments of WordNet::SenseRelate::AllWords

The goal of the performed experiments was to evaluate and assess the accuracy of the SR-AW algorithm by using a sample data of the corpus of objective sentences as an experimental data. Thus, a sample data of 130 objective sentences has been chosen from the created corpus to evaluate the accuracy of the disambiguation algorithm. The target words in these objective sentences are used in evaluating the performance of the applied algorithm. Each occurrence of a target word in the given objective sentences has been manually annotated by a human judge with the most appropriate WordNet sense for that context. The senses in the most recent version 2.1 of WordNet are utilised in annotating the utilised experimental data. All of the target words are known to WordNet as the applied algorithm only disambiguates the words that are defined in WordNet.

In the utilised data sample of the objective sentences, there are 29 target words with a total of 278 test instances (word occurrences), where these instances are divided among 25 different verbs (130 verb occurrences) and 4 different nouns (148 noun occurrences), where the part of speech of the target words is known to the disambiguating system. These target words are listed in Section 4.2.2.2.3. Each instance of a target word to be disambiguated consists of the sentence in which it occurs. The occurrences of a target word in the window of context are treated separately.

For disambiguating the POS-tagged text in the data sample, the SR-AW algorithm is given this text with the same instances of target words (without the human assigned senses) and the disambiguation output of this algorithm is what it considers to be the most appropriate senses for each of the target words in the provided objective sentences.

The extended '*lesk*' measure (extended gloss overlaps) (Banerjee and Pedersen 2003) is selected as a semantic relatedness measure for the experiments, where it estimates the semantic relatedness between a target word and its neighbours in an objective sentence. More specifically, this measure is used here to compare the sense definitions (glosses) of a target word in an objective sentence with the sense definitions of the other words in the same objective sentence (surrounding words in context). The sense definition with the highest number of words in common (overlaps) is considered to be the correct sense of that target word. This measure does not only compare the definitions of synsets but also the definitions of concepts (relations between synsets "e.g. hypernym, hyphonym, holonym etc.") that are related to that synset according to the WordNet concept hierarchies. As the SR-AW system includes several measures for determining the semantic relatedness between words (e.g. the adapted '*lesk*' measure (Banerjee and Pedersen 2003), context vectors (Patwardhan, 2003) and the '*hso*' (Hirst & St-Onge) measure (Hirst and St-Onge, 1998)), this study utilises the '*lesk*' measure specifically because the experiments conducted by some authors (e.g. Michelizzi, 2005; Pedersen and Kolhatkar, 2009) have shown that this measure outperforms the other semantic relatedness measures utilised in SR-AW.

In addition to the use of the extended '*lesk*' measure, the experiments also utilise a context window of size 8. The performed experiments showed that the increase in the size of the context window provides more data to the algorithm and improves the results of disambiguating the target words. The experiments were carried out with version 0.19 of the SR-AW disambiguation algorithm.

4.2.2.2.3 Experimental Results of WordNet::SenseRelate::AllWords

The accuracy of the SR-AW algorithm is estimated by comparing the results of the senses of target word occurrences that are produced by the algorithm with the human assigned senses. Table 8 presents the target words that appear in the utilised data sample, where these target words are broken down according to their word categories (verbs/nouns). This table also displays the number of test instances for each target word and reports the accuracy attained by the algorithm for disambiguating the senses for each target word. The accuracy is the number of test instances that have been correctly disambiguated with the correct WordNet senses divided by the total number of word occurrences for a word.

Nouns	**Correct Senses**	**Wrong Senses**	**Total Instances**	**Accuracy of each Word**
sales	34	28	62	0.55
purchases	2	5	7	0.29
gross-sales	55		55	1
gross-revenue	24		24	1
Verbs	**Correct Senses**	**Wrong Senses**	**Total Instances**	**Accuracy of each Word**
increase	12		12	1
augment	6	1	7	0.86
escalate	4		4	1
maximise	8	1	9	0.89
develop	2	2	4	0.50
boost	7	2	9	0.78
grow	7	1	8	0.87
raise		3	3	1
improve	6	2	8	0.75
enhance	8	1	9	0.89
gain	2	3	5	0.40
obtain	2	1	3	0.67
inherit	3		3	1
acquire	3		3	1
attain	1	2	3	0.33
achieve	9		9	1
accomplish	5	1	6	0.83
get	1	2	3	0.33
rise		3	3	1
expand	2		2	1
magnify	2	1	3	0.67
enlarge	1	1	2	0.50
inflate	1	2	3	0.33
generate	5		5	1
support	3	1	4	0.75

Table 8 Target Words, their Test Instances and the Accuracy of Disambiguating the Senses for Each Target Word

The accuracy achieved by the SR-AW disambiguation algorithm for each POS category (nouns/verbs) of the target words that appear in the given data sample as well as the overall accuracy achieved by the algorithm for WSD of all target words are presented in Table 9.

Correct Occurrences of Nouns	Total Noun Occurrences	Accuracy of Nouns
115	148	0.78
Correct Occurrences of Verbs	**Total Verb Occurrences**	**Accuracy of Verbs**
100	130	0.77
Correct Occurrences of All Target Words	**Total Occurrences of Target Words**	**Overall Accuracy of Target Words**
215	278	0.77

Table 9 WSD Evaluation Results

As shown in the above table, the SR-AW algorithm attains an overall accuracy of 77%, where 215 of 278 word occurrences are disambiguated precisely with the correct WordNet senses.

The idea is then to use all the contextual information and the disambiguation results produced by the applied SR-AW algorithm to determine the semantic classes of the target words in the objective sentences. This idea is explored in more detail in the following section.

4.2.2.3 Using WordNet::QueryData for Semantic Tagging

Section 4.2.2.2 demonstrated the task of disambiguating the senses of all words in the corpus of objective sentences. This section provides first a brief description of the WordNet::QueryData package including some illustrative examples. Then, it describes how the WordNet::QueryData module is exploited for the semantic tagging of target words in the corpus of objective sentences as well as presents the experimental evaluation results of applying WordNet::QueryData on this corpus.

4.2.2.3.1 Description of WordNet::QueryData

As described before in the literature review, the Perl interface called WordNet::QueryData (Rennie, 2000) is a well-known module for accessing the WordNet database files in order to retrieve useful information such as semantic (e.g. hypernyms, hyponyms, holonyms, meronyms**,** domain categories etc) and lexical (e.g. antonym, participle of verb, verb group etc) relations. It simply uses functions to access the WordNet database, such as 'querySense', 'queryWord', 'lexname' and some other functions. The 'querySense' function returns the semantic relations between senses, while the 'queryWord' function retrieves the lexical relations between words. The 'lexname' function returns the WordNet ‘lexname’ of a sense, where ‘lexname’ indicates the WordNet lexicographer file name (i.e. WordNet lexicographer semantic class) for each syntactic category (nouns, verbs, adjectives and adverbs) of a synset (e.g. noun.animal, noun.body, noun.possession, verb.emotion, verb.change, verb.creation etc). For example, the verb 'hate' is the synonym of the verb 'detest', where both of them share the same synset in WordNet which has the lexicographer semantic class 'verb.emotion'. Consequently, the WordNet 'lexname' for each of these two verbs ('hate' and 'detest') is 'verb.emotion', since both of them indicate feelings and emotions. The 'queryWord' and 'querySense' functions take one argument which is a query string if they have not been called to retrieve a relation. On the other hand, if they are called to return a relation between words or synset words; they require a second argument that specifies the relation name. For instance, consider the following Perl script example, where WordNet::QueryData is used to access the WordNet database:

Example 2:

```
“use WordNet::QueryData;
 my $wn = WordNet::QueryData->new;
print "Parts of Speech: ", join(", ", $wn->querySense("book")), "\n";”
```

In the third line of the above Perl script, the 'querySense' function is used to retrieve all POS categories for the word *“book”* from WordNet. Here the 'querySense' function takes one argument which is the query string *“book”*. If the above Perl script of *Example 2* is run on a Perl editor, the output would be as follows:

```
Parts of Speech: book#n, book#v
```

Here, the program prints the retrieved POS categories (noun '*n*', verb '*v*') for the word 'book'.

Suppose that the last line of the above Perl script of *Example 2* is *“print "Senses: ", join(", ", $wn->querySense("book#n")), "\n";”* the 'querySense' function will retrieve different result. Here the 'querySense' function also takes one argument which is the query string *"book"*, such that the POS category for this string is known and it is a noun ('*n*'). By executing this Perl script, the program gives the following output:

Senses: book#n#1, book#n#2, book#n#3, book#n#4, book#n#5, book#n#6, book#n#7, book#n#8, book#n#9, book#n#10, book#n#11.

As shown in the above output, the 'querySense' function returns a list of all senses for the noun *"book"*. Each number presented in the above output indicates the sense number for the noun *"book"* in the WordNet lexicon.

As another example, if the last line of the above Perl script of *Example 2* is *"print "Hyponyms: ", join(", ", $wn->querySense("cat#n#1", "hypo")), "\n";"*, the 'querySense' function will return a semantic relation for the specified synset (*"cat#n#1"*). Here the 'querySense' function takes two arguments (i.e. *"cat#n#1"* and *"hypo"*), where the first argument (*"cat#n#1"*) specifies a synset and includes the query string (*"cat"*), the word category (noun '*n*'), and the sense number of the word ('*1*'). The second argument identifies the relation name (*"hypo"*) to be called by the function. The *"hypo"* relation is a semantic relation that represents the 'is-a' relationship (e.g. "X is a hyponym of Y" indicates that X is a kind of Y) and it is used to return the hyponyms of a synset. By executing the modified Perl script, the program will provide the following result:

Hyponyms: domestic_cat#n#1, wildcat#n#3

The domestic cat and wildcat above are the retrieved hyponyms by the 'querySense' function for the specified synset of the word *"cat"* (i.e. domestic cat and wildcat are some types of cats).

4.2.2.3.2 Applying WordNet::QueryData for the Semantic Tagging of the Corpus

In this research, the WordNet::QueryData module is applied to determine the semantic classes of the target words that are commonly used to write SMART objectives. This demonstrates a task for partial WSD, in which target words in the given objective sentences are labelled using the lexicographer semantic classes available in WordNet. The following paragraphs illustrate in more detail how the WordNet::QueryData module is used in this research.

As explained earlier in the previous section (Section 4.2.2.2), the SR-AW algorithm has been applied to automatically find the senses of all words in objectives from WordNet based on the context in which they occur. However, this algorithm does not specify the relations between words or synsets as well as is not able to find the semantic categories of synsets. Since this study requires performing a useful text processing for the objectives by adding more general semantic information and disambiguating the words in objective sentences at the semantic class level, it is a necessity to use an approach for automatic semantic class classification of words in text.

Thus, the methodology applied here is based on using the disambiguation output produced by the SR-AW algorithm which includes all the words in the given text of objectives with the senses assigned. To be specific, the result of disambiguating the sense of each target word in the objectives by the SR-AW algorithm is provided to the WordNet::QueryData module for determining a semantic relation of domain category (class) for a specified synset by using information from WordNet.

In the previous section, the SR-AW algorithm has processed and disambiguated the objective sentence "*To increase PC gross-sales by 15% in 2009 by making discounts on the overstocked items*.", where the algorithm has retrieved the correct senses for the target words "*increase*" and "*gross-sales*" in this objective sentence (e.g. *increase#v#2* and *gross_sales#n#1*).

To be able to retrieve a semantic class for each target word (e.g. "*increase*", "*gross-sales*" etc) in the given objective sentences, the 'lexname' function in WordNet::QueryData is applied to access the WordNet database and return a semantic category (WordNet lexicographer semantic class) for a specified synset (e.g. *increase#v#2, gross_sales#n#1* etc).

Therefore, a Perl script is written for each occurrence of a target word in the given objective sentences based on the disambiguating result produced by the SR-AW algorithm for that target word occurrence, where the word category and the sense number of each target word occurrence is specified in the scripts. The following represents the created Perl script for the occurrence of the target word "*increase*" which appears in the above objective sentence:

Example 3:

```
"use WordNet::QueryData;
 my $wn = WordNet::QueryData->new;
print "Category: ", join(", ", $wn->lexname("increase#v#2", "dmnc")), "\n";
```

As it is shown in the above Perl script, the 'lexname' function takes two arguments (*"increase#v#2"* and *"dmnc"*), where the first argument (*"increase#v#2"*) states the retrieved meaning of the occurrence of the target word "*increase*" in the above objective by the SR-AW disambiguation algorithm. This argument includes the word "*increase*", the specified POS tag of the word "*increase*", which is a verb ('*v*'), and the specified sense number of the word "*increase*", which is '*2*'. The second argument states the relation name which is *"dmnc".* This semantic relation (*"dmnc"*) indicates the domain category of a synset (e.g. verb.change, verb.possession, verb.creation etc). The above Perl script of *Example 3* is run on a Perl editor and provided the following output:

```
Category: verb.change
```

Here the program prints the detected lexicographer semantic class of the target word "*increase*" which is "*verb.change*". The 'lexname' function in WordNet::QueryData has exploited the *"dmnc"* semantic relation to retrieve the lexicographer semantic class for the specified synset (*"increase#v#2"*) from WordNet. The WordNet lexicographer semantic category of the target word "*increase*" in the given sentence is specified correctly.

To determine the semantic class of the occurrence of the target word "*gross-sales*" in the above sentence, the last line of the above Perl script is set to *"print "Category: ",* *join(", ", $wn->lexname("gross_sales"#n#1", "dmnc")), "\n";"*. By executing the modified Perl script, the output is as follows:

Category: noun.possession

Here WordNet::QueryData has used the 'lexname' function and the semantic relation *"dmnc"* to return the lexicographer semantic category of the target word "*gross-sales"* from WordNet. Thus, the retrieved lexicographer semantic category of the specified synset (*"gross_sales"#n#1"*) is "*noun.possession*". The target word *"gross-sales"* in the given sentence has been disambiguated with the correct WordNet semantic class.

The objective sentence "*To attain at least £888 sales for mobiles by June of next year.*" has been processed and disambiguated by the SR-AW algorithm in the previous section. The disambiguation result showed that the target word *"attain"* has been assigned with the right sense ("*attain#v#1*"), while the target word *"sales"* has been disambiguated incorrectly ("*sale#n#3*") in this objective sentence.

The created Perl script for the occurrence of the target word *"attain"* which occurs in the above objective sentence is as follows:

Example 4:

```
"use WordNet::QueryData;
 my $wn = WordNet::QueryData->new;
print "Category: ", join(", ", $wn->lexname("attain#v#1", "dmnc")), "\n";
```

In the above Perl script, the 'lexname' function, which takes two arguments ("*attain#v#1"* and *"dmnc"*), is employed to access WordNet and retrieve the lexicographer semantic category of the synset *"attain#v#1"* by exploiting the semantic relation *"dmnc"*. The output of executing the above Perl script is:

```
Category: verb.social
```

Here the program prints the retrieved lexicographer semantic category of the target word "*attain*" which is "*verb.social*". The target word "*attain*" in the above sentence has been assigned with the correct WordNet semantic class.

To specify the semantic class of the occurrence of the target word "*sales*" that appears in the above objective sentence, the last line of the above Perl script of *Example 4* is set to *"print "Category: ", join(", ", $wn->lexname("sales"#n#3", "dmnc")), "\n";"*. Then, the modified Perl script is run on a Perl editor and produces the following output:

Category: noun.act

Here the program prints the detected lexicographer semantic class of the target word "*sales*" which is "*noun.act*". The retrieved semantic class of the target word "*sales*" (synset: *"sale#n#3"*) is incorrect in comparison to the context in which it occurs, since the word "*sales*" in the above sentence denotes possession and not acts or actions.

4.2.2.3.3 Experiments of WordNet::QueryData

In the experiments, the sample data of 130 objective sentences which has been utilised to evaluate the SR-AW algorithm is exploited again to evaluate the accuracy of WordNet::QueryData. Each occurrence of a target word in the given data sample has been manually annotated by a human judge with the most correspondent WordNet 2.1 semantic class for that context. The previously defined tagset of the semantic classes is utilised in the manual annotation of the target words in the data sample, where the noun semantic class called 'possession' is used to annotate the target nouns, while the verb semantic classes such as 'change', 'social', 'creation', 'motion', and 'possession' are used to tag the target verbs in this data sample. Similarly, the same set of target words is used again in evaluating WordNet::QueryData (i.e. 29 target words with a total of 278 word occurrences). The experiments were performed with the current version 1.49 of WordNet::QueryData. In this study, the WordNet version 2.1 and the Perl version 5.14.2 are installed to run the WordNet::QueryData Perl module. Furthermore, the Padre (Perl Application Development and Refactoring Environment) (version 0.94) text editor is used as an IDE to run and debug the Perl scripts.

4.2.2.3.4 Experimental Results of WordNet::QueryData

The semantic classes of the occurrences of target words that are specified by the WordNet::QueryData module are compared with the human assigned semantic classes in order to compute the accuracy of WordNet::QueryData. Table 10 also breaks down the target words according to their POS category and demonstrates the occurrences of target words as well as the accuracy obtained by the applied module for disambiguating the semantic classes of each target word. Here the accuracy is the number of occurrences of a target word that have been assigned with the correct WordNet semantic classes divided by the total number of occurrences for that target word.

Nouns	Correct Classes	Wrong Classes	Total Occurrences	Accuracy of each Word
sales	34	28	62	0.55
purchases	2	5	7	0.29
gross-sales	55		55	1
gross-revenue	24		24	1
Verbs	**Correct Classes**	**Wrong Classes**	**Total Occurrences**	**Accuracy of each Word**
increase	12		12	1
augment	7		7	1
escalate	4		4	1
maximise	9		9	1
develop	2	2	4	0.50
boost	7	2	9	0.78
grow	8		8	1
raise		3	3	1
improve	8		8	1
enhance	9		9	1
gain	3	2	5	0.60
obtain	2	1	3	0.67
inherit	3		3	1
acquire	3		3	1
attain	1	2	3	0.33
achieve	9		9	1
accomplish	5	1	6	0.83
get	1	2	3	0.33
rise		3	3	1
expand	2		2	1
magnify	3		3	1
enlarge	1	1	2	0.50
inflate	3		3	1
generate	5		5	1
support	3	1	4	0.75

Table 10 Target Words, their Occurrences and the Accuracy of Disambiguating the Semantic Classes for Each Target Word

Table 11 presents the accuracy achieved by WordNet::QueryData for each POS category (nouns/verbs) of the target words that appear in the given data sample of the objective sentences as well as the overall accuracy of determining the semantic classes for all target words in this data sample.

Correct Occurrences of Nouns	**Total Noun Occurrences**	**Accuracy of Nouns**
115	148	0.78
Correct Occurrences of Verbs	**Total Verb Occurrences**	**Accuracy of Verbs**
110	130	0.85
Correct Occurrences of All Target Words	**Total Occurrences of Target Words**	**Overall Accuracy of Target Words**
225	278	0.81

Table 11 Evaluation Results of Semantic Class Classification

As it is demonstrated in the above table, the WordNet::QueryData module has achieved an overall accuracy of 81%, where 225 of 278 word occurrences are disambiguated accurately with the correct WordNet semantic classes.

The results achieved by the WordNet::QueryData module are significant, since the majority of the semantic ambiguities and semantic tagging errors are solved by using this module. The use of semantic classes instead of senses allowed to disambiguate more correct word occurrences of target words and enhanced the accuracy of the semantic disambiguation (i.e. the overall accuracy achieved by applying SR-AW algorithm on a data sample of objectives for WSD is 77%, while the overall accuracy of classifying the target words in the same data sample based on their semantic classes using WordNet::QueryData is 81%). In particular, the task of disambiguating the words based on their semantic classes is a partial WSD step and is more general than the task of WSD. It identifies the domain categories of words (e.g. possession, emotion, body, animal, plant etc), since many words/synsets can share the same semantic class, while the task of WSD is based on assigning a specific meaning to each word in text. For instance, the verb "*maximise*" has only two senses in WordNet. This verb appears 9 times (9 occurrences) in the given data sample of objectives, since it has been disambiguated once with the wrong sense (*"maximise": make the most of; "He maximized his role"*), while with the other 8 instances it has been assigned with the correct sense (*"maximise": make as big or large as possible; "Maximize your profits!"*) by the SR-AW algorithm. By using the

WordNet::QueryData module, all occurrences of the verb "*maximise*" in the given data sample have been assigned with the correct verb semantic class, which is "*verb.change"*. This is because that the two senses of the verb "*maximise*" in WordNet are classified to the verb semantic class "*verb.change"*.

Based on the performed semantic analysis of the objectives, some important findings have been emerged and discovered. For example, the results of disambiguating the target words in the objectives semantically based on WordNet show that most of the action verbs, which are commonly used in writing SMART objectives (e.g. increase, maximise, grow, enhance, augment, escalate, develop, improve, enlarge, raise, rise, magnify, inflate, expand), have been classified into the verb semantic class called "change". However, there are also some other action verbs in the objectives, which also are used in structuring SMART objectives, have been classified into other verb semantic classes such as "social" (e.g. achieve, attain, accomplish), "possession" (e.g. gain, acquire, obtain, support, get, inherit), "creation" (e.g. generate) and "motion" (e.g. boost). Moreover, the common nouns, which are commonly used in writing SMART objectives to indicate the objective domain (e.g. sales, purchases, gross-sales, gross-revenue), have been classified into the noun semantic class called "possession".

Some examples of SMART objective sentences related to different domains (e.g. costs) have been examined semantically as well. Results show that the target words (action verbs, nouns that represent the objectives' domain) in these objectives are also classified into the same semantic classes as the target words in the developed corpus of objectives (which is related to the domain of sales). This point of view is illustrated in more detail in Chapter 6, where a corpus of objectives related to the costs domain has been constructed and then developed system in this research has been applied on this corpus in order to demonstrate the system ability to generalise and to be used in different domains.

This section has shown clearly how the SR-AW algorithm and WordNet::QueryData are employed to perform a semantic analysis of the objectives and to annotate the target words in these objectives with the semantic classes available in WordNet. This semantic analysis of the objectives has provided more useful semantic information which can be used by the applied machine learning technique for checking whether an objective is “specific”. In particular, it demonstrates which kinds of semantic classes (noun.possession [nouns denoting possession and transfer of possession], verb.change [verbs of size, temperature change, intensifying etc], verb.social [verbs of political and social activities and events], verb.creation [verbs of sewing, baking, painting, performing], verb.possession [verbs of buying, selling, owning] and verb.motion [verbs of walking, flying, swimming]) that must be used for choosing the relevant words to write effective SMART objectives.

The next chapter demonstrates how the performed syntactic and semantic analysis of the corpus of objective sentences will support a machine learning technique in learning a grammar for writing SMART objectives.

4.3 Summary

In this chapter, the methodology adopted for developing a corpus of objectives was presented. A corpus of objective sentences related to the sales domain was developed for evaluating the research. An illustration of the linguistic analysis performed on the developed corpus of objectives was given, where the GATE system was used for annotating the corpus, processing its text, and generating the POS tags of words in the sentences, NE annotations and some other semantic annotations. The use of the GATE system for the linguistic analysis of the corpus of objectives helps to specify some useful linguistic patterns which can be exploited by the applied machine learning method for checking whether the objectives are “specific”, “measurable”, and “time-related”. Furthermore, a detailed description of the semantic analysis performed on the corpus of objectives based on WordNet was given, where the SR-AW and WordNet::QueryData software have been employed to semantically annotate the target words in the developed corpus. Moreover, an experimental evaluation of the applied methods for the semantic analysis of the corpus was given and achieved good results in disambiguating the target words in the objective sentences semantically. The conducted semantic analysis of the corpus distinguishes the target words that are used commonly in writing SMART objectives by identifying the appropriate semantic classes for these target words.

In the next chapter, a detailed description of how ILP is used for learning a grammar for SMART objectives from the developed corpus will be presented. An explanation of how data mining techniques are used for assessing the objectives will be given. An example for illustrating the developed system will also be presented.

Chapter 5: The Development of the Framework Based on ILP and Data Mining Methods

The previous chapter described the development and the linguistic analysis of a corpus of objectives. This chapter builds upon Chapter 4, where it describes the other main components of the framework that are used to develop a system for assessing whether the written objectives are SMART based on the use of ILP and data mining techniques. The chapter is organised as follows:

- Section 5.1 describes the use of ILP to learn a grammar for writing SMART objectives.
- Section 5.2 illustrates the use of data mining techniques for assessing the objectives.
- Section 5.3 presents an example to illustrate the developed system.
- Section 5.4 presents a summary of this chapter.

5.1 Using ILP to Learn a Grammar for SMART Objectives

Chapter 4 described how the corpus of objective sentences has been POS and semantically tagged using GATE and WordNet for making the text in this corpus suitable for input to a machine learning algorithm. The structures of the POS and semantically tagged objectives make it necessary to utilise a machine learning method that is able to process relational background knowledge and describe the relations between the entities in the objectives and the SMART criteria in order to automatically learn grammar rules. This section describes how inductive logic programming (ILP) is applied to automatically infer a grammar for formulating SMART objectives.

The section is organised as follows. Section 5.1.1 summarises some specifications needed by ILP and the inductive machine learning method called ALEPH. Section 5.1.2 utilises the ILP algorithm ALEPH to learn a grammar for writing SMART objectives. Section 5.1.3 presents the validation of the rules learned by ALEPH against the training data.

5.1.1 Specifications Needed by ILP and ALEPH

As described in Chapter 2, ILP has been applied in different studies and applications to learn theories from relational data, given background knowledge and a set of examples. In this research, the ILP system called ALEPH, which has been applied to learn grammars and semantics, is utilised to infer transparent linguistic rules that help in modeling well-written SMART objectives.

Before describing how the ALEPH system is applied in this study, some main characteristics of ILP and the ALEPH system are illustrated in greater detail in the following paragraphs.

In general, the ILP system ALEPH is able to process relational background knowledge as well as positive and negative examples in order to generalise useful rules and theories. The learning process performed by ALEPH starts to search the space for the best hypothesis. During this step, ALEPH uses mode declarations which are provided by the user in the background knowledge file in order to constrain the search space. Mode declarations determine the predicates that define the relations as well as the type of data and the arguments for each predicate. They specify if a predicate can be utilised in the head (modeh declarations) or in the body of the generalised rule (modeb declarations). Furthermore, they can identify the form for calling predicates and the number of calls to the predicate when constructing the bottom clause (most specific clause) (i.e. this represents the ‘RecallNumber’ which is either an integer, n > 0, or ‘*’). A mode declaration also indicates whether an argument of a predicate may be an input variable, an output variable or a constant argument. The following example represents the mode declaration of a predicate called specific(A,B):

```
“:-modeh(*,specific(+vertex, -vertex)).”
```

The declaration indicates that the predicate specific(A,B) has to be used as a head predicate (modeh). Additionally, this predicate consists of two arguments “A” and “B” of type vertex (+vertex, -vertex), where the type vertex denotes the positions of words in a sentence. The symbol “+” indicates that the first argument is an input variable, while the symbol “-” states that the second argument is an output variable. If the symbol “#” appeared instead of “-” or “+”, this would mean that the argument is constant (ground term). The symbol “*” highlights that the ‘RecallNumber’ of the predicate “specific” is non-determined.

As well as mode declarations, ALEPH uses determinations which are also specified in the background knowledge file for describing the predicates in order to construct a theory. In particular, determinations identify the possible relationships between the target (head) and background (body) predicates. If there are no determination statements present in the background knowledge file, ALEPH will not build any clauses. The following example represents the possible format of determination for the relation (predicate) called “measurable (A, B)”:

```
“determination(measurable/2,money/2).”
```

In this example, there are two predicates, “measurable” and “money”, where the specified determination indicates that there is a relationship between these two predicates. The first predicate “measurable/2” represents the target predicate which appears in the head of the generalised rule. The arity “/2” means that the predicate “measurable” consists of two arguments. The second predicate “money/2” represents the background predicate which appears in the body of the generalised rule. The arity “/2” means that the predicate “money” consists of two arguments.

5.1.2 Using ILP to Learn a Grammar for Writing SMART Objectives

In this study, the ILP machine learning system ALEPH is applied on the corpus of the tagged objective sentences to induce general rules which specify all the linguistic patterns that can be used to express the features needed to ensure that an objective is “specific”, “measurable”, and “time-related”. Therefore, the POS tags of words and the annotations (NE annotations and some other semantic annotations) generated by GATE as well as the semantic class tags of target words specified by WordNet are the main contextual information which have been utilised by ALEPH in order to automatically learn the rules for formulating SMART objectives. All this information is exploited as background knowledge (i.e. includes logical definitions of relations) and ground facts (i.e. positive and negative examples) by ALEPH in the learning process to derive generic and meaningful hypotheses (rules) that are able to explain the given contextual information and the examples. ALEPH uses literals in the background knowledge to describe the linguistic pattern for each word in the given POS and semantically tagged objective sentences in order to parse these sentences and then generate rules in a logical form that represent the linguistic information about the patterns. The literals contain predicates written in the PROLOG language to describe relations that represent the POS and semantic tags of words in the objective sentences which have been added to the background knowledge file (file.b) (i.e. all the POS and semantic information of words in the objective sentences were converted into PROLOG facts in order to be used by ALEPH). The following represents an example of some clauses defined in the created background knowledge, where these clauses consist of sets of literals for describing the linguistic representation of words:

```
common_noun(W):- plural_common_noun_nns(W).
common_noun(W):- singular_common_noun_nn(W).
plural_common_noun_nns(W):-word(W,nns,_,_).
singular_common_noun_nn(W):- word(W,nn,_,_).
```

The above literals represent the POS literals which are used to describe the POS tags of words that appear in the given objective sentences, where the predicate (relation) called "common_noun(W)" indicates a common noun and the predicate called "plural_common_noun_nns(W)" means a common noun in the plural form, while the predicate "singular_common_noun_nn(W)" denotes a singular common noun. Thus, the first two clauses in the above literals mean that a common noun could be a plural common noun or a singular common noun. In PROLOG, the upper case letters (e.g. 'W') which appear between the brackets for a relation (e.g. plural_common_noun_nns(W)) refer to variables, while the lower case letters (e.g. 'nns') refer to atoms. The variable 'W' in the predicate called "plural_common_noun_nns(W)" indicates that this predicate consists of only one variable called 'W' which denotes a word in a sentence, since this word is a plural common noun. If the symbol '_' appeared in a predicate relation, this indicates an anonymous variable. The anonymous variable '_' is used when the variable occurs only once within a clause, such that each occurrence of '_' within a clause corresponds to a different variable. For instance, the predicate “word(W,nns,_,_)” which refers to the mode declaration “:- modeb(*,word(+word, #pos, +vertex, -vertex)).” (presented in the background knowledge file in Appendix C1) includes one variable ‘W’ which indicates a word in a sentence, a constant argument that refers to the POS tag of the word which is a plural common noun (‘nns’) here, and two anonymous variables (‘_,_’) that refer to the position of the word in the sentence (e.g. “word(gross_sales, nns, 3, 4)”). More specifically, the predicate “word(W,nns,_,_)” is used to define each word that represents a plural common noun in the given objective sentences.

The clause “common_noun(W):- plural_common_noun_nns(W).” includes two literals: “common_noun” and “plural_common_noun_nns”, where “common_noun” is called the immediate generalisation of “plural_common_noun_nns”, while “plural_common_noun_nns” is the immediate specialisation of “common_noun”. This clause means that a word in an objective sentence is considered as a common noun if it is a plural noun. Also, the word can be considered as a common noun if it is a singular noun (i.e. common_noun(W):- singular_common_noun_nn(W).).

The following clauses represent some of the semantic literals that are utilised to describe the semantic classes of target words in the objective sentences:

```
common_noun(_):- noun_possession_nns(_,_).
common_noun(_):- noun_possession_nn(_,_).
```

The predicate “noun_possession_nns(_,_)” refers to a plural noun (nns) which has been classified with the noun semantic class “possession” (e.g. ‘sales’), while the predicate “noun_possession_nn(_,_)” denotes a singular noun (nn) that has also been classified with the noun semantic class “possession” (e.g. ‘gross-revenue’). Thus, the above clauses mean that if a word is a common noun, then it can be classified into a plural noun of the semantic class “possession” or a singular noun of the semantic class “possession”.

To illustrate the background knowledge predicates that describe the literals of the body for the constructed clauses, consider the following example of an objective sentence presented in the background knowledge file:

Example 5: “To increase PC gross-sales by at least £18999 by 2008.”

The following represents the literals stored in the background knowledge file to describe the predicates (i.e. constituents) that define linguistic patterns for the words in *Example 5* including the POS and semantic tags of words as well as the positions between the constituents in the given objective sentence (*Example 5*):

```
to(0,1).
change_action_verb_vb(1,2).
product(2,3).
noun_possession_nns(3,4).
prepositional_phrase_pp(4,7).
money(7,9).
preposition_in(9,10).
date(10,11).
punc(11,12).
```

In this example, the predicate called “change_action_verb_vb (1, 2)” indicates a word in the given objective sentence (*Example 5*) that represents an action verb in the base form (VB) classified with the verb semantic class called “change” from word position “1” to word position “2” in the sentence.

The following presents the general linguistic representation of *Example 5* by showing how the words are segmented and tokenized in this sentence and Table 12 clarifies the positions between words in *Example 5*.

“$_{0}$ **To** $_{1}$ **increase** $_{2}$ **PC** $_{3}$ **gross-sales** $_{4}$ **by** $_{5}$ **at** $_{6}$ **least** $_{7}$ **£** $_{8}$ **18999** $_{9}$ **by** $_{10}$ **2008** $_{11}$ **.** $_{12}$”

From	To	Word
0	1	To
1	2	increase
2	3	PC
3	4	gross-sales
4	5	by
5	6	at
6	7	least
7	8	£
8	9	18999
9	10	by
10	11	2008
11	12	.

Table 12 Representation of Words and Words' Positions in *Example 5*

As described before, the ALEPH system has the ability to parse the sentences specified in the background knowledge file in order to infer theories that cover all the positive examples and none of the negative examples. Thus, ALEPH requires a set of positive and negative examples to induce generalised rules that explain the POS and semantic context for each objective sentence in order to distinguish the relevant objectives from the irrelevant ones. The relevant objectives are those objectives that include the features expected for an objective in order to be "specific", "measurable", and "time-related". Therefore, a set of positive examples which includes relevant ground facts has been created and added to the positive examples file (file.f) in order to be utilised by ALEPH in the learning process, such that all these examples must be covered by the generalised hypotheses. For instance, in order to learn the rules for the target predicate "measurable(A,B)", the following positive examples are stored in the file "file.f" and used by ALEPH:

```
measurable(7,9).
measurable(21,23).
measurable(33,35).
measurable(58,60).
```

As described above, the constituents of a sentence are represented as predicates in the ALEPH background knowledge file to describe a linguistic pattern for each word and the positions between words in the sentence. Based on that, the first positive example ("measurable(7,9)") presented in the above examples is used to refer to the measure or numeric indicator which appears in *Example 5*. This measure represents the predicate "money(7,9)" which has been specified in the background knowledge file to describe the words in *Example 5* that indicate money (£18999), from word position "7" to word position "9". Thus, the positive example "measurable(7,9)" is specified in the positive example file to call the predicate "money(7,9)" from the background knowledge file, since each them ("measurable(7,9)" and "money(7,9)") has the same word positions ("7,9") in the sentence.

The positive example "measurable(7,9)" means that the objective sentence is "measurable", from word position "7" to word position "9". By reviewing *Example 5*, the positive example "measurable(7,9)" represents a relevant example, since it covers the correct part in the *Example 5* sentence that indicates a measure (i.e. money(7,9)).

Moreover, a set of negative examples is also added to the negative examples file (file.n) in order to be used by ALEPH for generating hypotheses. The negative examples represent the irrelevant ground facts that should not be covered by the generalised hypotheses. For instance, in order to discover the rules for the target predicate "measurable(A,B)", the following negative examples are stored in the file "file.n" and used by ALEPH:

```
measurable(1,2).
measurable(2,3).
measurable(3,2).
measurable(19,20).
```

The negative example "measurable(1,2)" means that the objective sentence is "measurable", from word position "1" to word position "2". By reviewing *Example 5*, the negative example "measurable(1,2)" represents an irrelevant example, since it covers a part in the *Example 5* sentence that does not represent a measure or a numeric indicator (money).

Once the background knowledge, positive and negative examples are prepared as illustrated above, ALEPH is used to learn the rules for writing SMART objectives.

The generalised hypotheses (rules) by ALEPH explain all the positive examples and identify clearly the linguistic patterns in the objectives which can be utilised to express the elements required to check whether an objective is “specific”, “measurable”, and “time-related”.

The definition of the SMART approach (Hurd et al., 2008) presented in the literature review has shown that an objective is considered “specific” if it is well-defined and describes clearly the action to be taken to accomplish the objective by using an action verb (Rouillard, 2003) and by specifying the objective domain in the objective sentence (Carliner, 1998). In this research, ALEPH has learnt the grammar rules to ensure that an objective is “specific” by extracting the POS tags of words (e.g. plural noun (NNS), verb (VB), singular noun (NN) etc), the NE annotations (e.g. ‘Percent’ and ‘Money’), and the semantic classes of the target common nouns (e.g. possession) and verbs (e.g. change, social, creation, possession, motion) from the objective sentences as well as some other semantic information (i.e. product “e.g. PC/ mobile”).

ALEPH has induced several grammar rules from the corpus of objectives, including the following PROLOG rule for ensuring that an objective is “specific”:

```
specific(A,B) :-
        creation_action_verb_vb(A,C), percent(C,D), noun_possession_nn(D,E),
        preposition_in(E,F), product (F,B).
```

The above rule consists of the head of the rule (specific(A,B)), which is followed by the symbol “:-” that can be read as an “IF” in PROLOG, and the body of the rule which includes the body predicates (which describe the contextual information about the words that occur in the objective sentences) separated by commas “,” that indicate an “AND” in PROLOG. All the variables (e.g. A, C, D…) in the above rule represent the positions of predicates that describe linguistic patterns in a sentence. Thus, the generalised rule defines the “specific” relation and means that the objective sentence is “specific” from word position “A” to word position “B” if the sentence includes the following: an action verb in the base form (VB) classified with the verb semantic class “creation”, from word position “A” to word position “C”, the semantic annotation “Percent”, from word position “C” to word position “D”, a singular noun (NN) classified with the noun semantic class “possession”, from word position “D” to word position “E”, a preposition (IN), from word position “E” to word position “F”, and the semantic annotation “product”, from word position “F” to word position “B”. These predicates (creation_action_verb_vb/2, percent/2, noun_possession_nn/2, preposition_in/2, and product/2) must be specified in the ALEPH setting (determinations) in the background knowledge file in order to construct the theory.

Furthermore, a SMART objective must be “measurable” and normally include a specific evaluation score (e.g. a numeric indicator/measure) to evaluate the achievement of the objective (Hurd et al., 2008). ALEPH has inferred the grammar rules for ensuring that an objective is “measurable” by extracting the tags that represent a numeric indicator (e.g. “Percent” or “Money”) from the objective sentences. The following PROLOG rule is one of the induced grammar rules by ALEPH from the corpus of objectives for ensuring that an objective is “measurable”:

```
measurable(A,B):-
                percent(A,B).
```

The above rule means that the objective sentence is “measurable” from word position “A” to word position “B” if the sentence contains the semantic annotation “Percent” from word position “A” to word position “B”.

With regards to the SMART approach, an objective is considered “time-related” if the deadline for achieving it is stated in the objective (Hurd et al., 2008). ALEPH has discovered a grammar rule for ensuring that an objective is “time-related” by extracting the tags that represent dates (e.g. “Date”) from the objective sentences. The following PROLOG rule is derived by ALEPH from the corpus of objectives for ensuring that an objective is “time-related”:

```
time_related(A,B):-
                date (A, B).
```

The induced grammar rule means that the objective sentence is “time-related” from word position “A” to word position “B” if the sentence contains the semantic annotation “Date” from word position “A” to word position “B”.

To illustrate the grammar produced by ALEPH, consider the following example of an objective sentence from those objectives presented in the developed corpus.

Example 6: “To generate 12% gross-revenue for PCs within 2009 by offering many products for sale.”

By applying the above mentioned grammar rule for ensuring that an objective is “specific” to *Example 6,* the predicate creation_action_verb_vb(A,C) represents the action verb “generate” in *Example 6* which is classified with the verb semantic class “creation”. The predicate percent(C,D) represents the percentage of sales “12%” in *Example 6*. The predicate noun_possession_nn(D,E) indicates the target noun “gross-revenue” in *Example 6* which represents a singular common noun classified with the noun semantic class “possession”. The predicate preposition_in(E,F) refers to the preposition “for” in *Example 6*. Finally, the predicate product (F,B) represents the word “PCs” in *Example 6*. Thus, it is evident that part of the sentence of *Example 6* meets the requirements in order to be “specific”. Furthermore, by applying the above mentioned grammar rule for ensuring that an objective is “measurable” to *Example 6,* the predicate percent(A,B) represents the numeric indicator of sales percentage “12%” which appears in the objective sentence. As a result, part of the sentence of *Example 6* meets the requirements in order to be “measurable”. By applying the above mentioned grammar rule for ensuring that an objective is “time-related” to *Example 6,* the predicate date(A,B) represents the year “2009” which appears in the objective sentence. Thus, part of the sentence of *Example 6* meets the requirements in order to be “time-related”.

In general, the results of the generalised rules by ALEPH show that there is diversity in the rules inferred to ensure that an objective is “specific”. For example, the following rule has been also generalised by ALEPH to ensure that an objective is “specific”:

```
specific(A,B):-
        product(A,C), noun_possession_nns(C,D), modals(D,E),
        change_action_verb_vb(E,F), preposition_in(F,G), money(G,B).
```

This rule means that the objective sentence is "specific" if it includes the semantic annotation "product", followed by a plural common noun (NNS) classified with the noun semantic class "possession", a modal verb, an action verb in the base form (VB) classified with the verb semantic class "change", a preposition (IN), and the semantic annotation "Money". The induced rule describes clearly the type of the verb (action verb) and noun (plural common noun) used to write a SMART objective as well as some other semantic features. By applying the above rule to an objective sentence such as: "By the following year, mobile sales will boost to £819.", it is obvious that part of this objective sentence achieves the requirements in order to be "specific".

All of the constructed rules by the ALEPH learner represent consistent hypotheses which can be linguistically interpreted and all of them illustrate the contextual information (POS and semantic information) about the words in the given objective sentences as well as provide information about the respective position of each word in the objective sentences.

5.1.3 The Validation of the Rules Learned by ALEPH

For the experiments, the ALEPH algorithm (version 5) has been applied on a data sample of the developed corpus consisting of 70 POS and semantically annotated objective sentences. All of these sentences are provided to ALEPH as background knowledge. For ALEPH to learn, examples of "specific", "measurable", and "time-related" are needed. This is done by finding sub-parts of the 70 sentences and generating relevant positive and negative examples. In total, 210 positive examples are generated (70 examples for "specific", 70 examples for "measurable", and 70 examples for "time-related"), and 350 negative examples are created (150 examples for "specific", 100 examples for "measurable", and 100 examples for "time-related"). The created background knowledge file describing the relations of the words in the objective sentences is given in Appendix C1. The sets of examples generated from the objectives and used by ALEPH are given in Appendix C2.

Calls to the command “induce” in the ALEPH algorithm performs a general-to-specific search to construct theories. Moreover, the “induce” command estimates the performance on the training data as a confusion matrix.

Therefore, all the positive examples were covered and ALEPH performed 100% accuracy on the training data using the YAP PROLOG compiler (Santos Costa et al., 2000) (version 6.2.0). The total clauses constructed were 164 clauses.

From the given background knowledge and the positive and negative examples, ALEPH has successfully induced 26 different linguistic rules (All of them are believed to be correct and useful) for defining a grammar to write SMART objectives, which can be used to verify whether the objectives are “specific”, “measurable”, and “time-related”. More specifically, the generalised linguistic rules include: 23 grammar rules for ensuring that the objectives are “specific”, 2 grammar rules for checking whether the objectives are “measurable”, and 1 grammar rule for assessing whether the objectives are “time-related”. The learned rules cover at least 2 examples or more. The generalised rules by ALEPH are given in Appendix A1, as well as the number of positive (Pos cover) and negative (Neg cover) examples covered by each rule.

As ALEPH allows the setting of various parameters in constructing a model, some kinds of learning parameters have been utilised in the experiments performed by ALEPH in this study, where these parameters have been specified in the background knowledge file. Basically, these parameters have been chosen to improve the quality of learning for the applied ILP machine learning method, estimate the accuracy of the generalised theories, and reduce the search space. The following bullet points describe the parameter settings used by ALEPH in this study:

- The parameter called “minpos” is used for specifying the minimum number of positive examples to be covered by the generalised clause (i.e. this parameter affects the size of the search space). The “minpos” parameter was set to 2.
- The size of the search space and the running time of ALEPH are reduced by setting the parameter “clause length” to a maximum of 10 literals in the constructed clause.

- The “samplesize” parameter was set to the default value “0” for ensuring that the examples are saturated (selected) in a set order (the order of appearance) in the positive examples file. On the other hand, the non default value of the parameter “samplesize” is normally used for specifying the number of examples to be selected randomly for constructing a clause, where the best clause is then added to the theory. In general, the “samplesize” parameter affects the type of the search used by ALEPH.
- The number of search nodes was limited to 5000 for the “nodes” parameter (default 5000) in ALEPH to reduce the search space.
- The depth of variables in the bottom clause that entails (covers) the selected example was set to 99 for the “i” setting parameter (set (i,99)).
- The “minacc” parameter (default 0.0) was set to 0.80. This parameter specifies the minimum accuracy of an acceptable clause (i.e. the accuracy of a clause here represents the precision which is defined as the number of positive examples covered by a clause divided by the total number of positive and negative examples covered by the clause).
- To allow more generalisations, the “noise” parameter (default 0) is used to accept clauses which cover negative examples. The “noise” parameter was set to 50 to accept at maximum 50 negative examples (25 examples for “measurable”, 20 examples for “time-related” and 5 examples for “specific”) to be covered by the generalised clauses. Thus, the negative examples file includes 400 examples.
- The evaluation function for the search performed by ALEPH was set to “coverage” (set (evalfn, coverage)). The “coverage” is the ALEPH’s default clause evaluation function for searching clauses. This function set the clause utility to be P-N, where P and N represent the number of positive and negative examples covered by the clause.
- The parameters called “test_pos” and “test_neg” are both used for evaluating the accuracy of the learned theories by ALEPH on the testing data. For example, the parameter (set(test_pos,+V)) specifies the file name that includes the positive examples for testing; while the parameter (set(test_neg,+V)) specifies the file name that includes the negative examples for testing.

After determining the parameter settings needed to improve the quality of the learning for ALEPH, a validation of the rules learned by ALEPH against the training data is performed to evaluate the quality of the obtained rules. To carry out this evaluation, a 10-fold cross-validation test has been conducted on the created sets of 210 positive examples and 400 negative ones. Consequently, the set of examples (positive and negative examples) is split into ten folds, each of these folds is used as a testing set whilst the other folds are utilised as training set. This led to performing ten learning processes with these training datasets which are evaluated on the corresponding testing datasets. Then, the average accuracy of the learned theories is computed for all of the performed learning processes.

Table 13 demonstrates the proportion of the training and testing datasets used in the conducted evaluation and accuracies achieved by ALEPH for the training and testing datasets using 10-fold cross-validation.

	Proportion	**Average Accuracy**
Training Dataset	90%	92%
Testing Dataset	10%	89%

Table 13 Cross Validation Results (1)

The above table shows the results of 10-fold cross-validation for the rules learned by ALEPH, such that these results are obtained by using the parameter adjustments described earlier in this section and by allowing some noise for testing the learned theories. Thus, the estimated accuracy of the learned theories by ALEPH is 92% on the training data and 89% on the testing data. The validation of the quality of the grammar rules learned by ALEPH has achieved encouraging results, where the standard deviation obtained through the 10-fold cross-validation test is 0.0135.

This section has shown how the ALEPH machine learning method was able to automatically learn novel and general grammar rules from a set of objective sentences, where the learned rules cover all the intended information. The generalised grammar rules by ALEPH can then be used for writing SMART objectives and ensuring that the objectives are “specific”, “measurable”, and “time-related”. The following section explains the second part of the framework which is concerned with utilising data mining techniques for assessing the objectives.

5.2 Using Data Mining Techniques for Assessing the Objectives

This section describes how the data mining techniques are applied for assessing the objectives. Section 5.2.1 illustrates the use of a predication algorithm for assessing whether an objective is "achievable". Section 5.2.2 explains the use of a classification rule induction algorithm for ensuring that an objective is "realistic".

5.2.1 Using Data Mining for Assessing If an Objective is "Achievable"

An objective is considered "achievable" if it could be met within a specified period of time (Hurd et al., 2008). To set an "achievable" objective related to a domain such as sales using the SMART approach, an employee must determine a relevant amount of sales in the objective, such that this amount of sales could be achieved within a given timescale. This requires estimating the expected amount of sales that can be attained in the given timescale. To illustrate this aspect, consider the following example objective:

"To increase the sales of product 'A' to £90.000 by 2014."

The question is "Is it possible to achieve £90.000 of sales for the product 'A' in 2014, given that the current year is 2013 and the achieved amount of sales for the product 'A' in year 2013 is ranging from (£ 8000-£10.000)? ". Based on the studies surveyed in the literature review, which apply different data mining techniques for forecasting demand, making decisions and predicating future events such as sales, this question could be addressed by analysing the actual historical data of sales for the product 'A', and then forecasting the sales for this product in year 2014 by using such techniques. In case that the historical data of sales for the product 'A' is unavailable, then the subjective judgements of experts in sales can be utilised to give an approximate estimation for the expected sales amount of the product 'A' in the future.

In general, forecasting is a field in itself, and the aim here is to check if it could be used as part of the framework for assessing whether a stated objective is "achievable".

To carry out this assessment, the developed system in this research based on the framework has been applied to the domain of sales, and past historical data of sales has been utilised to forecast the amount of sales in a given objective in order to check whether this objective is "achievable". This task requires identifying the suitable timescale (i.e. date) to attain a measure (i.e. amount of sales) specified in an objective sentence. The previous section showed how the ILP learner ALEPH has induced grammar rules (the "measurable" and "time-related" rules) for extracting the measures and dates from the objective sentences. By extracting these features (measures and dates) from the objectives, the idea is then to use the extracted information by ALEPH in assessing whether a measure presented in an objective sentence is possible to be achieved within a given timescale in the same objective sentence.

Although there are several options, for simplicity, the linear regression algorithm (Anderson, Sweeney, and Williams, 2005; Han, Kamber, and Pei, 2011) is utilised in this research to assess whether an objective is "achievable". This statistical algorithm is applied within the WEKA toolkit (Hall et al., 2009; Witten, Frank, and Hall, 2011) (version 3.6) on the "U.S. Consumer Electronics Sales and Forecasts"[9] dataset, obtained from the Infochimps[10] repository. This dataset is released by the "Consumer Electronics Association" (CEA) in U.S, where it contains detailed information about sales of some electronic product categories from year 2003 to 2007, as well as the total sales of all products. In this study, the data for two products (i.e. PC and mobile) has been selected from this dataset and exploited by the applied algorithm.

Accordingly, the linear regression algorithm analyses the historical data of sales and years for a specified product (PC/mobile) in order to define the relationship between sales and time and estimate the expected product sales for a particular timeframe. This algorithm uses the information extracted by the rules generalised by ALEPH from the objectives (e.g. the "Money" and "Percent" patterns which refer to the "sales amount" as well as the "Date" patterns which indicate a "date" in the given objectives). Then, the developed system compares the value of the product sales set in an objective with the estimated one for a given timescale and checks whether the amount of product

[9] http://www.infochimps.com/datasets/us-consumer-electronics-sales-and-forecasts-2003-to-2007, Source: Consumer Electronics Association, Washington, DC, U.S. Consumer Electronics Sales and Forecasts: 2003 to 2007

[10] http://www.infochimps.com

sales in the objective is within the predicted sales interval for the specified timeframe. This implies verifying whether the amount of product sales in the objective is possible to be achieved within the proposed timeframe in the same objective based on the predicted product sales which have been estimated by the linear regression algorithm for that timeframe.

In this research, two experiments have been conducted by the linear regression algorithm using WEKA. The first experiment of this algorithm has been conducted on the sales dataset of mobile devices, whereas the second experiment has been performed on the sales dataset of PCs. These two experiments are discussed in the following paragraphs.

The dataset of mobile devices, which has been used in the experiments, is presented in Table 14, where this table specifies the actual sales of mobile from year 2003 to 2007.

Years (2003-2007) (x_i)	**Mobile Sales (y_i) (In Millions)**
1	417
2	521
3	552
4	679
5	789

Table 14 Mobile Sales Data (from 2003 to 2007)

The sales (y_i) of mobile devices from year (x_i) 1 to year 5 are demonstrated in the above table, where year ‘1’ represents year 2003 and the achieved sales in this year are 417, year ‘2’ indicates year 2004, and the attained sales in this year are 521, year ‘3’ refers to year 2005, and the accomplished sales in this year are 552, year ‘4’ means year 2006, and the obtained sales in this year are 679, and where year ‘5’ denotes year 2007, and the achieved sales in this year are 789.

Mobile dataset is prepared and converted to ARFF format in order to be compatible with the WEKA workbench which is employed in building the linear regression model. It contains 6 instances and 2 attributes named as “sales” and “year”, where the type of these attributes (“sales” and “year”) is numeric (i.e. real number). By using the

WEKA toolkit, the training dataset of mobile is used to fit the linear regression model, resulting in the linear equation:

$$sales = 321+90.2* year$$

where the variable "*sales*" represents the dependent variable y_i and the variable "*year*" indicates the independent variable x_i in the linear regression formula.

The above formula can then be used to estimate the expected sales of mobile for a specific timeframe. As an example, if we use the above equation to predict the mobile sales for year '6' which refers to year 2008, the result would be as follows:

Mobile sales for year '6': $sales = 321 + 90.2 * 6 = 862.2$

If the year is '7' (2009), then the estimated sales for mobile are:

Mobile sales for year '7': $sales = 321 + 90.2 * 7 = 952.4$

The performance of the linear regression model has been evaluated in WEKA on the training dataset of mobile sales. The model has achieved a correlation coefficient of 98% on the whole dataset of mobile sales and the estimated root mean squared error is 21.367.

As mentioned before, the second experiment of the linear regression algorithm has been conducted on a sales dataset of PC. This dataset is presented in Table 15, where this table demonstrates the actual sales of PC from year 2003 to 2007.

Years (2003-2007) (xi)	**PC Sales (yi) (In Millions)**
1	15.584
2	18.233
3	19.4
4	19.666
5	20.264

Table 15 PC Sales Data (from 2003 to 2007)

The sales (y_i) of PC from year (x_i) 1 to year 5 are presented in the above table, where year '1' represents year 2003 and the achieved sales in this year are 15.584, year '2' indicates year 2004, and the attained sales in this year are 18.233, year '3' refers to year 2005, and the accomplished sales in this year are 19.4, year '4' means year 2006, and the obtained sales in this year are 19.666, and where year '5' denotes year 2007, and the achieved sales in this year are 20.264.

PC sales dataset is also prepared in ARFF format and given to the WEKA toolkit, since it contains 6 instances and 2 attributes (i.e. "sales" and "year") of type numeric. By training the WEKA implementation of the linear regression algorithm on the PC sales data, a classifier model has been built and the following formula of the linear regression model has been generated to estimate the expected sales of PC for a given timescale:

$$sales = 15.3915 + 1.0793 * year$$

The performance of the linear regression model has been evaluated in WEKA using the training set of PC sales. The model has achieved a correlation coefficient of 92% on the whole dataset of PC sales and the estimated root mean squared error is 0.6516.

After generating the linear regression equations, the statistical prediction interval equation is then utilised to specify the sales interval of y_p for a given year x_p based on the generated linear regression equations. Before computing the prediction interval of an individual value of y_p, the standard deviation of y_p must be estimated first. The standard deviation of an individual value of y_p (Anderson, Sweeney, and Williams, 2005) is,

$$S_{ind} = s * \sqrt{(1 + 1/n + (x_p - \bar{x})^2 / \sum (x_i - \bar{x})^2)} \qquad (5.1)$$

where:

- n represents the number of observations (i.e. the sample size),
- x_p is the individual value of the independent variable x_i,

- $\bar{x}$ represents the mean of the independent variable x_i (the $\bar{x}$ equation is given below),
- *s* represents the standard error (the *s* equation is presented below).

The mean of the independent variable x_i is given by equation (5.2) (Anderson, Sweeney, and Williams, 2005):

$$\bar{x} = \text{sum } (x_i) / \text{ number of } (x_i) \tag{5.2}$$

The standard error *s* is given by equation (5.3) (Anderson, Sweeney, and Williams, 2005):

$$s = \sqrt{\frac{SSE}{n-2}} \tag{5.3}$$

where *SSE* represents the sum of squared errors (the difference between the observed value of the dependent variable y_i and the estimated value of the dependent variable *y*) (the *SSE* equation is given below).

The sum of squared errors *SSE* is given by equation (5.4) (Anderson, Sweeney, and Williams, 2005):

$$SSE = \sum (yi - y)^2 \tag{5.4}$$

The prediction interval of y_p (Anderson, Sweeney, and Williams, 2005) is,

$$y_p +- t_{\alpha/2} * S_{ind} \tag{5.5}$$

where:

- y_p is the estimated value of the dependent variable *y*,
- α represents the confidence coefficient,
- $t_{\alpha/2}$ represents the "t" distribution with *n-2* degree of freedom (*n* is the sample size).

Thus, the prediction interval equation is applied to estimate the sales interval for each of the mobile and PC datasets.

For the two applied datasets of sales (mobile and PC), the linear regression algorithm has been used to predict the sales up to two years (2008 and 2009), since the literature review has shown that the regression models are more effective in forecasting sales especially for medium-term (up to two years) forecasts (Reid and Bojanic, 2009). On the other hand, forecasting sales for long-term (more than 2 years) periods makes an objective less effective and more difficult to be achieved, since it will be too far in the future. As a result, the long-term forecasts of sales are better to be avoided when setting objectives related to the sales domain.

5.2.2 Using Data Mining for Assessing Whether an Objective is "Realistic"

With regards to the SMART approach, an objective is considered "realistic" if the sufficient resources are available to achieve it (Hurd et al., 2008). For example, to assess whether a business objective related to a domain such as sales is "realistic", this necessitates classifying the required resources that must be allocated to accomplish the desired sales within a specified period of time. The classification task here is based on a decision making process, such that the sales people can use some historical data to make accurate decisions in the future regarding to the quality, quantity and availability of the required resources to perform the expected sales. The required resources that should be available to achieve the target sales for a specific product may include items, staff, and time. There are several data mining techniques which can be employed for supporting the decision making process and classifying data by inducing decision and classification rules.

Thus, the aim here is to check if data mining methods could be also used as part of the framework to ensure whether an objective is "realistic" by classifying the required resources to achieve an objective, such as identifying the required number of items and staff in order to attain the desired sales within a given timeframe.

To perform this task, the JRIP classification data mining algorithm, which is the WEKA implementation of the RIPPER (Cohen, 1995) rule induction algorithm, is applied to assess whether an objective is “realistic”. In this study, JRIP has used some information extracted by ALEPH (i.e. ‘Product’ and ‘Date’ patterns) and some other additional information (i.e. the available staff and items) for inducing some classification rules that classify the product type with the required number of staff and items for a proposed date in order to obtain the expected sales.

For the experiments, the JRIP rule induction algorithm has been applied on a dataset of randomly selected examples created by using some suggested rules that describe some information about two products (i.e. PC and mobile)[11]. These rules are suggested to produce a dataset which can be exploited by the applied rule induction algorithm. The suggested rules are provided as input data to a Java program in order to generate the experimental dataset. Thus, a simple Java program is implemented to produce random positive and negative examples to be used by the JRIP algorithm.

The proposed rules are:

1. if (year = 2008) and (product = mobile) and (staff >= 100) and (staff <= 150) and (items >= 500) and (items <= 550) then realistic=yes
2. if (year = 2009) and (product = mobile) and (staff >= 170) and (staff <= 200) and (items <= 570) and (items >= 600) then realistic=yes
3. if (year = 2008) and (product = PC) and (staff >= 300) and (staff <= 350) and (items >= 1000) and (items <= 1300) then realistic=yes
4. if (year = 2009) and (product = PC) and (staff >= 400) and (staff <= 450) and (items >= 1500) and (items <= 1600) then realistic=yes

For the experiments, the created dataset is given to WEKA and it contains some attributes such as “product”, “staff”, “items” and “year”, while the target class attribute is named as “realistic”. More specifically, this dataset consists of 790 instances and 5 attributes (product, year, items, staff, and realistic), of which 3 are nominal (i.e. “product”, “realistic”, “year”) and 2 are numeric (i.e. “items”, “staff”). The instances in this dataset represent the examples which include 392 positive examples and 398 negative examples.

[11] "This was done since data was unavailable, though one can envision that such data could be collected if the kind of system proposed is deployed in practice."

By using WEKA, a classifier model has been built from the utilised dataset and a set of JRIP classification rules represented in a form of if-then rules has been generated. The following represents the output of the applied JRIP rule induction model within the WEKA toolkit:

1. (staff >= 101) and (staff <= 150) and (items >= 501) and (items <= 550) and (year = 2008) and (product = mobile) => realistic=yes (94.0/0.0)
2. (staff >= 170) and (staff <= 200) and (year = 2009) and (items <= 600) and (product = mobile) and (items >= 570) => realistic=yes (98.0/0.0)
3. (items >= 1012) and (staff >= 304) and (staff <= 342) and (items <= 1298) and (year = 2008) and (product = PC) => realistic=yes (84.0/0.0)
4. (staff >= 400) and (items >= 1506) and (items <= 1600) and (staff <= 450) and (product = PC) and (year = 2009) => realistic=yes (94.0/0.0)
5. (items >= 1007) and (items <= 1261) and (staff <= 350) and (staff >= 301) and (year = 2008) and (product = PC) => realistic=yes (14.0/0.0)
6. => realistic=no (406.0/8.0)

As it is shown above, the JRIP model has induced 6 classification rules from the applied training dataset, such that all these rules are explicit and understandable as well as they cover most of the examples. All these rules are constructed in a bottom-up fashion. The first rule can be read as:

If the year is equal to 2008 **and** the product is mobile **and** the number of staff is greater than or equal to 101 and less than or equal to 150 **and** the number of items is greater than or equal to 501 and less than or equal to 550, **then** the result is “realistic”. The number between the two brackets (94.0/0.0) means that 94 of the training examples are covered successfully by this rule.

The other rules can be read in a similar way and the last rule is activated if all the rules above it are not applicable.

The performance of the JRIP classification model has been evaluated using 10-fold cross-validation in WEKA in order to randomise the data instances and achieve representative classification accuracy. By using cross validation, the applied data is divided into two sets: a set of training examples that is used to train the model and a set of testing examples is utilised to evaluate performance of the model. The rule induction model has achieved a classification accuracy of 95%, where 752 instances have been correctly classified by the utilised model.

The aim above is to illustrate what could be possible in using a classification algorithm for assessing if an objective is “realistic” as part of the framework if relevant data were available. Clearly, in practice, the accuracy will depend on the availability of data and whether such data can be collected. If not, alternative subjective methods may need to be explored, such as using expert judgment.

5.3 An Illustrative Example for Describing the Developed System

The previous sections explained the main components of the framework. This section gives an illustrative example to show how an objective is assessed whether it is SMART by the developed system in this research.

Consider the following example of an objective presented in the developed corpus.

Example 8: “The sales department plans to achieve at least £1000 gross-sales for mobiles by 12/10/2009.”

The literature review presented in this research clarified the features of the SMART criteria, since the objective must be “specific”, “measurable”, “achievable”, “realistic”, and “time-related” in order to be SMART.

To check whether *Example 8* is “specific” according to the SMART approach, it must be focused and well-defined, where this includes the following characteristics (Hurd et al., 2008; Rouillard, 2003; Carliner, 1998):

- The objective must declare what is to be achieved.
- The objective must specify the accomplishment of the outcomes by using an action verb (e.g. increase, maximise, achieve…).

- The objective must state clearly to what it applies to (specify the objective domain, e.g. sales, costs, profits…).

In this case, *Example 8* is believed to be "specific", since it meets all the above-mentioned characteristics for the "specific" criterion.

The following describes how the developed system considers whether the objective is "specific".

First, the objective sentence of *Example 8* is annotated by GATE and this includes the following:

- The POS tags are specified for the words in this sentence.
- The entity names of proper nouns that indicate a numerical expression (numeric indicator) or a temporal expression (date) are recognised in this objective sentence by using the generated annotations 'Money' and 'Date'.
- The semantic annotations are extracted from the sentence, such as the 'Product' and 'Prepositionalphrase' annotations.

Then, the WordNet lexicon is utilised to perform the semantic analysis of the objective text. In particular, the SR-AW and WordNet::QueryData tools are applied to access the WordNet database in order to disambiguate the senses of words in the sentence and specify the WordNet semantic classes of the target words (the action verb and the common noun that represents the objective domain) in this sentence.

Once the sentence is analysed linguistically, the following grammar rule produced by ALEPH for assessing whether an objective is "specific" is used.

```
specific(A,B) :-
        social_action_verb_vb(A,C), prepositional_phrase_pp(C,D),
        money(D,E), noun_possession_nns(E,F),
        preposition_in(F,G), product(G,B).
```

By applying the above grammar rule, part of the sentence of *Example 8* is classified as being “specific”, since:

- The predicate social_action_verb_vb(A,C) is true given the action verb “achieve” which is classified with the verb semantic class “social”.
- The predicate prepositional_phrase_pp(C,D) is true given the prepositional phrase “at least”.
- The predicate money(D,E) is true given the sales amount “£1000”.
- The predicate noun_possession_nns(E,F) is true given the plural common noun “gross-sales”, which indicates the objective’s domain and is classified with the noun semantic class “possession”.
- The predicate preposition_in(F,G) refers to the preposition “for”.
- The predicate product(G,B) refers to the word “mobiles”.

The following clarifies which part of the *Example 8* sentence is considered “specific” (the bold and the underlined part of the sentence is “specific”):

> The sales department plans to **<u>achieve at least £1000 gross-sales for mobiles</u>** by 12/10/2009.

To assess whether *Example 8* is “measurable” according to the SMART criteria (Hurd et al., 2008), this requires checking if it specifies a measurement source/numeric indicator (e.g. sales amount) in the objective sentence which can be used to measure the achievement of the objective outcomes. Hence, *Example 8* is considered “measurable”, since it meets the above-mentioned feature for the “measurable” criterion.

To consider whether it is “measurable”, ALEPH utilises the linguistic information produced by GATE (e.g. the ‘Money’ annotation), which extracts the numerical expressions from the objectives text, and induces the following grammar rule:

```
measurable(A, B):-
          money(A,B).
```

By applying the above grammar rule to *Example 8,* part of the sentence is classified as "measurable". The predicate money(A,B) refers to the numeric indicator "£1000" which represents the sales amount (money). The following clarifies which part of the *Example 8* sentence is considered "measurable" (the bold and the underlined part of the sentence is "measurable"):

The sales department plans to achieve at least <u>**£1000**</u> gross-sales for mobiles by 12/10/2009.

To check whether *Example 8* is "time-related" according to the SMART approach (Hurd et al., 2008), this implies assessing if it states a particular timeframe (date) to attain the objective outcomes. Hence, *Example 8* is considered "time-related", where it achieves the above-mentioned characteristic for the "time-related" criterion.

To consider whether it is "time-related", ALEPH utilises the linguistic information generated by GATE (e.g. the 'Date' annotation), which extracts the temporal expressions from the objectives text, and derives the following grammar rule:

```
time_related(A, B):-
            date(A,B).
```

By applying the above grammar rule to *Example 8,* part of the sentence is classified as "time-related". The predicate date(A,B) refers to the proposed date "12/10/2009" in the sentence which represents the specified timeframe to obtain the desired sales. The following clarifies which part of the *Example 8* sentence is considered "time-related" (the bold and the underlined part of the sentence is "time-related"):

The sales department plans to achieve at least £1000 gross-sales for mobiles by <u>**12/10/2009**</u>.

To assess whether *Example 8* is "achievable" according to the SMART criteria (Hurd et al., 2008), this necessitates checking whether the amount of sales can be accomplished within the given timeframe (date).

Suppose that we use the past historical data of mobile sales which is previously mentioned in Table 14 (Section 5.2.1) to estimate the expected amount of mobile sales in *Example 8*. In this case, we must compare the amount of mobile sales from year 2003 to year 2007 in order to estimate the mobile sales in year 2009. The observations show that the amount of mobile sales is increasing every year. This will lead the sales' experts or observers to say that the amount of sales for mobiles may also increase during the next years. According to the dataset in Table 14, the amount of mobile sales in year 2007 is £789, while the amount of mobile sales in *Example 8* during year 2009 is £1000. Based on this dataset, we can say that the suggested amount of mobile sales (i.e. £1000) in *Example 8* is achievable during the proposed year (i.e. 2009).

The proposed method in this research for checking whether *Example 8* objective is "achievable" is presented below.

In the part of the framework related to assess whether an objective is "achievable", the WEKA implementation of the linear regression algorithm is applied on the mobile sales dataset presented in Table 14. Then, the idea is to compare the expected sales for a given timeframe with the suggested amount of sales in an objective for a proposed year. This requires checking whether the sales amount suggested in the objective for the specified year is within the estimated sales interval of mobiles for that year. By using the linear regression equation generated in WEKA for the mobile sales dataset to predict the amount of mobile sales for the proposed year (2009) in *Example 8*, the estimated amount of mobile sales y for year 2009 (seventh year, $xi=7$) is 952.4.

The statistical prediction interval formula (equation (5.5)), described in Section 5.2.1, is then used to specify the sales interval range y_p for a given year x_p depending on the linear regression equation generated for the mobile sales dataset.

The prediction interval for mobile sales for year 2009 is computed as:

$$952.4 \pm 146.877.$$

That is, the sales range of mobiles for year 2009 is from ‘£805.523’ to ‘£1099.277’. Therefore, *Example 8* is considered “achievable” because it is possible to achieve £1000 sales of mobile within the proposed year 2009.

To examine whether *Example 8* is “realistic” according to the SMART approach (Hurd et al., 2008) involves checking whether the required resources are available to achieve the objective outcomes.

To be able to carry out this evaluation, some additional information is needed about the available resources in the sales domain such as the quantity of items and staff required to obtain the desired product sales during a specified timescale. Thus, the resources that are needed to perform the desired sales of a product should be estimated for a given year. Suppose that the staff and items are available to achieve the amount of mobile sales suggested in *Example 8* for year 2009, then *Example 8* is considered “realistic”.

To consider whether this objective is “realistic”, the rules induced by using the rule induction system, JRIP, in WEKA (see Section 5.2.2 for description) are used. These rules are induced from the data and use the attributes year, product, number of staff, and number of items to classify if a given objective is “realistic”.

With regards to *Example 8*, suppose that the number of each of items and staff has been given as background information to *Example 8*, where the suggested number of available staff is 175 and the proposed number of available items is 590. By reviewing the rules obtained by JRIP, the following JRIP rule is used to classify the objective (*Example 8*) as “realistic”:

if (year = 2009) and (product = mobile) and (staff >= 170) and (staff <= 200) and (items <= 570) and (items >= 600) then realistic=yes

Thus, all five parts of the SMART (specific, measurable, achievable, realistic and time-related) criteria are satisfied in the *Example 8* objective, and as a result *Example 8* is a SMART objective.

5.4 Summary

In this chapter, a thorough explanation of using an ILP machine learning method to automatically derive a set of novel grammar rules for SMART objectives from a POS and semantically tagged corpus of objectives was given. The results of the rules obtained by the applied ILP learner ALEPH were given. The derived rules by ALEPH represent general, consistent, and meaningful hypotheses which can be used to help employees in structuring SMART objectives and in assessing whether the written objectives are “specific”, “measurable”, and “time-related”. A validation of the rules learned by ALEPH was given for estimating the quality of the obtained rules. The validation was carried out using 10-fold cross-validation and achieved encouraging results on 610 examples, where the accuracy of ALEPH on the training data was 92% and on the testing data was 89%. A comprehensive description of applying prediction and classification data mining methods in WEKA for assessing the objectives was presented. An experimental evaluation was also performed and showed promising results. The use of the prediction and classification techniques in the framework supports assessing whether the written objectives are “achievable” and “realistic”. Finally, an example objective was given to demonstrate how an objective is assessed whether it is SMART by the developed system based on the use of ILP and data mining techniques.

In the next chapter, a demonstration of the system implementation and a snapshot of the system interface will be presented. Moreover, an empirical evaluation of the developed system will be given. A second corpus of objectives related to the costs domain is presented together with an empirical evaluation. A comparison between the developed system in this research and related systems will also be presented.

Chapter 6: System Implementation and Empirical Evaluation

Chapters 4 and 5 described the development of a system based on the proposed framework for this research, where IE and NLP techniques are utilised to perform the linguistic analysis of the corpus of objectives, ILP is used to learn a grammar for writing SMART objectives and data mining techniques are applied for assessing the objectives. This chapter presents the system implementation and evaluation and is organised as follows. Section 6.1 describes the system implementation and presents a snapshot of the system interface with some illustrative objective examples. Section 6.2 presents an empirical evaluation of the developed system. Section 6.3 develops and describes another corpus of objectives and presents the results of applying the developed system on this corpus. Section 6.4 compares the developed system with the related systems. Section 6.5 gives a summary of the chapter.

6.1 System Implementation

As described before, this research develops a new framework that supports the setting of SMART objectives and aims to provide constructive feedback to employees regarding their written objectives.

In this research, the developed system based on the proposed framework has been implemented using the Java programming language (Gosling, Joy, Steele, and Bracha, 2005). In particular, the JDK (Java Development Kit)[12] (version 1.5) and NetBeans IDE[13] (version 6.8) open source software have been used for building the Java application of the developed system.

The structure of the application developed for the system is a model view controller (MVC) (Reenskaug, 2003; Curry and Grace 2008). MVC is a pattern architecture for designing GUI. It simplifies the design of the interface and facilitates the interactions between the user and the system. Moreover, it supports the process of maintaining and testing the application. MVC consists of the following three main elements:

[12] http://www.oracle.com/technetwork/java/archive-139210.html
[13] http://www.netbeans.org/

- *A Model (Business Logic),* which is concerned with representing, storing, managing and handling the data of the application domain.
- *A View (User Interface),* which manages the display of data of the application.
- *A Controller (User Input),* which interprets the user input events and converts the user interactions into actions that must be performed or modified by the model.

The user interface for the implemented application has been developed by using Java Swing (Loy et al., 2002). Java Swing is a GUI toolkit for providing graphical user interfaces for Java applications, where its components are platform independent, since they are completely written in Java. Furthermore, the MySQL[14] (version 5.5) open source database tool has been used as backend of the application to save the past historical data of sales utilised in this study in a database in order to retrieve the required information of sales from this database. As described in Chapter 5, there are two datasets of sales which have been used in the experiments conducted by the WEKA implementation of the linear regression algorithm: PC sales data and mobile sales data. Thus, a database and two tables have been created using MySQL. The first table represents the sales table, and includes the sales data of PC and mobile phones, while the second table is the products table, and includes the product types (e.g. PC, mobile).

The interface of the developed application for the system can provide good interaction between the user and the system itself. For example, the implemented system allows a user such as an employee to type a set of objective sentences on the system interface. Then, the system parses these objective sentences and assesses whether they are SMART. In addition, it provides the employee with the feedback and guidance on the written objectives in order to help him/her in formulating good objectives. In particular, the implemented system enables the employee to type an objective in each sentence text box and enter a suggested value for each of the available items and staff in the background information text boxes for items and staff. After entering and submitting an objective, the application will scan the entered objective sentence and its words in order to assess whether the required features for writing SMART

[14] http://dev.mysql.com/downloads

objectives are met for the entered objective. The assessment process of the application for the required features of the SMART objectives is based on using and applying the rules induced by the applied machine learning method and the classification rules and equations generated by the utilised data mining techniques to check whether the objectives are SMART.

Therefore, the grammar rules for SMART objectives which have been learned by ILP are implemented and used in the application in order to assess whether an objective is “specific”, “measurable”, and “time related”. In more detail, the developed system will apply the grammar rules induced by ALEPH to check whether an employee has entered some particular linguistic data patterns that are supposed to be present in an objective in order to be “measurable”, “time-related”, and “specific”. These linguistic textual patterns represent the numerical measures that indicate the amounts of sales (i.e. percent or money), timescales (i.e. dates), and the target words that are commonly utilised in writing SMART objectives, where these target words are the action verbs that describe the action taken to achieve an objective and the common nouns that identify the objective domain.

To check whether an entered objective sentence includes the action verb and the common noun required to write SMART objectives, the API of RiTa WordNet[15] is used to perform this task by accessing the WordNet database files to retrieve the needed semantic information. RiTa provides a WordNet library for Java which supports access to the WordNet lexical database to return lexical and semantic information such as synonyms, antonyms, hypernyms, hyponyms and verb-groups.

In the developed application, some fields and methods in the RiTa WordNet have been chosen to check whether the words in an objective sentence represent nouns and verbs, where the field “RiWordnet.NOUN” has been used to identify a string from noun part-of-speech and the field “RiWordnet.VERB” has been also utilised to indicate a string from verb part-of-speech. Then, the method “isVerb” is used to check whether a word in an objective sentence represents a verb. Moreover, the method “isNoun” is utilised to check whether a word in the objective sentence is a noun.

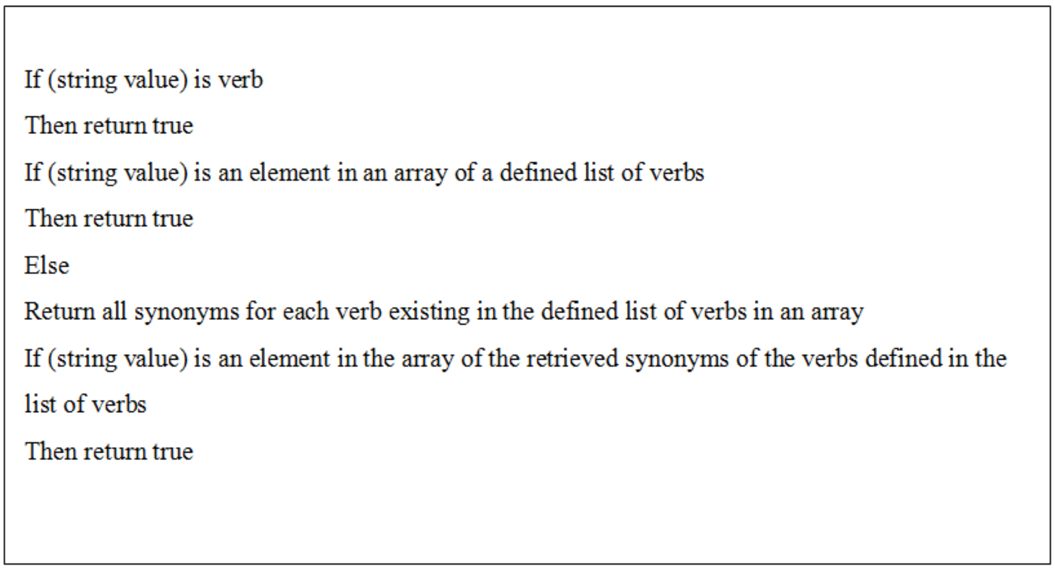
15 http://www.rednoise.org/rita/WordNet/documentation/ accessed date: 3/1/2013.

Figure 9 presents an outline of some rules that are used in the application to check whether a word in an objective sentence represents a noun and whether it is one of the target words defined in a list of nouns or one of their synonyms based on WordNet.

```
If (string value) is noun
Then return true
If (string value) is an element in an array of a defined list of nouns
Then return true
Else
Return all synonyms for each noun existing in the defined list of nouns in an array
If (string value) is an element in the array of the retrieved synonyms of the nouns defined in the
list of nouns
Then return true
```

Figure 9 An Outline of the Rules that Check If a Word in an Objective Represents a Noun

As shown in the above figure, in the case that a word in an objective represents a noun, the system will then check whether this noun is one of a set of nouns that have been defined in a created list of nouns, which includes some of the target common nouns (e.g. sales, purchases…). Otherwise, the system will check whether the noun in the objective sentence is one of the synonyms of the nouns defined in the created nouns list, by retrieving all the synonyms of the nouns in the list from WordNet in an array, then matching these synonyms with the noun in the given objective sentence. Here the method "getAllSynonyms" is used to return all the synonyms of a word. For example, if the noun in the objective sentence is "gross-sales", while the noun "sales" is one of the nouns in the created nouns list, the "getAllSynonyms" method will return "gross-revenue" and "gross-sales" as the synonyms of the noun "sales". Then, system will match the noun "gross-sales" which appears in the objective with the synonyms of the noun "sales" which are "gross-revenue" and "gross-sales". Since the noun "gross-sales" in the objective sentence has been matched with one of the synonyms of the noun "sales", then the system will return true. This means that the target word that represents the common noun has been correctly specified in the entered objective, where this noun indicates the objective domain (sales). Figure 10 presents an outline of some rules that are used in the application to check whether a word in an objective sentence represents a verb and whether it is one of the target words defined in a list of verbs or one of their synonyms based on WordNet.



```
If (string value) is verb
Then return true
If (string value) is an element in an array of a defined list of verbs
Then return true
Else
Return all synonyms for each verb existing in the defined list of verbs in an array
If (string value) is an element in the array of the retrieved synonyms of the verbs defined in the
list of verbs
Then return true
```

Figure 10 An Outline of the Rules that Check If a Word in an Objective Represents a Verb

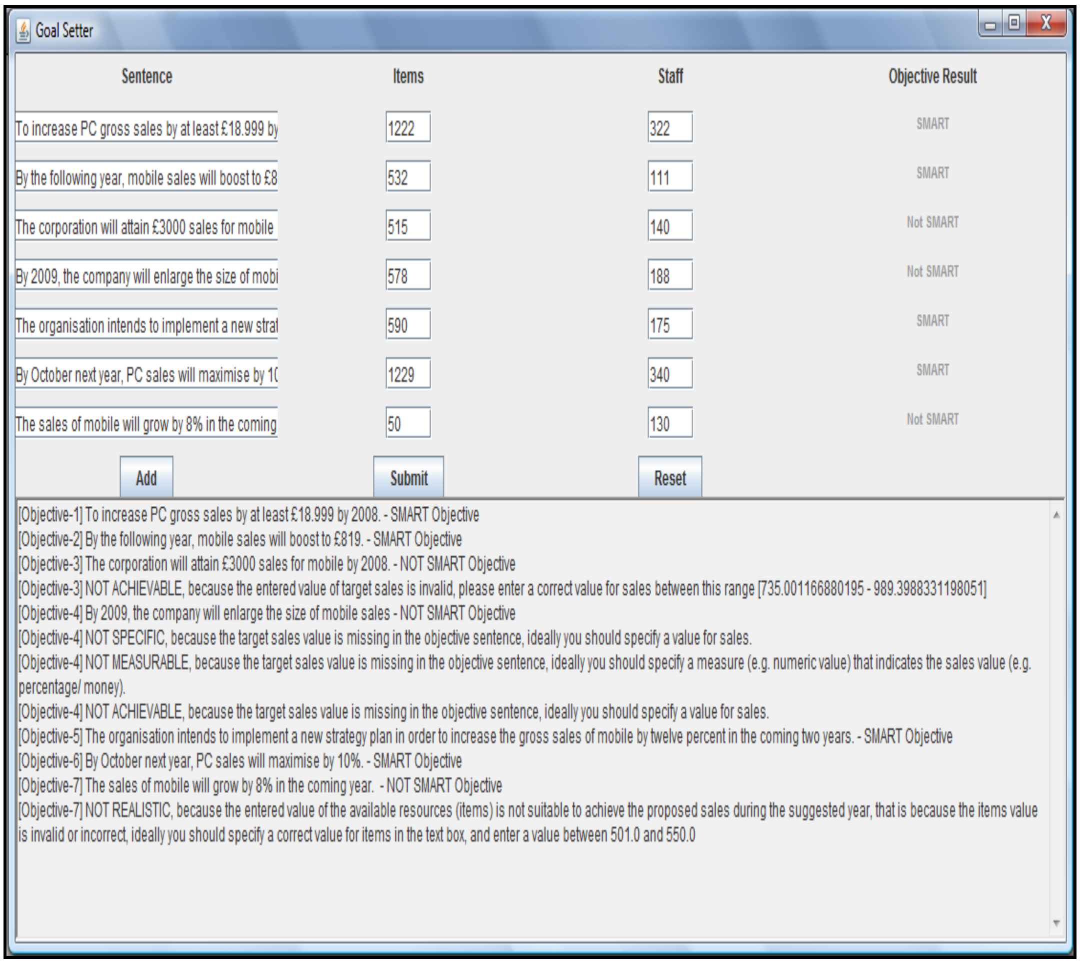
As shown it the above figure, in the case that the word in the objective sentence represents a verb, the system will then check whether this verb is one of a set of verbs that have been defined in a created list of verbs, where this list contains some of the target action verbs (e.g. increase, maximise, achieve, gain,…). Otherwise, the system will verify whether the verb in the objective sentence is one of the synonyms of the verbs defined in the created verbs list, by retrieving all the synonyms of the verbs in the list from WordNet in an array, then matching these synonyms with the verb in the entered objective sentence.

The linear regression algorithm has been implemented in Java for the developed application in order to ensure whether an objective is “achievable”. Thus, the system retrieves the data of product sales from the stored tables in the created database, in order to compute the linear regression and the least squares equations and then predict the product sales for the proposed year in a given objective. The prediction interval equation has been also implemented in Java to estimate the correct range of sales for a specific product within a given year. The implemented system will then compare the suggested amount of product sales in the objective with the predicted interval of product sales for a given year to check whether it is possible to accomplish the proposed amount of product sales within the given year in the objective sentence.

The rules induced by the JRIP rule induction algorithm are stored in an input data file. These rules are used by the application to validate the attributes: product, date, items, and staff with the provided data in the objective sentence (product and date) and the background information (items and staff) in order to classify whether an objective is “realistic”.

If all the conditions (specific, measurable, time-related, achievable, and realistic) are met for an objective, the system will then display the result of assessing this objective on the system interface, and inform the employee that he/she has entered a SMART objective. Otherwise the system will provide guidance and feedback messages to inform the employee what is missing or incorrect.

Figure 11 presents a snapshot of the system interface which includes a set of examples of objective sentences taken from the constructed corpus for this research and the results of assessing these objectives using the developed system. Additionally, the system interface presents some feedback messages regarding the written objectives.



Sentence	Items	Staff	Objective Result
To increase PC gross sales by at least £18.999 by	1222	322	SMART
By the following year, mobile sales will boost to £8	532	111	SMART
The corporation will attain £3000 sales for mobile	515	140	Not SMART
By 2009, the company will enlarge the size of mobi	578	188	Not SMART
The organisation intends to implement a new strat	590	175	SMART
By October next year, PC sales will maximise by 1(	1229	340	SMART
The sales of mobile will grow by 8% in the coming	50	130	Not SMART

```
[Objective-1] To increase PC gross sales by at least £18.999 by 2008. - SMART Objective
[Objective-2] By the following year, mobile sales will boost to £819. - SMART Objective
[Objective-3] The corporation will attain £3000 sales for mobile by 2008. - NOT SMART Objective
[Objective-3] NOT ACHIEVABLE, because the entered value of target sales is invalid, please enter a correct value for sales between this range [735.001166880195 - 989.3988331198051]
[Objective-4] By 2009, the company will enlarge the size of mobile sales - NOT SMART Objective
[Objective-4] NOT SPECIFIC, because the target sales value is missing in the objective sentence, ideally you should specify a value for sales.
[Objective-4] NOT MEASURABLE, because the target sales value is missing in the objective sentence, ideally you should specify a measure (e.g. numeric value) that indicates the sales value (e.g.
percentage/ money).
[Objective-4] NOT ACHIEVABLE, because the target sales value is missing in the objective sentence, ideally you should specify a value for sales.
[Objective-5] The organisation intends to implement a new strategy plan in order to increase the gross sales of mobile by twelve percent in the coming two years. - SMART Objective
[Objective-6] By October next year, PC sales will maximise by 10%. - SMART Objective
[Objective-7] The sales of mobile will grow by 8% in the coming year.  - NOT SMART Objective
[Objective-7] NOT REALISTIC, because the entered value of the available resources (items) is not suitable to achieve the proposed sales during the suggested year, that is because the items value
is invalid or incorrect, ideally you should specify a correct value for items in the text box, and enter a value between 501.0 and 550.0
```

Figure 11 A Snapshot of the System Interface (1)

In the above interface, the large text boxes are set for entering the objective sentences, while the small text boxes are specified for entering the background information such as the quantity of items and staff. In addition, the button "add" in the interface is for adding a new objective sentence in a new text box. The button "submit" is for sending the entered objectives and background information to be processed by the system. The button "reset" is for clearing up the entered text in the text boxes. As shown in the above figure, the system has parsed each objective sentence with the background information to indicate whether the entered objective represents a SMART objective. For example, the result of assessing the first objective sentence ("To increase PC gross-sales by at least £18.999 by 2008.") is SMART, since this objective has met all the requirements of the SMART criteria. The second objective sentence ("By the following year, mobile sales will boost to £819.") is also considered SMART. On the other hand, the result of evaluating the third objective sentence ("The corporation will attain £3000 sales for mobile by 2008.") is not SMART, since this objective is not "achievable". The reason identified for this is that the proposed amount of sales (£3000) in the given objective is not between the predicted range of the product (mobile) sales for the proposed year (2008). A feedback message appears to the employee for illustrating why the objective is not SMART and indicates the correct range of mobile sales for year 2008, which is from £735.001 to £989.398 (in Millions). The result of appraising the fourth objective sentence ("By 2009, the company will enlarge the size of mobile sales.") is not SMART, since this objective is not "measurable", not "specific" and not "achievable". The reason identified for this is that the sales amount which represents the numerical measure is missing in the objective sentence, where the "measurable" criterion requires stating a measure (sales amount) in the objective sentence for measuring the achievement of the objective outcomes. Furthermore, the "specific" criterion necessitates stating a well-defined objective sentence by specifying clearly what is to be achieved, and this implies stating some linguistic data patterns (target words and a measure) including a numeric indicator (sales amount) in the objective sentence. The "achievable" criterion involves specifying a measure (sales amount) in the objective sentence in order to compare it with the proposed date in the same objective sentence and then check whether it is possible to achieve this measure within the given date. For the fourth objective, a feedback message appears to the employee, explaining why this objective is not SMART. The system asks the employee to specify a value for the sales in the objective sentence. The fifth objective ("The organisation intends to implement a new strategy plan in order to increase the gross-sales of mobile by twelve percent in the coming two years.") and the sixth objective ("By October next year, PC sales will maximise by 10%.") are considered to be SMART. On the other hand, the seventh objective sentence ("The sales of mobile will grow by 8% in the coming year.") is not considered SMART, since this objective is not "realistic". The reason identified for this is that the entered value of the available resources (the number of items which has been specified as '50' in the background information text box) is not suitable to achieve the proposed sales amount (eight percent) during the suggested date ("coming year" which indicates here the year 2008) in the objective sentence. A feedback message appears to the employee for clarifying why the objective is not SMART. The system asks the employee to suggest a correct value for the items in the background information text box between the range [501 and 550].

The presented snapshot of the system interface has therefore demonstrated how the developed system for this research could help an employee to set SMART objectives and could support providing feedback to the employee regarding the written objectives.

6.2 An Empirical Evaluation of the Developed System

The above section demonstrated the system implementation. This section describes an empirical evaluation of the developed system.

As described in Chapter 4, a corpus of 300 objective sentences related to the domain of sales has been created to evaluate of the developed system. This corpus includes 130 SMART objective examples and 170 non-SMART objective examples, where each objective has been manually tagged with the help of some experts as SMART or non-SMART. For the empirical evaluation, the rules, which have been generated by the machine learning and data mining techniques, are applied on the created corpus of objectives (without the experts' assigned tags). That is, the developed corpus including the tags specified by the experts for the objective examples is used as a reference corpus and compared to the answers obtained for these objectives by the application of the rules generated by the ILP and the data mining techniques. This empirical evaluation has been performed to estimate the relevance of the decisions made by the applied rules concerning the classification of each objective example as SMART or non SMART.

To carry out an empirical evaluation of the quality of the grammar produced by ALEPH, 70 objectives were selected from the 300 objective examples, and ALEPH applied. As mentioned in Chapter 5, separate datasets were used to obtain the linear regression equations and the JRIP classification rules for assessing whether an objective is "achievable" and "realistic". The ALEPH rules together with the linear regression equations and the JRIP rules were then used on both the 300 objective examples and 230 objective examples (without the 70 used by ALEPH) to conduct the empirical evaluation and estimate the accuracy rate of the developed system. The accuracy rate is defined as the proportion of examples that are correctly classified (detected). Table 16 presents the empirical evaluation results on both the 300 objective examples and 230 objective examples.

Prediction Case	**All the 300 Examples**	**230 Examples**
SMART objectives detected SMART (TP)	100	30
Non-SMART objectives detected Non-SMART (TN)	150	150
SMART objectives detected Non-SMART (FN)	30	30
Non-SMART objectives detected SMART (FP)	20	20
Accuracy	**83%**	**78%**

Table 16 Empirical Evaluation on the Corpus of Objectives (Related to the Sales Domain)

In the above table, TP indicates the true positive examples, TN represents the true negative examples, FN refers to the false negative examples, and FP denotes the false positive examples. With regards to the empirical evaluation conducted on the 300 objective examples, the FP rate, that is the proportion of non-SMART objectives classified as SMART is 6.7%, and the FN rate, that is the proportion of SMART objectives classified as non-SMART is 10%. In the empirical evaluation performed on the 230 objective examples, the FP rate is 8.7%, while the FN rate is 13%.

When the results are analysed further by considering each part of the requirements, the accuracy for “specific” and “measurable” is 100%, while for the “achievable”, “realistic” and “time related” is not perfect. That is, 77% of the cases that are considered “achievable” are actually detected as “achievable”, and 88% of the cases that are considered not “achievable” are actually detected as not “achievable”. Moreover, 86% of the cases that are considered “realistic” are actually predicted as “realistic”, and 100% of the cases that are considered not “realistic” are actually detected as not “realistic”. Lastly, 88% of the cases that are considered “time-related” are actually predicted as “time-related”, and 100% of the cases that are considered not “time-related” are actually detected as not “time-related”.

Hence, the empirical evaluation on the corpus of objectives has achieved very promising results and high accuracy, since the estimated error rates of FN and FP are low. In addition, it has demonstrated experimentally the effectiveness of the developed system in assessing whether the objectives are SMART. The empirical evaluation has also shown that there are some objective sentences that have been incorrectly classified by the developed system. The following two examples illustrate two common misclassifications:

Example 9: "The company aims to generate 11% gross-sales for PCs over the next 20 months."

This example objective is SMART, but it has been detected by the system as not SMART. The feedback message that appeared when processing this example by the system was that this objective is not "achievable", not "realistic", and not "time-related" because the year is missing in the objective sentence. The problem that causes this error is that the system is not able to process some kinds of date unit formats such as months and weeks, as it is only able to process date unit formats that include a specified year (e.g. year, dd/mm/yyyy, mm/dd/yyyy, dd-MM-yyyy etc).

Likewise, the system is unable to accurately classify another form of the objective sentences such as the following:

*Example 10: "*To increase mobile sales from £1050 to £4000 by 2009."

This objective sentence is not SMART as it is not "achievable", but it has been detected by the system as SMART. The issue that causes this error is that the system only extracts the first numerical measure from the objective (*Example 10*) which represents the sales amount ('£1050') and compares this measure with the predicted range of the product (mobile) sales for the proposed year (2009). By reviewing the context of the objective sentence of *Example 10*, the system must extract the second measure of the sales amount ('£4000') and use it in the comparison with the estimated range of the product (mobiles) sales for the proposed year (2009). The estimated interval of mobile sales for 2009 is between '£805' and '£1099' (in Millions), and this means that it is not possible to achieve '£4000' during 2009 (not "achievable" objective). That is why *Example 10* should be not SMART. However, this type of objective sentences is rare in the utilised corpus (only 20 objective sentences)[16].

6.3 Applying the Developed System to Other Domains

The above section evaluated the developed system experimentally by carrying out an empirical evaluation on a corpus of objectives related to the sales domain. This section demonstrates the potential to apply the developed system to other domains and applications.

To demonstrate the effectiveness of the developed system on other domains, a second corpus of objectives has been created and used for showing the ability to apply the developed system in different domains. The second corpus represents an English textual data that consists of 150 objective sentences related to the costs domain (see Appendix C6). In particular, the second corpus includes 50 examples of SMART objectives and 100 examples of non-SMART objectives, where each objective example has been manually annotated with the help of some experts[17] as SMART or non-SMART. Similarly, the second corpus has been developed depending on some methodologies adopted for building corpora and by reviewing some SMART objective examples suggested by some authors (e.g. Carliner, 1998; Rouillard, 2003; Hurd et al., 2008; Desmond, 2011).

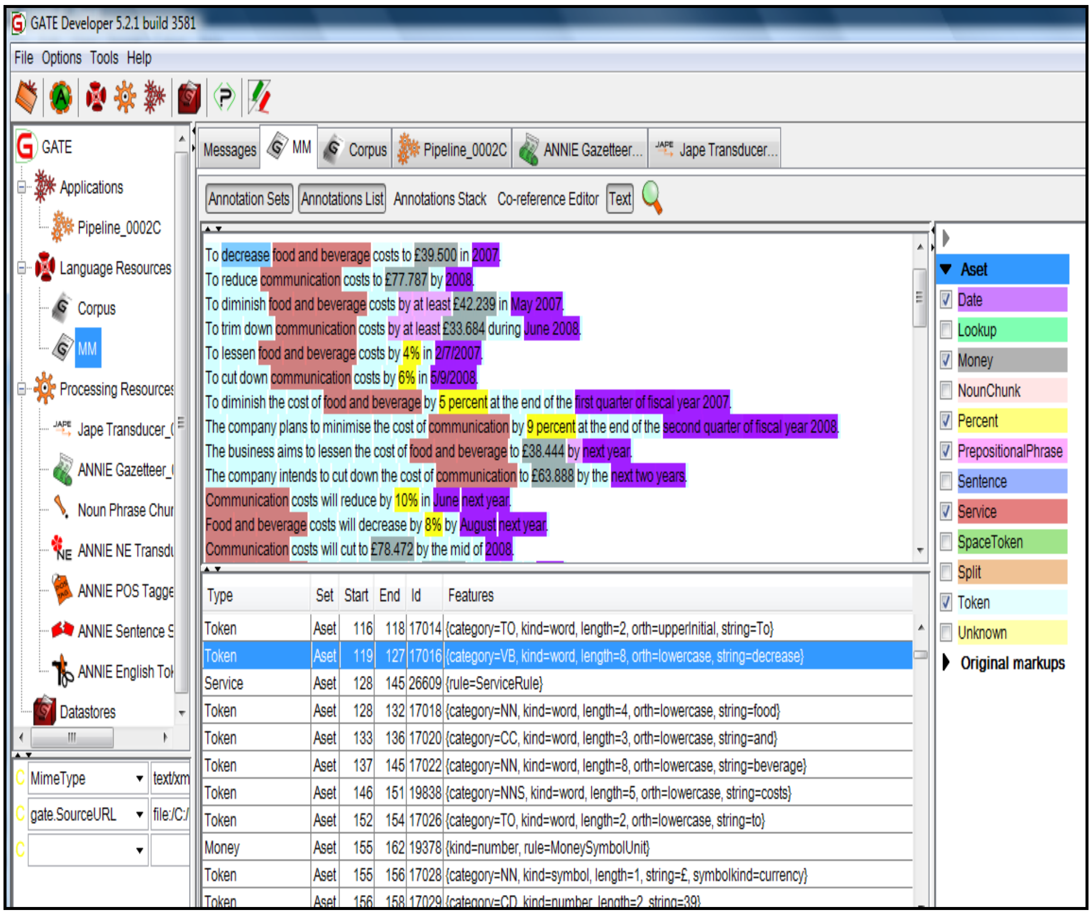
The following paragraphs describe how the system has been applied on the second corpus for assessing whether the given objectives in this corpus are SMART.

Before learning the rules and concepts of writing SMART objectives from the second corpus, this corpus includes unstructured natural language text of objectives which needs to be analysed and processed linguistically in order to be in a structured format that suits the machine learning method.

[16] A discussion on the common misclassifications of the objectives assessed by the developed system is presented in the future work section in Chapter 7.

[17] Experts from the human resource and the sales departments for the Jordan Techniques for Bio Detergent company in Jordan.

Thus, the GATE system is applied on the second corpus to process its text and provide text annotation. Particularly, the ANNIE IE system in GATE is used to perform these tasks, where some of the ANNIE processing resources (English Tokenizer, gazetteer, sentence splitter, POS tagger, NE transducer, and JAPE transducer) are arranged in a GATE pipeline with the NP Chunker plugin and then this pipeline is run over the second corpus to generate text annotation. Figure 12 presents a snapshot of the annotations generated by GATE from the second corpus.



To decrease food and beverage costs to £39.500 in 2007.
To reduce communication costs to £77.787 by 2008.
To diminish food and beverage costs by at least £42.239 in May 2007.
To trim down communication costs by at least £33.684 during June 2008.
To lessen food and beverage costs by 4% in 2/7/2007.
To cut down communication costs by 6% in 5/9/2008.
To diminish the cost of food and beverage by 5 percent at the end of the first quarter of fiscal year 2007.
The company plans to minimise the cost of communication by 9 percent at the end of the second quarter of fiscal year 2008.
The business aims to lessen the cost of food and beverage to £38.444 by next year.
The company intends to cut down the cost of communication to £63.888 by the next two years.
Communication costs will reduce by 10% in June next year.
Food and beverage costs will decrease by 8% by August next year.
Communication costs will cut to £78.472 by the mid of 2008.

Type	Set	Start	End	Id	Features
Token	Aset	116	118	17014	{category=TO, kind=word, length=2, orth=upperInitial, string=To}
Token	Aset	119	127	17016	{category=VB, kind=word, length=8, orth=lowercase, string=decrease}
Service	Aset	128	145	26609	{rule=ServiceRule}
Token	Aset	128	132	17018	{category=NN, kind=word, length=4, orth=lowercase, string=food}
Token	Aset	133	136	17020	{category=CC, kind=word, length=3, orth=lowercase, string=and}
Token	Aset	137	145	17022	{category=NN, kind=word, length=8, orth=lowercase, string=beverage}
Token	Aset	146	151	19838	{category=NNS, kind=word, length=5, orth=lowercase, string=costs}
Token	Aset	152	154	17026	{category=TO, kind=word, length=2, orth=lowercase, string=to}
Money	Aset	155	162	19378	{kind=number, rule=MoneySymbolUnit}
Token	Aset	155	156	17028	{category=NN, kind=symbol, length=1, string=£, symbolkind=currency}
Token	Aset	156	158	17029	{category=CD, kind=number, length=2, string=39}

Aset
- ✓ Date
- ☐ Lookup
- ✓ Money
- ☐ NounChunk
- ✓ Percent
- ✓ PrepositionalPhrase
- ☐ Sentence
- ✓ Service
- ☐ SpaceToken
- ☐ Split
- ✓ Token
- ☐ Unknown

Original markups

Figure 12 Typical Annotations of the Second Corpus Using GATE

As illustrated in the above figure, GATE has extracted some useful annotations from the objective sentences in the second corpus, where most of these annotations are similar to those extracted from the objective sentences in the first corpus (corpus of objectives related to the sales domain) by GATE (e.g. ‘Date’, ‘Money’, ‘Percent’, ‘Sentence’, ‘Lookup’, ‘Token’, ‘Split’, ‘SpaceToken’, and ‘NounChunk’ annotations). With regards to the ‘PrepositionalPhrase’ annotation, the JAPE grammar which was created before to extract the ‘PrepositionalPhrase’ annotations from the first corpus is applied again on the second corpus to extract the textual patterns that represent prepositional phrases. Moreover, the ‘Service’ annotation presented in Figure 12 has been extracted from the second corpus by exploiting the JAPE transducer processing resource, where a new JAPE grammar has been created to extract this annotation. Here the ‘Service’ annotation is used instead of the ‘Product’ annotation, where this type of annotation (‘Service’/‘Product’) is mainly based on the domain of objectives, since it indicates the type of product (e.g. ‘PC’ and ‘mobile’) in the first corpus, while it denotes the type of service (e.g. ‘communication’ and ‘food and beverage’) in the second corpus. The created JAPE grammar for the ‘Service’ annotation is presented in Appendix B1. The annotations extracted by GATE identify some significant linguistic patterns which can then be used by the applied machine learning method for learning the grammar rules for SMART objectives in order to ensure that the objectives are “specific”, “measurable”, and “time-related”.

As the second corpus has been annotated with different annotations (POS tags, NE annotations and some other annotations) by GATE, this corpus must be semantically analysed as well. The semantic analysis of the corpus is needed to classify the target words that are commonly utilised in writing SMART objectives and in formulating “specific” objectives. Therefore, the WordNet lexicon is applied to perform the semantic analysis of the objectives text in the second corpus, where the SR-AW software is utilised to access the WordNet database for retrieving the senses of words in the objective sentences based on the context in which they appear. In addition, the WordNet::QueryData module is used to specify the semantic classes of the target words in the second corpus based on WordNet (and on the disambiguation results of target words in the second corpus obtained by the SR-AW algorithm) in order to semantically identify the target words that are commonly used in structuring SMART objectives.

Based on WordNet, a tagset has been defined for the semantic tagging of the target words in the second corpus. This tagset includes the most generic semantic classes for the main POS categories (nouns and verbs) of the target words that appear in the second corpus. The specified semantic classes include the verb semantic class “change” and the noun semantic class “possession”.

To estimate the similarity between the results of the semantic analysis performed on the first corpus and the semantic analysis conducted on the second corpus, an experimental evaluation of the SR-AW algorithm is carried out on a data sample of 50 objective sentences which has been chosen from the second corpus. Each occurrence of a target word in this data sample has been manually annotated by a human judge with the most appropriate WordNet sense for that context. In the utilised data sample, there are 10 target words with a total of 104 word occurrences (50 verb occurrences and 54 noun occurrences). The experiments were carried out using the '*lesk*' semantic relatedness measure and a context window of size 8. The occurrences of target words (verbs, nouns, and all target words) and the accuracy achieved by the SR-AW algorithm for each POS category of the target words in the given data sample as well as the overall accuracy attained by the algorithm for all target words are presented in Table 17. The target words used in the experiments conducted by the SR-AW algorithm on the second corpus, the test instances for each target word, and the accuracy achieved by the algorithm for each target word are given in Appendix B2.

Correct Occurrences of Nouns	**Total Noun Occurrences**	**Accuracy of Nouns**
51	54	0.94
Correct Occurrences of Verbs	**Total Verb Occurrences**	**Accuracy of Verbs**
37	50	0.74
Correct Occurrences of All Target Words	**Total Occurrences of Target Words**	**Overall Accuracy of Target Words**
88	104	0.85

Table 17 Evaluation Results of WSD for the Second Corpus

With regards to the experiments conducted with WordNet::QueryData, the same data sample (50 objective sentences) of the second corpus is used again to estimate the accuracy of the semantic class classification performed by WordNet::QueryData. Furthermore, the idea also focuses on comparing the similarity in the results of semantic class classification obtained from the first corpus with those generated from the second corpus (comparing the similarity between the semantic classes of target words generated from both corpora). Each occurrence of a target word in the utilised data sample has been manually annotated by a human judge with the most correspondent WordNet semantic class for that context based on the defined tagset of the semantic classes for verbs and nouns. The occurrences of target words (verbs, nouns, and all target words) and the accuracy achieved for each POS category of the target words in the given data sample as well as the overall accuracy attained by WordNet QueryData for the semantic class classification of all target words are presented in Table 18. The target words used in the experiments performed by WordNet QueryData on the second corpus, the test instances for each target word, and the accuracy attained for each target word are given in Appendix B3.

Correct Occurrences of Nouns	**Total Noun Occurrences**	**Accuracy of Nouns**
51	54	0.94
Correct Occurrences of Verbs	**Total Verb Occurrences**	**Accuracy of Verbs**
43	50	0.86
Correct Occurrences of All Target Words	**Total Occurrences of Target Words**	**Overall Accuracy of Target Words**
94	104	0.90

Table 18 Evaluation Results of Semantic Class Classification for the Second Corpus

Hence, the semantic analysis of the target words in the second corpus based on WordNet shows that the action verbs (decrease, diminish, lessen, reduce, minimise, cut, cut down, trim down, trim), which appear in the objective sentences and are used in writing SMART objectives, have been classified into the verb semantic class "change". Furthermore, the common nouns, which appear in the objective sentences and also are used in writing SMART objectives to indicate the objective domain (e.g. cost/costs), have been classified into the noun semantic class "possession". It has been revealed that the semantic analysis results achieved from the second corpus are similar to those obtained previously from the first corpus using WordNet, SR-AW and WordNet::QueryData. To be specific, the semantic classification of the two corpora has specified the same WordNet semantic classes ("possession" and "change") of the target words (nouns and verbs) that appear in the objective sentences in both corpora, and this helps to distinguish in general the target words that are commonly utilised in formulating SMART objectives.

After processing the second corpus linguistically by GATE and WordNet, the rules of writing SMART objectives must be discovered. Therefore, this corpus is given to an ILP machine learning method to induce a grammar for structuring SMART objectives. The linguistic information of the POS and semantically tagged objective sentences in the second corpus are used as background knowledge and examples by the ILP learner ALEPH to automatically infer linguistic rules that explain the given examples. These rules can then be used to check whether an objective is “specific”, “measurable”, and “time-related”. The following is an example of an objective sentence from the second corpus:

Example 11: “The company plans to minimise the cost of communication by 9 percent at the end of the second quarter of fiscal year 2008.”

ALEPH has inferred several grammar rules from the second corpus, including the following PROLOG rule for ensuring that an objective is “specific”:

```
specific(A,B) :-
            change_action_verb_vb(A,C), determiner (C,D),
            noun_possession_nns (D,E), preposition_in (E,F),
            service (F,G), preposition_in (G,H), percent(H,B).
```

By applying the above grammar rule to *Example 11*, it is obvious that part of the sentence of *Example 11* meets the requirements in order to be “specific”.

Moreover, the following PROLOG rule is one of the induced grammar rules by ALEPH from the second corpus for ensuring that an objective is “measurable”:

```
measurable(A,B):-
            percent(A,B).
```

By applying the above grammar rule to *Example 11*, part of the sentence of *Example 11* achieves the requirements in order to be “measurable”.

The following PROLOG rule is derived by ALEPH from the second corpus for ensuring that an objective is “time-related”:

```
time_related(A,B):-
            date (A, B).
```

By applying the above grammar rule to *Example 11*, part of the sentence of *Example 11* meets the requirements in order to be “time-related”.

A validation of the grammar rules learned by ALEPH has been conducted on a data sample taken from the second corpus which includes 20 POS and semantically tagged objective sentences. The examples of “specific”, “measurable”, and “time-related” are generated from the 20 objective sentences. In total, 60 positive examples are generated (20 examples for “specific”, 20 examples for “measurable”, and 20 examples for “time-related”), and 100 negative examples are created (30 examples for “specific”, 30 examples for “measurable”, and 40 examples for “time-related”). Thus, ALEPH has generated 13 different grammar rules (10 rules for “specific”, 2 rules for “measurable”, and 1 rule for “time-related”) from the utilised data sample (background knowledge and examples) to formulate SMART objectives, since all the positive examples have been covered by the generalised rules. All the rules learned by ALEPH from the second corpus represent consistent hypotheses and can be utilised to check whether an objective is “specific”, “measurable”, and “time-related”. The background knowledge file describing the predicates of the words in the objective sentences that appear in the second corpus is presented in Appendix C3, while the created sets of examples used by ALEPH are given Appendix C4. The rules learned by ALEPH from the second corpus are given in Appendix A2.

To ensure a fair test for the induced rules, some parameter settings and noise data (here the noise parameter was set to 20 to accept at maximum 20 negative examples to be covered by the generalised clauses) have been utilised in the experiments performed by ALEPH (see background knowledge file for the second corpus in Appendix C3). Moreover, a validation of the grammar rules learned by ALEPH is carried out on the created sets of 60 positive examples and 120 negative ones using 10-fold cross-validation to estimate the quality of the obtained rules. Table 19 presents the proportion of the training and testing datasets used in the conducted evaluation for the second corpus and the accuracies achieved by ALEPH for the training and testing datasets using 10-fold cross-validation.

	Proportion	**Average Accuracy**
Training Dataset	90%	87%
Testing Dataset	10%	85%

Table 19 Cross Validation Results (2)

The validation of the quality of the grammar rules learned by ALEPH from the second corpus has therefore achieved promising results, such that the standard deviation obtained through the 10-fold cross-validation test is 0.00699.

By comparing the grammar rules learned by ALEPH from the second corpus with those also obtained by ALEPH from the first corpus, it is evident that the derived rules from both corpora are similar. To be specific, all the grammar rules generated from both corpora for ensuring that an objective is "measurable" and "time-related" are the same. However, the grammar rules induced from the first corpus for ensuring that an objective is "specific" are similar to those learned from the second corpus for also ensuring that an objective is "specific". In particular, there is only one predicate that is different in these rules, which is called the "product" predicate in the rules derived from the first corpus, while is called the "service" predicate in the rules learned from the second corpus. This difference is dependent on the domain of application for the given objectives. In addition, the number and diversity of the grammar rules learned by ALEPH from the first corpus for ensuring that an objective is "specific" are more than those inferred by ALEPH from the second corpus. This is based on the forms of the objective sentences in the given corpus and the utilised action verbs in the objectives. For example, it has been noticed that the grammar rules generated by ALEPH from the first corpus for ensuring that an objective is "specific" include different verb semantic classes, e.g. "change", "social", "possession", and "creation". On the other hand, the grammar rules learned by ALEPH from the second corpus for ensuring that an objective is "specific" include only the verb semantic class called "change". In general, it was shown that the verb semantic class "change" classifies most the action verbs used in the objective sentences in both corpora.

Thus, ALEPH has inferred novel and general rules from both corpora which define a grammar for writing SMART objectives, such that these grammar rules could be applied to any objective sentence from the other domains to ensure that this objective is “specific”, “measurable”, and “time-related”.

After the grammar rules for writing SMART objectives have been learned from the objectives in the second corpus to ensure that the objectives are “specific”, “measurable”, and “time-related”, these objectives must be assessed whether they could be met given the available resources and time. Therefore, prediction and classification algorithms in data mining are utilised for assessing whether the objectives in the second corpus are “achievable” and “realistic”.

To check whether an objective is “achievable”, the WEKA implementation of the linear regression algorithm has been applied on a past historical data of passenger airline cost[18], obtained from the Infochimps repository. This dataset includes the costs information of different services offered by an airline association from 1990 to 2006. In this study, the cost data for each of the communication and food and beverage services is chosen from this dataset for the experiments.

The applied linear regression algorithm utilises the historical data of costs for a specified service (communication/food and beverage) to predict the expected service cost for a given timeframe. The developed system then compares the value of the service cost suggested in an objective with the estimated one for a given timeframe and assesses whether the value of service cost in the objective is within the predicted cost interval of the service for the specified timeframe.

Two experiments have been carried out using the linear regression algorithm in the WEKA toolkit for ensuring that an objective is “achievable” by checking whether a suggested value of a service cost in an objective sentence is possible to be accomplished within a proposed timeframe in the objective. The first experiment of linear regression has been performed on the food and beverage data, while the second experiment has been conducted on the communication data. The cost data for each of food and beverage and communication services is presented in Appendix B4 and B5 respectively. Moreover, the classifier model of the linear regression formula which

[18] http://www.infochimps.com/datasets/passenger-airline-cost-indexes-1990-to-2006, Source: Air Transport Association of America, Washington, DC, U.S. Airline Cost Index

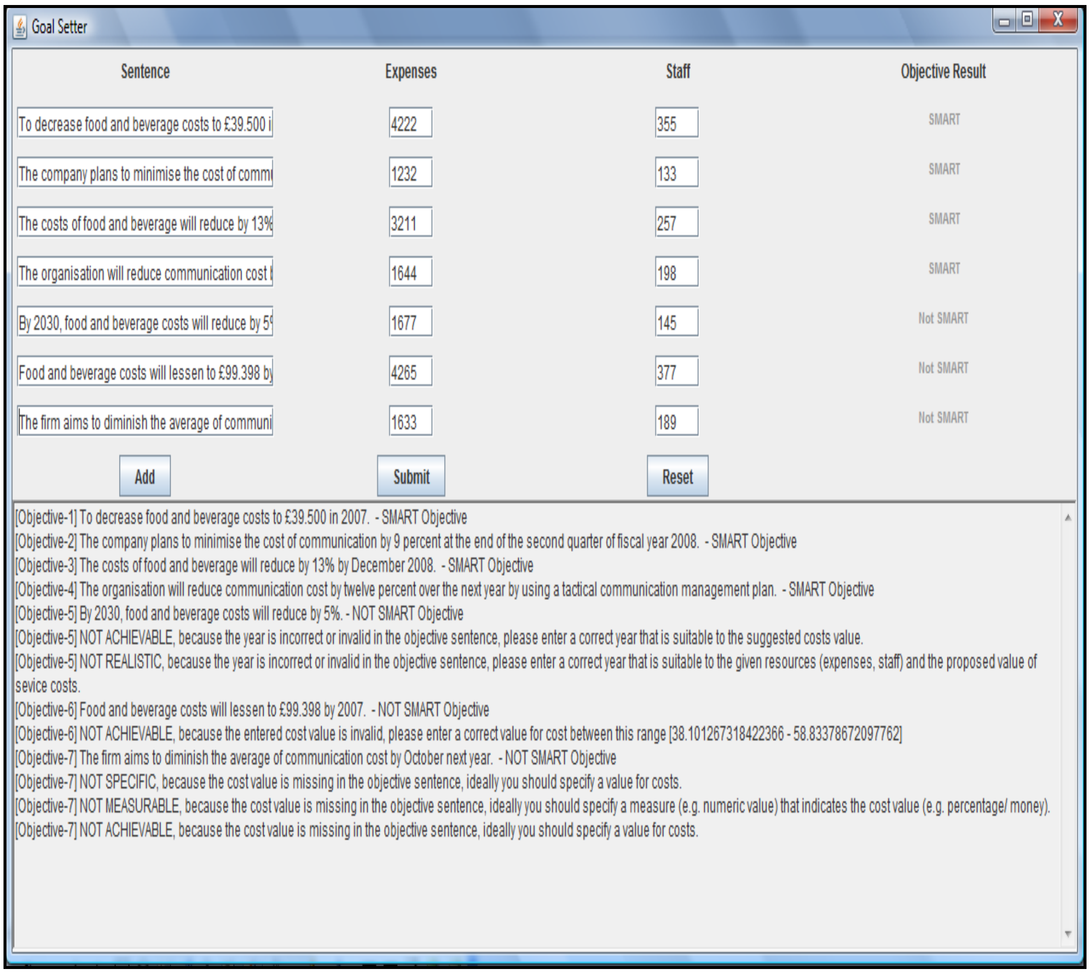
has been generated by the linear regression algorithm in WEKA for each of the food and beverage and communication datasets is presented in Appendix B6 and B7 respectively. The performance of the linear regression model has been evaluated in WEKA on the whole training datasets of food and beverage and communication services, where the model has achieved a correlation coefficient of 98% and 91% for each dataset respectively (and the root mean square error is 2.603 and 5.746 for each dataset respectively).

Hence, as the above illustrates, if it is possible to estimate parameters such as sales or costs, then the framework facilitates their use in assessing whether an objective is “achievable”.

To check whether an objective is “realistic”, the JRIP algorithm which is the WEKA implementation of RIPPER has been applied on a dataset of randomly selected examples created by using some suggested rules that describe some information about two services (food and beverage and communication). The suggested rules for the dataset which is used by JRIP in the experiments are given in Appendix B8.

For the experiments, the created dataset is given to WEKA, since it contains some attributes such as “expense”, “staff”, “service”, and “year”, while the target class attribute is named as “realistic”. This dataset includes 393 positive examples and 440 negative examples. By using the WEKA toolkit, a classifier model of JRIP has been built and 7 JRIP classification rules have been induced from the given dataset. The rules induced by the JRIP model from the utilised dataset are presented in Appendix B9. The performance of the JRIP model has been estimated using 10-fold cross-validation in WEKA, and achieved a classification accuracy of 98%, where 820 examples of a total of 833 examples have been correctly classified by the model.

Thus, the system has used all the grammar rules generated from the second corpus by the applied machine learning method together with the equations and the classification rules generated by the utilised data mining techniques to assess whether the written objectives are SMART. Figure 13 presents a snapshot of the system interface which includes a set of examples of objective sentences taken from the second corpus and the results of assessing these objectives using the developed system as well as the feedback messages on the written objectives.



Sentence	Expenses	Staff	Objective Result
To decrease food and beverage costs to £39.500 i	4222	355	SMART
The company plans to minimise the cost of comm	1232	133	SMART
The costs of food and beverage will reduce by 13%	3211	257	SMART
The organisation will reduce communication cost t	1644	198	SMART
By 2030, food and beverage costs will reduce by 5	1677	145	Not SMART
Food and beverage costs will lessen to £99.398 by	4265	377	Not SMART
The firm aims to diminish the average of communi	1633	189	Not SMART

```
[Objective-1] To decrease food and beverage costs to £39.500 in 2007.  - SMART Objective
[Objective-2] The company plans to minimise the cost of communication by 9 percent at the end of the second quarter of fiscal year 2008.  - SMART Objective
[Objective-3] The costs of food and beverage will reduce by 13% by December 2008.  - SMART Objective
[Objective-4] The organisation will reduce communication cost by twelve percent over the next year by using a tactical communication management plan.  - SMART Objective
[Objective-5] By 2030, food and beverage costs will reduce by 5%. - NOT SMART Objective
[Objective-5] NOT ACHIEVABLE, because the year is incorrect or invalid in the objective sentence, please enter a correct year that is suitable to the suggested costs value.
[Objective-5] NOT REALISTIC, because the year is incorrect or invalid in the objective sentence, please enter a correct year that is suitable to the given resources (expenses, staff) and the proposed value of
sevice costs.
[Objective-6] Food and beverage costs will lessen to £99.398 by 2007.  - NOT SMART Objective
[Objective-6] NOT ACHIEVABLE, because the entered cost value is invalid, please enter a correct value for cost between this range [38.101267318422366 - 58.83378672097762]
[Objective-7] The firm aims to diminish the average of communication cost by October next year.  - NOT SMART Objective
[Objective-7] NOT SPECIFIC, because the cost value is missing in the objective sentence, ideally you should specify a value for costs.
[Objective-7] NOT MEASURABLE, because the cost value is missing in the objective sentence, ideally you should specify a measure (e.g. numeric value) that indicates the cost value (e.g. percentage/ money).
[Objective-7] NOT ACHIEVABLE, because the cost value is missing in the objective sentence, ideally you should specify a value for costs.
```

Figure 13 A Snapshot of the System Interface (2)

To evaluate the effectiveness of the developed system on other domains, an empirical evaluation has been carried out on the second corpus. This empirical evaluation has been conducted after applying the developed system on the second corpus, which consists of 50 SMART objectives and 100 non-SMART objectives. In particular, the ALEPH grammar rules generated from the second corpus together with the linear regression equations and the JRIP rules were then used on both the 150 objective examples and 130 objective examples (without the 20 used by ALEPH) to conduct the empirical evaluation and estimate the system accuracy. The results obtained through the performed empirical evaluation on both the 150 objective examples (the second corpus) and 130 objective examples are presented in Table 20.

Prediction Case	All the 150 Examples	130 Examples
SMART objectives detected SMART (TP)	35	15
Non-SMART objectives detected Non-SMART (TN)	80	80
SMART objectives detected Non-SMART (FN)	15	15
Non-SMART objectives detected SMART (FP)	20	20
Accuracy	**77%**	**73%**

Table 20 Empirical Evaluation on the Second Corpus (Related to the Costs Domain)

Hence, the empirical evaluation on the second corpus has also attained very encouraging results. This empirical evaluation has shown experimentally the efficiency of the developed system in assessing whether the objectives are SMART as well as the ability to apply the developed system in other domains and applications.

6.4 Comparing the Developed System with Related Systems

The literature review presented in this research illustrated the importance of performance appraisal systems and their challenges. It also described the motivations of the goal setting process and the SMART approach for setting good and effective objectives. This thesis has shown how the developed system can support employee performance appraisals by facilitating the setting of SMART objectives and providing feedback based on the use of ILP and data mining techniques.

In general, performance appraisal has gained interest over the past few years and there are few studies that show interest in solving some problems of performance appraisal by applying automatic methods such as data mining methods for making significant decisions and classifying future data instances. For example, the study presented by Jantan et al. (2010) applies classification techniques in data mining to build a classification model which can be used for predicting the performance of employees in order to identify whether an employee is recommended for promotion or not based on his/her performance. Furthermore, the recent study proposed by Al-Radaideh and Al Nagi (2012) has also used data mining techniques to predict the performance of employees. This study utilises decision trees to build a classification model by using three datasets: personal information dataset (attributes: age, gender, marital status and number of kids), education information dataset (attributes: university type, general specialisation, degree and grade) and professional information dataset (attributes: number of experience years, job title, rank and salary). The constructed classification model classifies the most important factors (attributes) that might have the highest effect on the employee performance such as job title.

As well as to these studies, there are some commercial systems for performance appraisals and goal setting that have been developed such as Halogen eAppraisal[19], GoalsOnTrack[20] and Lifetick[21]. The Halogen eAppraisal software is a performance appraisal system that allows managing the progress of employee performance appraisal process by reviewing the employee appraisal reports as well as tracking the progress of employees' goals using goals report. Goal setting software such as GoalsOnTrack enables the user to write down his/her goals (or objectives) and facilitates the process of checking whether the goals are on track by tracking the progress of achieving the goals. The Lifetick software is a web based system which allows the user to create a set of goals (or objectives), tasks and reminders as well as enables him/her to track the progress of goals and tasks.

Even though the GoalsOnTrack and Lifetick systems support different tasks for setting goals and objectives, they do not have the ability to automatically assess whether an objective is SMART and do not have the potential for checking whether the resources and time are available to achieve the objective. Moreover, they do not perform any processing or analysing for the objectives text to automatically check the presence of the target words that are commonly used in writing SMART objectives as well as the presence of measures and dates in the objective sentences. Both systems assume that the user has knowledge about the SMART criteria, since a user of these systems has to ensure that an objective is SMART by himself/herself. For instance, the Lifetick software asks the user if he/she considers his/her objective as "specific", "measurable", "achievable", "relevant", and "time-specific", and prompts him/her to tick the appropriate box for each element of the SMART criteria. In the GoalsOnTrack system, the user must follow the goal creation form instructions to create a SMART objective. However, this system is not able to assess whether a

[19] http://www.halogensoftware.com/products/halogen-eappraisal/
[20] http://www.goalsontrack.com
[21] http://lifetick.com/

written objective is SMART, since the assessment process of the objective depends entirely on the user himself/herself.

Accordingly, the current commercial systems for setting objectives mainly focus on tracking objectives and do not support deeper aspects of the goal setting process such as assessing whether the objectives are SMART and providing feedback on the objectives.

In contrast, the developed system for this research is based on a novel framework that aims to support the writing of SMART objectives and providing feedback by performing the following tasks and contributions:

- Analyse and process the written text of an objective to automatically check whether the essential elements for formulating SMART objectives are specified in the objective (e.g. target words, measures and dates).
- Clarify the words that should be used when writing SMART objectives.
- Predict the objective outcomes (e.g. amount of sales or costs) and estimate the required resources that must be allocated to achieve the given objective based on past historical data.
- Automatically check whether an objective could be accomplished within the given resources and time based on past experience.
- Automatically assess whether the written objectives are SMART.
- Provide guidance and constructive feedback on structuring the objectives.

Thus, the work done in this research is novel and complements previous research and existing commercial systems.

Table 21 presents a comparison between the developed system for this research and the GoalsOnTrack and Lifetick systems for objectives setting by showing the main characteristics of each system:

System Characteristics	GoalsOnTrack	Lifetick	The Developed System
Enable a user to create objectives	√	√	√
Automatically assess whether the written objectives are SMART	X	X	√
Process the written objectives text to automatically check whether an objective includes the required features in order to be “specific”, “measurable” and “time-related”	X	X	√
Automatically check whether an objective could be achieved within the given resources and time	X	X	√
Clarify the words that should be used when writing SMART objectives	X	X	√
Predict the objective outcomes and the resources that must be allocated to achieve the objective	X	X	√
Provide constructive feedback regarding the written objectives	X	X	√
Track the progress of the objectives	√	√	X

Table 21 The Main Characteristics of the Developed System and Related Systems

6.5 Summary

In this chapter, a description of the system implementation was given. An explanation of some illustrative examples of objective sentences which have been assessed by the developed system for this research was presented in order to demonstrate how this system can be used to automate the process of setting SMART objectives and providing feedback based on AI techniques. The chapter included two empirical evaluations. First, an empirical evaluation for objectives from the sales domain was carried out. This included developing a corpus of objectives from the sales domain, and then applying the system on this corpus. The overall accuracy obtained for this corpus was 83%, with 33% TP rate and 50% TN rate. To test the generality of the framework, a second empirical evaluation for objectives from the costs domain was preformed. This included constructing another corpus of objectives from the costs domain, and then applying the system on this corpus. The overall accuracy achieved for the second corpus was 77%, with 23.3% TP rate and 53.3% TN rate. A comparison of the rules generated for both evaluations suggests that the approach is generic in terms of the checking whether the objectives are “specific”, “measurable” and “time-related”. The ability to assess if the objectives are “achievable” and “realistic” depends on the domain and prediction/classification methods used as well as the availability of data. Where this is not possible, this part of the framework will not, of course, be applicable. A comparison between the developed system and related systems was given to show the novelty and the main contributions of the developed system for this research by highlighting its efficiency and effectiveness in setting SMART objectives and providing feedback.

In the next chapter, a conclusion that summarises the achievements made in this research and how the research objectives have been addressed will be presented including the results obtained through the experimental evaluation of ILP and data mining methods as well as the results of the two empirical evaluations of the developed system. Furthermore, a summary of the limitations of the framework and directions for future work will also be proposed.

Chapter 7: Conclusions and Future Work

Employee performance appraisal is a vital part of an organisation's human resource strategy for assessing the performance of employees and making significant decisions regarding promotions, salary administration, and identification of training needs. Even though performance appraisal has advantages for organisations, it also has some challenges which make many employees in organisations to consider it less effective and based on unfair and subjective performance evaluations. These perceptions of the appraisal process can influence employees' performance, job satisfaction, productivity and ultimately their commitment to an organisation.

This thesis explores the use of natural language processing, machine learning and data mining techniques to support employee appraisal and goal setting systems by developing a new framework for employee performance appraisals. The main aim of the developed framework is to facilitate the setting of SMART objectives and providing feedback on objectives by automating the process of assessing whether the written objectives are SMART based on the use of ILP and data mining techniques.

In this final chapter, a summary of how the research objectives have been addressed is given Section 7.1 with the achievements of this research. The limitations of the developed framework and directions for future work are presented in Section 7.2.

7.1 The Research Objectives Revisited

This section presents the research objectives and reviews the extent to which they have been achieved.

1. **To develop a novel framework for supporting employee performance appraisal systems and the goal setting process so that employees are able to write SMART objectives:** A new semantic framework for supporting performance appraisals was proposed in Chapter 3. The proposed framework supports the automation of the process of assessing whether objectives are SMART and providing feedback. The research demonstrated that the process of setting SMART objectives can be supported by first discovering the rules of

writing SMART objectives and then assessing whether these objectives can be met. Therefore, research on what constitutes well written SMART objectives was first carried out and some real examples of SMART objectives were also surveyed in Chapter 2. Moreover, a comprehensive literature review of the most appropriate methods that could be employed to find the rules of structuring SMART objectives and to assess these objectives was performed. Machine learning and data mining techniques were used to develop a novel framework for setting SMART objectives and providing feedback. In particular, the framework utilises ILP for learning the rules for writing SMART objectives and applies classification and prediction techniques for assessing the objectives. Chapters 4, 5 and 6, the core of this thesis, present the development of a system that illustrates the viability of the proposed framework.

2. **To explore and assess the use of ILP and data mining techniques for developing a novel system and framework that support performance appraisals and goal setting to assess whether the objectives are SMART and provide feedback on the objectives:** Before ILP was used to learn the rules of writing SMART objectives, a corpus of 300 objective sentences was created to evaluate the developed system based on the proposed framework. The methodology adopted for constructing a corpus of objectives was presented in Chapter 4. A linguistic analysis of the corpus of objectives was performed by using the GATE system. A set of useful annotations was extracted based on a linguistic analysis of the corpus by using GATE. Then, a semantic analysis of the corpus of objectives was carried out using WordNet, where the SR-AW and WordNet::QueryData software were employed to semantically annotate the target words in the developed corpus with their WordNet semantic classes based on the context in which they occur. Chapter 4 described that the performed semantic analysis of the corpus helped to identify the target words that are commonly used in writing SMART objectives and in formulating “specific” objectives. A grammar for writing SMART objectives was automatically learned from a set of POS and semantically tagged objective sentences by using ILP. In particular, the ILP learner ALEPH was able to derive a set of novel and general grammar rules for SMART

objectives, where all of the generalised rules were successfully applied for checking whether the written objectives are “specific”, “measurable”, and “time-related”. A validation of the quality of the rules learned by ALEPH was conducted using a 10-fold cross-validation test on 210 positive examples and 400 negative ones. In each validation, 90% was used for training and 10% was used for testing. The performed 10-fold cross-validation test of ALEPH achieved an accuracy of 92% for the training data and 89% for the testing data. A description of how the ILP inductive learner ALEPH can be utilised to learn a grammar for writing SMART objectives from a corpus of objectives was given in Chapter 5. Chapter 5 also included a description of some of the induced grammar rules by ALEPH for ensuring that the objectives are “specific”, “measurable”, and “time-related”. The use of a prediction data mining algorithm for assessing whether an objective is “achievable” was presented in Chapter 5. Specifically, the WEKA implementation of the linear regression algorithm was used to predict the expected product sales for a given timeframe in order to check whether a suggested amount of product sales in an objective could be achieved within a proposed timeframe. The performance of the linear regression model was evaluated in WEKA on past historical datasets of PC and mobile sales, and achieved good results. The use of a classification rule induction algorithm for checking whether an objective is “realistic” was illustrated in Chapter 5. In particular, the WEKA implementation of the RIPPER rule induction algorithm (JRIP) was successfully applied for inducing some rules that classify whether there are sufficient resources (i.e. staff and items) to achieve the proposed amount of product sales in an objective during a given timeframe. Thus, the thesis demonstrates that ILP can be successfully used to learn a grammar for assessing whether objectives are “specific”, “measurable” and “time-related”. The use of data mining methods for classification and predication can be useful but the level of success will depend on the domain of application and the availability of data.

3. **To evaluate the framework by applying it to a particular domain and reviewing the outcomes in terms of system accuracy:** A system was developed in Java to test the viability of the framework and test examples that show its capability of assessing whether an objective is SMART and its ability

for providing feedback were presented in Chapter 6. The feasibility of the framework was explored and demonstrated in the domain of sales, where a corpus of objectives related to the domain of sales was used to evaluate the efficiency of developed system based on the framework. An empirical evaluation of the developed system on the corpus of objectives was given in Chapter 6, where the results obtained are promising with an overall accuracy of 83%. The developed system has shown experimentally its efficiency and effectiveness in assessing whether the written objectives are SMART and in providing constructive feedback to employees regarding the written objectives. It was demonstrated in Chapter 6 that the developed system can be applied to different domains, where a second empirical evaluation for objectives from the costs domain was conducted and achieved an overall accuracy of 77%. A comparison of the rules generated by ILP from both corpora of objectives of the two domains (sales and costs) was given in Chapter 6. The comparison demonstrates that the rules learned by ALEPH from both corpora for ensuring that the objectives are "measurable" and "time-related" are the same. However, the rules learned for ensuring that the objectives are "specific" are similar, where there is only one predicate that is different in these rules and this difference is dependent on the objectives domain (see Section 6.3). A comparison between the developed system and related systems was presented in Chapter 6. The comparison reveals the novelty and the main contributions of the developed system in this thesis over the existing systems and these contributions include: (i) the ability of the developed system to help employees by automatically assessing whether the written objectives are SMART and providing feedback on the written objectives, (ii) predicting the objective outcomes and the resources that must be allocated to achieve the objectives, and (iii) automatically checking whether an objective could be achieved within the given resources and time.

7.2 Limitations and Future Work

The use of ILP and data mining techniques in this research for developing a framework for SMART objective setting is one of the first of its kind and inevitably, there are several aspects that could be developed in the future, including the following:

1. **Generality of the framework:** Although the developed framework was applied successfully for setting SMART objectives in two different domains (i.e. sales and costs), there are some limitations of this framework, and in particular, there are some domains in which this framework will not be appropriate. In analysing the generality of the framework in Chapter 6, it was interesting that the rules produced by ILP were sufficiently general but the predictions needed for assessing whether an objective was “achievable” and “realistic” were dependent on the domain and availability of data. To address this in the future, more research is required on the type of domains where such predictability would be possible. So for example, in the academic domain, the task of assessing the number of papers that can be published by a member of the academic staff during a particular year is a complex research problem in its own right and depends on the discipline, stage of career, quality and resources available.
2. **Improving the empirical evaluation of the system**: The performed empirical evaluation of the system in this thesis was very useful; however, if time allowed, it could have been followed up with a user evaluation. A user evaluation can be undertaken by enabling a set of employees in a set of organisations to test the efficiency of developed system and investigate its usefulness by themselves. The idea here is to ask these employees to use the developed system for writing and setting business objectives in order to get their comments and opinions regarding the capability of the developed system in setting SMART objectives and providing feedback. In this type of evaluation, survey questionnaire templates could be utilised and distributed to be filled by these employees, who have already used the developed system for setting business objectives, in order to know the extent to which employees are satisfied with the developed system and to judge the effectiveness of this system.
3. **Enhancing the system accuracy and completeness:** As mentioned before, a corpus of objectives has been developed for this research to evaluate the developed system. This corpus is the first constructed for the problem of setting SMART objectives. One of the potential avenues for carrying out

further experiments of the developed system is to improve the corpus size and then repeat the experiments and the empirical evaluation of the developed system. Furthermore, the accuracy rate of the developed system could be potentially enhanced by performing further improvements on the system implementation. For example, it was demonstrated in Chapter 6 that there are some objective sentences that have been classified incorrectly by the developed system such as the following (from Chapter 6):

Example 9: “The company aims to generate 11% gross-sales for PCs over the next 20 months.”,

Example 10: “To increase mobile sales from £1050 to £4000 by 2009.”

The misclassification of sentences such as the one in *Example 9* could be solved by enabling the system to process more date formats (e.g. months, weeks, days) and using functions that convert months, weeks and days to years. The misclassification of sentence such as *Example 10* could be addressed by enabling the system to extract the second pattern of the proposed numerical measure in this example instead of extracting the first one, and then compare this measure with the predicted range of the product sales for the proposed year.

4. **Improving the system’s functionalities and characteristics**: As mentioned in the motivation presented in Chapter 1, there are some useful characteristics that could support performance appraisal and goal setting systems and improve their capabilities. Some of these characteristics were addressed and explored in this thesis, while others were left unaddressed due to a lack of time available for the thesis. One of these unaddressed aspects which is worth investigating is to support organisations in planning business strategy and evaluating the future risk factors of a business based on the suggested objectives. This aspect is based on using some effective business objectives in addressing the issue of supporting the business plans for organisations. Here classification and prediction data mining techniques could be used to address

this aspect and extend the developed framework presented in this thesis by using some useful data from a set of business objectives, which have been assessed by developed system as SMART, to predict the organisation's business future and forecast the future business risk. In other words, the specified objectives will help to guide the business decision making process by creating accurate and strategic decisions regarding the business strategy and therefore avoiding the future problems. Another important aspect presented in the motivation of this research is to measure the progress of a company and its employees by evaluating the progress of achieving the organisational goals. As described in Chapter 6, there are some commercial systems for goal setting that enable users to track the progress of their goals and objectives. The work presented in this thesis could be extended by adding the characteristic of tracking and measuring the progress of achieving the written objectives. It has been noticed that the characteristic of tracking the progress of the objectives in the current systems for goal setting is done manually by the user himself/herself. This characteristic could be supported by automating the process of tracking the progress of achieving objectives based on using data mining methods to estimate the extent to which the objectives are achieved and completed.

In conclusion, this research has proposed a new framework for employee appraisal that is based on the use of ILP and data mining to support the process of setting SMART objectives and providing feedback. A system was developed to show the feasibility of the approach and an empirical evaluation on two domains shows good results and potential. A survey and comparison with other systems suggests, that to the best of the author's knowledge, this is the first attempt to utilise NLP, IE, ILP and data mining to support this important application in business, and the author hopes that this provides a good basis for future research.

References

Agirre, E. and Edmonds, P. G. (Eds.) (2006). Word Sense Disambiguation: Algorithms and Applications. Vol.33. Springer, 364 pages.

Aitken, J. S. (2002). Learning Information Extraction Rules: An Inductive Logic Programming Approach. In Proceeding of the 15th European Conference on Artificial Intelligence, pp. 355-359.

Alon, I., Qi, M., and Sadowski, R. J. (2001). Forecasting Aggregate Retail Sales: a Comparison of Artificial Neural Networks and Traditional Methods. Journal of Retailing and Consumer Services, Vol. 8, No. 3, pp. 147-156.

Alpaydin, E. (2010). Introduction to Machine Learning. 2nd Edition. The MIT Press, 584 pages.

Al-Radaideh, Q. A., and Al Nagi, E. (2012). Using Data Mining Techniques to Build a Classification Model for Predicting Employees Performance. International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 3, No.2, pp. 144 - 151.

Anderson, D. R., Sweeney, D. J. and Williams, T. A. (2005). Statistics for Business and Economics. 9th Edition, South–WesternCollege, 1023 pages.

Antoni, C. (2005). Management by Objectives–an Effective Tool for Teamwork?. The International Journal of Human Resource Management, Vol. 16, No. 2, pp. 174-184.

Appelt, D. E. (1999). An Introduction to Information Extraction. Artificial Intelligence Communications, IOS Press, Vol. 12, No. 3, pp. 161-172.

Aqel, D., and Vadera, S. (2010). A Framework for Employee Appraisals Based on Sentiment Analysis. In Proceedings of the 1st International Conference on Intelligent Semantic Web – Services and Applications (ISWSA2010). ACM, Article 8, pp. 62-67.

Aqel, D., and Vadera, S. (2013a). A Framework for Employee Appraisals based on Inductive Logic Programming and Data Mining Methods. In Proceeding of the 18th International Conference on Applications of Natural Language to Information Systems (NLDB2013). Springer, Berlin Heidelberg, pp. 404-407.

Aqel, D., and Vadera, S. (2013b). Learning a Grammar for SMART Objectives Using Inductive Logic Programming. In Proceeding of the 5th European Conference in Intelligent Management Systems in Operations (IMSIO2013). The Operational Research Society, pp. 66-76.

Arias, M. and Khardon, R. (2004). Bottom-up ILP Using Large Refinement Steps. In Inductive Logic Programming, Springer Berlin Heidelberg, pp. 26-43.

Arvey, R. D. and Murphy, K. R. (1998). Performance Evaluation in Work Settings. Annual Review of Psychology, Vol. 49, No. 1, pp. 141-168.

Baldridge, J. (2005). The OpenNLP Project. URL: http://opennlp.apache.org/index.html, (accessed 2 February 2012).

Banerjee, S., and Pedersen, T. (2003). Extended Gloss Overlaps as a Measure of Semantic Relatedness. In International Joint Conference on Artificial Intelligence, pp. 805-810.

Barrick, M. R., Mount, M. K., and Strauss, J. P. (1993). Conscientiousness and Performance of Sales Representatives: Test of the Mediating Effects of Goal Setting. Journal of Applied Psychology, Vol. 78, No. 5, pp. 715-722.

Beer, M. (1982). Performance Appraisal: Dilemmas and Possibilities. Organizational Dynamics, Vol. 9, No. 3, pp. 24-36.

Berger, J., Harbring, C. and Sliwka, D. (2013). Performance Appraisals and the Impact of Forced Distribution-An Experimental Investigation. Management Science, Vol. 59, No. 1, pp. 54-68.

Berger, A. L., Pietra, V. J. and Pietra, S. A. (1996). A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics, Vol. 22, No. 1, pp. 39-71.

Bianco, V., Manca, O., and Nardini, S. (2009). Electricity Consumption Forecasting in Italy Using Linear Regression Models. Energy, Vol. 34, No. 9, pp. 1413-1421.

Bipp, T. and Kleingeld, A. (2011). Goal-Setting in Practice: The Effects of Personality and Perceptions of the Goal-Setting Process on Job Satisfaction and Goal Commitment. Personnel Review, Vol. 40, Issue: 3, pp. 306-323.

Bishop, C. M. (1995). Neural Networks for Pattern Recognition. Oxford University Press.

Blind, K. (2012). The Influence of Regulations on Innovation: A Quantitative Assessment for OECD Countries. Research Policy, Vol. 41, No. 2, pp. 391-400.

Boachie-mensah, F. and Seidu, P. A. (2012). Employees' Perception of Performance Appraisal System: A Case Study. International Journal of Business and Management, Vol. 7, No. 2, pp. 73-88.

Boice, D. F. and Kleiner, B. H. (1997). Designing Effective Performance Appraisal Systems. Work Study, MCB University Press, Vol. 46, No. 6, pp. 197-201.

Bontcheva, K., Tablan, V., Maynard, D., and Cunningham, H. (2004). Evolving GATE to Meet New Challenges in Language Engineering. Journal of Natural Language Engineering, Vol. 10, No. 3-4, pp. 349-373.

Bouillon, P., Claveau, V., Fabre, C., and Sébillot, P. (2001). Using Part-of-Speech and Semantic Tagging for the Corpus-Based Learning of Qualia Structure Elements. In First International Workshop on Generative Approaches to the Lexicon, GL'2001.

Bratko, I. (1990). PROLOG Programming for Artificial Intelligence. 2nd Edition, International Computer Science Series, Addison-Wesley, 597 pages.

Breiman, L. (2001). Random Forests. Machine Learning, Vol. 45, No. 1, pp. 5-32.

Brill, E., (1992). A Simple Rule-Based Part of Speech Tagger. In Proceedings of the Workshop on Speech and Natural Language. Association for Computational Linguistics, pp. 112- 116.

Bruce, R. and Wiebe, J. (1994). Word-Sense Disambiguation Using Decomposable Models. In Proceedings of the 32nd Annual Meeting on the Association for Computational Linguistics, pp. 139-146.

Buckland, M. K. and Gey, F. C. (1994). The Relationship between Recall and Precision. Journal of the American Society for Information Science. John Wiley & Sons, Vol. 45, No. 1, pp. 12-19.

Califf, M. E. and Mooney, R. J. (1999). Relational Learning of Pattern-Match Rules for Information Extraction. In Proceedings of the 16th National Conference on Artificial Intelligence. AAAI Press, pp. 328-334.

Carliner, S. (1998). Business Objectives: A Key Tool for Demonstrating the Value of Technical Communication Products. Technical Communication, Vol. 45, No. 3, pp. 380-384.

Carpuat, M. and Wu, D. (2007). Improving Statistical Machine Translation Using Word Sense Disambiguation. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 61-72.

Charniak, E., Blaheta, D., Ge, N., Hall, K., Hale, J., and Johnson, M. (2000). Bllip 1987-89 WSJ Corpus Release 1. Tech. rep. LDC2000T43. Linguistic Data Consortium, Philadelphia, PA.

Choon, L. K. and Embi, M. A. (2012). Subjectivity, Organizational Justice and Performance Appraisal: Understanding the Concept of Subjectivity in Leading towards Employees' Perception of Fairness in the Performance Appraisal. Procedia-

Social and Behavioral Sciences, Vol. 62, pp. 189-193.

Chor, D. (2010). Unpacking Sources of Comparative Advantage: A Quantitative Approach. Journal of International Economics, Vol. 82, No. 2, pp. 152-167.

Chu C-W., and Zhang G. P. (2003). A Comparative Study of Linear and Nonlinear Models for Aggregate Retail Sales Forecasting. International Journal of Production Economics, Elsevier, Vol. 86, No. 3, pp. 217-231.

Ciaramita, M., and Altun, Y. (2006). Broad-Coverage Sense Disambiguation and Information Extraction with a Supersense Sequence Tagger. In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp. 594-602.

Ciaramita, M. and Johnson, M. (2003). Supersense Tagging of Unknown Nouns in WordNet. In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 168-175.

Clark, P., and Boswell, R. (1991). Rule Induction with CN2: Some Recent Improvements. In Proceeding of the Working Session on Learning on Machine Learning (EWSL-91), Springer, Berlin Heidelberg, pp. 151-163.

Claveau, V., Sébillot, P., Fabre, C., and Bouillon, P. (2003). Learning Semantic Lexicons from a Part-of-Speech and Semantically Tagged Corpus Using Inductive Logic Programming. In Cussens, J. and Frisch, A. M. (Eds.). Journal of Machine Learning Research, MIT Press, Vol. 4, No. 4, pp. 493-525.

Cleveland J. N., Murphy, K. R. and Williams, R. E (1989). Multiple Uses of Performance Appraisal: Prevalence and Correlates. Journal of Applied Psychology, Vol. 74, No. 1, pp. 130-135.

Cohen, W.W. (1995). Fast Effective Rule Induction. In Proceedings of the Twelfth International Conference on Machine Learning (ML95), Morgan Kaufmann, pp. 115-123.

Collins, M. (2002). Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Vol. 10, pp. 1-8.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. The Journal of Machine Learning Research, Vol. 12, pp. 2493-2537.

Crammer, K. and Singer, Y. (2003). Ultraconservative Online Algorithms for Multiclass Problems. The Journal of Machine Learning Research, Vol. 3, pp. 951-991.

Cunningham, H. et al. (2013). Developing Language Processing Components with GATE. Version 7 (A User Guide), The University of Sheffield, Department of Computer Science (2001-2013), http://GATE.ac.uk/ , (accessed 2nd March 2013).

Curran, J. R. (2005). Supersense Tagging of Unknown Nouns Using Semantic Similarity. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 26-33.

Curry, E., and Grace, P. (2008). Flexible Self-Management Using the Model-View-Controller Pattern. Software, IEEE, Vol. 25, No. 3, pp. 84-90.

Cussens, J. (1997). Part-of-Speech Tagging Using Progol. In Inductive Logic Programming, Springer Berlin Heidelberg, pp. 93-108.

Cussens, J., Page, D., Muggleton, S., and Srinivasan, A. (1997). Using Inductive Logic Programming for Natural Language Processing. In Workshop Notes on Empirical Learning of Natural Language Tasks, pp. 25-34.

Dalrymple, D. J. (1975). Sales Forecasting: Methods and Accuracy. Elsevier: Business Horizons, Vol. 18, No. 6, pp. 69-73.

De Andrés, R., García-Lapresta, J. L. and González-Pachón, J. (2010). Performance Appraisal Based on Distance Function Methods. European Journal of Operational Research, Vol. 207, No. 3, pp. 1599-1607.

De Raedt, L. (Ed.) (2008). Logical and Relational Learning: from ILP to MRDM. Springer, 387 pages.

Dědek, J. (2010). Towards Semantic Annotation Supported by Dependency Linguistics and ILP. In Processing of the 9th International Semantic Web Conference (ISWC 2010), Springer-Verlag, pp. 297-304.

Desai, V. S. and Bharati, R. (1998). A Comparison of Linear Regression and Neural Network Methods for Predicting Excess Returns on Large Stocks. Annals of Operations Research, Vol. 78, pp. 127-163.

Desmond, C. (2011). ComSoc Guide to Managing Telecommunications Projects. Vol. 4, John Wiley & Sons, IEEE Press, 190 pages.

Dolsak, B., Bratko, I., and Jezernik, A. (1994). Finite-Element Mesh Design: An Engineering Domain for ILP Application. In Proceedings of the 4th International Workshop on Inductive Logic Programming (ILP-94), pp. 305-320.

Domingos, P. and Pazzani, M. (1997). On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. Machine learning, Vol. 29, No. 2-3, pp. 103-130.

Doran, G. T. (1981). There's a S.M.A.R.T. Way to Write Management's Goals and Objectives. Management Review, Vol. 70, No. 11, pp. 35-36.

Drucker, P. F. (1954). The Practice of Management. 1st Edition. New York: HarperCollins Publishers, 404 Pages.

Džeroski, S., Grbović, J., Walley, W. J., and Kompare, B. (1997). Using Machine Learning Techniques in the Construction of Models. II. Data Analysis with Rule Induction. Ecological Modelling, Vol. 95, No. 1, pp. 95-111.

Dzeroski, S. and Lavrac, N. (1996). Rule Induction and Instance-Based Learning Applied in Medical Diagnosis. Technology and Health Care, Vol. 4, No. 2, pp. 203-221.

Edmonds, P. and Cotton, S. (2001). Senseval-2: Overview. In Proceeding of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (Senseval-2), Association for Computational Linguistics, pp. 1-5.

Eisenstein, J., Clarke, J., Goldwasser, D., and Roth, D. (2009). Reading to Learn: Constructing Features from Semantic Abstracts. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Vol. 2, pp. 958-967.

Ek, T., Kirkegaard, C., Jonsson, H., and Nugues, P. (2011). Named Entity Recognition for Short Text Messages. Procedia-Social and Behavioral Sciences, Vol. 27, pp. 178-187.

Escudero, G., Màrquez, L., and Rigau, G. (2000). Naive Bayes and Exemplar-Based Approaches to Word Sense Disambiguation Revisited. In Proceedings of the 14th European Conference on Artificial Intelligence, pp. 421- 425.

Espinilla, M., De AndréS, R., Martfnez, F. J., and Martfnez, L. (2012). A 360-Degree Performance Appraisal Model Dealing with Heterogeneous Information and Dependent Criteria. Information Sciences, Vol. 222, pp. 459-471.

Farnham, A. (1989). The Trust Gap. Fortune, Vol. 120, No. 14, pp. 56-78.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996a). From Data Mining to Knowledge Discovery in Databases. American Association for Artificial Intelligence, AI Magazine, Vol. 17, No. 3, pp. 37 -54.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996b). Knowledge Discovery and Data Mining: Towards A Unifying Framework. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), AAAI Press, Vol.

32, pp. 82-88.

Feilmayr, C., Parzer, S. and Proll, B. (2009). Ontology-Based Information Extraction from Tourism Websites. Information Technology & Tourism, Vol. 11, No. 3, pp. 183-196.

Fellbaum, C. (1998). WordNet: An Electronic Lexical Database. In Fellbaum, C. (Ed.), with a preface by Miller, G., MIT Press, 423 pages.

Fiegenbaum, E. A. (1977). The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering. Computer Science Department, School of Humanities and Sciences, Stanford University. pp. 1014-1029.

Finlay, K. and McLaren, S. (2009). Does Appraisal Enhance Learning, Improve Practice and Encourage Continuing Professional Development? A Survey of General Practitioners' Experiences of Appraisal. Quality in Primary Care, Vol. 17, No. 6, pp. 387-395.

Flach, P. A. (1998). The Logic of Learning: A Brief Introduction to Inductive Logic Programming. In Proceedings of the CompulogNet Area Meeting on Computational Logic and Machine Learning, pp. 1-17.

Fletcher, C. (2001). Performance Appraisal and Management: the Developing Research Agenda. Journal of Occupational and Organizational Psychology, Vol. 74 No. 4, pp. 473-487.

Forst, F. G. (1992). Forecasting Restaurant Sales Using Multiple Regression and Box-Jenkins Analysis. Journal of Applied Business Research (JABR), Vol. 8, No. 2, pp. 15-19.

Frank, A. and Asuncion, A. (2010). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/Irvine, CA: University of California, School of Information and Computer Science.

Frank, E. Hall, M., Trigg, L., Holmes, G., and Witten, I. H. (2004). Data Mining in Bioinformatics Using WEKA. Bioinformatics, Oxford University Press, Vol. 20, No. 15, pp. 2479-2481.

Franklin, B. J. and Osborne, H. W. (Eds.) (1971). Research Methods: Issues and Insights. Wadsworth Publishing Company, 472 pages.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian Network Classifiers. Machine Learning, Vol. 29, No. 2-3, pp. 131-163.

Fukuda, K. Tsunoda, T., Tamura, A., and Takagi, T. (1998). Towards Information Extraction: Identifying Protein Names from Biological Papers. In Proceedings of the Pacific Symposium on Biocomputing, pp. 707-718.

Furnkranz, J. and Widmer, G. (1994). Incremental Reduced Error Pruning. In Proceeding of the 11th International Conference on Machine Learning, pp. 70-77.

Gale, W., Church, K. W., and Yarowsky, D. (1992). Estimating Upper and Lower Bounds on the Performance of Word Sense Disambiguation Programs. In Proceeding of the 30th Annual Meeting on Association for Computational Linguistics, pp. 249-256.

Gauss, C.F. (1809). Theoria Motus Corporum Coelesrium. English translation by C.H. Davis, reprinted 1963 Dover, NewYork.

Geisser, S. (1975). The Predictive Sample Reuse Method with Applications. Journal of the American Statistical Association, Vol. 70, No. 350, pp. 320-328.

Goker, A. and Davies, J. (Eds.). (2009). Information Retrieval: Searching in 21st Century. John Wiley & Sons, 320 pages.

Gosling, J., Joy, B., Steele, G., and Bracha, G. (2005). Java (TM) Language Specification. 3rd Edition, Addison-Wesley.

Graff, D. (Ed.). (2002). The AQUAINT Corpus of English News Text. Technical Report, Linguistic Data Consortium, Philadelphia, PA, USA.

Gruber, T. R. (1995). Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International journal of human computer studies, Vol. 43, No. 5, pp. 907-928.

Hahn, U., Romacker, M. and Schulz, S. (2002). MEDSYNDIKATE - a Natural Language System for the Extraction of Medical Information from Findings Reports. International Journal of Medical Informatics, Vol. 67, No. 1-3, pp. 63-74.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. ACM SIGKDD Explorations Newsletter, ACM, Vol. 11, Issue: 1, pp.10 -18.

Han, J., Kamber, M., and Pei, J. (2011). Data Mining: Concepts and Techniques. 3rd Edition, Morgan Kaufmann, Elsevier, 744 pages.

Han, J., Rodriguez, J. C., and Beheshti, M. (2008). Diabetes Data Analysis and Prediction Model Discovery Using RapidMiner. In Second International Conference on Future Generation Communication and Networking (FGCN'08), IEEE, Vol. 3, pp. 96-99.

Hart, P., Nilsson, N. and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics, Vol. 4, No.2, pp. 100-107.

Hepple, M. (2000). Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based Part-of-Speech Taggers. In Proceeding of the 38th Annual Meeting of the Association of Computational Linguistics, pp. 278-285.

Hirst, G. and St-Onge, D. (1998). Lexical Chains as Representations of Context for the Detection and Correction of Malapropisms. In WordNet: An Electronic Lexical Database, C. Fellbaum, (Ed.), MIT Press, pp. 305-332.

Holmes, T. A. (2005). How to Connect Diversity to Performance. Performance Improvement, Vol. 44, No. 5, pp. 13-17.

Hurd, A. R., Barcelona, R. J. and Meldrum, J. T. (2008). Leisure Service Management. Human Kinetics, 386 pages.

Huss, W. R. (1985). Comparative Analysis of Company Forecasts and Advanced Time Series Techniques Using Annual Electric Utility Energy Sales Data. International Journal of Forecasting, Vol. 1, No. 3, pp. 217-239.

Hutchins, J. (2007). Machine Translation: A Concise History. In Computer Aided Translation: Theory and Practice. C. S. Wai, Ed. Chinese University of Hong Kong.

Ide, N. and Suderman, K. (2006). Integrating Linguistic Resources: The American National Corpus Model. In Proceedings of the 5th Language Resources and Evaluation Conference (LREC, Genoa, Italy).

Jackson, P. and Moulinier, I. (2007). Natural Language Processing for Online Applications: Text retrieval, Extraction and Categorization. 2nd Revised Edition, John Benjamins, 247 pages.

Jantan, H., Hamdan, A. R., Othman, Z. A., and Puteh, M. (2010). Applying Data Mining Classification Techniques for Employee's Performance Prediction. In Knowledge Management 5th International Conference (KMICe2010), pp. 645-652.

Jaques, T. (2005). Systematic Objective Setting for Effective Issue Management. Journal of Public Affairs, Vol. 5, No. 1, pp. 33-42.

Jiang, J. and Conrath, D. (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In Proceedings of International Conference Research on Computational Linguistics (ROCLING X), pp. 19–33.

Joshi, P., Chaudhary, S., and Kumar, V. (2012). Information Extraction from Social Network for Agro-Produce Marketing. In the 2012 International Conference on

Communication Systems and Network Technologies (CSNT), IEEE, pp. 941-944.

Jovic, A. and Bogunovic, N. (2009). Feature Set Extension for Heart Rate Variability Analysis by Using Nonlinear, Statistical and Geometric Measures. In Proceedings of the 31st International Conference on Information Technology Interfaces (ITI '09), IEEE, pp. 35-40.

Judge, T. A. and Ferris, G. R. (1993). Social Context of Performance Evaluation Decisions. The Academy of Management Journal, Vol. 36, No. 1, pp. 80-105.

Jurafsky, D. and Martin, J. H. (2009). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. 2nd Edition, Prentice Hall, 988 pages.

Kang, N., van Mulligen, E. M. and Kors, J. A. (2011). Comparing and Combining Chunkers of Biomedical Text. Journal of Biomedical Informatics, Vol. 44, No. 2, pp. 354-360.

Khandani, A. E., Kim, A. J., and Lo, A. W. (2010). Consumer Credit-Risk Models via Machine-Learning Algorithms. Journal of Banking & Finance, Vol. 34, No. 11, pp. 2767-2787.

Khelif, K., Dieng-Kuntz, R., and Barbry, P. (2007). An Ontology-based Approach to Support Text Mining and Information Retrieval in the Biological Domain. In Journal of Universal Computer Science (JUCS), Vol. 13, No. 12, pp. 1881-1907.

Kilgarriff, A. and Yallop, C. (2000). What’s in a thesaurus?. In Proceedings of the 2nd Conference on Language Resources and Evaluation (LREC), pp.1371-1379.

Kline, T. J. and Sulsky, L. M. (2009). Measurement and Assessment Issues in Performance Appraisal. Canadian Psychological, Vol. 50, No. 3, pp. 161-171.

Klyuev, V. and Oleshchuk, V. (2010). A Novel Approach to Improve the Accuracy of Web Retrieval. In the Proceedings of the 5th IEE International Conference on Future

Information Technology. IEEE Conference Publications, pp. 1-5.

Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, Vol. 14, No. 2, pp. 1137-1145.

Kohomban, U. S. and Lee, W. S. (2005). Learning Semantic Classes for Word Sense Disambiguation. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 34-41.

Korhonen, A. (2002). Assigning Verbs to Semantic Classes via WordNet. In Proceedings of the 2002 Workshop on Building and Using Semantic Networks, Association for Computational Linguistics, Vol. 11, pp. 1-7.

Kothari, C. R. (2004). Research Methodology: Methods and Techniques. 2nd Edition, New Age International, 401 pages.

Kowalski, R. (1974). Predicate Logic as Programming Language. In IFIP congress, Vol. 74, pp. 569-574.

Kucera, H. and Francis, W. N. (1967). Computational Analysis of Present-Day American English. Dartmouth Publishing Group, Brown University Press, Providence, RI.

Kumar, C. M., and Mittal, M. L. (2012). Application of Artificial Neural Networks for Sales Forecasting in an Indian Automobile Manufacturing Company. Advances In Management, Vol. 5, No. 9, pp. 20-24.

Lam, S. and Schaubroeck, J. (1999). Total Quality Management and Performance Appraisal: An Experimental Study of Process versus Results and Group versus Individual Approaches. Journal of Organizational Behavior, Vol. 20, No. 4, pp. 445-457.

Langley, P. and Simon, H. A. (1995). Applications of Machine Learning and Rule Induction. Communications of the ACM, Vol. 38, No. 11, pp. 54-64.

Larrañaga, P. et al. (2006). Machine Learning in Bioinformatics. Briefings in Bioinformatics, Vol. 7, No. 1, pp. 86-112.

Latham, G. P. and Mann, S. (2006). Advances in the Science of Performance Appraisal: Implications for Practice. International Review of Industrial and Organizational Psychology, Vol. 21, pp. 295-338.

Le Cessie, S. and Van Houwelingen, J. C. (1992). Ridge Estimators in Logistic Regression. Applied Statistics, Vol. 41, No. 1, pp. 191-201.

Leacock, C. and Chodorow, M. (1998). Combining Local Context and WordNet Similarity for Word Sense Identification. In Fellbaum, C., (Ed.), WordNet: An Electronic Lexical Database, MIT Press, Vol. 49, No. 2, pp. 265-283.

Leacock, C., Towell, G., and Voorhees, E. (1993). Corpus-Based Statistical Sense Resolution. In Proceedings of the ARPA Workshop on Human Language Technology, Association for Computational Linguistics, pp. 260-265.

Lee, S. L. (2010). Commodity Recommendations of Retail Business Based on Decision Tree Induction. Expert Systems with Applications, Vol. 37, No. 5, pp. 3685-3694.

Legendre, A.M. (1805). Nouwelles Mithodes pour lu Diterminution des Orbites des Cometes. Paris: Courcier.

Lehnert, W. and Sundheim, B. (1991). A Performance Evaluation of Text-Analysis Technologies. Association for the Advancement of the Artificial Intelligence: AI Magazine, Vol. 12, No. 3, pp. 81-94.

Lesk, M. (1986). Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In Proceedings of the

5th Annual International Conference on Systems Documentation, ACM, pp. 24–26.

Levin, B. (1993). English Verb Classes and Alternations: A Preliminary Investigation. Vol. 348. Chicago: University of Chicago Press.

Lewis, D. D. and Jones, K. S. (1996). Natural Language Processing for Information Retrieval. Communications of the ACM, Vol. 39, No.1, pp. 92-101.

Lin, D. (1998). Automatic Retrieval and Clustering of Similar Words. In Proceedings of the 17th International Conference on Computational Linguistics, Association for Computational Linguistics, Vol. 2, pp.768-774.

Lindberg, N. and Eineborg, M. (1998). Learning Constraint Grammar-Style Disambiguation Rules Using Inductive Logic Programming. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Association of Computational Linguistics, Vol. 2, pp. 775-779.

Locke, E. A. and Latham, G. P. (1990). A Theory of Goal-setting and Task Performance. Prentice Hall, Englewood Cliffs, 413 pages.

Locke, E. A. and Latham, G. P. (2002). Building a Practically Useful Theory of Goal Setting and Task Motivation: A 35-Year Odyssey. American Psychologist, Vol. 57, No. 9, pp. 705-717.

Locke, E. A. and Latham, G. P. (2006). New Directions in Goal-Setting Theory. Current Directions in Psychological Science, Vol. 15, No. 5, pp. 265–268.

Locke, E. A. and Latham, G. P. (2009). Has Goal Setting Gone Wild, or Have Its Attackers Abandoned Good Scholarship?. The Academy of Management Perspectives, Vol. 23, No. 1, pp. 17−23.

Locke, E. A., Shaw, K. N., Saari, L. M. and Latham, G. P. (1981). Goals Setting and Task Performance: 1969-1980. Psychological Bulletin, Vol. 90, No.1, pp. 125-152.

London, M. and Beatty, R. W. (1993). 360-Degree Feedback as a Competitive Advantage. Human Resource Management, Vol. 32, No. 2-3, pp. 353–372.

Loper, E. and Bird, S. (2002). NLTK: The Natural Language Toolkit. In Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, Association for Computational Linguistics, Vol. 1, pp. 63-70, http://www.nltk.org/, (accessed 14th February 2012).

Loy, M., Eckstein, R., Wood, D., Elliott, J. and Cole, B. (2002). JAVA SWING. 2nd Edition, O'Reilly Media, Inc., 1280 pages.

Mallinson, S. (2002). Listening to Respondents: A Qualitative Assessment of the Short-Form 36 Health Status Questionnaire. Social Science and Medicine, Vol. 54, No. 1, pp.11-21.

Mannila, H. (1996). Data Mining: Machine Learning, Statistics, and Databases. In Proceedings of the Eighth International Conference on Scientific and Statistical Database Management Systems (SSDBM '96), IEEE, pp. 2-9.

Manning, C. D. and Schütze, H. (1999). Foundations of Statistical Natural Language Processing. MIT Press, 680 pages.

Mayer, R. C. and Davis, J. H. (1999). The Effect of the Performance Appraisal System on Trust for Management: A Field Quasi-experiment. Journal of Applied Psychology, Vol. 84, No. 1, pp. 123-136.

McClelland, J. L. and Kawamoto, A. H. (1986). Mechanisms of Sentence Processing: Assigning Roles to Constituents of Sentences. In McClelland, J. L. and Rumelhart, D. E., (Eds.), Parallel Distributed Processing, MIT Press, Vol. 2, pp. 318, 362.

McCulloch, W. and Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. The Bulletin of Mathematical Biophysics, Vol. 5, No. 4, pp. 115–133.

McEnery, T. and Wilson, A. (2001). Corpus Linguistics. 2nd Edition, Edinburgh University Press, 256 pages.

McKenzie, A., Matthews, M., Goodman, N., and Bayoumi, A (2010). Information Extraction from Helicopter Maintenance Records as a Springboard for the Future of Maintenance Text Analysis. In Trends in Applied Intelligent Systems, Springer-Heidelberg, pp. 590-600.

Meyer, C. F. (Ed.). (2002). English Corpus Linguistics: An Introduction. Cambridge University Press, 168 pages.

Michalski, R. S. (1983). A Theory and Methodology of Inductive Learning. Artificial Intelligence, Vol. 20, No. 2, pp. 111-161.

Michalski, R. S., Mozetic, I., Hong, J., and Lavrac, N. (1986). The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. In Proceeding of AAAI, Vol. 2, pp. 1041-1045.

Michelizzi, J. (2005). Semantic Relatedness Applied to All Words Sense Disambiguation. Master of Science Thesis, Department of Computer Science, University of Minnesota, Duluth, July.

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., and Euler, T. (2006). YALE: Rapid Prototyping for Complex Data Mining Tasks. In Proceedings of the 12th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD), pp. 935- 940.

Mihalcea, R. (2006). Knowledge-Based Methods for WSD. In Word Sense Disambiguation: Algorithms and Applications, E. Agirre and P. Edmonds, (Eds.), Springer, pp. 107-131.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography, Vol. 3, No. 4, pp. 235-244.

Miller, G. A., Leacock, C., Tengi, R., and Bunker, R. (1993). A Semantic Concordance. In Proceedings of the 3 DARPA Workshop on Human Language Technology, Association for Computational Linguistics, pp. 303-308.

Mitchell, T. M. (1997). Machine Learning. McGraw-Hill Series in Computer Science, 414 pages.

Moens, M-F. (2006). Information Extraction: Algorithms and Prospects in a Retrieval Context. Springer, 246 pages.

Montalvo, O., de Baker, R. S., Michael, A., Pedro, S., Nakama, A., and Gobert, J. D. (2010). Identifying Students' Inquiry Planning Using Machine Learning. In Proceedings of the 3rd International Conference on Educational Data Mining, pp. 141-150.

Mooney, R. (1996). Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 82-91.

Mooney, R. J. (1997). Inductive Logic Programming for Natural Language Processing. In Proceedings of 6th of International Workshop on Inductive Logic Programming. Springer-Verlag, pp. 3-22.

Morphet, C. S. (1991). Applying Multiple Regression Analysis to the Forecasting of Grocery Store Sales: An Application and Critical Appraisal. International Review of Retail, Distribution and Consumer Research, Vol. 1, No. 3, pp. 329-351.

Morris, J. and Hirst, G. (1991). Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text. Computational Linguistics, Vol. 17, No. 1, pp. 21-48.

Muggleton, S. (1995). Inverse Entailment and PROGOL. New Generation Computing, Special Issue on Inductive Logic Programming, Springer, Vol. 13, No. 3-4, pp. 245-286.

Muggleton, S. (1999). Inductive Logic Programming: Issues, Results and the Challenge of Learning Language in Logic. Artificial Intelligence, Vol. 114, No. 1, pp. 283-296.

Muggleton, S. and De Raedt, L. (1994). Inductive Logic Programming: Theory and Methods. In Proceedings of Journal of Logic Programming. Vol. 19, No. 20, pp. 629-679.

Muggleton, S. and Feng, C. (1992). Efficient Induction of Logic Programs. Inductive Logic Programming, Vol. 38, pp. 281-298.

Murphy, K. R. and Cleveland, J. (1991). Performance Appraisal: An Organizational Perspective. 3rd Edition, Allyn and Bacon, 349 pages.

Murphy, K. R. and Cleveland, J. (1995). Understanding Performance Appraisal: Social, Organizational, and Goal-based Perspectives. Sage, 502 pages.

Navigli, R. (2009). Word Sense Disambiguation: A Survey. ACM Computing Surverys, Vol. 41, Issue: 2, pp. 1-69.

Ng, H. T. (1997). Exemplar-Based Word Sense Disambiguation: Some Recent Improvements. In Proceedings of the Second Conference on Empirical methods in Natural Language Processing, pp. 208-213.

Ng, H. T. and Lee, H. B. (1996). Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-based Approach. In Proceedings of the 34th Annual Meeting on the Association for Computational Linguistics, ACL, pp. 40-47.

Ordonez, L. D., Schweitzer, M. E., Galinsky, A. D., and Bazerman, M. H. (2009). Goals Gone Wild: The Systematic Side Effects of Overprescribing Goal Setting. The Academy of Management Perspectives, Vol. 23, No. 1, pp. 6−16.

Pang, B., and Lee, L. (2008). Opinion Mining and Sentiment Analysis. Foundations and Trends in Information Retrieval, Vol. 2, No.1-2, pp. 1-135.

Pantel, P. and Lin, D. (2002). Discovering Word Senses from Text. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Edmonton, Alta., Canada, pp. 613-619.

Papadimitriou, C. H. and Steiglitz, K. (1982). Combinatorial Optimisation: Algorithm and Complexity. Prentice-Hall, Edgewood-Cliffs, NJ, 512 pages.

Patwardhan, S. (2003). Incorporating Dictionary and Corpus Information into a Context Vector Measure of Semantic Relatedness. Doctoral Dissertation, University of Minnesota, Duluth.

Pedersen, T. (2000). A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. In Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference. Association for Computational Linguistics, pp. 63-69.

Pedersen, T. (2006). Unsupervised Corpus-Based Methods for WSD. In Word Sense Disambiguation: Algorithms and Applications, E. Agirre and P. Edmonds, (Eds.), Springer, pp. 133-166.

Pedersen, T. and Bruce, R. (1997). Distinguishing Word Senses in Untagged Text. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP), Vol. 2, pp. 197-207.

Pedersen, T. and Kolhatkar, V. (2009). WordNet::SenseRelate::AllWords - A Broad Coverage Word Sense Tagger that Maximizes Semantic Relatedness. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association of Computational Linguistics, pp. 17-20.

Pedersen, T., Patwardhan, S. and Michelizzi, J. (2004). Wordnet::Similarity - Measuring the Relatedness of Concepts. In Demonstration Papers at HLT-NAACL, Association for Computational Linguistics, pp. 38-41.

Peng, Y., Wang, G., Kou, G., and Shi, Y. (2011). An Empirical Study of

Classification Algorithm Evaluation for Financial Risk Prediction. Applied Soft Computing, Vol. 11, No. 2, pp. 2906-2915.

Plarre, K. et al. (2011). Continuous Inference of Psychological Stress from Sensory Measurements Collected in the Natural Environment. In Proceeding of the 10th International Conference on Information Processing in Sensor Networks (IPSN), IEEE, pp. 97-108.

Platt, J. C. (1999). Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In: B. Schotolkopf, C. J. C. Burges, A. Smola (Eds.), Advances in Kernel Methods-Support Vector Learning, MIT press, pp. 185-208.

Plotkin, G. D. (1970). A Note on Inductive Generalization. Machine Intelligence, Vol. 5, No. 1, pp. 153-163.

Proctor, P. (Ed.). (1978). Longman Dictionary of Contemporary English. 1st Edition, Longman Group, Harlow, U.K.

Provost, F. and Kohavi, R. (1998). Guest Editors' Introduction: On Applied Research in Machine Learning. In Editorial for the Special Issue on Applications of Machine Learning and the Knowledge Discovery Process, Machine Learning, Vol. 30, No. 2-3, pp. 127-132.

Purandare, A. and Pedersen, T. (2004). Improving Word Sense Discrimination with Gloss Augmented Feature Vectors. In Proceedings of the Workshop on Lexical Resources for the Web and Word Sense Disambiguation, Puebla, Mexico.

Quinlan, J. R. (1986). Induction of Decision Trees. Machine Learning. Vol. 1, No. 1, pp. 81-106.

Quinlan, J. R. (1990). Learning Logical Definitions from Relations. Machine Learning, Kluwer Academic, Vol. 5, No. 3, pp. 239 – 266.

Quinlan, J. R. (1993). C4. 5: Programs for Machine Learning. Morgan Kaufmann, 302 pages.

Quinlan, J. R. (1994). Past Tenses of Verbs and First-Order Learning. In Proceedings of the Seventh Australian Joint Conference on Artificial Intelligence, World Scientific, Singapore, pp. 13-20.

Ramakrishnan, G., Joshi, S., Balakrishnan, S., and Srinivasan, A. (2008). Using ILP to Construct Features for Information Extraction from Semi-structured Text. In Inductive Logic Programming, Springer, Heidelberg, pp. 211–224.

Ramanathan, U. (2012). Supply Chain Collaboration for Improved Forecast Accuracy of Promotional Sales. International Journal of Operations & Production Management, Vol. 32, No. 6, pp. 676-695.

Rampersad, H. K. (2001). A Visionary Management Model. The TQM Magazine, MCB University Press, Vol. 13, No. 4, pp. 211-223.

Ratnaparkhi, A. (1999). Learning to Parse Natural Language with Maximum Entropy Models. Machine learning, Vol. 34, No. 1-3, pp. 151-175.

Reenskaug, T. (2003). The Model-View-Controller (MVC), Its Past and Present. JavaZone, University of Oslo, Norway. http://heim.ifi.uio.no/~trygver/2003/javazone-jaoo/MVC_pattern.pdf (accessed 8th December 2012).

Reid, R. D. and Bojanic, D. C. (2009). Hospitality Marketing Management. 5th Edition, John Wiley & Sons, 672 pages.

Rennie, J. (2000). WordNet::QueryData: A Perl Module for Accessing the WordNet Database. http://search.cpan.org/dist/WordNet-QueryData/, (accessed 6th April 2013).

Riloff, E. (1996). Automatically Generating Extraction Patterns from Untagged Text. In Proceedings of the National Conference on Artificial Intelligence, pp. 1044-1049.

Roberts, G. E. (2003). Employee Performance Appraisal System Participation: A

Technique That Works. Public Personnel Management, Vol. 32, No.1, pp. 89-98.

Roget, P. M. (1911). Roget's International Thesaurus. 1st Edition. Cromwell, New York, NY.

Rouillard, L. A. (2003). Goals and Goal Setting: Achieving Measured Objectives. 3rd Edition, Cengage Learning, 106 pages.

Rowland, C. A. and Hall, R. D. (2012). Organizational Justice and Performance: is Appraisal Fair?. EuroMed Journal of Business, Vol. 7, Issue: 3, pp. 280-293.

Russell, S. J. and Norvig, P. (2010). Artificial Intelligence: A Modern Approach. 3rd Edition. Prentice Hall, 1132 pages.

Ryan, J. (2011). Personal Financial Literacy. 2nd Edition. South-Western CENGAGE Learning, 448 pages.

Saggion, H., Funk, A., Maynard, D., and Bontcheva, K. (2007). Ontology-Based Information Extraction for Business Intelligence. In Proceedings of the 6th International the Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference. Springer-Verlag, pp. 843-856.

Saito, M. and Hagiwara, M. (2010). Natural Language Processing Neural Network for Analogical Inference. In Proceedings of the International Joint Conference on Neural Networks, IEEE, pp. 1-7.

Samanovic, M., Cukusic, M., and Jadric, M. (2011). Influence of Various Business Regulations on the Amount of Foreign Direct Investments. WSEAS Transactions on Business and Economics, Vol. 8, No. 1, pp. 27-37.

Santorini, B. (1990). Part of Speech Tagging Guidelines for the Penn Treebank Project. (3rd Revision). Department of Computer and Information Science, University of Pennsylvania.

Santos, J. A., Nassif, H., Page, D., Muggleton, S., and Sternberg, M. E. (2012). Automated Identification of Protein-Ligand Interaction Features Using Inductive Logic Programming: A Hexose Binding Case Study. BMC Bioinformatics. Vol. 13, No. 1, pp. 162-172.

Santos Costa, V., Damas, L., Reis, R., and Azevedo, R. (2000). YAP User's Manual. Universidade do Porto. http://www.ncc.up.pt/~vsc/Yap (accessed 7th May 2013).

Savage, N. (2012). Better Medicine through Machine Learning. Communications of the ACM, Vol. 55, No. 1, pp. 17-19.

Schütze, H. (1998). Automatic Word Sense Discrimination. Computational Linguistics, Vol. 24, No. 1, pp. 97-124.

Schütze, H. and Pedersen, J. (1995). Information Retrieval Based on Word Senses. In Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95), pp. 161-175.

Schwab, D. P., Heneman, H. G., and Decotiis, T. A. (1975). Behaviourally Anchored Rating Scales: A Review of the Literature. Personnel Psychology, Vol. 28, No. 4, pp. 549-562.

Scullen, S. E., Bergey, P. K. and Aiman-Smith, L. (2005). Forced Distribution Rating Systems and the Improvement of Workforce Potential: A Baseline Simulation. Personnel Psychology, Vol. 58, No. 1, pp. 1-32.

Shabanzadeh, M., Nematbakhsh, M. A., and Nematbakhsh, N. (2010). A Semantic Based Query Expansion to Search. International Conference on Intelligent Control and Information Processing (ICICIP), IEEE, pp. 523-528.

Siltepavet, A., Sinthupinyo, S., and Chongstitvatana, P. (2012). Improving Quality of Products in Hard Drive Manufacturing by Decision Tree Technique. International Journal of Computer Science Issues, Vol. 9, No. 3, pp. 29-34.

Smither, J. W., London, M. and Reilly, R. R. (2005). Does Performance Improve Following Multisource Feedback? A Theoretical Model, Meta-analysis, and Review of Empirical Findings. Personnel Psychology, Vol. 58, No. 1, pp. 33- 66.

Snyder, B. and Palmer, M. (2004). The English All- Words Tasks. In Proceedings of Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, pp. 41-43.

Soderland, S., Fisher, D., Aseltine, J., and Lehnert, W. (1995). CRYSTAL: Inducing a Conceptual Dictionary. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95), pp. 1314–1319.

Sokolova, M., Japkowicz, N., and Szpakowicz, S. (2006). Beyond Accuracy, F-score and ROC: A family of Discriminant Measures for Performance Evaluation. In AI 2006: Advances in Artificial Intelligence, Springer Berlin Heidelberg, pp. 1015-1021.

Srinivasan, A. (1999). The ALEPH Manual. Available at: http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html (accessed 5th April 2013).

Stoyanchev, S., Song, Y. C. and Lahti, W. (2008). Exact Phrases in Information Retrieval for Question Answering. In Coling 2008: Proceedings of the 2nd Workshop on Information Retrieval for Question Answering, Association for Computational Linguistics, pp. 9-16.

Sung, N. and Chang, Y. (2004). Business Information Extraction from Semi-Structured Webpages. Expert Systems with Applications, Vol. 26, No. 4, pp. 575-582.

Tanabe, L., and Wilbur, W. J. (2004). Generation of a Large Gene/Protein Lexicon by Morphological Pattern Analysis. Journal of Bioinformatics and Computational Biology, Vol. 1, No. 4, pp. 611-626.

Tanner, L. et al. (2008). Decision Tree Algorithms Predict the Diagnosis and Outcome of Dengue Fever in the Early Phase of Illness. PLoS Neglected Tropical

Diseases, Vol. 2, No. 3, pp. 196-207.

Towell, G. and Voorhees, E. (1998). Disambiguating Highly Ambiguous Words. Computational Linguistics, Vol. 24, No. 1, pp. 125-145.

Tsumoto, S. and Hirano, S. (2010). Risk Mining in Medicine: Application of Data Mining to Medical Risk Management. Fundamenta Informaticae, Vol. 98, No. 1, pp. 107-121.

Ulutaşdemir, N. and Dağlı, Ö. (2010). Evaluation of Risk of Death in Hepatitis by Rule Induction Algorithms. Scientific Research and Essays. Vol. 5, No.20, pp. 3059-3062.

Vasilescu, F., Langlais, P., and Lapalme, G. (2004). Evaluating Variants of the Lesk Approach for Disambiguating Words. In Proceedings of the Conference of Language Resources and Evaluations (LREC 2004), pp. 633-636.

V´eronis, J., and Ide, N. M. (1990). Word Sense Disambiguation with Very Large Neural Networks Extracted from Machine Readable Dictionaries. In Proceedings of the 13th Conference on Association Computational Linguistics, Vol. 2, pp. 389-394.

Waddell, D. and Sohal, A. S. (1994). Forecasting: the Key to Managerial Decision Making. Management Decision, Vol. 32, No. 1, pp. 41-49.

Wahl, H., Winiwarter, W. and Quirchmayr, G. (2010). Natural Language Processing Technologies for Developing a Language Learning Environment. In Proceedings of the 12th International Conference on Information Integration and Web-based Applications and Services, ACM, pp. 381-388.

Waldman, D. A., Bass, B. M. and Einstein, W. O. (1987). Leadership and Outcomes of Performance Appraisal Processes. Journal of Occupational Psychology. British Psychological Society, Vol. 60, Issue: 3, pp.177-186.

Wang, G., and Ma, J. (2012). A Hybrid Ensemble Approach for Enterprise Credit

Risk Assessment Based on Support Vector Machine. Expert Systems with Applications, Vol. 39, No. 5, pp. 5325-5331.

Weaver, W. (1955). Translation. In Machine Translation of Languages: Fourteen Essays, W. N. Locke and A. D. Booth, (Eds.) Technology Press of MIT, Cambridge, MA, and John Wiley & Sons, New York, NY, Vol. 14, pp.15-23.

Whidden, C. (2005). Simple and Effective Question Processing using Regular Expressions and WordNet. Dalhousie FCS Technical Report, pp. 1-13.

Williams, J. and Curtis, T. (2007). Marketing Management in Practice. Elsevier/Butterworth- Heinemann, 376 pages.

Witten, I., Frank, E. and Hall, M. (2011). Data Mining: Practical Machine Learning Tools and Techniques. 3rd Edition. Morgan Kaufmann, Elsevier, 664 pages.

Witten, I. H., Frank, E., Trigg, L., Hall, M., Holmes, G., and Cunningham, S. J. (1999). Weka: Practical Machine Learning Tools and Techniques with Java Implementations. In: Emerging Knowledge Engineering and Connectionist-Based Information Systems, pp. 192-196.

Wubben, S. (2010). UvT: Memory-based Pairwise Ranking of Paraphrasing Verbs. In Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, pp. 260-263.

Xue, M. and Zhu, C. (2009). A Study and Application on Machine Learning of Artificial Intelligence. In International Joint Conference on Artificial Intelligence, (JCAI'09), IEEE, pp. 272-274.

Ye, Q., Zhang, Z., and Law, R. (2009). Sentiment Classification of Online Reviews to Travel Destinations by Supervised Machine Learning Approaches. Expert Systems with Applications, Vol. 36, No. 3, pp. 6527-6535.

Yeh, C. H., and Chang, Y. H. (2009). Modeling Subjective Evaluation for Fuzzy Group Multicriteria Decision Making. European Journal of Operational Research, Vol. 194, No. 2, pp. 464-473.

Zelle, J. and Mooney, R. (1993). Learning Semantic Grammars with Constructive Inductive Logic Programming. In Proceedings of the National Conference on Artificial Intelligence (AAAI-93). John Wiley & Sons LTD, pp. 817-822.

Zouaghi, A., Merhbene, L., and Zrigui, M. (2012). Combination of Information Retrieval Methods with LESK Algorithm for Arabic Word Sense Disambiguation. Artificial Intelligence Review, Vol. 38, No. 4, pp. 257-269.

Appendix A

A1 The Grammar Rules Learned by ALEPH from the First Corpus

The following presents the grammar rules for SMART objectives learned by ALEPH from the corpus of objectives related to the sales domain (first corpus); given background knowledge file (file.b) and positive (file.f) and negative (file.n) examples to ensure that the objectives are “specific”, “measurable”, and “time-related”.

```
[Rule 1] [Pos cover = 30 Neg cover = 0]
1. measurable(A,B):-
                    money(A,B).

 [Rule 2] [Pos cover = 40 Neg cover = 0]
2. time_related(A,B):-
              date (A, B).

[Rule 3] [Pos cover = 70 Neg cover = 0]
3. measurable(A,B):-
             percent(A,B).

[Rule 4] [Pos cover = 3 Neg cover = 0]
4. specific(A,B) :-
            creation_action_verb_vb(A,C), product (C,D), noun_possession_nns(D,E),
            prepositional_ phrase_pp(E,F), money (F,B).

[Rule 5] [Pos cover = 5 Neg cover = 0]
5. specific(A,B) :-
            change_action_verb_vb(A,C), determiner(C,D),
            noun_possession_nn(D,E), preposition_in(E,F), product(F,G),
            prepositional_phrase_pp(G,H), money(H,B).

[Rule 6] [Pos cover = 10 Neg cover = 0]
6. specific(A,B) :-
            change_action_verb_vb(A,C), product (C,D), noun_possession_nns(D,E),
            preposition_in(E,F), percent (F,B).
```

```
[Rule 7] [Pos cover = 2 Neg cover = 0]
7.  specific(A,B) :-
            possession _action_verb_vb(A,C), money(C,D),
            prepositional_phrase_pp(D,E), noun_possession_nns(E,F),
            preposition_in(F,G), product(G,B).

[Rule 8] [Pos cover = 2 Neg cover = 0]
8.  specific(A,B) :-
            social_action_verb_vb(A,C), noun_possession_nns(C,D),
            preposition_in(D,E), money(E,F), preposition_in(F,G),
            product(G,B).

[Rule 9] [Pos cover = 2 Neg cover = 0]
9.  specific(A,B) :-
            social_action_verb_vb(A,C), noun_possession_nns(C,D),
            preposition_in(D,E), percent(E,F), preposition_in(F,G),
            product(G,B).

[Rule 10] [Pos cover = 2 Neg cover = 0]
10. specific(A,B) :-
             social_action_verb_vb(A,C), money(C,D), noun_possession_nns(D,E),
             preposition_in(E,F), product(F,B).

[Rule 11] [Pos cover = 2 Neg cover = 0]
11. specific(A,B) :-
             social_action_verb_vb(A,C), percent(C,D), noun_possession_nns(D,E),
             preposition_in(E,F), product(F,B).

[Rule 12] [Pos cover = 2 Neg cover = 0]
12.  specific(A,B) :-
             social_action_verb_vb(A,C), prepositional_phrase_pp(C,D),
              money(D,E), noun_possession_nns(E,F),
              preposition_in(F,G), product(G,B).
```

```
[Rule 13] [Pos cover = 2 Neg cover = 0]
13.  specific(A,B) :-
                    possession_action_verb_vb(A,C), percent(C,D),
                    noun_possession_nns(D,E), preposition_in(E,F), product(F,B).

[Rule 14] [Pos cover = 2 Neg cover = 0]
14.  specific(A,B) :-
                    possession_action_verb_vb(A,C), prepositional_phrase_pp(C,D),
                    money(D,E), noun_possession_nns(E,F), preposition_in(F,G),
                    product(G,B).

[Rule 15] [Pos cover = 7 Neg cover = 0]
 15. specific(A,B):-
                    product(A,C), noun_possession_nns(C,D), modals(D,E),
                    change_action_verb_vb(E,F), preposition_in(F,G), percent(G,B).

[Rule 16] [Pos cover = 3 Neg cover = 0]
16. specific(A,B):-
                   change_action_verb_vb(A,C), product(C,D), noun_possession_nns(D,E),
                   preposition_in(E,F), money(F,B).

[Rule 17] [Pos cover = 2 Neg cover = 0]
17. specific(A,B):-
                   product(A,C), noun_possession_nns(C,D), modals(D,E),
                   change_action_verb_vb(E,F), preposition_in(F,G), money(G,B).

[Rule 18] [Pos cover = 4 Neg cover = 0]
18. specific(A,B) :-
                   change_action_verb_vb(A,C), noun_phrase(C,D),
                   preposition_in(D,E), product(E,F), noun_possession_nns(F,G),
                   preposition_in(G,H), percent(H,B).

[Rule 19] [Pos cover = 3 Neg cover = 0]
19. specific(A,B) :-
                  change_action_verb_vb(A,C), noun_phrase(C,D),
                  preposition_in(D,E), product(E,F), noun_possession_nns (F,G),
                  preposition_in(G,H), money(H,B).
```

```
[Rule 20] [Pos cover = 2 Neg cover = 0]
20. specific(A,B) :-
          change_action_verb_vb(A,C), noun_phrase(C,D), product(D,E),
          noun_possession_nns(E,F), preposition_in(F,G), percent(G,B).

[Rule 21] [Pos cover = 2 Neg cover = 0]
 21. specific(A,B):-
           creation_action_verb_vb(A,C), money(C,D), noun_possession_nns(D,E),
           preposition_in(E,F), product(F,B).

[Rule 22] [Pos cover = 2 Neg cover = 0]
22. specific(A,B):-
          creation_action_verb_vb(A,C), percent(C,D), noun_possession_nns(D,E),
          preposition_in(E,F), product(F,B).

[Rule 23] [Pos cover = 5 Neg cover = 0]
 23.  specific(A,B) :-
           determiner(A,C), noun_possession_nns(C,D), preposition_in(D,E),
           product(E,F), modals(F,G), change_action_verb_vb(G,H),
           preposition_in(H,I), percent(I,B).

[Rule 24] [Pos cover = 2 Neg cover = 0]
24. specific(A,B):-
          possession_action_verb_vb(A,C),product(C,D),
          noun_possession_nns(D,E), preposition_in(E,F), percent(F,B).

[Rule 25] [Pos cover = 2 Neg cover = 0]
25. specific(A,B):-
          determiner(A,C),noun_possession_nns(C,D),preposition_in(D,E),
          product(E,F),modals(F,G),change_action_verb_vb(G,H),
          preposition_in(H,I), money(I,B).

[Rule 26] [Pos cover = 2 Neg cover = 0]
26. specific(A,B):-
          change_action_verb_vb(A,C),determiner(C,D),
          noun_possession_nns(D,E), preposition_in(E,F), product(F,G),
          preposition_in(G,H), percent(H,B).
```

A2 The Grammar Rules Learned by ALEPH from the Second Corpus

The following presents the grammar rules for SMART objectives learned by ALEPH from the corpus of objectives related to the costs domain (second corpus); given background knowledge (file2.b) and positive (file2.f) and negative (file2.n) examples to ensure that the objectives are “specific”, “measurable”, and “time-related”.

```
[Rule 1] [Pos cover = 12 Neg cover = 0]
1. measurable(A,B) :-
                money(A,B).

[Rule 2] [Pos cover = 8 Neg cover = 0]
2. measurable(A,B) :-
                percent(A,B).

[Rule 3] [Pos cover = 20 Neg cover = 0]
3. time_related(A,B) :-
                 date(A,B).

[Rule 4] [Pos cover = 2 Neg cover = 0]
4. specific(A,B) :-
           change_action_verb_vb(A,C), service(C,D), noun_possession_nns(D,E),
           preposition_in(E,F), money(F,B).

[Rule 5] [Pos cover = 2 Neg cover = 0]
5. specific(A,B) :-
           change_action_verb_vb(A,C), service(C,D), noun_possession_nns(D,E),
           prepositional_phrase_pp(E,F), money(F,B).

[Rule 6] [Pos cover = 2 Neg cover = 0]
6. specific(A,B) :-
           change_action_verb_vb(A,C), service(C,D), noun_possession_nns(D,E),
           preposition_in(E,F), percent(F,B).
```

```
[Rule 7] [Pos cover = 2 Neg cover = 0]
7. specific(A,B) :-
        change_action_verb_vb(A,C), determiner(C,D),
        noun_possession_nns(D,E), preposition_in(E,F), service(F,G),
        preposition_in(G,H), percent(H,B).

[Rule 8] [Pos cover = 2 Neg cover = 0]
8. specific(A,B) :-
        change_action_verb_vb(A,C), determiner(C,D),
        noun_possession_nns(D,E), preposition_in(E,F), service(F,G),
        preposition_in(G,H), money(H,B).

[Rule 9] [Pos cover = 2 Neg cover = 0]
9. specific(A,B) :-
        service(A,C), noun_possession_nns(C,D), modals(D,E),
        change_action_verb_vb(E,F), preposition_in(F,G), percent(G,B).

[Rule 10] [Pos cover = 2 Neg cover = 0]
10. specific(A,B) :-
         service(A,C), noun_possession_nns(C,D), modals(D,E),
         change_action_verb_vb(E,F), preposition_in(F,G), money(G,B).

[Rule 11] [Pos cover = 2 Neg cover = 0]
11. specific(A,B) :-
         determiner(A,C), noun_possession_nns(C,D), preposition_in(D,E),
         service(E,F), modals(F,G), change_action_verb_vb(G,H),
         preposition_in(H,I), percent(I,B).

[Rule 12] [Pos cover = 2 Neg cover = 0]
12. specific(A,B) :-
         determiner(A,C), noun_possession_nns(C,D), preposition_in(D,E),
         service(E,F), modals(F,G), change_action_verb_vb(G,H),
         preposition_in(H,I), money(I,B).
```

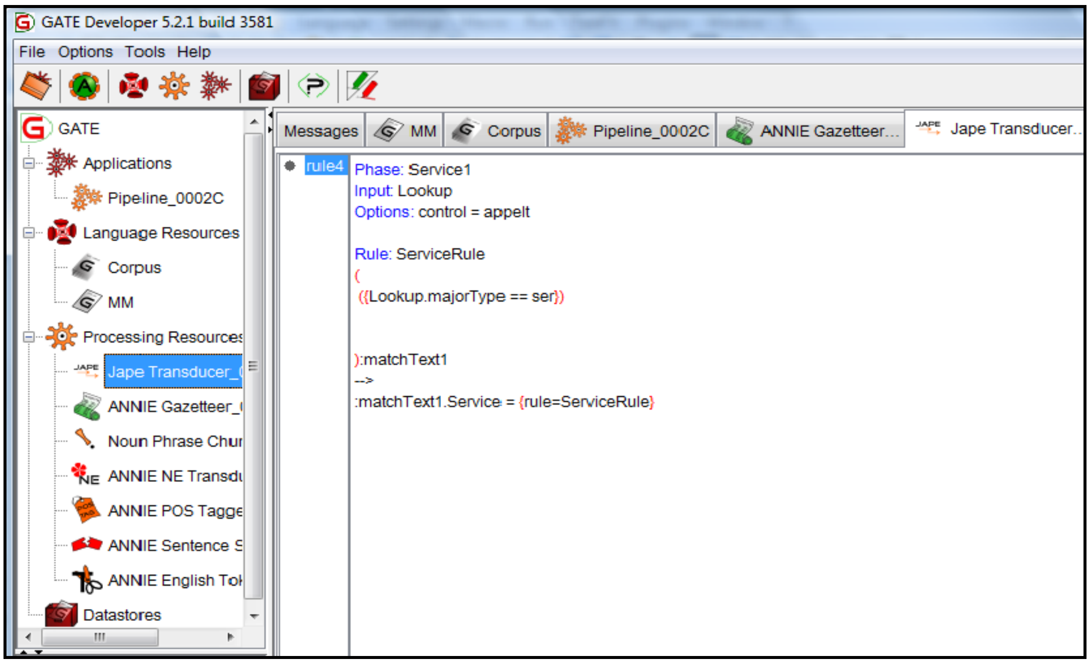
[Rule 13] [Pos cover = 2 Neg cover = 0]

```
13. specific(A,B) :-
          change_action_verb_vb(A,C), noun_phrase(C,D), preposition_in(D,E),
          service(E,F), noun_possession_nns(F,G), preposition_in(G,H),
          money(H,B).
```

Appendix B

B1 The Created JAPE Grammar for the ‘Service’ Annotation

The created JAPE grammar for the ‘Service’ annotation is presented in Figure B1.1. This grammar includes a rule called ‘ServiceRule’ which is created to match any pattern that represents a ‘Lookup’ annotation with a ‘majorType’ of ‘ser’. A list of service types has been created and added to the gazetteer lists for matching occurrences of specific strings in text. This list includes some keywords (e.g. ‘communication’, ‘food and beverage’, ‘Communication’, ‘Food and beverage’). Particularly, the created grammar rule extracts the ‘Service’ annotation by matching strings in text with the keywords defined in the created gazetteer list of service types.



```
Phase: Service1
Input: Lookup
Options: control = appelt

Rule: ServiceRule
(
 ({Lookup.majorType == ser})


):matchText1
-->
:matchText1.Service = {rule=ServiceRule}
```

Figure B1. 1 The Created JAPE Grammar for the 'Service' Annotation

B2 The Target Words Used in the Experiments Conducted by SR-AW on the Second Corpus

The target words used in the experiments conducted by the SR-AW algorithm on the second corpus, the test instances for each target word, and the accuracy achieved by the algorithm for each target word are given in Table B2.2.

Nouns	**Correct Senses**	**Wrong Senses**	**Total Instances**	**Accuracy of each Word**
costs	51	3	54	0.94
Verbs	**Correct Senses**	**Wrong Senses**	**Total Instances**	**Accuracy of each Word**
reduce	7	2	9	0.78
minimise	3	4	7	0.43
decrease	6	2	8	0.75
lessen	3	2	5	0.60
diminish	5		5	1
cut	2	3	5	0.40
cut down	4		4	1
trim	3		3	1
trim down	4		4	1

Table B2.2 Target Words, their Test Instances and the Accuracy of Disambiguating the Senses for Each Target Word

B3 The Target Words Used in the Experiments Conducted by WordNet QueryData on the Second Corpus

The target words used in the experiments performed by WordNet QueryData on the second corpus, the test instances for each target word, and the accuracy attained for each target word are given in Table B3.3.

Nouns	**Correct Classes**	**Wrong Classes**	**Total Instances**	**Accuracy of each Word**
costs	51	3	54	0.94
Verbs	**Correct Classes**	**Wrong Classes**	**Total Instances**	**Accuracy of each Word**
reduce	9		9	1
minimise	3	4	7	0.43
decrease	8		8	1
lessen	5		5	1
diminish	5		5	1
cut	2	3	5	0.40
cut down	4		4	1
trim	3		3	1
trim down	4		4	1

Table B3.3 Target Words, their Occurrences and the Accuracy of Disambiguating the Semantic Classes for Each Target Word

B4 Food and Beverage Dataset

The dataset of food and beverage costs which has been utilised in the experiments performed with linear regression in WEKA is presented in Table B4.4, where this table specifies the actual costs of food and beverage from year 2000 to 2006.

Years (2000-2006) (xi)	**Food and Beverage Cost (yi) (In Millions)**
2000 (1)	100
2001 (2)	92.184
2002 (3)	86.790
2003 (4)	72.699
2004 (5)	67.278
2005 (6)	62.6
2006 (7)	59.1

Table B4.4 Food and Beverage Cost Data (from 2000 to 2006)

B5 Communication Dataset

The dataset of communication costs which has been used in the experiments carried out with linear regression in WEKA is presented in Table B5.5, where this table specifies the actual costs of communication from year 2000 to 2006.

Years (2000-2006) (x_i)	**Communication Cost (y_i) (In Millions)**
2000 (1)	100
2001 (2)	110.926
2002 (3)	101.423
2003 (4)	85.118
2004 (5)	77.322
2005 (6)	74.4
2006 (7)	70.7

Table B5.5 Communication Cost Data (from 2000 to 2006)

B6 The Generated Linear Regression Formula for Food and Beverage Dataset

The following presents the linear regression formula generated in the WEKA toolkit from the food and beverage dataset, where this formula can be used to estimate the expected costs of food and beverage for a specified timeframe:

$Cost = -7.1922 * year + 106.005$

B7 The Generated Linear Regression Formula for Communication Dataset

The following presents the linear regression formula generated in WEKA from the communication dataset, where this formula can be used to estimate the expected costs of communication for a specified timeframe:

$Cost = -6.6091 * year + 114.9921$

B8 The Suggested Rules for Generating the Dataset Used by JRIP

The following presents the suggested rules for producing the dataset which is used by JRIP in the experiments.

1. if (staff >= 350) and (staff <= 400) and (expenses >= 4000) and (expenses <= 4500) and (year = 2007) and (service= food and beverage) then realistic=yes

2. if (staff >= 250) and (staff <= 300) and (expenses >= 3000) and (expenses <= 3500) and (year = 2008) and (service = food and beverage) then realistic=yes

3. if (staff >= 160) and (staff <= 200) and (expenses >= 1600) and (expenses <= 2000) and (year = 2007) and (service = communication) then realistic=yes

4. if (staff >= 100) and (staff <= 150) and (expenses >= 1000) and (expenses <= 1500) and (year = 2008) and (service = communication) then realistic=yes

B9 The Classification Rules Induced by JRIP in WEKA

The following presents the classification rules induced by the JRIP model from the given dataset using WEKA.

1. (Expenses >= 1002) and (Staff >= 101) and (Staff <= 149) and (Expenses <= 1500) and (service = Communication) and (year = 2008) => realistic=yes (97.0/0.0)

2. (Expenses >= 1601) and (Staff >= 161) and (Staff <= 199) and (Expenses <= 2000) and (service = Communication) and (year = 2007) => realistic=yes (98.0/0.0)

3. (Expenses >= 3002) and (Staff >= 250) and (Expenses <= 3500) and (Staff <= 300) and (service = Food_and_beverage) and (year = 2008) => realistic=yes (99.0/0.0)

4. (Expenses >= 4000) and (Expenses <= 4274) and (Staff <= 399) and (Staff >= 350) and (year = 2007) and (service = Food_and_beverage) => realistic=yes (54.0/0.0)

5. (Expenses >= 4170) and (Expenses <= 4500) and (Staff >= 350) and (service = Food_and_beverage) and (Staff <= 400) and (year = 2007) => realistic=yes (42.0/0.0)

6. (Staff <= 100) and (Staff >= 100) => realistic=yes (2.0/0.0)

7. => realistic=no (441.0/1.0)

Appendix C

The following documents are on the attached DVD and are named as specified.

C1 Background Knowledge File (file.b) for the First Corpus

C2 Examples Used by ALEPH from the First Corpus

C3 Background Knowledge File (file2.b) for the Second Corpus

C4 Examples Used by ALEPH from the Second Corpus

C5 The corpus of objectives related to the sales domain

C6 The corpus of objectives related to the costs domain