Concept Learning Supported Semantic Search Using Multi-Agent System Based on Social Networks

by

Shimaa Mansour Moailak ElSherif

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA
April, 2014

© Shimaa Mansour Moailak ElSherif 2014
Abstract

In this research, we propose an agent-based semantic search system supported by ontological concept learning and contents annotation. Our system consists of a group of multi-agent systems (MASs), each controlling a repository of structured and unstructured documents. Each repository has its own concept hierarchy, i.e. ontology. Each MAS consists of a group of agents, each with its own responsibilities and an assigned tasks to perform. All agents in each MAS cooperate with each other to perform more general tasks. Agents of different MASs communicate with each other by developing a common understanding of concepts used during communication. Agents communicate with each other via a social network. Strengths of ties between agents in the social network represent how close/far two agents are to each other.

In our system, there are two major modules: a semantic search module and a concept learning module. In the semantic search module, MASs cooperate with each other to perform semantic search and return results back to the user. The second major module in our system is the concept learning module. During the semantic search process, a MAS may discover that it needs to learn new concepts. The Concept Learning module helps that MAS to learn the new concepts from other MASs. A social network is used in communication between agents from different MASs.

In this research, we define two case studies to test the system. These case studies evaluate the efficiency of using social networks in representing relationships between agents in different MASs in learning new concepts from several teachers. We also introduce a novel approach of calculating tie strengths in social networks using Hidden Markov Model (HMM).

The results obtained show that using social networks in communicating between agents in different MASs has a positive effect in our system. During leaning new con-
cepts, using social networks between the learner and the teachers gives better accuracies in all concepts learnt and with different machine learning techniques used. On the other hand, using social networks decrease the negative effect of increasing number of teachers in the concept learning process.
Acknowledgements

This research project would not have been possible without the support of many people. It is the result of more than five years of work whereby I have been accompanied and supported by many people. I would like to thank everybody who has accompanied me to go through the whole process of graduation.

First and foremost, I would like to express my sincere appreciation to my supervisor, Dr. Behrouz Far, for his support and guidance throughout the span of my graduate studies. I owe him lots of gratitude for having me shown this way of research until I gained expertise in this research area. I would like to thank him from the bottom of my heart for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study. I am very pound to have been his student.

I would like also to thank my co-supervisor Dr. Armin Eberlein who kept an eye on the progress of my work He was abundantly helpful and offered invaluable assistance, support and guidance. Thank you for always being available when I needed your advices.

I also wish to express my deep appreciation to the examiners of this thesis, Dr. Mahmoud Maussavi, Department of Electrical and Computer Engineering, Dr. D. Krishnamurthy, Department of Electrical and Computer Engineering, Dr. Reda Elhajj, Department of Computer Science, and Dr. Malek Mouhoub, University of Regina

for their generous assistance and patience in reviewing my work and providing advice

Thanks to my fellow members of this research team: Mohsen Afsharchi, Zilan Yang and Cheng Zhong who have obtained their degrees and left this team earlier. Although we did not have much time to work together but their thesis gave me a lot of inspirations.
I also would like express my appreciation to another member, Mohammad Moshirpour for his cooperation.

I feel a deep sense of gratitude to my family. For my mother who supports me always, she has such high expectations on me. The happy memory of my late father still provides inspirations for my life, I hope he was here with me. I am grateful for my brother Tamer for his understanding and endless love, through the duration of my studies.

To My dear husband Sherif, Thank you very much for you love and patience during my PhD period and for giving me strength when I showed my weakness. Thank you for your continuous support and taking care of our daughters to let me concentrating on my research. There is no way I can ever thank you properly for everything that you have done to support me. At last, I would like to thank my two little angels Farah and Lily for being such good girls and for their relatively good behaviour during my study.
Dedication

To my father’s soul
To my mother
To my blessed husband Sherif
And to my beloved daughters
Farah and Lily
# Table of Contents

Abstract ........................................................................................................ ii  
Acknowledgements ....................................................................................... iv 
**Dedication** .................................................................................................. vi  
Table of Contents ........................................................................................... vii  
List of Tables .................................................................................................. x  
List of Figures ................................................................................................ xi  
1 Introduction ................................................................................................. 1  
  1.1 Goal ......................................................................................................... 2  
  1.2 Research Motivation ................................................................................ 6  
  1.3 Proposed System ...................................................................................... 7  
    1.3.1 Semantic Search Module ................................................................... 8  
    1.3.2 Concept learning Module .................................................................. 9  
  1.4 Contributions .......................................................................................... 10  
  1.5 Thesis overview ....................................................................................... 11  
2 Background and Literature Review ............................................................. 14  
  2.1 Basic Concepts ....................................................................................... 14  
    2.1.1 Agent ................................................................................................. 14  
    2.1.2 Multi-Agent System ......................................................................... 17  
    2.1.3 Ontology ............................................................................................ 19  
    2.1.4 Concept .............................................................................................. 26  
    2.1.5 Semantic web ..................................................................................... 26  
  2.2 UIMA ....................................................................................................... 28  
  2.3 Machine learning .................................................................................... 31  
    2.3.1 K - Nearest Neighbour (K-NN) ......................................................... 34  
    2.3.2 Naïve Bayes Learning: ................................................................. 35  
    2.3.3 Support Vector Machine (SVM) ....................................................... 36  
  2.4 Social Networks ...................................................................................... 38  
    2.4.1 Data representation in social networks ............................................ 38  
    2.4.2 Relationship levels in social networks .......................................... 38  
    2.4.3 Graphical representation of social network .................................... 39  
    2.4.4 Social network properties .............................................................. 43  
    2.4.5 Social networks in our research ..................................................... 45  
  2.5 Related works ......................................................................................... 45  
    2.5.1 Semantic Search .............................................................................. 45  
    2.5.2 Learning Approaches in MAS ....................................................... 51  
  2.6 Summary ................................................................................................. 59  
3 Proposed System: Concept Learning Supported Semantic Search Using  
MAS Based On Social Networks ................................................................. 60  
  3.1 Proposed System ..................................................................................... 60  
    3.1.1 System Architecture ....................................................................... 61  
    3.1.2 Key assumptions ............................................................................ 62
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.3</td>
<td>General interaction scheme</td>
<td>63</td>
</tr>
<tr>
<td>3.1.4</td>
<td>System workflow</td>
<td>64</td>
</tr>
<tr>
<td>3.1.5</td>
<td>System Infrastructure</td>
<td>67</td>
</tr>
<tr>
<td>3.2</td>
<td>Semantic Search Module</td>
<td>68</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Semantic Interoperability</td>
<td>69</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Semantic Search Process</td>
<td>72</td>
</tr>
<tr>
<td>3.3</td>
<td>Concept Learning Module</td>
<td>75</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Illustrative Example</td>
<td>77</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Concept learning process</td>
<td>78</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Concept learning workflow</td>
<td>81</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Conflict Resolution</td>
<td>83</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Selecting positive and negative examples</td>
<td>85</td>
</tr>
<tr>
<td>3.4</td>
<td>Prototype of our System</td>
<td>87</td>
</tr>
<tr>
<td>3.4.1</td>
<td>System Requirements</td>
<td>88</td>
</tr>
<tr>
<td>3.4.2</td>
<td>GAIA Analysis Process</td>
<td>89</td>
</tr>
<tr>
<td>3.4.3</td>
<td>GAIA Design Process</td>
<td>96</td>
</tr>
<tr>
<td>3.5</td>
<td>Summary</td>
<td>99</td>
</tr>
<tr>
<td>4</td>
<td>Calculating Tie Strengths in a Social Network Using Hidden Markov Model</td>
<td>100</td>
</tr>
<tr>
<td>4.1</td>
<td>Hidden Markov Models (HMM)</td>
<td>100</td>
</tr>
<tr>
<td>4.2</td>
<td>Tie strength in social networks</td>
<td>102</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Measuring similarity between Ontologies</td>
<td>105</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Measuring Neighbourhood overlap</td>
<td>109</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Encoding Network Structure</td>
<td>112</td>
</tr>
<tr>
<td>4.3</td>
<td>Using HMM to Calculate Tie Strength (our proposed methodology)</td>
<td>113</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Calculating the predictive variable that affects strengths of ties</td>
<td>115</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Calculating Tie Strength</td>
<td>116</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Validity of Our Methodology</td>
<td>117</td>
</tr>
<tr>
<td>4.4</td>
<td>Summary</td>
<td>120</td>
</tr>
<tr>
<td>5</td>
<td>An Example Application</td>
<td>122</td>
</tr>
<tr>
<td>5.1</td>
<td>Knowledge base domain</td>
<td>122</td>
</tr>
<tr>
<td>5.2</td>
<td>Document Classification</td>
<td>125</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Document Preprocessing</td>
<td>126</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Document Representation</td>
<td>127</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Document Categorization</td>
<td>127</td>
</tr>
<tr>
<td>5.3</td>
<td>Our case studies</td>
<td>128</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Case study I</td>
<td>129</td>
</tr>
<tr>
<td>5.3.1.1</td>
<td>Data set</td>
<td>129</td>
</tr>
<tr>
<td>5.3.1.2</td>
<td>Test scenarios</td>
<td>131</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Case Study II</td>
<td>141</td>
</tr>
<tr>
<td>5.3.2.1</td>
<td>Data Set</td>
<td>141</td>
</tr>
<tr>
<td>5.3.2.2</td>
<td>Test Scenarios</td>
<td>142</td>
</tr>
<tr>
<td>5.4</td>
<td>Accuracy of the learnt concept</td>
<td>148</td>
</tr>
<tr>
<td>5.5</td>
<td>Summary</td>
<td>149</td>
</tr>
<tr>
<td>6</td>
<td>Test Results</td>
<td>151</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.1</td>
<td>Case study I</td>
<td>151</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Test scenario I</td>
<td>151</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Test scenario II</td>
<td>159</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Test Scenario III</td>
<td>165</td>
</tr>
<tr>
<td>6.1.4</td>
<td>Test Scenario IV</td>
<td>167</td>
</tr>
<tr>
<td>6.2</td>
<td>Case study II</td>
<td>175</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Test scenario I</td>
<td>178</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Test scenario II</td>
<td>181</td>
</tr>
<tr>
<td>6.2.3</td>
<td>Test scenario III</td>
<td>187</td>
</tr>
<tr>
<td>6.3</td>
<td>Summary</td>
<td>196</td>
</tr>
<tr>
<td>7</td>
<td>Conclusions and Recommendations for Future Work</td>
<td>198</td>
</tr>
<tr>
<td>7.1</td>
<td>Research Summary</td>
<td>198</td>
</tr>
<tr>
<td>7.2</td>
<td>Contribution</td>
<td>202</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Enhancing system workflow</td>
<td>203</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Using social networks in our system</td>
<td>203</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Increasing number of teachers</td>
<td>204</td>
</tr>
<tr>
<td>7.2.4</td>
<td>Calculating tie strengths</td>
<td>205</td>
</tr>
<tr>
<td>7.3</td>
<td>Future Work</td>
<td>205</td>
</tr>
<tr>
<td>References</td>
<td></td>
<td>208</td>
</tr>
</tbody>
</table>
## List of Tables

2.1 Comparison of elements of ontology spectrum ................................. 25

3.1 An example of some fruit concepts ................................................. 77

3.2 Major services of the semantic search system ................................. 98

4.1 Sample data represent the effect of changing the value of closeness on the tie strength and the probabilities of all variables that affect the tie strength118

5.1 Domain characteristics for three universities (Afsharchi, 2007) ........ 123

5.2 Confusion matrix ................................................................. 148

6.1 Number of returned documents and similarity values of the search results for some concepts (The search keywords are ("computer science" OR "program language")). .................................................. 153

6.2 Confusion matrices and learning accuracies for learning concept "Computer Science" to an empty learner and no social networks used (CS: Computer Science, n-CS: non Computer Science) .................................................. 154

6.3 Similarity values of the search results and the similarity values between feature vectors and average value of both $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ and similarity of feature vector (F.V.: feature vectors; AVG: average) .................................................. 156

6.4 Number of returned documents, total number of documents and similarity values of the search results for some concepts. The search annotation is ("program language" | C++ | JAVA) .................................................. 158

6.5 Confusion matrices and learning accuracies for learning concept "Programming Language" to an empty learner and no social networks used (PL: Programming Language, n-PL: non Programming Language) .................................................. 159

6.6 The strength of ties between $Ag_L$ and teachers $Ag_C$, $Ag_M$ and $Ag_W$ after learning the concept "Computer Science" .................................................. 161

6.7 Number of positive and negative examples selected from each teacher for teaching "Computer Science" concept to $Ag_L$ .................................................. 161

6.8 Confusion matrices and learning accuracies for learning concept "Computer Science" to $Ag_L$ with a social network used (CS: Computer Science, n-CS: non Computer Science) .................................................. 162

6.9 Updated tie strength between $Ag_L$ and teachers: $Ag_C$, $Ag_M$ and $Ag_W$ after relearning the concept "Computer Science" .................................................. 163

6.10 Number of positive and negative examples selected from each teacher for teaching the concept "Programming Language" to $Ag_L$ .................................................. 163

6.11 Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_L$ with a social network used (PL: Programming Language, n-PL: non Programming Language) .................................................. 164

6.12 Part of the feature vector of the concept "Computer Science" in $Ag_C$'s ontology .................................................. 165
6.13 Confusion matrices and learning accuracies for learning concept "Computer Science" to $Ag_G$ with no social networks used (CS: Computer Science, n-CS: non Computer Science) 166
6.14 Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_G$ with no social networks used (PL: Programming Language, n-PL: non Programming Language) 167
6.15 Initial tie strengths values between $Ag_G$ and each teacher: $Ag_C$, $Ag_M$ and $Ag_W$ 168
6.16 Number of positive and negative examples selected by each teacher for teaching the concept "Computer Science" to $Ag_G$ 169
6.17 Confusion matrices and learning accuracies for learning concept "Computer Science" to $Ag_G$ with a social network used (CS: Computer Science, n-CS: non Computer Science) 170
6.18 New tie strengths values between $Ag_G$ and all teachers after relearning concept "Computer Science" 170
6.19 Number of positive and negative examples selected from each teacher for teaching the concept "Programming Language" to $Ag_G$ 172
6.20 Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_G$ with a social network used (PL: Programming Language, n-PL: non Programming Language) 172
6.21 Summary of the learning accuracies for learning different concepts without using social networks 174
6.22 Summary of the learning accuracies for learning different concepts by using social networks 174
6.23 Number of returned documents from the search and value of $sim(q_{spec}, C_{best})$ for the selected $C_{best}$ in all teachers 177
6.24 Summary of the learning accuracies for learning $Ag_L$ the new concept "Computer Science" from different number of teachers without defining any social networks in communication between learner and teachers 179
6.25 Summary of learning accuracies for concept "Computer Science" from different number of teachers with defining a social network in communication between $Ag_L$ and the teachers using three learning techniques 183
6.26 Learning accuracies for learning concept "Computer Science" from different number of teachers without defining any social networks in communication between $Ag_G$ and the teachers having $Ag_G$ controls a non-empty repository using three learning techniques 189
6.27 Tie strengths values between $Ag_G$ and all teachers involved in the learning process 191
6.28 Learning accuracies for learning concept "Computer Science" from different number of teachers with defining a social network in communication between $Ag_G$ and the teachers having $Ag_G$ controls a non-empty repository using three learning techniques 192
6.29 Summary of the learning accuracies of Case Study I (SN: Social Networks; CS: Computer Science; PL: Programming Language) 197
# List of Figures and Illustrations

1.1 Outline of general contributions of our research team members .......................... 3
1.2 Simple representation of the spiral-like procedural for semantic search and concept learning (Zhong, 2008) ........................................ 4

2.1 The Ontology spectrum (Daconta et al., 2003) ............................................. 20
2.2 An example of a simple taxonomy .............................................................. 21
2.3 An example of Thesaurus ............................................................................. 23
2.4 General workflow of annotation (Lally et al., 2008) ..................................... 31
2.5 A graphical representation of a binary graph of a social network ................. 40
2.6 A graphical representation of a signed graph of a social network ................. 40
2.7 A graphical representation of a valued graph of a social network ................. 41
2.8 A directed graph of a social network ............................................................ 42
2.9 A graphical representation of multiplex relationships between nodes in a social network ........................................................................ 42
2.10 A graphical representation of dyad relationship in a social network .......... 43
2.11 A graphical representation of triad relationship in a social network .......... 44

3.1 System architecture ....................................................................................... 61
3.2 Roles of different agents in each multi-agent system ................................... 63
3.3 The Spiral-like workflow of the Concept Learning and Semantic Search ..... 65
3.4 An Illustrative example using different tie strength to connect agents in a social network ........................................................................ 68
3.5 Different layers of semantic interoperability ................................................. 70
3.6 Communication between two peers based on Semantic Interoperability .... 71
3.7 The relationship between layers of semantic interoperability and ontology spectrum ..................................................................................... 72
3.8 Screen shot of semantic search process interface ............................................ 74
3.9 Illustration of Semantic Search Procedure (Zhong, 2008) ............................. 75
3.10 Representation of example space (Afsharchi, 2007) ..................................... 79
3.11 Flow diagram of tasks in the concept learning module ................................. 81
3.12 Illustration of conflict occurs during concept learning ............................... 84
3.13 Selection of positive and negative examples if $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ is greater than selected threshold ........................................................................ 86
3.14 Selection of positive and negative examples if $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ is less than selected threshold ................................................................. 87
3.15 Use case diagram of the system .................................................................. 90
3.16 The agent model of the system .................................................................. 97
3.17 The Acquaintance Model of the System ...................................................... 99

4.1 Illustrative example of case 1 in calculating similarity vector $(sv)$ for a concept $C_A$ with respect to ontology $O_B$ .................................................. 106
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Illustrative example of case 2 in calculating similarity vector ((sv)) for a concept (C_A) with respect to ontology (O_B) .................................................. 107</td>
</tr>
<tr>
<td>4.3</td>
<td>Illustrative example of case 3 in calculating similarity vector ((sv)) for a concept (C_A) with respect to ontology (O_B) .................................................. 108</td>
</tr>
<tr>
<td>4.4</td>
<td>Illustrative example of case 4 in calculating similarity vector ((sv)) for a concept (C_A) with respect to ontology (O_B) .................................................. 109</td>
</tr>
<tr>
<td>4.5</td>
<td>A representation of social network between two agents (Ag_i) and (Ag_j) with no common friends ................................................................. 111</td>
</tr>
<tr>
<td>4.6</td>
<td>A representation of social network between two agents (Ag_i) and (Ag_j) with all their friends are common ...................................................... 111</td>
</tr>
<tr>
<td>4.7</td>
<td>A representation of social network between two agents (Ag_i) and (Ag_j) with some common friends ................................................................. 112</td>
</tr>
<tr>
<td>4.8</td>
<td>Illustration of using Hidden Markov Model (HMM) to calculate tie strength of a social network ................................................................. 113</td>
</tr>
<tr>
<td>4.9</td>
<td>Illustration of using Hidden Markov Model (HMM) to calculate tie strength of a social network ................................................................. 115</td>
</tr>
<tr>
<td>4.10</td>
<td>Graphical representation of the effect of changing the value of ontology similarity on the estimated probability of the closeness variable ........ 119</td>
</tr>
<tr>
<td>4.11</td>
<td>Graphical representation of the relation between tie strength and ontology similarity ................................................................. 119</td>
</tr>
<tr>
<td>5.1</td>
<td>Illustration of document classification stages ................................................................. 126</td>
</tr>
<tr>
<td>5.2</td>
<td>Screen shot of our experiment using RapidMiner ................................................................. 128</td>
</tr>
<tr>
<td>5.3</td>
<td>Taxonomy of Cornell University ................................................................. 130</td>
</tr>
<tr>
<td>5.4</td>
<td>Taxonomy of the University of Michigan ................................................................. 130</td>
</tr>
<tr>
<td>5.5</td>
<td>Taxonomy of the University of Washington ................................................................. 130</td>
</tr>
<tr>
<td>5.6</td>
<td>Summary of test scenarios in case study I ................................................................. 150</td>
</tr>
<tr>
<td>5.7</td>
<td>Summary of test scenarios in case study II ................................................................. 150</td>
</tr>
<tr>
<td>6.1</td>
<td>A graph that represent the relationship between increasing the number of teachers and the leaning accuracy for learning a new concept to an empty learner with no social networks used and using three learning techniques ... 180</td>
</tr>
<tr>
<td>6.2</td>
<td>A graph that represent the relationship between increasing the number of teachers and the leaning accuracy for learning a new concept to an empty learner while using a social network using three learning techniques 185</td>
</tr>
<tr>
<td>6.3</td>
<td>A graph compares between learning accuracies of learning a new concept to (Ag_L) with and without social networks using K-NN learning algorithm (SN: social network) ................................................................. 186</td>
</tr>
<tr>
<td>6.4</td>
<td>A graph compares between learning accuracies of learning a new concept to (Ag_L) with and without social networks using Naïve Bayes learning algorithm (SN: social network) ................................................................. 186</td>
</tr>
<tr>
<td>6.5</td>
<td>A graph compares between learning accuracies of learning a new concept to (Ag_L) with and without social networks using SVM learning algorithm (SN: social network) ................................................................. 187</td>
</tr>
</tbody>
</table>
6.6 A graph that represents the relationship between increasing the number of teachers and the learning accuracy of learning a new concept to $Ag_G$ with no social networks used ................................................. 190

6.7 A graph that represents the relationship between increasing the number of teachers and the learning accuracy while using a social network for a non-empty learner ................................................. 193

6.8 A graph compares between the learning accuracies with and without social networks using K-NN learning algorithm for a non-empty learner (SN: social network) ................................................. 194

6.9 A graph compares between the learning accuracies with and without social networks using Naïve Bayes learning algorithm for a non-empty learner (SN: social network) ................................................. 194

6.10 A graph compares between the learning accuracies with and without social networks using SVM learning algorithm for a non-empty learner (SN: social network) ................................................. 195
Publications


Chapter 1

Introduction

The World Wide Web (WWW) has enabled users to access information from all over the world. With the increase of amount of information available, the burden on the user to search, filter and select the desired information increases drastically. One possible solution is to enhance the WWW infrastructure by adding semantics to the search. This is the motivation for the semantic web (Berners-Lee and Fischetti, 2000) which inherits the decentralized architecture of the traditional Internet. On one hand, decentralization of the web has advantages as it scales up easily and there is no single point of failure. On the other hand, this decentralization causes several problems, such as the problem of understandability and heterogeneity of ontologies, communication and negotiation complexity. Semantic web extends the World Wide Web with semantics, which adds meanings to terms used in documents and relates them to each other. In this research, we argue that, semantic integration, semantic search, concept learning and agent technology are fundamental components of an efficient Knowledge Management (KM) solution. An agent-based semantic search system supported by ontological concept learning and contents annotation is proposed.

In this proposed system, software agents: employ ontologies to organize contents in their corresponding repositories; improve their own search capability by finding relevant peers and learn new concepts from each other; conduct search on behalf of and deliver customized results to the users; and encapsulate complexity of search and concept learning process from the users. A unique feature of this system is that the semantic search agents form a social network. We use Hidden Markov Model (HMM) to calculate the tie strengths between agents that use different ontologies.
In this work, a social network is a set of actors (e.g. human, agent, document repository, multi-agent system etc.) and relationships between them. It can be represented as a set of nodes that have one or more kinds of relationships (ties) between them (El-Sherif et al., 2011). The strength of ties helps refine the search and improve accuracy of the results in both semantic search and concept learning process. Using social networks gives us flexibility in dealing with the concepts in ontologies. It allows agents to understand the meaning of the same concept even though its definition might be slightly different in each agent’s ontology.

Using the social networks paradigm in a semantic search model improves the quality of ontology based concept learning and search. The strength of ties connecting agents changes dynamically based on several factors as will be indicated later in Chapter 4.

In this chapter, in Section 1.1, we discuss the goals of our research. Section 1.2 shows the research motivation. We give a brief introduction of our proposed system in Section 1.3. Afterwards, we introduce our contributions in this work in Section 1.4. Finally, the thesis overview is presented in Section 1.5.

1.1 Goal

Our team provides an implementation of a multi-agent system for semantic search. This semantic search is integrated with a concept learning system. This concept learning system helps an agent to learn new concepts from several other agents. The goal of this research is to advance the current semantic search system by using social networks to help communication between different agents in different Multi-Agent Systems (MAS) to improve both the learning process and ranking the results of the semantic search. The work done by our team members is shown in figure 1.1.
As shown in Figure 1.1, for each research goal, some work has been done and published. Algorithms for concept learning are covered in (M. Afsharchi and Denzinger, 2006), (Z. Yang and Far, 2008) and (El-Sherif et al., 2010b). (M. Afsharchi and Denzinger, 2006) focused on devising a learning mechanism to solve the semantic heterogeneity problem in multi-agent systems where individual autonomous agents learn ontology concepts from several teacher agents to better communicate and share information. (Z. Yang and Far, 2008) started to utilize this learning mechanism into the semantic search application and enrich its practicability with some modifications. (El-Sherif et al., 2010b) used social networks in communicating between agents with different tie strengths and depended on the tie strength between the learner and the teachers to resolve conflicts occurred during the learning. In (Cheng Zhong and Far, 2008) and (Far et al., 2009), a conceptual model for semantic interoperation between concept learning and semantic search has been proposed, and a prototype MAS conforming to this model has been designed and implemented. As depicted in Figure 1.2, a semantic search engine keeps interacting with the concept learning in the form of a spiral-like workflow, and finally improves the semantic search quality. In (Far et al., 2009), experiments have been performed to verify the
contributions of concept learning and annotation to improve semantic search quality and positive results were obtained. In (El-Sherif et al., 2012b) and (El-Sherif et al., 2012a), the learning accuracy has been improved by using social networks in communicating between the learner and the teachers. In order to measure the tie strengths between agents in the social network, a new methodology to calculate the tie strength in a non-human social networks is implemented in (El-Sherif et al., 2011).

Figure 1.2: Simple representation of the spiral-like procedural for semantic search and concept learning (Zhong, 2008)

This research makes a contribution to the areas of: multi-agent and ontology-guided concept learning, distributed artificial intelligence, social networks and semantic search engine.

*In order to achieve this goal, we define four requirements in this research:*

1. Learners and teachers must be able to communicate with each other even if they use different representations of concepts (different ontologies).

2. Individual autonomous agent (learner) must be able to learn new concepts
from several other agents (teachers) by interacting with them using examples to illustrate the meaning of the new concept.

3. The learner must be able to resolve conflicts that occur during the learning process using the social network paradigm. The learner communicates with other teachers. When a conflict occurs, the learner makes use of the strength of ties with other teachers to reach the correct decision about this conflict.

4. The agent must be able to rank the results of the semantic search by using social networks in communication among agents in different multi-agent systems.

To realize the first requirement, the problem of ontology heterogeneity among agents in multi-agent systems must be taken into consideration. This problem is directly caused by the fact that any ontology specific to a certain domain evolves according to the context of that domain. The ontology, then, has its own concepts and concept features which are different from other ontologies. Most of the previous research assumes that agents have a complete common understanding of concepts that are used to depict the domain knowledge base and/or common language used to represent concepts. In the real world, different developers use different ontologies to represent their knowledge in their local repositories. So the same concept name may represent different meanings according to the context. For example, concept, ”Apple” in a data repository that represents food or restaurants refers to a kind of fruit. In another data repository that represents technology, ”Apple” refers to a computer or name of a company. On the other hand, two different concept names may refer to the same meaning. So this is a very important problem that needs to be solved before agents can understand each other. This means that agents that communicate for the first time need to have a common conceptualization and a common
association of meaning to the concepts. This point is well summarized in (Afsharchi, 2007).

To realize the second and third requirements, we depend on social networks in the interaction between agents in different multi-agent systems. Social networks have a lot of advantages. The most important one to us is that they allow actors (agents in our proposed system) to communicate with each other by different tie strengths according to a closeness metrics defined between them. For our system, we can set the strength of ties between agents depending on similarity between conceptualization of their repositories. Also the strength of the tie can change according to the frequency of interactions and the number of concepts learnt between the two agents (other features that affect tie strengths in social networks, used in our proposed system, will be discussed in more detail in Chapter 4). On one hand, using social networks helps improve quality of the learning process. On the other hand, it increases the confidence in the accuracy of the definition of the newly learnt concept.

1.2 Research Motivation

This research is motivated by four main factors:

First: Due to the huge amount of information available in the web (and its rapid growth) and the increasing need to access it fast and in the most efficient way and due to the distribution of this information among different repositories, we need to develop a good system to make the best use of these data repositories during semantic search. This enables users to get the best possible results for their search query. Semantic search is considered the future of the World Wide Web (Berners-Lee et al., 2001). This can be achieved by enabling the learning of new concepts between agents.

Second: The accuracy of learning new concepts needs to be improved by making use
of multi-agent systems in the concept learning process. This can be done by enabling one agent (learner) to accurately learn new concepts from several other agents (teachers).

**Third**: A very important problem, which is usually ignored by most researchers in the area of concept learning, is how to initialize the learning process, and how an agent knows that it needs to learn a new concept. We suggest an integration of semantic search and concept learning to make use of semantic search in initiating the concept learning process if the agent discovers that it does not know a concept within a search phrase. We believe that semantic search and concept learning are highly correlated with each other.

**Fourth**: The final motivation of this research is the importance of the social network paradigm. We know that social networks enable the participants in the network to have desirable relationships with one another. The strength of these relationships can vary among participants and by time. We can make use of the pros of social networks in our semantic search by increasing the strength of the relationship between related repositories to improve the quality of the search processes. In the learning process, social networks can help in improving learning accuracy and in solving conflicts that may occur during the learning process due to the diversity in the concept meanings within different repositories. The learner may depend on teachers according to strengths of ties between them.

### 1.3 Proposed System

We aim at solving some of the problems of traditional searching by suggesting a semantic search system based on a Multi-Agent System (MAS) that supports the decentralization required for the semantic web. The system uses a concept learning algorithm in order to solve the problem of ontology heterogeneity among agents to ease communication between them.
Our proposed system consists of two basic modules: semantic search module; and concept learning module.

1.3.1 Semantic Search Module

Semantic search is the result of combining semantic web with a specialized search engine. Semantic search tries to make the best use of semantic web. Semantic search engines aim at understanding the meaning of concepts used in search queries which in turn understand the context of the desired search in order to improve the search results by reducing ambiguity and increasing relevance. A semantic search module was proposed by our team in (El-Sherif et al., 2010a) (Far et al., 2009). In this module, each concept defined in an ontology is specified by a set of features (attributes) and corresponding values for those features.

Using social networks in the communication between agents in different multi-agent systems positively affects the results of the search process. When a user sends a search query to one agent (local agent), this local agent searches its own repository for the results. At the same time, the local agent sends this query to all peer agents (remote agents) that connect with it in a relationship. Each remote agent searches its own repository to get some results. All remote agents send their results back to the local agent. The local agent collects the results from all remote agents and ranks them according to the strengths of ties between the local agent and each remote agent.

If the search query contains a new keyword (concept) that the local agent does not know (not included in its repository), the local agent (becomes a learner in this case) initializes the concept learning module. The learner sends a learning request containing all possible information about this new concept to all peer agents (teachers in this case) it communicates with.
1.3.2 Concept learning Module

The ability of human beings to communicate with each other is one of the most important enablers for cooperation. In order for human beings to communicate efficiently, a common language that assigns the same meaning to terms of the language and a basic understanding of world concepts and communication protocols are required. The ability to learn/teach new concepts from/to each other is the skill that humans have mastered better than any other hominoid. In the research of information sharing among multiple repositories, it is assumed that both a common language as well as a common understanding of concepts exists. In practice, repositories are designed, implemented and evolved by different developers with different ontologies for representing concepts; therefore, the assumption of a common ontology is unrealistic.

In this research, we introduce a concept learning module that overcomes the above mentioned problem based on social capabilities of multi-agent systems. In our module, each multi-agent system (MAS) is associated with a repository and the concept learning process is realized when one agent in a MAS (learner) sends queries about the concept to be learnt to other agents in different MASs (teachers). This query contains information available about this concept (e.g. concept name, keywords, examples of the concept). Learning new concepts from several teachers, each with its own definition of this concept and its own ontology reduces the accuracy of the learnt concept. By increasing the number of teachers, the diversity of information used to learn a new concept increases, so learning accuracy decreases. In order to overcome the degradation of the learning and to improve the accuracy, inspired by the social networks, a set of peers is selected based on the strength of relationships between them and the learner. Repositories that have a similar conceptual structure are consulted first. Those teachers then try to find the best match of the queried concept and send positive and negative examples describing this concept back to the learner in order to help it learn the new concept.
1.4 Contributions

This research presents new advances in the state of the art in the following aspects:

- Enhancing the architecture of defining the integration between concept learning and semantic search. This integration can be represented by a spiral-like workflow in which the concept learning and semantic search are treated with the same degree of importance (El-Sherif et al., 2010a).

- Introducing the social networks paradigm in the concept learning algorithm. A social network is helpful in both the learning stage and the conflict resolution stage (El-Sherif et al., 2010b). Using social networks enables learners resolve these conflicts. Social networks allow an agent to communicate with other agents with different tie strengths. Depending on the strength of ties, the learner can resolve the conflict occurred.

- Studying the effect of using social network on the communication between MASs during the concept learning process by having the learner learns new concepts first without and then with using social networks. The accuracy of the learnt concept is then compared to see if introducing the social networks paradigm has a positive or negative effect on learning new concepts from several teachers in MAS (El-Sherif et al., 2012a) (El-Sherif et al., 2012b).

- Studying the effect of increasing the number of teachers on the learning accuracy during the concept learning process. We increase the number of teachers and learn the same concept again and calculate the accuracy of the concept learning each time. Afterwards, we use a social network in communicating between the learner and the teachers. We study the effect of increasing the number of teachers on the learning accuracy after using
social networks to show if it positively or negatively affects the accuracy of concept learning process.

- The strength of ties between nodes (agents in our system) is a very important factor in both concept learning and semantic search modules. The strength of ties between two nodes depends on several factors and is changing dynamically based on changes in ontologies and interactions done between agents. Our contribution is introducing a novel approach in calculating the strength of ties between nodes in a social networks using HMM (El-Sherif et al., 2011).

- Introducing the social network paradigm during semantic search. The strength of ties connecting agents changes dynamically according to both closeness between agents and the interaction between them. The strength of ties helps in defining the accuracy of the result obtained from each agent. When agent $A_{g1}$ sends a search query to other agents, each of the other agents will search this query locally and send back the result. Agent $A_{g1}$ then collects the results and is able to rank them according to their accuracy (El-Sherif et al., 2010a).

### 1.5 Thesis overview

In Chapter 1, we have outlined the basic problem which this research addresses. We declared our research goal, introduced the main motivation for our research and summarized our main contributions.

In Chapter 2, we define some general notions that we use throughout this research. We provide definitions for agent, multi-agent system, concept, ontology and semantic web. We give a brief introduction to Unstructured Information Management Architecture
(UIMA) that is used in document annotation. Afterwards, we give a brief definition of machine learning and introduce the three main machine learning techniques used throughout our case studies: K Nearest Neighbour (K-NN), Naïve Bayes and Support Vector Machine (SVM). A general definition of social networks is introduced in addition to defining some basic concepts related to it. Finally, we provide a literature review of related work done by other researchers in the areas of concept learning and semantic search.

The methodology of our proposed system of multi-agent system semantic search based on concept learning using social networks is proposed in Chapter 3. This includes the architecture of our system, the key assumptions considered in the system, the general interaction scheme in addition to the system workflow and its architecture. Finally, an analysis of our system is done using the GAIA methodology.

Chapter 4 contains the implementation of a mathematical model using HMM to measure strength of ties between nodes in social networks. It shows factors that affect tie strength in social networks and how we use them in defining our model. Some illustrative results are introduced regarding tie strengths.

In Chapter 5, we define our knowledge domain and document classification process which we used to deal with our examples (documents). Afterwards, we illustrate our case studies used to test the effect of using social networks on our system by testing the accuracy of concept learning. Our case studies also tests the effect of increasing the number of teachers on the learning process with and without using social networks.

In Chapter 6, we introduce the results of our test scenarios. In addition, we compare the results obtained with and without using social networks in order to show the positive effect of using social networks in the learning process on general and also while increasing the number of teachers.

A summary of the research in addition to conclusions drawn from the results are
provided in Chapter 7. Finally, some suggestions for future works are made.
Chapter 2

Background and Literature Review

As described in chapter one, our proposed system of semantic search based on a multi-agent system involves multiple research areas. In this chapter we highlight some of the important areas and also describe the related work to our research. In Section 2.1, we define some basic concepts used during our research. In Section 2.2, we mention the Unstructured Information Management Applications (UIMA). Section 2.3 provides basic information about machine learning and the learning techniques used in our research. A brief description of social networks as applicable to this research can be found in Section 2.4. Finally, in Section 2.5, we illustrate the related works done by other researchers in both semantic search and concept learning in multi-agent system.

2.1 Basic Concepts

In this section, concepts that play a vital role in this research are introduced.

2.1.1 Agent

The concept of agent has many different definitions. Russell and Norving define an agent as: "Anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators. For example robotic agents might have cameras and infrared range finders for sensors, and various motors for actuators. A software agent receives keystrokes, file contents and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files and sending network packets" (Russell and Norvig, 2003).
An agent in any system can interact with its environment through sensors (to get inputs) and effectors (to produce outputs). The same is valid for an software agent. It has encoded software as its precepts and actions (Russell and Norvig, 2010).

Agents can be classified into four types:

1. **Simple reflex agents** An agent of this type specifies its actions according to current sensor inputs only and gives no attention to the history of inputs.

2. **Model-based reflex agents** An agent of this type selects its action based on both current sensor inputs and its internal state. This agent keeps track of its internal state which reflects part of the world objects it cannot see now.

3. **Goal-based agents** An agent of this type not only keeps track of its internal state. It also has a goal to be achieved, which helps the agent to decide the next action. In some cases, the goal is a straightforward one and can be achieved by only one action. In other cases, the goal is more difficult and requires planning techniques.

4. **Utility-based agents** An agent of this type tries not only to know its goal and decide an action (or sequence of actions) to achieve it, but is also able to select between several paths to achieve that goal. A goal-based agent specifies if the action helps it achieve its goals or not, which is not enough; because in some cases, conflicts between goals occurred or it is not possible for the system to achieve all goals. In these cases, the system has to compare number of goals to be achieved in order to be able to select between different actions to perform. More scientifically, a utility-based agent has a utility function that is able to represent the current state of the agent by a real number that represents how close this agent is to achieve its goals. The
utility function allows the agent to deal with situations in which not all goals can be achieved:

1. When there are conflicts between goals, only some of them will be achieved.

2. When there are many goals to be achieved, an utility function helps the agent to increase the likelihood of success of some of the goals according to their importance.

The architecture of an agent depends on four parameters,

\[ A_g = (\text{Sit}, \text{Act}, \text{Dat}, f_{A_g}) \]  

Where: \( \text{Sit} \) is the set of situations an agent can be in, which represents effectors on the agent; \( \text{Act} \) is the set of actions an agent can perform; \( \text{Dat} \) is the set of internal data areas of an agent; There are functions for that agent: \( f_{A_g} : \text{Sit} \times \text{Dat} \rightarrow \text{Act} \), which enable an agent to select the most suitable action to perform according to current situation and internal knowledge of the agent (Denzinger and Ennis, 2002).

In this dissertation, we focus on knowledge representation used by an agent (\( \text{Dat} \)). We assume that each agent in our system contains an ontology (\( O \)) to represent agents knowledge of concepts. Other areas of knowledge may exist in (\( \text{Dat} \)) rather than (\( O \)). These areas may contain additional information about the agent itself, other agents it interacts with, etc. In our research, we concentrate only on ontology aspects of \( \text{Dat} \) (\( O \)).

The actions of agents (\( \text{Act} \)) depend on the application it is used in. In our work, we need an agent to communicate with other agents using its ontology. Our agent need also to be able to manipulate its ontology based on information received from other agents include learning actions, searching its ontology, adding new concepts and update existing ontology.
(Sit) represents observations of the environment and other agents. In our research, (Sit) contains all messages sent and received by other agents since the last situation the agent was in.

In our research, agents are implemented in a Multi-Agent System (MAS). Each agent interoperates with other agents to perform a general goal. On the other hand, agents from different multi-agent systems need to communicate with each other in both semantic search and concept learning processes. Therefore, an agents architecture must be able to distinguish between its own internal data Datown and the internal data of other agents Datother it communicates with.

2.1.2 Multi-Agent System

Introduction
The advent of research in Distributed Artificial Intelligent (DAI) provided a necessary foundation for critical concepts used in Multi-Agent System (MAS). Work on MAS started in the early 1980s when problems emerged that were beyond the capabilities of one agent and required multiple agents to solve them. MAS provides modularity needed in solving problems for complex, large and unpredictable domains. Decomposition of a problem into modules (agents), allows each agent to use the most-suitable paradigm for solving each part of the problem. Agents can communicate with each other to reach the required solution of the problem (Sycara, 1998).

In real world problems, a systems structure itself may change dynamically. Thus, object structures within the system may not be known in advance. Such a system is called an open system (e.g. Internet) (Hewitt, 1986). Agents play an important role in defining these systems. Agents can change, appear and/or disappear unexpectedly.
Multi-Agent System (MAS)

MAS can be defined as: “a loosely coupled network of problem solvers that interact to solve problems that are beyond the individual capabilities or knowledge of each problem solver” (Durfee and Lesser, 1989). MAS is therefore a collection of heterogeneous agents, each of which with its own problem solving strategy. Those agents are able to interact and coordinate with each other (Wang and Tianfield, 2005).

In order to cope with business changes required during operating a system, MAS allows interoperation between multiple existing legacy systems (Genesereth and Ketchpel, 1994). MAS is also capable of solving problems that affect society (such as scheduling of meetings) by coordinating between different agents to solve these problems (Garrido and Sycara, 1996) (Dent et al., 1992). MAS is the most suitable system in handling problems of distributed nature (e.g. health care systems) (Lewis and Sycara, 1993).

Modularity of MAS and the use of different agents with various capabilities enhance the computational performance, efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility and reusability of systems. On the other hand, due to lack of global system control, MAS has limitations due to the conflict between intentions of agents in the system. One of these limitations is the possibility of unstable system behaviour due to some local decisions. It is also difficult to define protocols and communication languages in order to interoperate heterogeneous agents (Bond and Gasser, 1988).

Multi-agent systems constitute a number of software layers particular to each agent, in order to deal with different layers of abstraction. The most common architecture is composed of three layers. The bottom layer (reactive layer) deals with decision making based on raw sensor input. The middle layer (Knowledge layer) makes use of symbolic representation of knowledge view of the agent. The top layer of this architecture deals with social aspects, such as coordination between other agents Bonasso et al. (1997);
In our research, MAS provides modularity in solving problems. It is helpful in allowing several agents to use different ontologies in representing their knowledge. Using MAS simulates the web in the diversity of data types and representations. Agents, in our research, can communicate with each other in both learning and search processes.

2.1.3 Ontology

In area of Artificial Intelligent (AI), an agent has a view to its environment to understand it. This view is called “conceptualization” or “ontology” (Genesereth and Nilsson, 1987) (Gruber, 1991). Ontologies have received lots of attention in recent years. They are widely used in AI, Knowledge engineering, education, e-commerce, semantic web, etc. There are several definitions of ontology. We use Daconta’s definition (Daconta et al., 2003) which is relevant to our work: “Ontology defines the common words and concepts (meanings) used to describe and represent an area of knowledge, and so standardizes the meanings. Ontologies are used by people, databases, and applications that need to share domain information (a domain is just a specific subject area or area of knowledge, like medicine, counterterrorism, imagery, automobile repair, etc.) Ontologies include computer usable definitions of basic concepts in the domain and the relationships among them.”

If two agents use the same ontology or are able to understand each others ontology, communication between them is possible. Daconta classifies ontologies according to their richness into four groups as presented in Figure 2.1. The lower group has the weakest ontology semantic and the upper one has the strongest.
Taxonomy

Taxonomy is "The classification of information entities in the form of a hierarchy, according to the presumed relationships of the real-world entities that they represent" (Daconta et al., 2003). This means that taxonomy is just a hierarchical classification of entities of knowledge. Taxonomy is very important in information retrieval as it is used to categorize information in a hierarchical way which is helpful in making semantic search easier, faster and more accurate. It is also helpful for mapping between two different ontologies for learning new concepts and placing them in their proper position in the hierarchy.

Taxonomy uses two main relationships. The first one is "subclassification of": when the relation is from a child entity to a parent entity. The second one is "superclassification of": when the relation is from a parent entity to a child entity. However, these types of relations are not well defined. For example, the subclassification of relation can be an "is part of" relation (i.e. aggregation) or a "subclass of" relation (i.e. inheritance) or even an unspecified type of relations, such as "belongs-to", etc, see Figure 2.2.
In addition to entities and relationships between them, taxonomy also contains attributes and attribute values related to each entity to discriminate it from other entities. We can deduce that the parent entity has some general attributes. Those attributes are general to all its children and, at the same time, discriminate it (and its children) from other entities.

Thesaurus
A thesaurus can be defined as: "a controlled vocabulary arranged in a known order and structured so that equivalence, homographic, hierarchical, and associative relationships among terms are displayed clearly and identified by standardized relationship indicators. The primary purposes of a thesaurus are to facilitate retrieval of documents and to achieve consistency in the indexing of written or otherwise recorded documents and other items."

(Daconta et al., 2003). A thesaurus is concerned with the relationships between terms defined in a taxonomy, i.e. we can consider a thesaurus as a taxonomy plus some semantic relations between terms.

A thesaurus defines five basic semantic relationships between entities:
1. **Synonym:** two entities have the same meaning.

2. **Homonym:** Two entities have the same spelling but different meaning according to the context.

3. **Broader Than:** The first entity has a broader meaning than the second one (a relation from a parent entity to a child entity).

4. **Narrower Than:** The first entity has a narrower meaning than the second one (a relation from a child entity to a parent one).

5. **Associated:** Two entities are associated with each other without any specific relationship (e.g. a nail is associated with a hammer).

For technologists, Wordnet (Miller et al., 1990) is an important thesaurus. Wordnet uses only three basic relationships:

1. **Synonyms:** The ”*is a*” relationship.

2. **Hypernyms:** The ”*parent to child*” relationship.

3. **Hyponyms:** The ”*child to parent*” relationship.

A term is a label or a string that is used to represent the underlying meaning which is called concept. Thesaurus deals with concepts rather than terms as two terms might refer to the same concept, for example bungalow and single story house. On the other hand, there may be one term that refers to two or more different concepts according to the context, for example bank which may refer to a financial institution or to a sloping land beside water. Thesaurus adds attributes, attribute values and relationships to those concepts to distinguish them from each other to be easier to understand the semantic meaning of used concept. Figure 2.3 shows an example of a thesaurus.
A conceptual model can be defined as: "a model of a subject area or area of knowledge, sometimes called a domain" (Daconta et al., 2003). Sometimes, it is called domain of knowledge. A conceptual model is used to express entities, relationships between them, attributes and their values, and rules associated with entities. This means that a conceptual model is more complex and stronger than a thesaurus in representing semantic meaning.

Rules identified in the conceptual model are similar to the production rules used in an expert system. A rule is structured in the following way:

\[
\text{If } X \text{ is true Then } Y \text{ must be also true} \tag{2.2}
\]

The first part (if part) is called antecedent of the rule. The second part (then part) is called consequent of the rule. If the condition of one rule attached to an entity is true then the rule will be executed which may also fire another rule to be executed. According
to the rules, new concepts may be defined or altered.

In a conceptual model, "subclass of" relation is defined more strictly than in a thesaurus or taxonomy, which makes this category stronger in defining semantics of an ontology.

Local Domain Theory

A local domain is the richest representation of an ontology which can be directly interpreted by the machine. Semantic relationships can be expressed in logical theory by its highest possible degree. Logical theory is built on axioms (set of statements asserted to be true) and inference rules. These inference rules and axioms are used together to prove theorems about a domain represented by an ontology. Logical theory refers to axioms, inference rules and theorems all together.

Similarities and differences of the elements of the ontology spectrum are summarized in Table 2.1.

The ontology in our research can be represented using equation 2.3 (Stumme, 2001):

\[ O := (C, \leq_C, R, \sigma, \leq_R) \]  \hspace{1cm} (2.3)

Where: \( O \) is the core ontology; \( C \) is a concept set; \( R \) is a relation set; \( \sigma \) is a signature of the ontology and \( (\sigma : R \rightarrow C) \); \( \leq_C \) is a partial order of \( C \) and called concept hierarchy (taxonomy); \( \leq_R \) is a partial order of \( R \) and called relation hierarchy.

For concepts \( c_1 \) and \( c_2 \in C \) (\( C \) is the set of all concepts in ontology \( O \)), if \( c_1 \leq_C c_2 \), then \( c_1 \) is called subconcept of \( c_2 \) and \( c_2 \) is called superconcept of \( c_1 \).

As was stated earlier, if two agents use the same ontology, they can communicate with each other. Committing to a single ontology between different agents produced by different developers is too restrictive and unrealistic. Agents use diverse ontologies. For two agents \( Ag_1 \) and \( Ag_2 \), their ontologies are diverse if at least one member of tuple
Table 2.1: Comparison of elements of ontology spectrum

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Taxonomy</th>
<th>Thesaurus</th>
<th>Conceptual Model</th>
<th>Local Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strength of semantics</td>
<td>Weakest</td>
<td>Weak</td>
<td>Strong</td>
<td>Strongest</td>
</tr>
<tr>
<td>Types of relationship used</td>
<td>General superclass-of and subclass-of</td>
<td>Five types of relations: (Synonym, Homonym, Broader than, Narrower than, Associated)</td>
<td>Truly defined subclass-of and superclass-of relationships which are semantically well specified at metamodel level</td>
<td>Well defined disjoint superclass-of and subclass-of relations with transitivity property</td>
</tr>
<tr>
<td>Entity representation</td>
<td>Entity represented by attributes and attributes values</td>
<td>Entity represented by attributes, attributes values and relations with other entities</td>
<td>Entity represented by attributes, attributes values, relations with other entities and associated production rules</td>
<td>Entities are built on axioms and associated inference rules that are used to prove theorem</td>
</tr>
</tbody>
</table>
Two ontologies \((O_1 \text{ and } O_2)\) may be different if they use different sets of concepts \((C_1 \text{ and } C_2)\). Sometimes, the same concepts can have different names in different ontologies. The most familiar case is that the same name can be used to describe different concepts. Relations between concepts \((R_1 \text{ and } R_2)\) also can be different. If two ontologies use the same \(C\) and the same \(R\), the way they use relations to associate concepts \((\sigma_1 \text{ and } \sigma_2)\) may be different too.

2.1.4 Concept

A concept is a unit of thoughts consisting of two parts, \textit{extension} and \textit{intension}. The \textit{extension} covers all objects belonging to this concept. The \textit{intension} comprises all features valid for all those objects. Many machine learning researchers define concepts as a collection of objects that share certain features. In our research, we follow this definition. In our work, we have a set of features \(F = \{f_1, \cdots , f_n\}\) where the values of feature \(f_i\) are from domain \(D_i = \{V_{i1}, \cdots , V_{im}\}\). We can characterize any concept by using its feature/value pairs. A concept \(C_k = \{[f_i = V_i], \cdots , [f_n = V_n]\}\) where: \(V_i = \{v_{i1}, \cdots , v_{ij}\} \in D_i\). An object \(o = \{[f_1 = v_1], \cdots , [f_n = v_n]\}\) is covered by concept \(C_k\) if for all \(i\) we have \(v_i \in V_i\).

2.1.5 Semantic web

As the name indicates, semantic web extends World Wide Web with semantics, which in turn is the understanding of the meaning of concepts used in the search query. Semantic web can be understood as an evolution of the Web by adding semantics of information and services. The fundamental idea of semantic web was first introduced by Tim Berners-Lee (Berners-Lee and Fischetti, 2000) as the next generation of the web. A more developed form of semantic web was introduced by Berners-Lee in (Berners-Lee et al., 2001) as a universal medium for data, information and knowledge exchange. The last few years
have seen intense activity in developing these ideas. The World Wide Web Consortium (W3C) is one of the main developers of the semantic web. It develops the main ideas and suggests standard languages to support the semantic web and also encourages research in this field (W3C, W3C). The amount of data and information exchanged via the web is drastically increases from year to year. We need a semantic web application to be able to extract the required information and filter out the rest and to help the user understand relationships between several types of information and also to enable him/her to process them the way s/he wants.

Not only interpretation of data is required, but also the integration of this data is an important challenge in the new web. Integration is important to reach a better understanding of data with several views. At the same time, sharing of data between several different services is required.

The main purpose of semantic web, as previously stated, is to understand the meaning of information regardless of its type and to be able to process it in a way required by the user. This purpose can be achieved by creating and using *semantic metadata* which tells us the content of a document together with additional information about it (Davies et al., 2006). Metadata exists in two levels:

1. Metadata that describes a whole document (e.g. web page) or a part of a document (e.g. paragraph).

2. Metadata that describes entities within a documents (e.g. persons or organizations) and relations between them (e.g. works for, located at).

Metadata is very important in the area of semantic web research as it enables search by meaning. Metadata is also important in data integration from heterogeneous sources. Typically, different schemas are used to describe and classify information, and different terminologies are used within this information. By mapping between different schemas,
it is possible to create a unified view and to achieve interoperability between processes which utilize information.

Semantic web does not intend to make data smarter. Semantic web does not need smart data. It needs to get the right data in the right place, so that the semantic web applications can process this data as needed. Availability and consistency of machine processable web data is a main issue in semantic web (Parfeni, 2009).

2.2 UIMA

UIMA (Unstructured Information Management Applications) is a software system that analyzes unstructured information in order to discover knowledge relevant to the end user (UIMA, UIMA). For example, it can ingest plain text to identify entities (e.g. persons, companies) or relations (e.g. work for, locate at). Unstructured information can be defined as a direct output from human communication. It is called unstructured because it is raw information which is not accessible by the machine. It lacks explicit semantic (structure) and is only understood by humans. Examples of unstructured information are: plain documents, natural language emails, etc. This type of information needs some tools to transform it to a form accessible by software tools (Lally et al., 2008).

This unstructured information represents the most current and important source of knowledge for organizations, governments and any human communications. It requires a tool to structure this information. Structuring means adding some structures and semantics to data to be interpreted correctly by target applications. In UIMA, an unstructured piece of information is called "Artefact". Adding structure to it is called "Analyze". A tool that adds this structure is called "Analytic". The output of the analysis process (structured data) is called "Metadata".

UIMA consists of three main components: frameworks (such as Java or C++ frame-
works), *components* (such as annotators or repositories) and *infrastructure* (such as servers or repository supported tools).

UIMA decomposes an application into components (e.g. language identification, entity detection and sentence detection). Each component implements interfaces defined by a framework and provides self-describing metadata using XML descriptor files. The framework works as a manager between these components and manages flow of data among them. UIMA can wrap these components into network services. UIMA can also scale to large volumes by replicating processing pipelines over a cluster of networked nodes.

Seven main elements constituting UIMA (Lally et al., 2008)

1. **Common Analysis Structure (CAS):** It is a common data structure shared by all UIMA analytics. CAS is used to represent artefacts and metadata artefacts. UIMA defines two fundamental types of objects in CAS:
   
   (a) *Sofa (Subject of analysis):* It holds the artefacts.
   
   (b) *Annotations:* It is a type of artefacts metadata. It points to a region of a Sofa (i.e. required data) and annotates/labels it.

2. **Type System Model:** It is a collection of inter-related type definitions. Each type describes the structure of a set of objects that are instances of this type. Type definition defines attributes of a specific type and also specifies filters to those attributes. UIMA does not provide any standard data types. All data types are user defined in order to be applicable to any domain or industry.
3. **Base Type System**: UIMA defines some commonly used, domain independent types such as Sofa and Annotation. A Base Type System also includes primitive types, views and source document information.

4. **Abstract Interfaces**: they define standard component types and operations that UIMA services implement. Processing Element (PE) is the super type of all UIMA components. One of the commonly used subtypes of PE is the analytic which performs analysis of CASes. Analytics has two subtypes:

   (a) **Analyzer**: It processes CAS and possibly updates its contents.

   (b) **Multiplier**: It processes CAS and possibly creates new CASes. Input may be one CAS and the multiplier may divide it into pieces each of which is a new CAS. It also can be used to merge information within multiple CASes and produce one CAS.

5. **Behavioural Metadata**: Behavioural metadata of an analytic declaratively describes what the analytic does. Behavioural metadata must describe in detail the behaviour of the analytic to enable both human and automatic follow up of the analysis process.

6. **Processing Element Metadata**: It describes Processing Elements (PE) used for analysis.

7. **WSDL Service Descriptor**: It describes frameworks and interfaces used for web services.
UIMA is used in our system to do the annotation. It annotates documents based on keywords entered by a user in order to perform a semantic search. Figure 2.4 shows the annotation workflow using UIMA.

![Figure 2.4: General workflow of annotation (Lally et al., 2008)](image)

2.3 Machine learning

Machine learning is a branch of artificial intelligence and is defined according to Simon as a "Field of study that gives computers the ability to learn without being explicitly programmed" (Simon, 2013). Tom M. Mitchel provides a more operational definition to machine learning as: "A computer program is said to learn from experience E with respect to some class of tasks T and perform measure P, if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997a).

The need for machine learning arises when facing some applications where no specific algorithm is available to solve it, there are only some example data. Machine learning is very powerful to make use of stored data by analyzing it and turning it into information which can be used to make predictions. We do believe that there is a process underlying
the analysis of data, but we do not know exactly the steps (algorithm) of this process. The best way to overcome that is by using examples to learn the process and be able to analyze new data. For example, spam emails change by time and differ from one person to another. In this case, a system can be trained for spam emails and regular emails. For new emails, the system can learn to distinguish between spam and regular emails. In this case, there is no specific algorithm. The system creates its own algorithm (Alpaydın, 2010).

Machine learning is not just an algorithm to solve database problems; it is a part of Artificial Intelligence (AI). It aims at making systems more intelligent. Being intelligent means: in spite of a changing environment, the system is able to learn and use its knowledge to perform a required task (e.g. analysis, prediction). For example, every day we can recognize our family members and friends by looking at their faces even with changing poses and hair styles. This is a type of task that we do effortlessly and we cannot describe how. Using machine learning, we can enable computers to perform face recognition. Pattern recognition is one application of machine learning.

Machine learning builds a mathematical model. Its task is to make inference from a sample. First, we need a powerful training algorithm to solve an optimization problem and also to store and process a large amount of data used during the training stage in order to create a mathematical model. Second, after training, the created model needs a powerful representation and algorithmic solution for inference needs (Alpaydın, 2010). Learning a rule from training data is known as "knowledge extraction". We can define a rule as a simple model that explains input data. Another important role of machine learning is to find instances that do not follow an extracted rule. This is called "outlier detection".

There are two major categories of machine learning:

**Supervised learning** The aim of supervised learning is to map specific inputs (training
data) to their corresponding outputs under supervision.

**Unsupervised learning** In unsupervised learning, no supervision exists. The aim is to find regularities in inputs by arranging the inputs into clusters using general features of those clusters and trying to assign each input to one cluster.

With the increasing amount of data available, the need for smart data analysis techniques increases. Some researchers characterize learning problems based on types of data they use. When problems arise related to similar data types, they may be solved using similar techniques (Smola and Vishwanathan, 2008).

One of the challenges in dealing with vectors in machine learning techniques is the length of feature vectors. In order to overcome this problem, data normalization is used.

Types of data:

Types of data used during training in a machine learning process can be one of the following:

**Lists** They describe each data entry by a set of features.

**Sets** They are used whenever there is a large number of potential causes of an effect. They infer the properties of an object given a set of features.

**Matrices** They are used for representing pairwise relationships.

**Images** Sometimes, they are considered as a two dimensional matrix.

**Videos** They add temporal dimension to images.

**Trees and Graphs** They are most used in describing relations between different objects.
Strings They are the most used types in areas of bioinformatics and natural language processing. They may be used as inputs for classifying spam emails process, or searching for specific names or texts within documents. They can also be outputs to some automatic translation process and answering natural language queries.

Compound structures They are a mixture of different data types. They are the most frequently occurring data types.

In our research, we use machine learning for the learning of concepts from several teachers. Data type used in our system is string data. In order to test our system, we use three different learning techniques in our case studies: K-Nearest Neighbour (K-NN); Naïve Bayes; and Support Vector Machine (SVM).

2.3.1 K - Nearest Neighbour (K-NN)

k-NN can be defined as ”non-parametric method for classifying objects based on closest training examples in the feature space” (Altman, 1992). We can say that, it is a non-parametric algorithm for classifying an object based on distance between it and all objects in training data.

It is the most basis estimator. It assigns labels to nearest neighbour of an observation \((x)\). They measure distance between pairs of observations \(d(x,x')\). The distance does not need to be symmetric, so, this classification can be extremely flexible (Zhang and Zhou, 2007).

If data is noisy, estimation of nearest neighbour classification can be very noisy as well. For example, if one spam email is labeled as non-spam by mistake, all similar emails will be indicated as non-spam too.

K-NN of \(x\) uses a majority votes to decide the class that \(x\) belongs to. If good distance measures are used, K-NN may lead to perfect results.
**Algorithm of K-NN classification (Zhang and Zhou, 2007):**

```
Classify (X, Y, x) read documents X, labels Y and query x
for i:=1 to m do
    ComputeDistance d(x_i, x)
End for
Compute set I containing indices for the k smallest distance d(x_i, x)
Return majority label of y_i where i ∈ I
```

K-NN is one of the most popular and successful algorithm in test categorization (Manning and Schütze, 1999). It can achieve high performance rate in different data sets (Yang and Liu, 1999) (Joachims, 1998). One of the cons of k-NN algorithm is its efficiency. It needs to measure the distance between the test objects and all training samples, which may consume time if the size of the training set is large.

**In document classification:** In order to decide which class the test document belongs to, k-NN uses equation 2.4.

\[
y(d_i) = \arg \max_{x_j \in K_{-NN}} \sum_{x_j} \text{sim}(d_i, x_j)y(x_j, c_k) \tag{2.4}
\]

Where: \(d_i\) is a test document; \(x_j\) is one of the neighbors in the training set, \(y(x_j, c_k) \in \{0, 1\}\) indicates whether \(x_j\) belongs to class \(c_k\); \(\text{sim}(d_i, x_j)\) is a similarity function for \(d_i\) and \(x_j\).

This equation means that, the predicted class is the one with maximal sum of similarity (Li et al., 2003).

2.3.2 Naïve Bayes Learning:

It is a simple probabilistic classifier based on Bayes rule with strong (Naïve) independence assumptions. The Baysian classifier depends on the Baysian theorem with independent
assumption between predictors. Naïve Bayes is an easy to build model as it does not require complicated iterative parameter estimation. It is useful in very large data sets (Zhang, 2004).

**Bayes rule:**

In a supervised learning, in order to get the value of $P(Y \mid X)$, where $X$ and $Y$ are random variables (Mitchell, 1997a) (Smola and Vishwanathan, 2008):

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)} \tag{2.5}$$

Where: $P(Y \mid X)$ is the posterior probability of class $Y$ given predictor $(X)$; $P(Y)$ is the prior probability of class $(Y)$; $P(X \mid Y)$ is the probability of predictor $(X)$ given class $(Y)$; $P(X)$ is prior probability of predictor $(X)$.

Bayes rule of $P(Y = y_i \mid X)$ can be represented as:

$$P(Y = y_i \mid X = x_i) = \frac{P(X = x_k \mid Y = y_i)P(Y = y_i)}{\sum_{y_j} P(X = x_k \mid Y = y_j)P(Y = y_j)} \tag{2.6}$$

Where: $y_m$ is the $m^{th}$ possible value of $Y$, $x_k$ is $k^{th}$ possible value of $X$.

To learn $P(Y \mid X)$, we can use training data to estimate $P(X \mid Y)$ and $P(Y)$ together with Baysian rule to determine $P(Y \mid X = x_k)$ for an instance $x_k$.

2.3.3 Support Vector Machine (SVM)

SVM was initially proposed by Cortes and Vapnik. The main description of SVM was introduced in Vapnik (1995, 1999) with the same basic idea. For a given learning process with a finite amount of training data, the best performance can be achieved when achieving a balance between the accuracy obtained and the machine capacity. That is the ability of machine to learn any amount of data without errors.

SVM is a supervised learning model with a learning algorithm that analyzes data
and recognizes patterns. SVM is used for classification and regression analysis. SVM constructs a hyperplane of set of planes in a large space with an infinite number of dimensions. A good hyperplane separation can be achieved by the largest distance to the nearest training object of all data classes (functional margin). The larger the functional margin, the lower the generalization error (Burges, 1998).

Originally, the problem was stated in a finite space. Later, it is discovered that some sets cannot be linearly separated in that space. Afterwards, it was proposed to map into higher dimensional space to make separation easier in that space. In order to keep computational load reasonable, mapping used by SVM is designed to ensure that the dot product can be computed easily in terms of variables in original space. This can be achieved by defining those variables in terms of a selected kernel function $K(x, y)$.

The hyperplanes are the set of points whose dot product with a vector in that space is constant. Vectors that define those hyperplanes are chosen to be a linear combination with parameter $\alpha_i$ of the feature vectors of a database (Press, 2007).

The point $x$ in a feature space that are mapped into the hyperplane are defined by relation:

$$\sum_{\forall i} \alpha_i k(x_i, x) = \text{constant} \quad (2.7)$$

Note that, if $k(x, y)$ becomes smaller when $y$ gets further away from $x$, then each element in the above sum represent the closeness between test point $x$ and corresponding database point $x_i$. In this way, we can conclude that, equation $2.7$ represents the distance between test point $x$ and each data point in each class which helps in mapping the test point $x$ to the nearest class.
2.4 Social Networks

Today, social networks have attracted considerable attention (Scott, 2000), especially after the ubiquitous use of the Internet as a communication medium. A social network is a set of individuals and relationships between them. It can be represented as a set of actors or nodes (agents in our system) that have one or more kind of relationships (ties) among them. The importance of social networks is due to their effect in diffusion of information among actors included within a network (Hanneman and Riddle, 2005).

2.4.1 Data representation in social networks

Data represented by social networks (Wasserman and Faust, 1994) is slightly different from regular statistical data. Regular statistical data can be represented by an array in which rows represent an instance or an observation, columns represent attributes of this instance and cell contents represent a value of each attribute with respect to each instance. On the other hand, social network data can be represented by a square array in which rows are corresponding to individuals participating in this network, columns are the same individuals and cell contents represent relations between the individuals.

2.4.2 Relationship levels in social networks

There are different levels to represent the strength of relationships in social networks.

**Binary relations** In this level of measurement, a relationship either exists or not. If it exists the relation takes the value of 1. Otherwise, it takes 0. It is the simplest way to represent relations in social networks and many types of analyses depend on it.

**Multiple-category nominal relations** In this measurement, several different types of relations are presented in the network to provide more than one type
of relation for each actor. For example, the actor may specify different relations for other actors such as a friend, partner, room-mate...etc. The main advantage of this level of measurement is that the resultant network can be divided into several different binary social networks each of which represents one kind of relation. It also can be binarized by stating only the existence of a relation.

**Grouped ordinal relation** This level is similar to binary relation, but the relation is represented by three values (-1, 0, 1) which are dislike, neutral or like.

**Full-rank ordinal relations** It is possible for an actor to rank the strength of his/her relations with other actors from strongest to weakest by giving them ordered numbers. In this level of measurement, the difference in strength between each two successive ranks may not be the same.

**Interval relations** It is like full rank ordinal relation in defining different ranks of a relation. But in this type of measurement, the difference between ranks is the same. This type of relation depends on observation. For example, relation between organizations can be represented by the amount of trading between them.

2.4.3 Graphical representation of social network

The graph used to represent a social network (Wasserman and Faust, 1994) consists of several nodes that represent actors of the network and arrows or lines to represent relationships between actors.

Level of measurements (binary, signed or valued graphs)

In binary graphs, lines or arrows between two nodes indicate the existence of a relationship between these two actors, see Figure 2.5. In this network, there are relationships
between nodes $A$ and $B$, $A$ and $C$, $B$ and $D$, $C$ and $D$ and $D$ and $E$ but no other relationships.

![Graph](image)

Figure 2.5: A graphical representation of a binary graph of a social network

For signed data, a positive sign (+) is written with the arrow to represent positive feeling of the relation. A negative sign (-) represents negative feeling of the relation. No sign represents neutral feeling or no relation, see Figure 2.6. In this network, there are positive relationships between nodes: $A$ and $B$, $A$ and $C$, $B$ and $D$, $C$ and $D$ and $D$ and $E$, negative relationships between nodes: $A$ and $E$, $A$ and $D$ and $C$ and $E$ and natural relation between nodes: $B$ and $E$ and $B$ and $C$.

![Graph](image)

Figure 2.6: A graphical representation of a signed graph of a social network
In a valued graph, a number is combined with an arrow to represent the value of the relationship, see Figure 2.7.

Figure 2.7: A graphical representation of a valued graph of a social network

Directed or undirected graph

A directed graph allows the representation of a direction of a relation between actors. A relation from one actor (ego) to another one (alter) is indicated by an arrow from the ego to the alter. In undirected graphs, a relation is represented by a line which corresponds to the existence of a relation, and is called co-presence, co-occurrence or bonded tie.

If we need to represent two relations between actor A and actor B and back between actor B and actor A by a directed graph, it can be represented either by double arrows or by a two headed arrow, see Figure 2.8. In this network, there are relationships between nodes: A and B, B and A, A and C, C and A, B and C, D and B, C and D, D and E and E and D only.
Simplex and multiplex relations

If a graph represents one type of relation, it is called a *simplex graph*. If it represents multiple types of relations, it is called *multiplex graph*. Sometimes, there are many types of relations within a single graph which leads to complexity of the graph. The analyst may divide a complex graph into several simplex graphs or use different line thicknesses to represent a number of relations between two actors, see Figure 2.9.

Figure 2.8: A directed graph of a social network

Figure 2.9: A graphical representation of multiplex relationships between nodes in a social network
2.4.4 Social network properties

Connection between nodes

One of the most important features of a social network is connection between its nodes. As we mentioned previously, there are two types of connections (ties) between actors in a social network: directed and undirected.

For directed connection, actors embedded within this connection have different roles. An actor that sends the tie is called (source). An actor that receives the tie is called (sink).

Types of relations

It is necessary to identify the way actors are embedded within a relation. There are many different types of relations. The most important types of relations are: dyad relation (two actors are involved in the relation, see Figure 2.10) or triad relation (three actors are involved in the relation, see Figure 2.11). In dyad relation between actors A and B, there are four possibilities: A has a relation with B, B has a relation with A, neither A nor B has relations with each other or both A and B has a mutual relation.

Figure 2.10: A graphical representation of dyad relationship in a social network
Size of social networks
The size of a social network is the number of nodes within the network. Density of a social network is the ratio between existing connections in the network to all possible connections. The number of possible connections within a network is $k(k - 1)$, where $k$ is the number of nodes in the network in the case of a directed graph or asymmetric network. In the case of a symmetric network, the number of possible connections equals $(k(k - 1))/2$ because in this case, the connection between $A$ and $B$ is the same as the connection between $B$ and $A$.

Degree of nodes
The degree of a node is the number of connections in which this node is involved. In the case of a directed network, there are two types of degrees, in-degree and out-degree, representing the number of relations towards and outwards from a node respectively. The degree of a node is a very important property of the node. The in-degree represents how powerful the node is. It corresponds to the amount of knowledge a node has. The larger the in-degree the more knowledge the node has. The out-degree represents how influential this node is. The larger the out-degree of a node the more it affects the surrounding nodes.
Reachability in social networks

Reachability of one actor to another is the ability of one actor to reach another one by any sequence of steps. In directed graphs, there may be a case that actor $A$ can reach actor $B$ but $B$ cannot reach $A$. In symmetric data, if two actors cannot reach each other, there will be a kind of division within the network.

Distance in a social network

The distance between two nodes is the number of steps required to exchange information between them. This is an important property of the nodes in a social network because it specifies how effective a node is and how far knowledge at a node can propagate.

2.4.5 Social networks in our research

In our system, we introduce the concept of social networks to define relationships between nodes. In our work, nodes of social networks are agents in different MASs. Relationships between them are asymmetric. For two agents, $Ag_1$ and $Ag_2$, there may be a relation from $Ag_1$ to $Ag_2$ and no relations from $Ag_2$ to $Ag_1$. Strengths of relationships in our work are full-rank-ordinal relations. All relations in our system are dyad relations (each relation is between two actors/agents only).

2.5 Related works

In this section, we introduce work done by other researchers in areas of "Semantic Search" and "Concept Learning".

2.5.1 Semantic Search

Semantic search is the integration between a semantic web and a search engine. Semantic search is considered as the centerpiece of semantic web. It aims at understanding the
meaning of concepts used in search queries which in turn understands the context of desired search in order to improve the results of those queries. Semantic search is preferred over ordinary search as it reduces the ambiguity of a query. It also increases the relevance of search results.

We can trace the evolution of search engines and classify them according to their development, into five categories:

1. **Horizontal Search Engine:** It is a generic search engine such as Google (Google, a) and Yahoo (Yahoo, Yahoo). It is called a keyword based search engine. Using these search engines, a user may be required to go through several result pages to reach the desired result page.

2. **Vertical Search Engine:** The context of these search engines is more specialized by location or by topic. Results of search queries are more relevant. Examples of search engine of this type are: Genie Knows (GenieKnows, GenieKnows) which is a Canadian vertical search engine company concentrating on niche markets: health search, video games search, and local business directory search; Wazap (Wazap, Wazap) is a Japanese vertical search engine, video game database and social networking site.

3. **Blended Search Engine:** Some horizontal search engines such as Google and Yahoo embedded some features from vertical search engines technologies to make their results more relevant to the user. They did so by re categorizing the results by grouping them by their attributes or types.

4. **Self-Help Search Engine:** It is a relatively new type of search engine. They contain a database to enable the user to upload any material and specify attributes or keywords to make them searchable. Google released
its GoogleBase in 2008 (Google, b). According to the relevance of uploaded materials, they can be searched by ordinary Google search engine. In this type of search engines, the user can make his/her material searchable instead of passively respond to search queries and manage them locally.

5. **Offline Search Engine:** As there may be a problem accessing your local documents on the hard drive that are bonded by their location or their name, Google releases an offline engine called Google Desktop (Google, c). It indexes the local files such as PDFs, textual files and emails. It stores them in a local index which is constantly updated. Also, Windows updated its offline search engine to increase its productivity to help users quickly find and retrieve emails messages, documents and many other files located in their PCs or corporate networks.

A relatively new approach of semantic search that comes as an add-on with Firefox is called Semanti (Parfeni, 2009). It is a blend of social search and personal search powered with some easy tools. It uses methods to enable the user to specify exactly the meaning of his search terms. It is not a search engine, it is just an add-on. It uses established search engines such as Google and Yahoo to give their search more semantics.

For the development of a search engine, Berners-Lee suggests in (Berners-Lee et al., 2001) creating results dynamically by creating direct connectivity between low-level pieces of information. The challenge here is to find a way to represent all data types. Using this representation, when data is called, it links to all other relevant information. This process is called ”tagging”. Tagging increases connectivity between data items rather than pages. This technique faces some barriers that are difficult to be overcome. One of those barriers is the huge amount of dynamic information added by web sites around the world and the difficulty of keeping track of them and managing them all the time.
In (Oram, 2003), Andy Orman suggests an architecture for search engines that depends on metadata and is therefore called meta-search engine. The architecture uses a peer-to-peer system. It sends search queries to multiple search engines and other data resources. It then collects results and formats them for proper display. It reads HTML pages and extracts the required text. It is a meta-search engine, not a semantic search engine. It depends on results of existing search engines rather than doing an actual search.

In (Ding, 2006), Ding uses semantic search in a Peer-to-Peer (P2P) based digital library. He takes into consideration three aspects in online connected digital libraries.

**First:** What network topology should be used to cope with searching in a highly distributed network? He shows that the P2P topology is preferable over a centralized network topology in dealing with the significant growth of the number of digital libraries. The client/server method would be a bottle neck for performance. In addition, P2P provides scalability. Any peer (digital library) can join or leave the network at any time. This strengthens the system robustness.

**Second:** Ding takes into consideration semantic heterogeneity of metadata from several different libraries. Due to different ontologies used with those libraries, he uses a global ontology (RDF) to represent metadata (data about data). He applies conversion techniques to convert from diverse ontologies used into the global ontology RDF. For simplicity, he assumes that all record sets in data repositories are XML formatted. For ontology integration in his P2P network, he uses two methods for querying heterogeneous records: Global as View (GaV) method; and Local as View (LaV) method.

**Last:** Ding uses semantic search to infer the meaning of keywords used in the query from its context. He applies a lexical database (WordNet (WordNet, WordNet)) to get the required information about meaning of the used term. Broad, narrow or related terms may be used if no results are returned by searching for the current term. The system
may exploit ontology to rewrite the user query and propagate it through the network to achieve better results.

Ding considers in his work proposal that all metadata used are XML formatted. Furthermore, he converts all used ontologies into one global ontology (RDF). These are an unacceptable restriction for real systems.

In (Renuga et al., 2009), Renuga suggests a system in which a semantic search engine is able to combine content based search techniques with spread activation techniques. First, an instance graph is created concatenating properties of a node. The user expresses his/her search by keywords. The core idea of this technique is to extract knowledge of each concept of a used ontology and its relationships with all other concepts. Renuga then represents these relations by a weighted value. He applies a content-based search technique to the extracted instance graph to obtain a set of nodes corresponding to the query. These nodes are used as input in a spread activation algorithm. The same order of obtaining these nodes is used in the spread activation technique. The node with the highest activation value is considered as a concept explorer. A searched concept can be inferred depending on the strength of the relations between instances in a knowledge base. This is calculated by a weight mapping function.

Renugas suggested system depends on centralization of knowledge. The user searches only one knowledge base. In a real search engine, there are several knowledge bases to be searched to get better results. The search query is propagated to these knowledge bases and results are collected to be sent back to the user.

Ben Mustapha et al propose a framework in (Ben-Mustapha et al., 2009) to describe a semantic web search for ontology learning called ”MetaOntology”. In their approach, search query can be translated in an ontology module (a part of an ontology). It can be used by other users to guide their search using similar queries. It can also be used by the ontology itself to enrich it by learning new concepts. Contextual search system will be
multi-contextual and adaptable by queries of the users. Searchers also can participate in teaching new concepts. It is done by selecting documents relevant to a specific concept, to be used in the learning process of this concept.

Ben Mustaphas approach depends also on centralization of knowledge bases. If the user tries to search for a new concept that does not exist in the knowledge base, no results will return back. The only way for a concept to be added to the ontology is by the user himself/herself. That is not practical, because the user may not understand the concept perfectly; he/she may add a redundant concept or add incorrect definition or examples of the concept. Moreover, the user searches for a concept, which may mean that s/he does not know it.

Passadore (Passadore et al., 2009) implements a new system called "AgentSeeker". It is a search engine based on a multi-agent system. The aim of this search engine is to make document retrieval smarter and to enable finding texts semantically hidden within the user query. This system aims at indexing Internet documents and local files. It focuses especially on an enterprise context where the value of digital information is high.

AgentSeeker is not a real search engine. It is just a Meta search engine that indexes the results of other search engine such as Yahoo and Google and shows the results to the user.

Understanding the intention of a search request in a general purpose context is a very difficult process. Weber et al (Weber et al., 2009) implement a new system called "motoso.de" that tackles a semantic search problem in a small domain regarding a specific field. They choose this field to be the car domain. The system is a multi-agent based system. It depends on a domain specific ontology to analyze and interpret a users intention in the domain of cars.

Weber et al depends in their research on a single data repository to be searched. They do not specify any learning mechanism to enable adding new concepts to the data
repository. If a user searches for a concept that does not exist in the repository, no result will be returned to the user.

Bast et al introduce in (Bast et al., 2013) a semantic search engine called "Brocoli". The search in Brocoli operates on four categories: ordinary words, classes, instances and relations. The proposed system combines full text search and ontology search. For full text search, they compare document contents with exact search keywords. In ontology search, they search a database for entities (nodes) and relations (edges). Basts search engine depends on a predefined ontology and a set of documents. They do not mention updating this ontology or changing the number of documents searched. They did not indicate what happens when the search engine does not understand the meaning of a search concept.

Our proposed semantic search module depends on a multi-agent system. It is also supported by a concept learning module in case there is a new concept in the search query that the search system does not understand. This concept learning module enables learning new concepts.

2.5.2 Learning Approaches in MAS

In order for agents in Multi-agent systems (MAS) to communicate with each other, they need first to understand each other. Many researchers consider (for simplicity) that all agents in a MAS use the same ontology. This is not the case in the real world. In applications, each agent in a multi-agent system may use a different knowledge representation (ontology) compared to other agents. The real challenge here is to enable agents to understand each other. Many researchers work on these topics with different strategies, but they all argue that the main issue for an agent is to make sure that other agents have the same understanding of a concept.

In (Steels, 1998), Steels et al use in work a distributed multi-agent system that has no
central control. They also considered that the language community is open. Any agent can enter it at any time and can easily converge to the used ontology. The new agent can adopt the convention of the group of old agents and in the same way the group can adopt any new conventions added by the new agent. Steels et al also evolve all basic principles of multi-agent systems, as agents have limited knowledge and do not know anything about internal states of other agents. Agents also have their own knowledge and decide how to communicate or divide their domains.

Steels et al describe a "language game", in which each agent has to create its own ontology based on its experience of its environment. Agents communicate with each other to agree on a common ontology or shared set of lexicons to be used in their interactions. Steels et al divided agents according to their roles as a speaker (teacher) and a listener (learner).

First of all, a speaker agent decides to share the meaning of an object with other agents. It tries to create a definition of this object using some words and gives some context of that object. The speaker agent then specifies features of this object in order to conceptualize it and distinguish it from other objects in the context. The speaker sends this definition to listener agents. The listener agents in turn try to decode the definition to some interpretations and select the best one that fits with the current situation. If they follow this scenario then the game succeeds. Otherwise the game fails and a repair action should be taken.

If failure occurs, the speaker agent has to redefine that object with more discriminating words that represent the object by creating new distinctions or refining the existing one. The listeners create new linguistic conventions by creating new words or adopting the words used by the speaker. The listeners then record the success of the word and set the most preferable words that have the most success in order to be used more frequently in future.
In this methodology, agents try to agree on a common ontology or set of lexicons rather than keep their own ontology. Using a common ontology is not suitable in those situations in which an agent prefers to keep its own ontology. On the other hand, the created ontology depends on the definition of objects rather than concepts (class of objects).

Another methodology implemented by Sen (Sen and Kar, 2002) describes how an agent can teach another agent a new concept. In this methodology, a teacher agent knows nothing about the internal knowledge representation of a learner agent. This methodology does not require internal knowledge representation of the two agents (teacher and learner) to be the same.

First, a teacher chooses one of its internal concept descriptions to teach a learner agent by an iterative learning process. Initially, the teacher agent selects the most discriminating examples that belong and do not belong to the selected concept description. The teacher agent divides these examples into two sets: a training set and a testing set. The learner uses the training set to understand the concept and tries to create its description by its internal knowledge representation. The learner agent then uses the test set to classify new examples and sends the resultant classification back to the teacher for evaluation. The learner agent checks the results and prepares the next learning and testing sets. This iterative process continues until the frequency of errors done by the learner is below a specific threshold value.

Sens work has two major weaknesses. First, Sen assumes that every teacher agent knows all discriminating features describing concepts being learned, which means that all agents should use the same set of features to learn this concept. Second, the collaboration between agents is restricted to agent-to-agent communication rather than a full multi-agent collaboration.

Williams in (Williams, 2004) implements a new methodology called DOGGIE (Dis-
tributed Ontology Gathering Group Integration Environment) in order to help agents with diverse ontologies to communicate with each other. In his system, an agent communicates with its neighbour agents in order to make sure that they know a specific concept in its ontology.

The DOGGIE procedure flow is:

1. The first agent $Ag_1$ sends a message (with the name of a specific concept and some example objects that correspond to this concept) to all other agents.

2. Each agent searches its own repository for that concept name and matches the example objects sent with its own; in order to make sure that the concept name in the first agent $Ag_1$ represents the same concept in its ontology.

3. All agents reply back to $Ag_1$ by either yes, maybe, or no.

4. If the answer is no, nothing happens. If the answer is yes, the answer is accompanied with extra examples to assure $Ag_1$ that the other agent knows exactly the same concept. Agent $Ag_1$ adds to its knowledge base a new record that "Agent X knows concept Y". If the answer is maybe, agent $Ag_1$ sends some discriminating features of the concept in order to teach the other agent this concept. This can be done using Recursive Semantic Context Role Learning (RSCRL) (Williams and Ren, 2001).

In this technique, no learning occurs if the other agent does not know the concept. It also considers no overlapping between concept definitions. On the other hand, Williams considers that teacher agents know all discriminating features describing the object. Furthermore, in his work the collaboration between agents is restricted to agent-to-agent communication rather than a full multi-agent activity.
Palmisano (Palmisano et al., 2006) uses a machine learning approach to accomplish an algorithm for creating a shared ontology among different agents that use different ontologies. He tries to achieve some important aspects of a created ontology such as being automatable. Applications can integrate with each other without human intervention. Also, the system should be flexible so that any agent can communicate with the system by its own ontology. The suggested system has to be on-demand and is limited in scope.

The main purpose of Palmisano’s work is to make each agent maintain its own ontology and at the same time keep track of concept meanings of other agents it communicates with. He uses Peer-To-Peer (P2P) communication topology. He tries to build a representation of an object belonging to another ontology using its own terminology. To achieve this, he employs a concept learning paradigm (Mitchell, 1997b). He also assumes that some primitive concepts must be shared initially among agents in order to make learning of new concepts possible. Palmisano divides his data into two sets: one for examples and the other one for counter examples. He also applies a similarity measurement technique to measure the similarity between a newly learnt concept and all existing concepts in a local repository. Measuring similarity shows agents any equivalence to the new concept.

Palmisano’s work falls into ontology mapping. It maps a concept of one ontology to another concept in another ontology. There is no real learning of concepts. Also, it uses agent-to-agent communication not a full multi-agent collaboration.

Afsharchi et al (Afsharchi and Far, 2006a) try to enable an agent to learn new concepts from a group of agents. Afterwards, the learner agent represents this new concept in its own terminology without the need of a shared ontology (Afsharchi et al., 2006). They deal with a multi-agent system as a group of agents with different ontologies. Any agent of this group can learn a new concept by asking the other agents about this concept. It can learn this new concept from all of them and represent it using its own taxonomies.

The learner agent in Afsharchi’s work tries to learn a new concept that it thinks is
important to know. It sends to all agents it communicates with, the proposed features and set of examples that represent this concept. The supervisor (teacher) agent sends back some positive and negative examples of the concept in question. The learner agent uses a concept learning technique to learn this new concept. The learning can be done iteratively until a suitable definition of the concept is reached. In order to improve the efficiency of the learning, the supervisor agents may use some relations with other concepts in their repositories.

Afsharchi et al also developed a technique that can be used to improve the selection of positive examples used in the learning process (Afsharchi and Far, 2006b). In this technique, the teacher agent expands features it uses in its ontology to describe the concept by additional features to represent different viewpoints of this concept. These new viewpoints may be closer to the learner agent. Each teacher agent selects some of these examples together with negative examples in the learning process.

In the proposed system, Afsharchi et al do not explain how this concept learning system is initiated. How does the learner agent know about a new concept to ask other agents to learn it? Also, the learner agent deals with all other agents equally. We argue that there must be some kind of ranking for the teacher agents, with respect to the learner agent, according to closeness of their knowledge bases. This ranking helps in improving the quality of the learning process. It enables the learner agent resolve any possible conflict especially for large number of teacher agents with diverse ontologies and diverse domains.

Diggelen in (Diggelen et al., 2007) develops a layered approach called ANEMONE (AN Efficient Minimal Ontology Negotiation Environment). This approach establishes a minimal shared ontology between agents to be able to communicate with each other if they use different ontologies. He also assumes that agents share some minimal common ground which can then be used to learn new concepts (Diggelen et al., 2004).
Diggelen assumes that agents can communicate using on a layered approach (Diggelen et al., 2006). The upper layer uses the Normal Communication Protocol (NCP) which is used for normal communication between agents when no ontology problems arise.

If there are problems, a teacher agent selects one of its internal concepts to teach other agents. In this case, two extra layers will be added:

The first layer uses the Concept Definition Protocol (CDP). In this layer, the teacher agent tries to teach a learner agent a new concept. The teacher agent specifies a set of relations between this new concept and other shared concepts. The teacher agent then sends these taxonomical relations to the learner in order to be able to describe this new concept. If there are not sufficient shared concepts between the teacher and the learner, communication goes to the higher layer of communication uses the Concept Explication Protocol (CEP). In this layer, the teacher agent tries to teach the learner agent the new concept by specifying some positive and negative examples of this concept. The learner agent uses its own concept classifier to classify the new concept.

An important point about Diggelen’s work is that it is an environment that facilitates mapping between ontologies rather than a concept-learning environment. Furthermore, Diggelen’s work does not talk about feature diversity. Again, it uses agent-to-agent communication rather than a full multi-agent collaboration.

Bourgne et al deal in (Bourgne et al., 2012) with the problem of collaborative learning in MAS when agents have a limited storage capacity (small memory). In this case, the agent cannot keep all examples received from other agents. Their approach aims at reducing cost of accessing examples used in small and mobile devices such as smart phones, tablets, etc. Bourgne et al are more interested in what they call online concept learning using examples. In this learning process, the output is a hypothesis that is supported to cover positive examples and reject negative examples of concepts to be learnt. In their online learning technique, they depend on a stream of examples for
a concept and try to classify them. A misclassified example is considered a counter
example. Their agent is an incremental learner, i.e. if there is a new example, the agent
revises its theory rather than building a new one from scratch.

In Brougn et al research, they try to solve the problem of limited resource (bounded)
agents with limited memory by accepting only positive examples for the target concept
and reject all negative examples. In this case, the learner will not be able to precisely
define a concepts boundaries that differentiate it from other similar concepts, i.e. chances
of misclassified examples increase.

Moreover, when the number of examples exceeds a certain limit, they start removing
some old examples which may be essential in defining the concept to be learnt. They
also consider the learner to be familiar with all concepts. The learner only uses examples
to revise their definitions. Bourgn et al never speak about learning new concepts from
scratch in which the learner will not be able to classify their examples themselves.

Didandeh et al in (Didandeh et al., 2013) discuss the problem of concept learning in
multi-agent systems under a game theoretic view. They consider both learner and teacher
agents to be in competition. The teacher agent will not give all required information to
the learner agent in response to its request to learn a new concept. At the same time,
the learner agent will not send all possible information about the required concept to the
teacher as the teacher may make use of the information sent.

Didandeh’s approach is based on determining values of information pieces that agents
(learner or teacher) are going to share with other agents. They put scores on each
piece of information each agent wants to share and try to gain higher scores from learnt
information than lost information.

Didandeh et al, in their work, only discuss a special case of concept learning (learning
from competitors), where both the learner and the teacher agents do not want to share
information with each other. They do not mention if their proposed system works in
regular concept learning or not. They also claim that their concept learning approach is working in a multi-agent system environment, but their case study depends only on agent-to-agent learning and no multi-agent collaboration.

2.6 Summary

This chapter first provided definitions and examples of some basic concepts that are vital in our research, such as: agent, multi-agent system, ontology, concept and semantic web. UIMA that is used in document annotation in our system was also introduced briefly throughout this chapter. Basic information about machine learning and social networks, as applicable to our research, was also provided in this chapter.

At the end of this chapter, a comprehensive review to the relevant research has been given. The review is split into two sections. The first one was about semantic search while the second one discussed learning new concepts from different teachers. Some limitations of the different studies in the literature have been explained. The review has shown the need to new system that better employ multi-agent system in both semantic search and concept learning.

The next chapter explains our proposed system for semantic search based on concept learning using multi-agent system and depending on social networks in communications between agents of different multi-agent systems.
Chapter 3

Proposed System: Concept Learning Supported Semantic Search Using MAS Based On Social Networks

In this chapter, we provide a description of our proposed system, concept learning supported semantic search using MAS based on social networks. Our system consists of two major modules: the Concept Learning module and the Semantic Search module. Both are interleaved.

This chapter is organized as follow: Section 3.1 gives an overview of the proposed system in general; in Sections 3.2 and 3.3, we give more details about semantic search module and concept learning module respectively; finally, an analysis and design of our system is done using GAIA methodology in Section 3.4.

3.1 Proposed System

In this section, we describe our proposed system. First, we illustrate the architecture of our system. The assumptions set for our system is then presented. Afterwards, interactions between agents in multi-agent systems used in our system are illustrated. After that, we introduce the workflow for our system. Finally, we introduce the system infrastructure.
3.1.1 System Architecture

Our system consists of a group of MASs, each controlling a repository of structured and unstructured documents, see Figure 3.1. Each repository has its own concept hierarchy, i.e. ontology, and objects associated with each concept. We do not restrict the repositories to a single ontology. Agents of different MASs communicate with each other by developing a common understanding of concepts used during communication. Agents of different MASs communicate with each other via a social network. Strengths of ties between agents in the social network represent how close/far two agents are to each other. The tie strengths are updated dynamically after communication between agents.

Our proposed system is designated for local semantic search and can be used effectively in an intranet (e.g. a small scale network between different companies or universities). Our proposed system is a small scale one (the number of MASs used < 100).

During building the system for the first time, all agents from different MASs are communicating with each other in a P2P network. In order to apply social networks in our system, initial values of tie strengths are calculated by measuring the ontology similarities between ontologies used by each two agents. A threshold is set to decide if all relations in the social networks are acceptable or some of them are very weak and need to be removed. For a new MAS to join the system, an initial tie strengths values are
calculated with all existing MASs in the network to decide the friends of the new MAS.

In this thesis, we assume that the MAS for each repository is built and already exists. The development and deployment of agency are out of scope of this thesis.

3.1.2 Key assumptions

In our work, we set some key assumptions that are much weaker than assumptions set by other researchers in the area of concept learning and semantic search. Our key assumptions are:

- Ontologies used by MASs are diverse. Each MAS may use a different set of concepts and relationships and also different concept arrangements.

- MAS may use different feature sets to describe its concepts. Among these features there is a subset of some base features that can be recognized by all MASs.

- There are some common base concepts that are known by name and base feature values and/or the objects (examples) covered by them.

- MASs uses example objects as part of their knowledge structure and they can learn concepts from their example objects.

- Agents in different MASs are connected to each other via a Peer-to-Peer (P2P) network using the social network paradigm for describing their relationships.

- Each MAS is able to annotate its local documents in order to perform a semantic search.
3.1.3 General interaction scheme

As we said earlier, our system consists of a group of MASs communicating with each other. Each MAS consists of a group of agents, each with its own responsibility and an assigned task to perform. All agents in each MAS cooperate with each other to perform a general task. The main tasks of MASs in our system are: control a repository of knowledge bases; perform a semantic search with the help of other MASs in the system and return results back to the user; learn new concepts from other MASs in order to improve the performance of semantic search results, update the concept hierarchy after learning new concepts or update existing ones. Figure 3.2 shows different agents’ roles in each MAS in our proposed system (El-Sherif et al., 2010a). These roles can be defined as follows:

Figure 3.2: Roles of different agents in each multi-agent system

**PA**  This module is responsible for interactions between the user and the system.

**Query Handler**  This role involves accepting a search query and processing it by extracting concepts from it. Also it is responsible for passing the query to
Peer Finder.

**Concept Manager** This role involves finding the new concepts in the search query and broadcasting it to all neighbouring MASs to help the current MAS to learn them.

**Concept Learner** This is the agent that is responsible for the concept learning process. This role involves maintaining and confirming newly learnt concepts, including creation of taxonomies of domain of interest. Moreover, the concept learner rearranges local repositories with the newly learned concepts.

**Document Annotator** This role involves annotating documents in the local repository and filtering them according to search keywords.

**Peer Finder** This role involves detecting cooperative peers (agents) that communicate with the current MAS during both concept learning process (by sending the learning request to learn the required concept) and semantic search process (by broadcasting the query statement to search for it).

**Tie Manager** This agent is responsible for the social network tasks. This role involves keeping track of common concepts between peers and the interactions that occur between those peers in the learning process. This allows change of tie strength between peers dynamically. It is also responsible for setting the initial tie strength between agents in different MASs.

3.1.4 System workflow

A spiral-like workflow has been suggested by our team in (Far et al., 2009). This workflow incorporates both semantic search and concept learning. This workflow deals with both semantic search and concept learning as independent processes. The concept learning
process starts after the semantic search is finished. In the suggested workflow, the annotation procedure of search process in the semantic search module cannot be done on the newly learnt concepts because the learning process is initialized after finishing the semantic search process. Instead, the agent that is responsible for the semantic search, deals only with a fixed number of predefined concepts in the ontology. This case is not the best representation of our suggested system.

The workflow suggested in (Far et al., 2009) needs to better represent the process of our system. The workflow needs to be enhanced to better represent the newly learnt concepts of the concept learning module. The modified spiral workflow and its suggested scenario are shown in Figure 3.3.

![Figure 3.3: The Spiral-like workflow of the Concept Learning and Semantic Search](image)

The process of this workflow works as follows (El-Sherif et al., 2010a):

- The system is initiated by the user when s/he sends a search query to an agent within one of MASs in the system (we call it local agent). This step is the initialization of our system.

- Concepts are extracted from the search query by the local agent.
• The concepts in the search query are compared with those defined in the ontology in the local repository. This comparison is essential to enable the local agent identify the new concepts in the search query that are not defined in the agent’s local repository.

• If new concepts are found in the search query, the local agent (learner) requests other agents in other MASs (teachers) to teach it these new concepts. (This step represents the initialization of the concept learning process.)

• The concept learning process is repeated until the local agent learns all new concepts adequately.

• After learning all new concepts, the concept hierarchy in the local repository is reorganized to add the new concepts in their proper position.

• The annotation procedure is then performed on the fly on all the concepts in the local repository, old and newly learned concepts. UIMA is used to enable search and classification within repository.

• At the same time, the local agent broadcasts the search query to agents in the other MASs (remote agents in this case) to search their local repositories and send back the search results.

• The local agent collects the returned results from the remote agents and ranks them using social networks before sending them to the user.

This workflow represents the interaction between the concept learning module and the semantic search module in our system as long as there are new concepts in the search query need to be learned. If the search query entered by the user does not contain any new concepts to be learned, this workflow is suspended and the semantic search module operates regularly without interacting with the concept learning module.
3.1.5 System Infrastructure

In our system, we need a powerful communication system to enable agents in different multi-agent systems to interact with each other. We will use the decentralized Peer-to-Peer (P2P) network topology. P2P network provides more flexibility for any MAS to get in or out from the system. It also avoids the data transfer bottleneck if the number of MASs in the system increases. We also use the social networks paradigm in defining relationships between agents in different MASs in our system.

Figure 3.4 shows an illustrative example of ten agents connected with each other in a social network. In this example, we omit relationships between agents if tie strength between them is very low (i.e. a very weak relationship). The thickness of the line represents the strength of tie between each two agents (i.e. the thicker the line the stronger the tie). For example, the tie strength between agent A and agent C is higher than it between agent A and agent B and much higher than tie between agent B and agent D. We can notice also that the strength of ties is not similar between two agents. For example, strength of ties from agent A to agent B is strong but strength of ties from agent B to agent A is much weaker (asymmetric social network).
Social networks is very helpful in solving problems that may be encountered in either semantic search process (e.g. ranking returned documents) or concept learning process (e.g. resolving conflicts and deciding number of positive and negative examples from each teacher).

### 3.2 Semantic Search Module

The key requirements for our proposed semantic search module are given below:

1. Agents from different MASs must be able to communicate with each other and be able to exchange information.

2. Agents must be responsible for organizing data in their own repository by annotating documents in their local repositories.

3. Agents must be able to cooperate with each other to propagate the search query among each other.
4. Agents must be able to rank the result documents, based on strength of ties between them, before send it to the user.

5. Agents should be able to hide the complexity of semantic search process from the user.

3.2.1 Semantic Interoperability

Traditional search engines depend on the number of occurrences of a word in a document. Semantic search depends on understanding the meaning of concepts used in the context of other words. It then tries to retrieve documents related to these concepts. The backbone of semantic search is semantic interoperability (Euzenat, 2001). Semantic interoperability is the main ingredient for notation extraction from a search phrase.

In order to enable multiple agents with different ontologies to communicate with each other, they need to understand each other. Euzenat (Euzenat, 2001) defines semantic interoperability as: ”the faculty of interpreting knowledge imported from other languages at the semantic level, i.e. to ascribe to each imported piece of knowledge the correct interpretation or set of models”. Euzenat defines five layers of semantic interoperability to understand the expression coming from a system by another one with different ontology. The layers of semantic interoperability are (see Figure 3.5):
**Encoding Layer** It is the base layer which provides the ability to segment representation in characters, i.e. it defines an encoding format. ASCII and Unicode are mainly encoding formats.

**Lexical Layer** It provides the ability to segment representation in words/symbols, i.e. tokenize the search phrase. At this layer, important identifiers of ontology components are identified.

**Syntactical Layer** It provides the ability to structure the representation in structured sentences (formula or assertions). This layer gives the ability to identify concepts by structuring words following a grammar. It is able to extract the concepts from a structured representation.

**Semantic Layer** It provides the ability to understand propositional meaning of representation.

**Semiotic Layer** It provides the ability to understand meaning in context (specific domain) of representation.

The human communication style follows the above architecture in understanding each
other. Each layer can be reached only if the lower layers are traversed. For example, people cannot understand each other unless they have a common language to communicate with. They have to clarify the meaning of concepts used and use them in a correctly structured sentence. Our proposed system tries to simulate that architecture in communicating between two agents that use different ontologies. Figure 3.6 represents a communication between two peers/agents based on semantic interoperability layers. This layered architecture reduces the complexity by breaking the complex semantic interoperability problem into smaller problems.

![Communication between two peers based on Semantic Interoperability](image)

The communication between two peers is governed by some rules:

- Each layer can talk only with the peer layer on the remote side. A conversation conforms to the rules of the agreed language of communication, grammar representation and encoding standards.

- A search phrase can be initiated at any layer. The search phrase is relayed to the lower layers until it reaches the encoding layer which is responsible
for direct communication between two peers.

- Each layer adds annotation information to the search phrase. At the end, the whole package is sent to the remote side. At the remote side, each layer interprets the added information of the peer layer.

- Each layer relies on ontology located in the same layer of ontology spectrum (Daconta et al., 2003). Figure 3.7 represents the adjacent relations between each layer in semantic interoperability and its corresponding layer in ontology spectrum.

![Figure 3.7: The relationship between layers of semantic interoperability and ontology spectrum](image-url)

3.2.2 Semantic Search Process

As we introduced in Chapter 2, IBM (later Apache’s) UIMA (UIMA, UIMA) offers a practical development platform to support dynamic document annotation within a semantic search engine. UIMA supports configuring and running pipelines of annotator components. These annotator components do the actual work of analyzing the unstructured information. Users can write their own annotators, or configure and use pre-existing
annotators. For example, in order to identify documents that belong to "Computer Programming", we define the annotation logic as:

\[
\text{< Language + Program | C++ | JAVA >}
\]

The logic can be interpreted as: "if term Language and Program, or C++ or Java have been found in the document, then this document is determined to be a target". To implement this logic in the UIMA annotation system, an aggregate annotator can be created which includes several primitive annotators. In addition to some pre-existing natural language processing annotators, such as tokenizer annotator to create token and sentence annotations, dictionary annotator to handle synonyms, etc, users write their own annotator to implement this logic to detect entities for "Computer Programming". The UIMA framework manages these annotators and the data flow between them to achieve the final aggregate result. In addition, UIMA provides capabilities to wrap annotators as network services. That is, repositories can utilize publicly accessible annotators to annotate their own documents locally (Zhong, 2008).

In our research, each repository uses its own ontology. Annotations can be defined for each concept in the repository using the UIMA annotation system. For example, in the domain of university courses catalogue, we can have the concept "Computer Science" which includes all the courses offered in a Computer Science department. Under this concept, different annotations can be built to further clarify some particular specializations the user wants to search, such as "Computer Programming", "Database System", etc.

In this section, we present an example to show the importance of tie strengths in our semantic search system. Figure 3.8 illustrates the process of semantic search followed in our system.
The semantic search is initiated by a user. A user starts by entering the keywords s/he wants to search for using our semantic search module. The keywords are provided to an agent in one of the MASs (local agent) in the system. The local agent first checks the keywords to identify any new concepts. If the local agent found new concepts, it initializes the concept-learning module by asking other agents (teachers in this case) from neighbour MASs for this new concept. The choice of teachers depends on the tie strengths between the local agent and its neighbours. The teachers teach the local agent the new concept by sending illustrative examples representing this concept. More details about learning new concepts from different teachers (our proposed concept learning module) are illustrated in the next section.

The learning process ends when the local agent is sure that all keywords exist in its ontology. The next phase of the system then starts, which is the document annotation. The local agent annotates its own repository using keywords in the search statement. At the same time, the local agent propagates the search query to all other agents in neighbouring MASs (remote agents in this case). The local agent searches the annotated
repository for the search query and returns the results (local results). Each of the remote agents searches its own repository by annotating it with the search keywords. Then each remote agent returns more results (remote results).

After gathering all search results from all remote agents, the local agent removes any redundant documents, if any. Afterwards, the local agent tries to rank the results. The purpose of this ranking is to decide the order in which the local agent delivers the results to the user. The ranking process depends also on the tie strength between the local agent and each of the remote agents. The stronger the tie between the local agent and the remote agent, the higher the rank of the result documents returned by that remote agent. Finally, the local agent returns all results (local results and remote results) to the user. Figure 3.9 is a screen shot of our semantic search interface.

![Illustration of Semantic Search Procedure](Image)

Figure 3.9: Illustration of Semantic Search Procedure (Zhong, 2008)

### 3.3 Concept Learning Module

The key requirements for our proposed concept-learning module are given below:

1. Agents in different MASs must be able to communicate with each other
and be able to exchange information.

2. Agents must be able to learn and teach new concepts from each other.

3. Agents must be able to identify positive and negative examples for each concept defined in their ontologies.

4. Agents must be responsible for organizing data in their own repositories based on new learnt concepts and updated concepts.

5. Agents should be able to hide the complexity of concept learning process from the user.

In our framework, an object contained in an ontology is specified by a set of features (attributes) and values corresponding to those features. A feature denoted by \( f_i \) is a discrete valued variable. The set of values that feature \( f_i \) can have is called the domain of that feature, \( D_i \). The domain \( D_i \) of a certain feature contains all the possible values for that feature \( D_i = \{v_{i1}^1, v_{i2}^2, v_{i3}^3 \cdots v_{in}^n\} \). To represent an object within an ontology, it is represented by a number of feature/value pairs. For example, an object \( o \) can be represented as: \( o = \{(f_1 = v_{i1}^1), (f_2 = v_{i2}^2), (f_3 = v_{i3}^3), \cdots \} \) where \( f_1, f_2 \) and \( f_3 \) are features and \( v_{i1}^1, v_{i2}^2 \) and \( v_{i3}^3 \) are values from the domains \( D_1, D_2 \) and \( D_3 \) respectively.

In order to represent a concept in an ontology, it will be as a simple extension of object representation allowing internal disjunction of values within a feature. This addition is necessary to cover more than a single object by this concept. A concept \( C \) using this ontology is defined as \( C = \{(f_1 = v_1), (f_2 = v_2), (f_3 = v_3), \cdots \} \). Value \( v_i \) is either empty to represent a "don’t care" condition or has one or more values of those of the domain of this feature.

Under this representation, an object is an instance of a concept and object description language is a subset of concept description language.
3.3.1 Illustrative Example

For clarification, we provide a simple example to make this representation of concepts in an ontology clear. To represent the concept fruit in an ontology, we consider the set of features \{ color, size, pulp and rind \} as the discriminating features of concepts in our ontology with domains \{ red, green, orange, yellow \}, \{ small, medium, large \}, \{ juicy, dry \}, and \{ thin, thick, very thick \}, respectively. Definitions of some fruits can be found in table 3.1.

Table 3.1: An example of some fruit concepts

<table>
<thead>
<tr>
<th>Object (Fruit)</th>
<th>Color</th>
<th>Size</th>
<th>Pulp</th>
<th>Rind</th>
</tr>
</thead>
<tbody>
<tr>
<td>Water-melon</td>
<td>Green</td>
<td>Large</td>
<td>Juicy</td>
<td>Very thick</td>
</tr>
<tr>
<td>Orange</td>
<td>Orange</td>
<td>Medium</td>
<td>Juicy</td>
<td>Thick</td>
</tr>
<tr>
<td>Banana</td>
<td>Yellow</td>
<td>Medium</td>
<td>Dry</td>
<td>Thick</td>
</tr>
<tr>
<td>Grapefruit</td>
<td>Yellow</td>
<td>Medium</td>
<td>Juicy</td>
<td>Thick</td>
</tr>
<tr>
<td>Grape</td>
<td>red</td>
<td>small</td>
<td>juicy</td>
<td>Thin</td>
</tr>
</tbody>
</table>

To express "Orange" from the above ontology:

\[ \text{Orange} = \{(\text{color} = \text{orange}), (\text{size} = \text{medium}), (\text{pulp} = \text{juicy}), (\text{rind} = \text{thick})\} \]

For simplicity, we may omit feature names and state only their values. In this case, the orange object will be:

\[ \text{Orange} = \{\text{orange}, \text{medium}, \text{juicy}, \text{thick}\} \]

On the other hand, we can represent a concept "Citrus" that contains both orange and grapefruit as:

\[ C_1 = \{(\text{color} = \{}), (\text{size} = \text{medium}), (\text{pulp} = \text{juicy}), (\text{rind} = \text{thick})\} \]

If we want to add "Lemon" to this ontology, the definition of this concept will be as follows:
$C_1 = \{(\text{color} = \{}), (\text{size} = \text{small, medium}), (\text{pulp} = \text{juicy}), (\text{rind} = \text{thick})\}$

3.3.2 Concept learning process

Despite the fact that all MASs can teach or learn from other MASs, for simplicity, we consider one MAS ($Ag_L$) as the one that wants to learn a new concept (learner) and the other MASs ($Ag_1, Ag_2, \ldots, Ag_n$) are its teachers. $Ag_L$ has an ontology $O_L$ and knows some features $F_L$. On the other hand, each teacher $Ag_i$ has its own ontology $O_i$ and knows some other features $F_i$. $Ag_L$ wants to learn a new concept $C$ which is known to teacher $Ag_i$ in its ontology $O_i$. $C$ in $O_i$ has its own data area and a set of positive examples $pex_i^C$ which is a subset of all objects in the world ($U$). $Ag_i$ uses $pex_i^C$ to teach $Ag_L$ the concept $C$. $Ag_i$ uses also some negative examples $nex_i^C (= \{U - pex_i^C\})$ to complete the learning process by defining the border of the learnt concept.

We can say that in a society of $n$ agents $Ag_1, Ag_2, Ag_n$ in different MASs, each agent $Ag_i$ has a concept $C$ and has some features ($f_i^C$) to clarify this concept and possesses some supporting examples ($E_i^C$) of this concept. In addition, each agent has its own ontology ($O_i$) to represent the concept $C$. Each agent uses a learning algorithm ($L_i$) to conceptualize that concept. The agent $Ag_i$ can be represented as:

$$Ag_i = \{L_i, E_i^C, f_i^C, O_i\} \quad (3.1)$$

Learning Algorithm ($L_i$)

Each agent (learner) may learn new concepts from other agents (teachers) in the system that has a direct connection with the learner. The learning process in our system is a supervised learning technique. This supervision is accomplished by the teacher to be able to monitor the learning process and to make sure that the learner has adequately learnt the concept. Moreover, this supervision gives the teacher the authority to answer incoming queries regarding the classification of certain training examples.
Set of Examples ($E_i^C$)

The teacher uses a set of training examples (objects) to teach the learner about new concepts. An example set is divided into positive and negative examples (Afsharchi and Far, 2006b). Figure 3.10 shows a representation of the example space. This figure shows that negative examples of a concept are much larger than the positive examples. The justification is that most other concepts can be considered as negative example source to the current concept. Careful selection of positive and negative examples during learning is very important. In Figure 3.10, the example $e_2$ is a border positive example. It is near to the negative example space. Using $e_2$ in the training set is important to correctly clarify the border of the concept. A proper selection of positive examples increases the accuracy of definition of the learnt concept. In selecting positive examples we need lots of core examples (e.g. $e_1$) and also some border examples (e.g. $e_2$) in order to correctly define the border of the concept to be learnt. The same has to be taking into consideration in selecting negative examples. Each teacher needs to select some border negative examples (e.g. $c_j$) in order to help clarifying the outer border of concept definition. It also may select some general negative examples. We will discuss selecting negative examples in detail later in section 3.3.4.

![Figure 3.10: Representation of example space (Afsharchi, 2007)](image)

After completing the learning process, the learner can label these examples and store
them in its local repository accompanying the newly learnt concept.

Set of Features ($f^C_i$)
As stated previously, representation of an object is done using feature/value pairs. Features here play a vital role in the learning algorithm. Each agent uses a different ontology. A teacher agent represents its understanding of a concept by the features it considers for training examples.

Ontology ($O_i$)
We believe that using a common ontology within different multi-agent systems (MAS) to represent concepts is neither realistic nor practical. In the real world, each MAS uses its own ontology to represent its knowledge which differs from other ontologies used by other MASs. Ontology in our proposed system uses a conjunction of meta-concepts and fine-grained concepts. Meta-concepts are those concepts that represent more general objects. Fine-grained concepts are the leaves of ontology hierarchy that refer to more detailed objects.
3.3.3 Concept learning workflow

Figure 3.11: Flow diagram of tasks in the concept learning module

Figure 3.11 is a flow diagram of the concept learning module. The steps of the concept learning process are as follow:

- The concept learning process is initialized by a learner. The learner sends a request to all agents it is related to in a social network. The learning request contains information available about the concept to be learnt (e.g. its name, identifying keywords, conceptual information, etc).

- Each teacher receives the learning request.

- Each teacher searches its local repository for the best matching concept. In order to select the best matching concept, each teacher calculates for each concept in its local repository the value of $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ which is the ratio between number of examples that satisfy the search keywords and total number of examples that represent each concept. The best matching
concept is the one with the highest value for \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \).

\[
\text{sim}(q_{\text{spec}}, C_{\text{best}}) = \max_{i} \frac{\text{number of examples satisfy search keywords in concept } C_i}{\text{total number of examples for concept } C_i}
\] (3.2)

- After finding the best matching concept, each teacher agent selects some positive (+ve) and negative (-ve) examples to represent this concept.

- The teachers send their example sets to the learner agent.

- The learner collects all example sets from all teachers.

- The learner tries to detect if there are any conflicts between the examples. In order to resolve this conflict, the learner sends another request to all teachers to vote against examples of conflict (i.e. decide if those examples are positive or negative examples according to their ontologies) (Afsharchi et al., 2006).

- All teachers vote against the examples of conflict.

- All teachers send their votes back to the learner.

- The learner collects all votes and resolves the detected conflicts (the technique used for conflict resolution is explained later in this section).

- After resolving all conflicts, the learner uses the positive and negative example sets to learn the new concept.

- The learner updates its local repository by adding the newly learnt concept to its proper location in the ontology.

- The learner updates the strengths of all ties between it and all teachers based on the interactions that occurred between them during the learning
and the closeness between their ontologies. (More details about calculating tie strengths between agents in social networks and the factors that affect it are presented in chapter 4.)

3.3.4 Conflict Resolution

Figure 3.12 is an illustration of a conflict that may occur during the learning of a new concept from several teachers. In this case, there are three teachers $Ag_1$, $Ag_2$, $Ag_3$. Each of them sends a positive example set represented by an oval in Figure 3.12. Any example outside an oval is considered either a negative example or an undecided one with respect to that teacher. The shaded area in Figure 3.12 (the intersection area) is the set of examples that are considered as positive examples by all the three teachers. The figure contains also some conflict areas that contain examples that may be considered as positive examples by one teacher but as negative or undecided examples by other teacher/s. Any example that happens to be in any of these areas are considered as a conflict example and need to be resolved with the help of all teachers.
The learner can resolve this conflict using social networks. A social network allows different agents from different MASs to communicate with each other by different tie strengths (Gilbert and Karahalios, 2009a) (Onnela et al., 2007). In the case of the conflict stated above, the learner may ask all teachers to vote for this example. The learner counts votes according to tie strengths with each teacher, using equations 3.3 and 3.4.

\[ PosV^E = \sum_{vi} \omega_{1i} PosV_i^E \]  \tag{3.3} \]

\[ NegV^E = \sum_{vi} \omega_{1i} NegV_i^E \]  \tag{3.4} \]

Where, \( PosV^E \) and \( NegV^E \) are the total positive and negative votes for example \( E \) respectively; \( \omega_{1i} \) is the tie strength between the learner and teacher \( Ag_i \). \( PosV_i^E \) and \( NegV_i^E \) are positive and negative votes of teacher \( Ag_i \) for example \( E \). Using positive

---

Figure 3.12: Illustration of conflict occurs during concept learning
and negative vote results, the learner $Ag_L$ can resolve this conflict and can also decide if example $E$ is a positive or a negative example from the perspective of the newly learnt concept and completely define the concept $C$.

3.3.5 Selecting positive and negative examples

After receiving the initial query to learn a new concept $C_{goal}$ from a learner $Ag_L$, the teacher ($Ag_T$) tries to find a concept $C_{best}$ that best matches the information in the initial query, in order to answer the learning request received. After deciding $C_{best}$ the teacher $Ag_T$ needs to select some positive and negative examples in order to help the learner $Ag_L$ to learn the required concept $C_{goal}$ as we discussed earlier in this section.

Selecting positive and negative examples is a crucial step that affects the accuracy of the learning process. Generally, the traditional way to select positive examples is random selection from a pool of candidate objects (i.e. examples) that represent the chosen concept $C_{best}$ taking into consideration a proper distribution of examples selected to represent the complete meaning of $C_{best}$. Both core examples and border examples need to be included in the example set as described earlier in Figure 3.10. For negative examples, the traditional way is to use the structure of the ontology to select objects from siblings of the selected concept $C_{best}$. This works well when the selected concept $C_{best}$ has the same meaning (or is very close to) the required concept $C_{goal}$. Otherwise, the selected positive and negative examples could negatively affect the learning process. By choosing positive and negative examples, we follow the mechanism proposed in (Yang, 2010). This mechanism depends on the value of $\text{sim}(q_{spec}, C_{best})$ (equation 3.2), which represents the similarity between $C_{best}$ and the information of $C_{goal}$ sent by the learner agent. A threshold value is set.

If $\text{sim}(q_{spec}, C_{best})$ is higher than this threshold, $C_{best}$ is very similar to $C_{goal}$. In this case, we can choose positive examples randomly from positive objects that represent
$C_{\text{best}}$. Negative examples are chosen from positive objects of siblings of $C_{\text{best}}$ as shown in Figure 3.13.

![Diagram](image.png)

Figure 3.13: Selection of positive and negative examples if $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ is greater than selected threshold

On the other hand, if $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ of chosen $C_{\text{best}}$ is less than the specified threshold, there is some intersection in meanings between $C_{\text{best}}$ and $C_{\text{goal}}$, but they are not very close to each other. This means that some objects of $C_{\text{best}}$ represent the meaning of $C_{\text{goal}}$ but others do not. In this case, we select the positive examples from those objects that satisfy the search query. The negative examples are chosen from the rest of the positive objects that represent $C_{\text{best}}$ as shown in Figure 3.14.
Figure 3.14: Selection of positive and negative examples if $\text{sim}(q_{\text{spec}}, C_{\text{best}})$ is less than selected threshold

The number of positive and negative examples chosen from each teacher $Ag_T$ depends on how close this teacher is to the learner $Ag_L$ (i.e. the strength of tie between $Ag_L$ and $Ag_T$), because tie strength reflects how much $Ag_L$ trusts and depends on $Ag_T$. The number of positive and negative examples sent by each teacher is proportional to the tie strength between the learner and that teacher. The stronger the tie between the learner and the teacher, the larger the number of positive and negative examples sent by that teacher. After selecting a positive and a negative example set, each teacher sends its sets to $Ag_L$ to complete the learning process.

3.4 Prototype of our System

In this section, we introduce an analysis of our proposed system. We choose the GAIA methodology (Wooldridge et al., 2000) to guide the development of the system. GAIA is
a methodology for agent based system analysis and design. It emphasizes on social-level abstraction. It is detailed enough to help us go directly from the analysis and design stage to the implementation stage in a systematic way. GAIA encourages us to think of the system as a process of organizational design. Using GAIA we obtain main models of analysis and design of our proposed system, starting from system requirements to acquaintance model.

3.4.1 System Requirements

The main requirements for the system are:

- Multi-agent systems must be able to form cooperative groups through social activities. They should be able to:
  
  1. Register to a group.
  
  2. Access each other to get information required to initialize a conversation.

- Multi-agent systems must be responsible in organizing data in their own repository. They must be able to:
  
  1. Annotate documents in the local repository statically or dynamically using keywords.
  
  2. Recategorize the repository.

- Multi-agent systems should cooperate with each other to be able to learn and teach concepts properly. At the same time, they should hide the complexity from the user. They must be able to:
  
  1. Learn new concepts from other teachers.
2. Update concepts’ definitions in their repositories with new definitions or extra features or examples.

3. Teach concepts to other learners using their learning algorithms.

4. Recategorize their local repository according to updates that occur in concepts (add new concept or update old concept).

3.4.2 GAIA Analysis Process

As a first step in our system analysis, we need to specify some operational guidelines of our system to be taken into consideration while identifying the roles of the system.

- Identify stakeholders of the system, i.e. those who will interact with the system and those inside the system that will be responsible of this interaction.

- Identify the roles of the system parts that perform the major functions.

- Specify if there are secondary roles to support or help the main roles or not.

Use Case Diagram

Using the above guidelines, we can identify the main roles of the system as: the Searcher, the QueryHandler, the ConceptLearner, the PeerFinder, the ConceptManager, the GroupManager, the Annotator, the TieRanker and the TieOperator. Before using GAIA methodology to analyze the system, we present here the Use Case diagram of the system showing the main roles of the system listed above and their main use cases (Figure 3.15).
Based on these roles, we build our system models using the GAIA modeling system. In the Analysis Phase, we define two models according to the GAIA methodology: the Role Model and the Interaction Model.

Figure 3.15: Use case diagram of the system
The Role Model

The role model is used to describe the key roles in the system. A role can be considered as an abstract description of the expected function of the entity. Here, we present the role schema of four major components of the system: the QueryHandler, the ConceptLearner, the Annotator and the TieOperator.

<table>
<thead>
<tr>
<th>Role Schema: QueryHandler (QH)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td>This role involves accepting search query and processing it by extracting concepts from it. Also it is responsible for broadcasting the query statement to all the neighbour repositories.</td>
</tr>
<tr>
<td><strong>Protocols and Activities:</strong></td>
</tr>
<tr>
<td>ExtractConcepts, AcceptSearchQuery, ReturnResults, BroadcastSearchQuery</td>
</tr>
<tr>
<td><strong>Permissions:</strong></td>
</tr>
<tr>
<td>Reads Query statement //What user wants to search for</td>
</tr>
<tr>
<td>Sends Query Statements //The search Query; to be searched for</td>
</tr>
<tr>
<td>Returns Documents //The documents that satisfy user requirements</td>
</tr>
<tr>
<td><strong>Responsibilities:</strong></td>
</tr>
<tr>
<td>Liveness:</td>
</tr>
<tr>
<td>QueryHandler = (AcceptSearchQuery</td>
</tr>
<tr>
<td>Safety:</td>
</tr>
<tr>
<td>QueryStatement &lt;&gt; nil</td>
</tr>
<tr>
<td>Role Schema: ConceptLearner (CL)</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td>Description:</td>
</tr>
<tr>
<td>This role involves finding the new concepts in the search query and broadcast it to all neighbour agents in order to be learnt.</td>
</tr>
<tr>
<td>Protocols and Activities:</td>
</tr>
<tr>
<td>SpecifyNewConcepts, LearnNewConcept, FindPeerInformation</td>
</tr>
<tr>
<td>Permissions:</td>
</tr>
<tr>
<td>Generates NewConcepts</td>
</tr>
<tr>
<td>Sends NewConcepts</td>
</tr>
<tr>
<td>Responsibilities:</td>
</tr>
<tr>
<td>Liveness:</td>
</tr>
<tr>
<td>ConceptLearner = (SpecifyNewConcepts</td>
</tr>
<tr>
<td>Safety:</td>
</tr>
<tr>
<td>QueryStatement Is Legal = false =&gt; Documents = nil</td>
</tr>
<tr>
<td>QueryStatement &lt;&gt; nil</td>
</tr>
<tr>
<td>Role Schema: Annotator (AN)</td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td>Description:</td>
</tr>
<tr>
<td>This role involves annotating the documents in the local repository and filtering them according to the search keywords, then returning the filtered documents.</td>
</tr>
<tr>
<td>Protocols and Activities:</td>
</tr>
<tr>
<td>AnnotateDocuments, ReturnDocuments, RearrangeLocalRepository</td>
</tr>
<tr>
<td>Permissions:</td>
</tr>
<tr>
<td>Reads QueryPhrase //The entered query statement</td>
</tr>
<tr>
<td>Generates AnnotatedDocuments //All annotated Documents</td>
</tr>
<tr>
<td>Changes LocalRepository //According to the keywords used for annotation</td>
</tr>
<tr>
<td>Responsibilities:</td>
</tr>
<tr>
<td>Liveness:</td>
</tr>
<tr>
<td>Annotator = (AnnotateDocument</td>
</tr>
<tr>
<td>Safety:</td>
</tr>
<tr>
<td>Search Query &lt;&gt; nil</td>
</tr>
<tr>
<td>QueryPhrase IsLegal = false =&gt; Documents = nil</td>
</tr>
</tbody>
</table>
Role Schema: TieOperator (TO)

Description:
This role involves changing the strength of tie between two repositories according to common concepts and the interaction between them.

Protocols and Activities:
IncreaseTieStrength, DecreaseTieStrength

Permissions:
Changes TieStrength //Strength of ties between peers

Responsibilities:
Liveness:
\[ \text{TieOperator} = (\text{IncreaseTieStrength} \mid \text{DecreaseTieStrength}) \]

Safety:
True

The Interaction Model
This model represents the dependencies and relationships between various roles in the system. This interaction between various roles is expressed by protocols. In order to define the interaction between agents, we need to define protocols associated with each role.

Here we identify some protocols associated with the ConceptLearner and the Annotator roles:

**ConceptLearner (CL):**
LearnNewConcept:
GetConcept
QDB  CL
Get the concepts in the search statement

SpecifyNewConcepts
QDB  CL
Identify the new concepts

FindPeers
CL  PF
Find peers of the local actor

LearnNewConcept
CL  CM
Send new concept to be learnt

*Annotator (AN):*

ReturnAnnotatedDoc:
3.4.3 GAIA Design Process

GAIA design process involves three models: the Agent Model, the Service Model and the Acquaintance Model.

The Agent Model

The purpose of this model is to specify the various agent types that will be used in each MAS in the system and the number of agents released for each agent type during runtime. The agent model is represented in Figure 3.16.
The Service Model

This model aims at specifying the main services associated with each role. The main purpose of this model is to specify the main properties of each service. Some of the major services of the system are indicated in Table 3.2.
<table>
<thead>
<tr>
<th>Service</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Pre-Condition</th>
<th>Post-Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExtractConcepts</td>
<td>Search Statement</td>
<td>Concepts</td>
<td>Search Statement &lt;&gt; nil</td>
<td>Concepts &lt;&gt; nil</td>
</tr>
<tr>
<td>LearnNewConcept</td>
<td>Positive and Negative Examples</td>
<td>Definition of New Concept</td>
<td>Examples &lt;&gt; nil</td>
<td>Definition &lt;&gt; nil</td>
</tr>
<tr>
<td>AddConcept</td>
<td>New Concept</td>
<td>Updated Taxonomy</td>
<td>New Concept &lt;&gt; nil</td>
<td>True</td>
</tr>
<tr>
<td>DoAnnotation</td>
<td>Row Documents &amp; Keywords</td>
<td>Annotated Documents</td>
<td>Documents Satisfy Search Condition</td>
<td>True</td>
</tr>
<tr>
<td>Increase</td>
<td>Old Tie Strength &amp; Interaction Info &amp; Similarity degree</td>
<td>Tie Strength Increased</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>TieStrength</td>
<td>Annotated Documents</td>
<td>Filtered Documents</td>
<td>Annotated Documents &lt;&gt; nil</td>
<td>Filtered Documents &lt;&gt; nil</td>
</tr>
</tbody>
</table>

The Acquaintance Model

This model defines the communication links exists between agent types. The system acquaintance model is shown in Figure 3.17.
3.5 Summary

This chapter explained our proposed system of concept learning supported semantic search using multi-agent system based on social networks by introducing system architecture, main assumptions, interaction scheme, suggested spiral like workflow and system infrastructure using social networks. Our system consists of two modules: semantic search module and concept learning module. Both modules have been explained in detail throughout this chapter. At the end of this chapter, part of analysis and design done for our system has been presented, such as: system requirements, use case diagram, role models of major components of the system, some interaction scheme models of some roles in the system, agent model, service model and acquaintance model.

The interactions between agents in our system are done using social networks. These interactions depend on tie strengths between each two agents. A mathematical model using Hidden Markov Model is implemented in next chapter to measure tie strengths between nodes in a social network.
Chapter 4

Calculating Tie Strengths in a Social Network Using Hidden Markov Model

In our research, we propose a multi-agent-based semantic search system supported by ontological concept learning and contents annotation. The agents in this system are communicating with each other via a social network. One of the most interesting features of social networks is the tie strength between the actors/nodes of the network. The strength of ties (relationships) between actors in a social network changes dynamically to represent the closeness between each two actors in the network. In this chapter, we discuss how to calculate the tie strength in a social network we used in our system and the factors that affect its value. We use Hidden Markov Model (HMM) to calculate the tie strengths between agents and their corresponding ontologies. This chapter is organized as follow: in Section 4.1, we introduce a brief introduction about Hidden Markov Model; Section 4.2 illustrates factors that affect strengths of ties between two actors in a social network; finally, in Section 4.3, we discuss our proposed methodology of calculating tie strength in a social network using HMM.

4.1 Hidden Markov Models (HMM)

The basic theory of HMMs was published by Baum et al in Baum and Petrie (1966, 1971) in the 1960s and 1970s. The use of hidden states in the model and the simple structure of Markov chains make HMMs generic enough to solve a variety of complex problems.

A Hidden Markov Model can be defined as ”A doubly stochastic process with an underlying stochastic process that is not directly observable (it is ”hidden”) but can be
observed only through another stochastic process that produces the sequence of observations” (Cappé et al., 2005). For simplicity, we can say that HMM is a Markov chain \( X \) that is observed in a noise. This Markov chain \( X \) is hidden. What is observed by the observer is another stochastic process \( Y \) linked to \( X \). All statistical analysis should be done on \( Y \) as \( X \) is hidden.

Basic elements of a HMM are:

1. The number of nodes in the model \( S \).
2. Matrix of transitions probability \( A = \{a_{ij}\} \).
3. Matrix of observations value \( B = \{b_{jk}\} \).
4. The initial state values \( \pi = \{\pi_i\} \).

The Hidden Markov Model (HMM) can be defined as:

\[
\lambda = (S, A, B, \pi)
\]  

(4.1)

HMM can be used to solve three basic types of problems (Rabiner, 1989):

1. Calculating the probability of the observation sequence given the model.
   Given the observation sequence \( O = \{o_1o_2\cdots o_t\} \) and the model \( \lambda = (A, B, \pi) \), the model tries to calculate the probability of the observation sequence given the model \( P(O \mid \lambda) \). This is an evaluation problem. Both the model and the observations are available and we want to make sure that these observations can be obtained by the given model.

2. Calculating sequence state explaining the observation. Given the observation sequence \( O = \{o_1o_2\cdots o_t\} \) and the model \( \lambda = (A, B, \pi) \), the model tries to get the sequence state \( Q = \{q_1q_2\cdots q_t\} \) that best explains the observations. In this problem, the model tries to uncover the hidden part,
represented by the state transition matrix, by finding the correct state sequence.

3. Optimizing the parameters of the system to best describe how the observations come about. Given the observation sequence \( O = \{o_1 o_2 \cdots o_t\} \), the model parameters \( \lambda = (A, B, \pi) \) have to be adjusted to maximize \( P(O \mid \lambda) \). In this type of problem, the model attempts to optimize the parameters of the system to best describe how the observations come about. This is the type of problem we are aiming to solve using HMM in our system.

### 4.2 Tie strength in social networks

In our system, we try to enable agents from different MASs to communicate with each other and to understand each other by interacting through a social network. In this social network, the strengths of relationships (ties) between agents are not the same. The strength differs from a tie to another depending on several factors, as will be indicated later in this section. The closer the tie indicates the better contents can be shared between agents and more understanding during their communication. Another important property of a social network is that the strength of the ties in the network is dynamic. It can change over time.

Earlier social networks only consider the existence of a tie (i.e. relationship) between two actors/agents (the strength of ties between two actors is either one or zero). Recently, more attention has been paid to the tie strength. The strength of ties in a social network is an intrinsic property of the network. The values of the tie strengths do not change with the application in which the social network is used.

The strength of the tie is affected by several factors. Granovetter in Granovetter (1982, 1995) proposed four dimensions that may affect the tie strength: the duration of
the relationship; the intimacy between the two individuals participating in the relationship; the intensity of their communication with each other; and the reciprocal services they provide to each other. Wellman and Wortley (Wellman and Wortley, 1990) argue that emotional support and helping each other strongly affect the strength of the tie between any two members in a social network. Other factors, such as socioeconomic status, educational level, political affiliation, race and gender are also considered to affect the strength of ties (Lin and Ensel, 1981). Ronal Burt in his research (Burt, 1995) also indicates that the structural factors, such as network topology and information about social circles, may affect the tie strength.

Gilbert et al (Gilbert and Karahalios, 2009b) suggest other variables that may affect the tie strength in social networks. The first variable is the intensity variable which measures the total number of words and messages traded between two friends in a social network. This is a good indicator of how often the two friends communicate with each other which indicates how strong their relationship is.

The second factor suggested by Gilbert et al is the days passed since the last communication between two friends. This variable measures the recency of the communication between two members and can indicate how much friends still trust each other, and how they depend on each other.

The third variable is the duration variable. This variable measures the time since the first communication which refers to the length of the friendship. This variable combined with the previous two (i.e. days since last communication and the intensity variable) are considered time-related variables. These variables reflect how strong a relationship is.

Another important variable that may affect the strength of a tie between two members in a social network is the neighborhood overlap variable (Onnela et al., 2007). This variable refers to the number of common groups and friends the two members have. This is an effective variable because if the number of common groups in which members
participate increases, the two members have more things in common. The same is valid for common friends.

Petrczi et al. (Petrczi et al., 2006) introduce an important factor that affects the strength of the tie between two members in a social network. This factor is the mutual confiding between the members. To show the importance of this factor, we can consider as an example the relationship between a star and fans. This is a one-sided relationship, i.e., the communication is from one side and the messages are mainly sent from one side (i.e. the fans). This kind of relationship is a weaker relationship compared to the relationship between two friends who mutually communicate with each other.

Here we need to apply the factors that affect tie strength, used in a human social network, in our social network. For the closeness factor, we can measure the closeness between the agents by measuring similarity between their ontologies. The intensity variable can be represented in our system by the number of messages traded between two agents and how many concepts are learnt from each other which brings their ontologies closer to each other. The recency factor can indicate how much agents still depend on each other in learning new concepts or searching for keywords. The neighborhood overlap can be reflected as the overlap of neighborhood circles (i.e. number of common neighbours) of two agents.

In most social networks, the relationship between members is not symmetric. Actor A can consider actor B as his/her best friend. At the same time, actor B considers his/her relationship with A as an acquaintance or classmate relationship. That means the strength of the tie between A and B is not necessarily equal to the strength of the tie between B and A. The same consideration is valid in the social network we use in our system. Agent Ag1 may have a strong relationship with agent Ag2, but the relationship between Ag2 and Ag1 is not as strong.

In the following subsection (4.2.1), we describe how to measure similarity between
different ontologies used by different MASs. In subsection 4.2.2, we illustrate how neighbourhood overlap affects strengths of ties between two actors in a social network and how to measure it. A brief illustration about network structure factor is presented in subsection 4.2.3.

4.2.1 Measuring similarity between Ontologies

In our system, in order to calculate similarity between ontologies, we depend on the methodology proposed by Olivares-Ceja et al in (Olivares-Ceja and Guzmán-Arenas, 2004). For two agents $Ag_A$ and $Ag_B$, in order to calculate how similar ontology $O_A$ of agent $Ag_A$ to ontology $O_B$ of agent $Ag_B$, we measure the degree of understanding of agent $Ag_B$ to ontology $O_A$ ($du(Ag_B, O_A)$). "Degree of understanding of agent $Ag_B$ to ontology $O_A$ is how much $Ag_B$ understand about what $Ag_A$ knows, how well each concept of $O_A$ maps into the corresponding (most similar) concept in $O_B$" (Olivares-Ceja and Guzmán-Arenas, 2004). The degree of understanding is not a symmetric property. The degree of understanding of agent $Ag_A$ to ontology $O_B$ ($du(Ag_A, O_B)$) is not necessary to be the same as degree of understanding of agent $Ag_B$ to ontology $O_A$ ($du(Ag_B, O_A)$). Each ontology ($O_A$ and $O_B$) defines a different set of concepts in a hierarchical representation. Each concept is characterized by a set of feature/value pairs. To measure how similar $O_A$ is to $O_B$, we first need to know for each concept $C_A$ in $O_A$ the most similar concept $C_B$ in $O_B$.

This can be done by dealing with name, feature/value pairs, parent and/or grandparent of concept $C_A$ and compare it to all concepts in ontology $O_B$. Ontology $O_B$ tries to find the most similar concept $C_B$ to the concept $C_A$. The similarity vector ($sv$) between $C_A$ and $C_B$ is then calculated. The values of $sv$ are between 0 and 1. This similarity vector ($sv$) depends on the similarity between the concepts themselves (i.e. $C_A$ and $C_B$), the similarity between their parents/grandparents, the similarity between their feature/value
pairs and/or similarity between their children (if any). $sv$ is not a symmetric value. If $C_B$ is the most similar concept to $C_A$ in ontology $O_B$ that does not mean that it is necessary that $C_A$ is the most similar concept to $C_B$ in ontology $O_A$.

In order to explain how this methodology works, consider the following example. Suppose agent $Ag_A$ wants to know the most similar concept $C_B$ in ontology $O_B$ to its concept $C_A$. $Ag_A$ sends to agent $Ag_B$ the words of concept $C_A$ along with its parent concept $P_A$ in its ontology $O_A$. Agent $Ag_B$ tries to find the most matching concept $C_B$ and calculate the similarity vector ($sv$) between $C_B$ and $C_A$. During the process of finding concept $C_B$, there are four cases $Ag_B$ can face.

**Case 1:** in this case, concept $C_B$ coincides with concept $C_A$; concept $P_B$ coincides with concept $P_A$ and $P_B$ is father, grandfather or great grandfather of concept $C_B$. If $Ag_B$ reaches this case, the search stops and $Ag_B$ returns concept $C_B$ to $Ag_A$ as best matching concept and $sv = 1$. See Figure 4.1.

![Figure 4.1](image)

**Figure 4.1:** Illustrative example of case 1 in calculating similarity vector ($sv$) for a concept $C_A$ with respect to ontology $O_B$

**Case 2:** in this case, agent $Ag_B$ founds a matching concept $P_B$ to the parent concept $P_A$, but not matching for concept $C_A$ itself. If $P_B$ is the root of ontology $O_B$ (the "Thing"
concept), the search stops and $Ag_B$ returns not found and $sv = 0$. Otherwise, if $P_B$ is not the root of ontology $O_B$, $Ag_B$ tries to find the most similar concept of $C_A$ and called it $C_B'$. $Ag_B$ now checks for similarity of the feature/value pairs of concept $C_A$ to all ancestors of $P_B$ and calculate $sv$ for each concept it checks. $sv$ in this case is fraction of features pairs of $C_B'$ coincides with corresponding features pairs of $C_A$. if the similarity value is so low (under a specific threshold), this concept is rejected. This operation is recursively repeated until finding the highest $sv$ value for all ancestors of $P_B$. In this case, $Ag_B$ returns the concept $C_B'$ and the calculated $sv$. On the other hand, if no such a concept is found (i.e. $sv$ values calculated for all ancestor concepts of $P_B$ is below the predetermined threshold), $Ag_B$ returns a "son of $P_B$" and $sv = 0.5$, see Figure 4.2. For example, if $Ag_A$ sends concept "Kiwi" and parent concept "Fruit". $Ag_B$ searches its ontology and found concept "Fruit" but not such a concept "Kiwi" or any matching to it. $Ag_B$ marks this concept as "son of Fruit" which means some Fruit that I do not know.

![Diagram](image)

Figure 4.2: Illustrative example of case 2 in calculating similarity vector ($sv$) for a concept $C_A$ with respect to ontology $O_B$

**Case 3**: in this case, agent $Ag_B$ found a matching concept $C_B$ but did not find a
matching concept to $P_A$. In this case, $Ag_B$ tries to match feature/value pairs of $C_A$ with those of $C_B$ and also children of $C_A$ to children of $C_B$ (if any). If most features of $C_B$ coincide with most features of $C_A$ and most children of $C_B$ coincide with those of $C_A$ (based on preselected thresholds), $C_B$ is chosen as the most similar concept to $C_A$ in ontology $O_B$ and $sv$ is fraction of features and children of $C_B$ that match with their correspondings of $C_A$, see Figure 4.3. On the other hand, if neither features nor children match, $Ag_B$ returns not found and $sv = 0$.

![Diagram](image)

**Figure 4.3:** Illustrative example of case 3 in calculating similarity vector ($sv$) for a concept $C_A$ with respect to ontology $O_B$

*Case 4:* in this case, neither $C_B$ nor $P_B$ could be found to match $C_A$ and $P_A$ respectively. In this case, $Ag_B$ returns not found and $sv = 0$, see Figure 4.4.
After calculating $sv$ for all concepts in $O_A$, we can calculate how similar $O_A$ is to $O_B$. This can be done by calculating the degree of understanding that agent $Ag_B$ has of the ontology $O_A$. This is defined in (4.2) (Olivares-Ceja and Guzmán-Arenas, 2004):

$$du(Ag_B, O_A) = \frac{\sum_{\forall C_A} sv_{C_A}}{|O_A|}$$ (4.2)

The degree of understanding can be considered as the average of all similarity values for all concepts in ontology $O_A$. This is the same as if agent $Ag_A$ asks agent $Ag_B$ to get similarity values for all concepts in its ontology. The value of $du$ is between 0 and 1. When similarity between concepts in $O_A$ with respect to $O_B$ increases, $du$ increases and vice versa. $du$ is a good representation of similarity between two ontologies.

4.2.2 Measuring Neighbourhood overlap

The "strength of weak ties" hypothesis (Granovetter, 1982) states that, "the strength of a tie between two nodes (i and j) in a social network increases with the overlap of their
friendship circles”. That means, if the number of common friends between two nodes in a social network increases, the strength of the tie between them increases as well.

Onnela et at (Onnela et al., 2007) prove that neighbourhood overlapping parameter \(O_{ij}\) is proportional to the strength of ties between each two actors \(i\) and \(j\) in a social network. The higher the value of \(O_{ij}\) (more number of common friends between actors \(i\) and \(j\)), the stronger the tie between them, and the lower the value of \(O_{ij}\) (less number of common friends between actors \(i\) and \(j\)) the weaker the tie between them. The opposite is also true: the stronger the tie between actors \(i\) and \(j\), the higher the value of \(O_{ij}\) (more number of common friends between them), and the weaker the tie between actors \(i\) and \(j\), the lower the value of \(O_{ij}\) (less number of common friends between them).

In order to quantitatively measure the overlap of neighborhood circles of the two agents \(Ag_i\) and \(Ag_j\), we can use the following equation (Onnela et al., 2007):

\[
O_{ij} = \frac{n_{ij}}{(k_i - 1) + (k_j - 1) - n_{ij}} \tag{4.3}
\]

Where: \(n_{ij}\) is the number of common network neighbors of agents \(Ag_i\) and \(Ag_j\); \(k_i\) and \(k_j\) denotes the number of edges of agents \(Ag_i\) and \(Ag_j\) respectively. If agents \(Ag_i\) and \(Ag_j\) have no common friends between them, \(O_{ij} = 0\). In this case, the relationship between \(Ag_i\) and \(Ag_j\) represents a potential bridge between two different communities. If all friends are common between agents \(Ag_i\) and \(Ag_j\), so the value of \(O_{ij} = 1\). \(Ag_i\) and \(Ag_j\) are part of the same circle of friends. Otherwise, the value of \(O_{ij}\) varies from 0 to 1 based on number of common friends as indicated in (4.3).

From figure 4.5, each of agents \(Ag_i\) and \(Ag_j\) has two other friends but they have no common friends. In this case, the number of common friends \((n_{ij})\) is 0, the number of friends of agents \(Ag_i\) \((k_i)\) is 3 and number of friends of agent \(Ag_j\) \((k_j)\) is 3, applying (4.3) we found that the neighborhood overlap between them is: \(O_{ij} = 0\).
From figure 4.6, we can see that the two agents $Ag_i$ and $Ag_j$ each has six more friends and all of those friends are common between the two agents. In this case, the number of common friends ($n_{ij}$) is 6, the number of friends of agents $Ag_i$ ($k_i$) is 7 and the number of friends of agent $Ag_j$ ($k_j$) is 7. Again, (4.3) can be applied to calculate the neighborhood overlap which is in this case: $O_{ij} = 1$.

From figure 4.7, each of the two agents $Ag_i$ and $Ag_j$ has four other friends; two of them are common friends between the two agents. In this case, the number of common friends ($n_{ij}$) is 2, the number of friends of agents $Ag_i$ ($k_i$) is 5 and number of friends of
agent $Ag_j (k_j)$ is 5. Applying (4.3) we can find that the neighborhood overlap between them is: $O_{ij} = 1/3$.

![Figure 4.7: A representation of social network between two agents $Ag_i$ and $Ag_j$ with some common friends](image)

4.2.3 Encoding Network Structure

The network structure mainly encodes the idea that the tie strength of a relationship does not depend on the history of the relationship itself only, but depends also on the strength of the ties of some mutual friends (Gilbert and Karahalios, 2009b). In other words, if $A$ and $B$ are friends, their relationship will be stronger if they have mutual friends. Also, if their relationship with those mutual friends is strong, that makes the relationship between $A$ and $B$ even stronger.

To represent this network structure dimension, we depend on the descriptors of the ties strength distribution over the mutual friends: mean, median, variance, skew, kurtosis, minimum and maximum. We can calculate the network structure dimension using the following equation (Gilbert and Karahalios, 2009b):

\[
N_{ij} = \lambda_0 \mu_M + \lambda_1 med_M + \sum_{k=2}^{4} \sum_{TS \in M} \lambda_k (TS - \mu_M)^k + \lambda_5 min_M + \lambda_6 max_M \tag{4.4}
\]

Where:
\[ M = \{TS_{ij} : i \text{ and } j \text{ are Mutual Friends} \} \] (4.5)

Where \( TS \) is the tie strength between \( i \) and \( j \). As we can see from the above equation, \( N_{ij} \) introduces a dependency model. Each tie strength depends on the tie strengths with other mutual friends. The question is: how to initialize those tie strengths in order to get the descriptors of the tie strength to calculate the network structure term \( (N_{ij}) \). To set the initial value of a tie strength, we can use ontologies similarity as initial value of tie strength.

4.3 Using HMM to Calculate Tie Strength (our proposed methodology)

We choose HMM in measuring tie strengths in social network because HMM is a generative and probabilistic model where we can try to model the distribution over the sequence of observations. In our methodology, we use HMM to estimate the weights of the factors affecting tie strengths in a social network. In order to do so, it attempts to learn the values of the factors that affect tie strengths.

![Figure 4.8: Illustration of using Hidden Markov Model (HMM) to calculate tie strength of a social network](image)

From Figure 4.8, each \( y(t) \) represents an observation \( y \) at time \( t \) (in our model, a value of a factor that affect the strength of a tie). The hidden states \( x \) are corresponding to the transition probability (the weight of each value). In addition, we assume that our model
follows the Markov property: the current state depends on the previous state (Rabiner, 1989). This assumption coincides with the fact that, a tie strength may change by time based on the current and previous interactions done between the actors participating in the relation/tie.

In our system, tie strength between agents of different MASs is a good indicator of how close these MASs are to each other. It helps in both semantic search and concept learning modules.

We chose to use Hidden Markov Model (HMM) in our system to calculate the tie strength. This model leverages the known data of the system and uses it to solve the system and to estimate the unknown information. In our system, the unknown parameter that we try to calculate is the strength of the ties between agents in our social network.

As indicated before, HMM can be declared by four parameters:

S: is the set of nodes in the network.

A: is the transition probability matrix. In our system, A represents the weights of the factors that affect the relations between agents (this is the unknown parameter that we try to find).

B: is the observations matrix. In our system, the observation that we can assign to the model is the values of the factors that affect the tie strength.

π: is the initial state of the system. We have two options for setting the initial values of the system.

The first option is to start with an equilibrium state in which all tie strengths have the same value. In this case, we assume that agents of all MASs connect to each other through relationships and the strengths of these relationships are all equal. We apply HMM to enhance values of the strengths according to other parameters. In this case, many calculations are needed as the network is fully connected, especially, if the system contains a large number of MASs that manage a variety of knowledge repositories.
The second option assumes simplified representations of the tie strengths to be the initial values of weights of the relationships in the system. In this case, we use the degree of similarity between ontologies of MASs to be our initial state values. Using ontologies similarity as initial state reduces the calculations required by HMM to get to the required solution. The system starts already from a point at which the weights of the relations are calculated based on the similarity of ontologies. This similarity is a good indicator for the closeness between agents which in turn is one of the main factors that affect the strength of ties between agents.

Figure 4.9 is an illustration of using HMM in calculating tie strength. It indicates the values of initial states, the variables that affect the observation matrix and the calculated transition probability matrix and its usage to calculate tie strength.

Figure 4.9: Illustration of using Hidden Markov Model (HMM) to calculate tie strength of a social network

4.3.1 Calculating the predictive variable that affects strengths of ties

As described earlier, tie strength depends on several factors. These factors can be grouped into three general variables:
• **Closeness variable**: This variable is related to measuring how close two agents are to each other. This can be done by measuring the degree of similarity between the two ontologies used by the two agents participating in the relationship.

• **Time-related variable**: This variable combines all time factors that affect the strength of the relationship. The first factor is the duration of the relationship. The second factor is the frequency of communication between the two agents. The third factor is the time passed since the last communication.

• **Mutual confidence variable**: This variable clarifies the nature of the relationship under measure, if it is a one-sided relationship or a mutual relationship.

We combine these variables in a linear combination to measure the predictive variable term that affects the strength of the tie in our social network. The predictive variable can be calculated as:

\[ X_{ij} = \alpha C_{ij} + \beta T_{ij} + \gamma MC_{ij} \]  

(4.6)

Where: \( X_{ij} \) is the predictive variable to be calculated between agents \( Ag_i \) and \( Ag_j \); \( C_{ij} \) is the closeness between the two agents; \( T_{ij} \) is the time-related variable between the two agents; and \( MC_{ij} \) is the mutual confidence variable between them.

4.3.2 Calculating Tie Strength

The strengths of ties between agents in a social network depend on pair-wise interaction, neighborhood size and the network structure of the system, each with a certain probability. In order to find out these probabilities, we can apply HMM to figure out the required
probabilities. After calculating the required probabilities we can then calculate the tie strength as (El-Sherif et al., 2011)

\[ TS_{ij} = P(X_{ij}).X_{ij} + P(O_{ij}).O_{ij} + P(N_{ij}).N_{ij} \]  \hspace{1em} (4.7)

Where: \( TS_{ij} \) is the strength of tie between two friends (agents) \( Ag_i \) and \( Ag_j \); \( X_{ij} \) is the predictive variable calculated for the two agents; \( O_{ij} \) is the network overlap factor between them; \( N_{ij} \) is the network structure factor between them; and \( P(X_{ij}), P(O_{ij}) \) and \( P(N_{ij}) \) are probabilities of predictive variable, network overlap and network structure respectively, they are calculated using HMM.

4.3.3 Validity of Our Methodology

The actors of our social network are non-humans, so we faced a problem of finding an existing data to test our methodology. All available social networks are human social networks which is not the case in our proposed network. We have to build our own test data to validate our methodology.

In this section, we discuss the effect of ontology similarity on the strength of tie between two agents in social networks. As indicated above, tie strength depends on three main variables: predictive variables; neighborhood overlap variable; and network structure variable. The first variable (i.e. predictive variable) consists of three other variables: closeness; time-related variables; and mutual confidence. All in all, in our model, tie strength depends on five variables (closeness, time-related, mutual confidence, neighborhood overlap and network structure). In order to study the effect of the closeness variable on the tie strength, we assume that all other factors will remain constant. We change the value of the closeness variable only and apply our HMM model to determine the predicted probabilities of all variables, then apply them in equations (4.6) and (4.7) to get the corresponding value of the tie strength.
Table 4.1 illustrates the effect of changing the value of the closeness variable on the estimated probabilities of all variables in the model and also on the corresponding strengths of ties. As it can be noticed, by increasing the closeness values between two agents, the estimated probability of closeness variable increases which means that, the effect of the closeness value on the strength of the tie increases. This agrees with our assumption: "the more similar the ontologies used by two agents the stronger the relationship between them". This relationship can be viewed in Figure 4.10.

Table 4.1: Sample data represent the effect of changing the value of closeness on the tie strength and the probabilities of all variables that affect the tie strength

<table>
<thead>
<tr>
<th>$C_{ij}$</th>
<th>$TS_{ij}$</th>
<th>$P(C_{ij})$</th>
<th>$P(T_{ij})$</th>
<th>$P(MC_{ij})$</th>
<th>$P(O_{ij})$</th>
<th>$P(N_{ij})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.6787</td>
<td>0.04</td>
<td>0.2133</td>
<td>0.3467</td>
<td>0.1867</td>
<td>0.2133</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6543</td>
<td>0.08</td>
<td>0.2033</td>
<td>0.3133</td>
<td>0.1867</td>
<td>0.2167</td>
</tr>
<tr>
<td>0.3</td>
<td>0.6557</td>
<td>0.1133</td>
<td>0.1933</td>
<td>0.3167</td>
<td>0.1767</td>
<td>0.2</td>
</tr>
<tr>
<td>0.4</td>
<td>0.6747</td>
<td>0.1433</td>
<td>0.1667</td>
<td>0.3467</td>
<td>0.1667</td>
<td>0.1667</td>
</tr>
<tr>
<td>0.5</td>
<td>0.683</td>
<td>0.1767</td>
<td>0.13</td>
<td>0.33</td>
<td>0.1833</td>
<td>0.18</td>
</tr>
<tr>
<td>0.6</td>
<td>0.711</td>
<td>0.1933</td>
<td>0.1567</td>
<td>0.3533</td>
<td>0.1467</td>
<td>0.15</td>
</tr>
<tr>
<td>0.7</td>
<td>0.729</td>
<td>0.2233</td>
<td>0.1167</td>
<td>0.3233</td>
<td>0.11</td>
<td>0.2267</td>
</tr>
<tr>
<td>0.8</td>
<td>0.7427</td>
<td>0.2533</td>
<td>0.1367</td>
<td>0.2967</td>
<td>0.13</td>
<td>0.1833</td>
</tr>
<tr>
<td>0.9</td>
<td>0.7653</td>
<td>0.26</td>
<td>0.1433</td>
<td>0.2933</td>
<td>0.1567</td>
<td>0.1467</td>
</tr>
<tr>
<td>1</td>
<td>0.825</td>
<td>0.3</td>
<td>0.0933</td>
<td>0.3167</td>
<td>0.1233</td>
<td>0.1667</td>
</tr>
</tbody>
</table>
Figure 4.10: Graphical representation of the effect of changing the value of ontology similarity on the estimated probability of the closeness variable.

Figure 4.11 represents the relation between the tie strength and the closeness values. From this figure, we can notice that at the beginning the tie strength starts with a relatively high value and by increasing the closeness value, the strength of the tie decreases. Afterwards, the strength of the tie starts to increase again by increasing the closeness value.

Figure 4.11: Graphical representation of the relation between tie strength and ontology similarity.
To explain this we have to consider the effect of other variables on the strength of the tie between two agents. At the beginning, the value of the closeness variable \( C_{ij} \) is low, so the effect of this variable on the strength of the tie is low. On the other hand, the remaining four variables (time-related, mutual confidence, neighborhood overlap and network structure) have higher effects on the tie strength. According to (4.6) and (4.7) and based on the values of the four variables, the value of the tie strength is relatively high.

By increasing the ontology similarity between agents, the value of \( C_{ij} \) increases and the probability of this variable \( P(C_{ij}) \) increases too. From Table 4.1, we can notice that the value of the closeness variable \( C_{ij} \) increases (e.g. 0.05, 0.1, 0.15 ) and the resultant value of its probability \( P(C_{ij}) \) increases also (e.g. 0.0267, 0.04, 0.05, ). These values are relatively low with respect to the values of other variables in (4.6) and (4.7). Applying these values in (4.6) and (4.7), the value of tie strengths decrease as indicated in Table 4.1 (e.g. 0.686, 0.6787, 0.6752, ) At a certain point (in this case when closeness variable is 0.4) the probability of the closeness variable \( P(C_{ij}) \) increases relatively (e.g. 0.1433, 0.1533, 0.1767, ) and value of the closeness variable itself \( C_{ij} \) increases also (e.g. 0.4, 0.45, 0.5, ) so the effect of the closeness variable on the tie strength increases which make values of the tie strengths get higher (e.g. 0.6747, 0.6748, 0.683, ).

4.4 Summary

This chapter explained our methodology in calculating strengths of ties between nodes in a social network using Hidden Markov Model. This chapter explained factors that affect the tie strengths in a social network individually. The factors were grouped into three major groups: predictive variable, neighbourhood overlap and network structure. This chapter explained in detail how each factor can be measured. Afterwards, our proposed
methodology of calculating a tie strength in a social networks using HMM was discussed along with some illustrative results. In the next chapter, an example application is set to test the concept learning module of our system.
Chapter 5

An Example Application

We have now laid out our proposed system and considered the practical issues involved in its implementation as well as the methodology suggested for calculating tie strengths in the social network used in our system. This chapter focuses on describing the case studies we have conducted in order to test the concept learning module of our system. These case studies illustrate all aspects related to our proposed module which enables agents to learn new concepts. Moreover, they evaluate the efficiency of using social networks in representing relationships between agents in different MASs as well as the effect of increasing the number of teachers on the learning accuracies.

This chapter is organized as follow: in Section 5.1, we describe the knowledge domain used in this example; Section 5.2 introduces a brief description of techniques used in our system for document classification; we present our two case studies in detail along with test scenarios suggested for each case study in Section 5.3; finally, in Section 5.4, we describe the confusion matrix used in calculating learning accuracies in our case studies.

5.1 Knowledge base domain

In order to evaluate our concept learning module of our system illustrated in Chapter 3, we have chosen the course catalogue domain. This is a very popular domain in ontology research. Concepts of this ontology domain are courses offered by all faculties and departments in a specific university. The objects (examples) of each concept in this ontology consist of text files that describe the courses offered. The domain is additionally structured according to the university units of these universities, which creates different
ontologies for each of them.

Building our knowledge bases was the most time consuming work done in this research. First, information of all courses offered by each department and faculty of each selected university is created in unstructured form. Second, the ontology is built using Portege. In our example, we created ontologies for course syllabi of twenty seven universities. Three of these ontologies were built by Afsharchi (Afsharchi, 2007). There are twenty six universities are considered as teachers and one as a learner. An empty ontology (no concepts are initially defined in it) is also considered to represent another learner. For each university, we build a hierarchy of all faculties and departments in this university as an ontology of this university. The nodes of each ontology are faculties, departments and sub-departments of the university. Courses offered by each department are the leafs of the ontology. Some unstructured text documents are created to define each course. These documents are the positive examples describing the courses. Table 5.1 shows the characteristics of part of our domain.

<table>
<thead>
<tr>
<th>Ontologies</th>
<th>Cornell</th>
<th>Michigan</th>
<th>Washington</th>
</tr>
</thead>
<tbody>
<tr>
<td># of concepts</td>
<td>176</td>
<td>174</td>
<td>166</td>
</tr>
<tr>
<td># of non-leaf concepts</td>
<td>27</td>
<td>21</td>
<td>25</td>
</tr>
<tr>
<td>depth</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td># of objects in ontology</td>
<td>4360</td>
<td>7744</td>
<td>6957</td>
</tr>
<tr>
<td>max # of objects at a leaf</td>
<td>161</td>
<td>293</td>
<td>214</td>
</tr>
</tbody>
</table>

During selecting the universities participating in our learning module, we took into consideration having a diverse selection. some of the chosen universities are high ranked universities with different faculties and departments. Others are medium and low ranked
universities. The reason behind this is to build diverse ontologies, so our system have different types of teachers. some teachers use powerful and well structured ontologies with well described concepts. other teachers use weaker ontologies with some concepts that are not well defined.

In building our ontologies, we used Protege (Protege, Protege). Protege is a platform that provides a suite of tools to construct domain models and knowledge-based applications with ontologies. At its core, Protege implements a rich set of knowledge modeling structures and actions that support the creation, visualization, and manipulation of ontologies in various representation formats. Fortunately Protege ontologies can be exported into a variety of formats including OWL which was our target format.

Each course file (example) contains a course identifier, course description and prerequisites of the course (if any). For our concepts (courses), there are three main features representing the concept. The first feature is the course identifier ($f_{id}$). It is a string containing the name of the course. The second main feature are course prerequisites ($f_{prereq}$). It contains identifiers of other courses that are considered as prerequisites of the current course. As each university in our case studies uses a different ontology, each university also uses different identifiers to the courses it offers. The previous two features are not helpful for the learning of new concepts. The third feature is the course description ($f_{desc}$). It is a simple text-based description of the course. This feature determines the main units of the course, content and organization of that course and a brief description of it. This feature is a good, unique and ontology-independent representative of each course and gives us the ability to conduct a successful concept learning process.

Using the natural language text feature as an identifying feature of a concept faces lots of challenges, but this area has quite a lot of interest and practical applications. The first challenge we faced is to represent the text documents for information retrieval. One way to represent a text document is using the bag-of-words technique (Ko, 2012).
In this technique, each document is parsed into a set of keywords (tokens) with term frequencies to represent the number of occurrences of each keyword in the document. Unfortunately, this technique is not general enough to be used in our system. The output of this technique is a single word vector with all words contained in the document. That requires that the search keywords are sent by the learner to contain all keywords of that concept, which is too restrictive for our system. Another technique to define such a feature is to group the keywords to look for a certain words or word combination (Sahami, 1996) (Peng and Schuurmans, 2003).

In our course description, we have a set of $K$ keywords. We have also a set of features for each possible subset of $K$. The most challenging difficulty here is the creation of $K$ keywords. Since the bag of words that represents each concept is so large (high dimensionality of keyword space), each concept can have hundreds of thousands of keywords to represent its content. We need to reduce this set of keywords. An easy way to do so is to select some keywords randomly. This solution is not as intelligent as required for our concept learning module. We need a preprocessing stage in our module in which we use some information retrieval technique to reduce the keyword space dimensionality.

5.2 Document Classification

As explained previously in Section 5.1, we need to classify the documents used in all repositories. Document classification means assigning documents to one or more categories using data mining methods. In order to classify the documents, some steps need to be followed (see Figure 5.1). Document classification consists of three stages: document preprocessing; document representation; and document categorization. The following sub-sections describe each stage in more details.
5.2.1 Document Preprocessing

The main purpose of document preprocessing is feature extraction. Feature extraction is the selection of words (features) that best describe the document.

The steps of document preprocessing used in our research are as follow:

1. At the beginning, documents are tokenized to get all words in the documents.

2. Then a feature reduction technique is used to delete non-informative words (words that are irrelevant to a document’s meaning), such as common English words like: a, an, in, for, the, etc. Also, we can create our own list of words that do not affect the document categorization or that are common in most documents (e.g. semester, fall, winter, etc).

3. The next step in document preprocessing is word stemming. Word stemming is applied to reduce the length of the feature set by removing suffix and prefix of words. In this case, we treat words with the same stem
as a single word. For example: mathematics and mathematical can be considered as the same feature in the document (mathematics).

4. Finally, using a statistical metric produces a feature vector metric that best represents a group of documents by calculating the relevance of each feature to the group it belongs to. In our experiment, we use Term Frequency plus Inverse Document Frequency ($TF \times IDF$) for feature extraction.

5.2.2 Document Representation

After finishing document preprocessing, we have feature vectors, each one representing a concept. A feature vector consists of several words that best represents documents/examples of each concept. However, the created feature vectors tend to be very long. We need to reduce their size in order to make our system faster. We select the top $n$ words created for each category with the highest scores accompanied with their weight as a feature vector for each category.

In our case studies, we set the value of $n$ to 30. We select the top 30 words to represent the feature vector for each concept. We noticed that after 30 words, the weight of the words in documents/examples of a concept is very low. That means the occurrence of these words in documents representing a concept is very rare. Eliminating them from feature vectors will not affect the concept representation. At the same time, deleting them will simplify dealing with the feature vectors.

5.2.3 Document Categorization

Document categorization is the core of our learning process. In this stage, we try to assign each group of documents to a certain category (learning process) and calculate the accuracy of that learning process. In our work, we use three different learning techniques: K Nearest Neighbour (K-NN); Naïve Bayes; and Support Vector Machine (SVM) to learn
new concepts.

In this experiment, we adopt RapidMiner (Rapid-I, Rapid-I) software tool to classify documents. It is a free data mining software tool that offers a variety of methods for document preprocessing and categorization. Figure 5.2 is a screen shot of our implementation of the preprocessing stage using RapidMiner 5.0.

Figure 5.2: Screen shot of our experiment using RapidMiner

5.3 Our case studies

In order to test our concept learning module, we carried out two case studies presented in this section. In the following case studies, we measure the learning accuracy for each new concept learnt from several teachers. Therefore, we can see the influence of each teacher in the learning process. In selecting positive and negative examples, we follow the strategy explained in Chapter 3. In the following case studies, the learner uses majority voting to
resolve the conflicts.

5.3.1 Case study I

In this case study, we want to show the effect of using social networks in defining the relationship between agents from different MASs on the accuracy of the learning process. In this case study, we have three teachers and two learners. One of the learners $Ag_L$ controls an empty repository. The second learner $Ag_G$ controls a non-empty repository. In all test scenarios, the learner learns some concepts from all teachers. The learning accuracy is then calculated.

5.3.1.1 Data set

Our data set in this case study consists of structured hierarchy ontologies of course syllabi at three universities: Cornell University, the University of Michigan and the University of Washington, $Ag_C$, $Ag_M$ and $Ag_W$ respectively. Each of them is an agent in a MAS that controls a repository that contains one of the three ontologies for the three universities’ course syllabi. The new concepts the learner wants to learn in this experiment are "Computer Science", "Programming Language", "Chemistry", "Germany", "Linguistics", "Mathematics" and "Physics".

We chose to start with learning the concept "Computer Science" because it has different representations in the three ontologies. Cornell University considers "Computer Science" as an engineering discipline that is, however, completely independent from "Electrical and Computer Engineering" (see Figure 5.3). The University of Michigan organizes "Computer Science" also as an engineering discipline but as a joint program with "Electrical Engineering", as shown in Figure 5.4. The University of Washington organizes "Computer Science" as an engineering discipline but independent from "Electrical Engineering" and as a joint program with "Computer Engineering", as shown in Figure 5.5.
Although the concept "Programming Language" is not an existing course (not a stan-
dalone concept in any of ontology hierarchies of the three teachers), we use it to show how teachers can deal with a new concept and teach it to the learner.

5.3.1.2 Test scenarios
In this case study, we have four test scenarios. In the first two test scenarios, the learner, $Ag_L$, controls an empty repository. In the last two test scenarios, the learner, $Ag_G$, controls a repository that contains an ontology hierarchy of courses offered by the University of Calgary.

Test Scenario I
In the first test scenario, no social networks are used, i.e. all teachers have the same effect on the learner. In this test scenario, the repository of the learner $Ag_L$ is empty at the beginning. This test scenario consists of three stages:

Stage 1: The first stage of this test scenario is to learn a new concept "Computer Science" from all teachers. The learner uses keywords only to describe the concept. The steps of this stage are as follow:

1. Only keywords are used to search for the best matching concept $C_{best}$ in the teachers' ontologies. The keywords used are ("computer science" OR "program language").

2. Teachers search their ontologies for the best matching concept, $C_{best}$, that has the highest value of $sim(q_{spec}, C_{best})$.

3. Each teacher creates sets of positive and negative examples. The number of positive and negative examples selected by each teacher is the same.

4. Each teacher sends its set of positive and negative examples to $Ag_L$. 

131
5. $Ag_L$ creates a new definition of the concept "Computer Science" based on the received examples using three machine learning techniques (K-NN, Na"ive Bayes and SVM). It calculates the learning accuracy each time.

6. $Ag_L$ extracts a feature vector of the newly learnt concept "Computer Science".

Stage 2: The second stage of this test scenario is to refine the definition of the newly learnt concept "Computer Science" from all teachers by relearning it. The steps of this stage are as follows:

1. $Ag_L$ now has an initial definition of the concept "Computer Science" in its repository.

2. $Ag_L$ sends another learning request to all teachers. The new learning request contains same keywords used in stage 1 ("computer science" OR "program language") plus the feature vector extracted of the existing concept "Computer Science". The aim of this step is to enable teachers to enhance their searching for the best matching concept in order to enhance the learning accuracy.

3. Teachers search their ontologies for the best matching concept, $C_{best}$, that have the highest value of $\text{sim}(q_{spec}, C_{best})$.

4. Each teacher creates sets of positive and negative examples. The number of positive and negative examples selected by each teacher is the same.

5. Each teacher sends its set of positive and negative examples to $Ag_L$.

6. $Ag_L$ creates a new definition of the concept "Computer Science" based on the received examples using three machine learning techniques (K-NN, Na"ive Bayes and SVM).
Naïve Bayes and SVM). It calculates the learning accuracy each time.

7. $A_{gL}$ updates the feature vector of the newly updated concept "Computer Science".

Stage 3: The third stage of this test scenario is to learn a new concept "Programming Language" that is not a standalone concept in the teachers’ ontologies. It is a child of the newly learnt concept "Computer Science". The steps of this stage are as follow:

1. $A_{gL}$ defines an annotation for the "Programming Language" concept which is a child of the learnt concept "Computer Science". The annotation specifies the concept "Programming Language" is: ("program language" | "C++" | "Java").

2. $A_{gL}$ sends the annotation of step 1 in addition to the feature vector created for the learnt concept "Computer Science" to teacher agents to search their ontologies for the new best matching concept $C_{best}$ based on the value of $\text{sim}(q_{spec}, C_{best})$.

3. Each teacher creates sets of positive and negative examples. The number of positive and negative examples selected by each teacher is the same.

4. Each teacher sends its set of positive and negative examples to $A_{gL}$.

5. $A_{gL}$ creates a definition of the concept "Programming Language" based on the received examples using three machine learning techniques (K-NN, Naïve Bayes and SVM). It calculates the learning accuracy each time.

6. $A_{gL}$ extracts a feature vector of the newly learnt concept "Programming Language" and updates its repository.
Test Scenario II

In this test scenario, we introduce a social network for defining relationships between the learner $AgL$ and teachers in our system. At the beginning, the strengths of ties between $AgL$ and all teachers are the same and they are updated during the learning process. The learner’s repository is also empty at the beginning of this test scenario. This test scenario consists of three stages:

Stage 1: The first stage of this test scenario is to learn the concept "Computer Science" from all teachers. $AgL$ uses only keywords to describe the concept. The steps of this stage are as follows:

1. Only keywords are used to search for the best matching concept, $C_{best}$, in the teachers’ ontologies. The keywords used are ("computer science" OR "program language").

2. Teachers search their ontologies for the best matching concept, $C_{best}$, that have the highest value of $sim(q_{spec}; C_{best})$.

3. Each teacher creates sets of positive and negative examples. The number of positive and negative examples selected by each teacher is the same.

4. Each teacher sends its set of positive and negative examples to $AgL$.

5. $AgL$ creates a definition of the concept "Computer Science" based on the received examples using three machine learning techniques (K-NN, Naïve Bayes and SVM). It calculates the learning accuracy each time.

6. $AgL$ extracts a feature vector of the newly learnt concept "Computer Science".
Stage 2: The second stage of this test scenario is to refine the definition of the newly learnt concept “Computer Science” by relearning it from all teachers. The steps of this stage are as follow:

1. Calculate the closeness between the feature vector created for the newly learnt concept and feature vectors of each $C_{\text{best}}$ chosen by each teacher.

2. Depending on the closeness value calculated in the previous step, set an initial tie strength between $Ag_L$ and each teacher.

3. $Ag_L$ sends keywords (”computer science” OR ”program language”) and the feature vector obtained for the newly defined concept ”Computer Science” to all teachers it communicates with to search for the best matching concept $C_{\text{best}}$ in their ontologies.

4. Teachers search their ontologies for $C_{\text{best}}$, that have the highest value of $\text{sim}(q_{\text{spec}}, C_{\text{best}})$.

5. After finding $C_{\text{best}}$, each teacher selects positive and negative examples to represent this concept. The number of positive and negative examples selected from each teacher is proportional to the strengths of ties calculated between each teacher and $Ag_L$.

6. Each teacher sends its set of positive and negative examples to the learner in order to help it to learn the concept ”Computer Science”.

7. After completing the learning process, calculate the learning accuracy.

8. Update the feature vector of the learnt concept ”Computer Science”.
9. Based on the feature vector of the newly updated concept "Computer Science" in $Ag_L$, we can update the values of tie strengths between $Ag_L$ and each teacher.

Stage 3: The third stage of this test scenario is to learn a new concept "Programming Language". The steps of this stage are as follow:

1. Use the annotation ("program language" | "C++" | "Java") for the concept "Programming Language" to learn this concept.

2. $Ag_L$ sends the annotation from step 1 in addition to the feature vector created for the learnt concept "Computer Science" to the teachers to search their ontologies for the new best matching concept $C_{best}$.

3. Teachers search their ontologies for $C_{best}$, that have the highest value of $sim(q_{spec}, C_{best})$.

4. Each teacher selects positive and negative examples to represent the new concept "Programming Language" and sends it back to the learner. The number of positive and negative examples sent by each teacher depends on the updated value of the tie strengths between $Ag_L$ and each teacher.

5. $Ag_L$ uses the positive and negative examples sent to learn the new concept "Programming Language".

6. $Ag_L$ calculates the learning accuracy and creates a feature vector for the concept "Programming Language".

7. $Ag_L$ updates the tie strengths with all teachers.
Test Scenario III

In this test scenario, no social network is defined in the system. The learner agent, $A_g$, contains a predefined ontology of the courses of the University of Calgary with an initial definition of the concept “Computer Science”. This test scenario consists of two stages:

Stage 1: The first stage of this test scenario is to enhance the definition of the concept “Computer Science” defined in the learner’s ontology. The steps of this stage are as follow:

1. Create a feature vector for the concept “Computer Science” in $A_g$’s ontology.
2. The learner sends the keywords (“computer science” OR “program language”) and the feature vector obtained in step 1 to all teachers to search their repositories for the best matching concept $C_{best}$.
3. Teachers search their ontologies for $C_{best}$, that have the highest value of $\text{sim}(q_{spec}, C_{best})$.
4. Each teacher selects positive and negative examples that represent the selected $C_{best}$ and sends them to the learner agent. The number of positive and negative examples selected by each teacher is the same.
5. $A_g$ uses positive and negative examples to relearn the concept “Computer Science” using three learning technique (K-NN, Naïve Bayes and SVM).
6. After completing the learning process, calculate the learning accuracy each time.
7. $A_g$ updates the feature vector of the newly updated concept “Computer Science”.

137
Stage 2: The second stage of this test scenario is to learn the new concept "Programming Language" that is not a standalone concept in the learner’s and teachers’ ontologies. This concept is a child of the newly learnt concept "Computer Science". The steps of this stage are as follow:

1. Use the annotation ("program language" | "C++" | "Java") to define the concept "Programming Language".

2. $Ag_G$ sends the annotation from step 1 in addition to the feature vector created for the updated concept "Computer Science" in stage 1 to all teachers to search their ontologies for the new best matching concept $C_{\text{best}}$.

3. Teachers search their ontologies for $C_{\text{best}}$, that have the highest value of $\text{sim}(q_{\text{spec}}, C_{\text{best}})$.

4. Each teacher selects a set of positive and negative examples to represent the new concept "Programming Language" and sends it to the learner. The number of positive and negative examples selected by each teacher is the same.

5. $Ag_G$ uses positive and negative examples sent from all teachers to learn the new concept "Programming Language".

6. $Ag_G$ calculates the learning accuracy and creates a feature vector for the concept "Programming Language".

Test Scenario IV

In this test scenario, we introduce a social network in defining relationships between $Ag_G$ and teachers in our system. The learner agent contains a predefined ontology of the courses of the University of Calgary with an initial definition of the concept "Computer Science". This test scenario consists of two stages:
Stage 1: The first stage of this test scenario is to enhance the definition of the concept "Computer Science" defined in the learner’s ontology. The steps of this stage are as follows:

1. Create a feature vector for the concept "Computer Science" in AgG’s ontology.

2. Calculate the closeness between AgG and each teacher. Use these closeness values to set up initial values of tie strengths between AgG and each teacher.

3. AgG sends the keywords ("computer science" OR "program language") and the feature vector obtained in step 1 to all teachers to search their repositories for the best matching concept $C_{best}$.

4. Teachers search their ontologies for $C_{best}$, that have the highest value of $sim(q_{spec}, C_{best})$.

5. Each teacher selects positive and negative examples that represent the selected $C_{best}$ and sends them to AgG. The number of positive and negative examples is proportional to the strength of ties between each teacher and AgG.

6. AgG uses positive and negative examples to relearn the concept "Computer Science" using three learning technique (K-NN, Naïve Bayes and SVM).

7. After completing the learning process, calculate the learning accuracy each time.

8. AgG updates feature vector of newly updated concept "Computer Science".

9. Update the values of tie strengths between AgG and each teacher.
Stage 2: The second stage of this test scenario is to learn the new concept "Programming Language". The steps of this stage are as follows:

1. Use the annotation ("program language" | "C++" | "Java") to define the concept "Programming Language."

2. $Ag_G$ sends the annotation from step 1 in addition to the feature vector created for the updated concept "Computer Science" in stage 1 to all teachers to search their ontologies for the new best matching concept $C_{best}$. 

3. Teachers search their ontologies for $C_{best}$, that have the highest value of $sim(q_{spec}, C_{best})$. 

4. Each teacher selects a set of positive and negative examples to represent the new concept "Programming Language" and sends it to the learner. The number of positive and negative examples sent by each teacher depends on the updated value of the tie strengths between the learner and each teacher. 

5. $Ag_G$ uses positive and negative examples sent from all teachers to learn the new concept "Programming Language". 

6. $Ag_G$ calculates the learning accuracy and creates a feature vector for the concept "Programming Language". 

7. $Ag_G$ updates the values of tie strengths with all teachers. 

In order to clarify the effect of using the social networks in our system, the learner learns some more concepts with and without using of social networks. The concepts to be learnt are: "Chemistry", "German", "Linguistics", "Mathematics" and "Physics".
5.3.2 Case Study II

In this case study, we want to show the effect of increasing the number of teachers on the learning accuracy. On the other hand, we want to show the effect of using social networks in defining the relationship between agents on the learning accuracy and if the quality of the learnt concept is affected by increasing the number of teachers when using social networks. In this case study we start with four teachers and increase the number of teachers one by one. Each time, the learner learns the same concept "Computer Science" from all teachers using three learning techniques: K-NN, Naïve Bayes and SVM. The learning accuracy is then calculated.

5.3.2.1 Data Set

Our data set consists of structured hierarchy ontologies of course syllabi at twenty six universities: Cornell University, the University of Michigan, the University of Washington, the University of Victoria, Simon Fraser University, the University of Northern British Columbia, Dalhousie University, the University of New Brunswick, the University of Manitoba, the University of Winnipeg, Memorial University, the University of British Columbia, the University of Alberta, Vancouver Island University, the University of the Fraser Valley, Mount Allison University, Mount Royal University, Athabasca University, Kings University, University of Lethbridge, Kwantlen Polytechnic University, Thompson Rivers University, Trinity Western University, Acadia University, Mount Saint Vincent University, Saint Marys University. Course syllabus information for each course is contained in a text document. In order to test our proposed concept learning module, we use MASs to control the repositories that contain those ontologies. These MASs are the teachers in our system. Each MAS controls a repository that contains one of the twenty six ontologies for the twenty six universities’ course syllabi. We use the same two learners ($Ag_L$ and $Ag_G$) used in case study I to learn new concepts from the above teachers.
5.3.2.2 Test Scenarios

In our case study we have three test scenarios. In the first two test scenarios, the learner ($A_{gL}$) controls an empty repository. In the third test scenario, the learner ($A_{gG}$) controls a non-empty repository.

Test scenario I

In this test scenario, no social networks are defined, i.e. all teachers have the same effect on $A_{gL}$. This test scenario consists of two stages:

Stage 1: In this stage, $A_{gL}$ needs to learn the new concept ”Computer Science” from all teachers. $A_{gL}$ uses keywords only to describe the concept. The steps of this stage are as follows:

1. The number of teachers is four at the beginning.
2. Keywords that used to describe the concept ”Computer Science” are (”computer science” OR ”program language”). These keywords are used by all teachers to search for the best matching concept $C_{best}$ in their ontologies.
3. Teachers search their ontologies for the best matching concept, $C_{best}$ that has the highest matching with the sent keywords (i.e. the highest value of $sim(q_{spec}, C_{best})$).
4. Each teacher selects sets of positive and negative examples. The number of positive and negative examples selected by each teacher is the same.
5. Each teacher sends its set of positive and negative examples to $A_{gL}$.
6. $A_{gL}$ learns the concept ”Computer Science” based on the received examples then calculates the learning accuracy.
7. Create feature vector for the newly learnt concept ”Computer Science”.

142
Stage 2: In this stage, $Ag_L$ tries to enhance the definition of the newly learnt concept "Computer Science" by relearning it from all teachers. The steps of this stage are as follows:

8. $Ag_L$ tries to refine the definition of the learnt concept by sending another learning request containing the same keywords ("computer science" OR "program language") along with the feature vector created in the previous stage.

9. All teachers try to find the best matching concept $C_{best}'$ for the new learning request sent.

10. After finding new $C_{best}'$, each teacher selects the same number of positive and negative examples and sends them back to $Ag_L$ to learn the new concept.

11. $Ag_L$ collects all positive and negative examples sent by all teachers and learns the concept "Computer Science" and updates its definition.

12. Calculate the learning accuracy each time.

13. $Ag_L$ extracts the feature vector of the newly learnt concept "Computer Science".

14. Add one more teacher.

15. Repeat this test scenario again starting from step 2 in stage 1 until all teachers defined in our case study are involved.

Test scenario II

In this test scenario, $Ag_L$ controls an empty repository. We define a social network to represent relationships between $Ag_L$ and teachers in the system. The initial tie strength
between $Ag_L$ and all teachers are the same. This test scenario consists of two stages:

Stage 1: In this stage, $Ag_L$ needs to learn a new concept "Computer Science" from all teachers. $Ag_L$ uses only keywords to describe the concept. The steps of this stage are as follows:

1. The number of teachers is four at the beginning.

2. Keywords used to describe the concept "Computer Science" are ("computer science" OR "program language"). The keywords are used by all teachers to search for the best matching concept, $C_{best}$, in their ontologies.

3. Teachers search their ontologies for the best matching concept, $C_{best}$ that has the highest value of $sim(q_{spec}, C_{best})$. Each teacher creates sets of positive and negative examples.

4. Ties strengths between $Ag_L$ and all teachers are the same, so the number of positive and negative examples selected by each teacher is the same.

5. Each teacher sends its set of positive and negative examples to $Ag_L$.

6. $Ag_L$ learns the concept "Computer Science" based on the received examples. It calculates the learning accuracy.

7. $Ag_L$ Creates feature vector for the newly learnt concept "Computer Science".

8. Calculate the closeness between the ontology defined in $Ag_L$ and ontologies of each teacher to get the new tie strength between $Ag_L$ and each teacher in the system.
Stage 2: In this stage, \( Ag_L \) tries to enhance the definition of the newly learnt concept "Computer Science" by relearning it from all teachers. The steps of this stage are as follow:

9. \( Ag_L \) tries to refine the definition of the learnt concept by sending another learning request containing the same keywords ("computer science" OR "program language") along with the feature vector extracted in the previous stage.

10. All teachers try to find the best matching concept \( C_{best} \) for the new learning request sent based on value of \( \text{sim}(q_{spec}, C_{best}) \).

11. After finding the best matching concept \( C_{best} \), each teacher selects positive and negative examples to represent this concept. The number of positive and negative examples selected from each teacher is proportional to the strength of tie between each teacher and \( Ag_L \).

12. Each teacher sends its set of positive and negative examples to \( Ag_L \) in order to enhance the definition of the concept "Computer Science" and calculate the learning accuracy.

13. After completing the learning process, \( Ag_L \) updates the definition of the concept "Computer Science" in its ontology.

14. \( Ag_L \) updates the feature vector of the updated concept "Computer Science".

15. Update the strengths of ties between \( Ag_L \) and all teachers.

16. Add one more teacher.
17. Repeat this test scenario starting from step 2 in stage 1 until all teachers defined in our case study are involved.

Test scenario III
In this test scenario, our learner agent, $Ag_G$, controls a non-empty repository. In this test scenario, we need to show if the effect of using social networks in communicating between MASs is still valid or not even if the learner is non-empty and has predefined concepts. $Ag_G$ tries to enhance the definition of the concept ”Computer Science” by relearning it from all teachers. This test scenario consists of two stages:

Stage 1: In this stage, no social networks are used, i.e. all teachers have the same effect on $Ag_G$. The steps of this stage are as follows:

1. The number of teachers is four at the beginning.

2. $Ag_G$ in this stage has an initial definition of concept ”Computer Science”. $Ag_G$ creates a feature vector of that concept.

3. $Ag_G$ sends a learning request to all teachers to learn concept ”Computer Science”. The learning request contains the keywords that describe that concepts (”computer science” OR ”program language”) along with the feature vector created in step 1.

4. Teachers search their ontologies for the best matching concept, $C_{best}$ based on value of $sim(q_{spec}, C_{best})$.

5. Each teacher selects sets of positive and negative examples. the number of positive and negative examples selected by each teacher are the same.

6. Each teacher sends its set of positive and negative examples to $Ag_G$. 

146
7. $Ag_G$ relearn concept "Computer Science" based on the received examples. It calculates the learning accuracy.

8. Add one more teacher.

9. Repeat this stage starting from step 2 until all teachers defined in our case study are involved.

Stage 2: In this stage, we introduce a social network for defining relationships between $Ag_G$ and teachers in our system. The steps of this stage are as follows:

1. Starting with four teachers.

2. Create the feature vector of the already defined concept "Computer Science" in $Ag_G$’s ontology.

3. Calculate the closeness between the ontology defined in $Ag_G$ and ontologies of each teacher.

4. Depending on the closeness value calculated in the previous step, set the initial tie strengths between $Ag_G$ and each teacher.

5. $Ag_G$ sends a learning request to all teachers to relearn concept "Computer Science". It sends the same keywords ("computer science" OR "program language") along with the feature vector created in step 2 to search for the best matching concept $C_{best}$ in their ontologies.

6. After finding $C_{best}$, each teacher selects positive and negative examples to represent this concept. The number of positive and negative examples selected by each teacher is proportional to the strengths of ties between $Ag_G$ and each teacher.
7. Each teacher sends its set of positive and negative examples to $Ag_G$ in order to learn the concept "Computer Science" and calculate the learning accuracies.

8. After completing the learning process, $Ag_G$ updates the definition of the concept "Computer Science" in its ontology.

9. Update strengths of ties between $Ag_G$ and the teachers.

10. Add one more teacher.

11. Repeat this stage starting from step 2 until all teachers defined in our system are involved.

5.4 Accuracy of the learnt concept

In order to measure the learning accuracy in our case studies, we use a confusion matrix, (see Table 5.2) to measure the proportion of true results (i.e. true positive and true negative).

Table 5.2: Confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>True Positive</th>
<th>False Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Negative</td>
<td></td>
<td>True Negative</td>
</tr>
</tbody>
</table>

\[
\text{accuracy} = \frac{(#\text{of True Positive} + #\text{of True Negative})}{(#\text{of True Positive} + #\text{of False Positive} + #\text{of False Negative} + #\text{of True Negative})}
\]  \hspace{1cm} (5.1)

Where:

\textit{# of True Positive}: It is the number of positive examples that were correctly categorized
as positive examples during the learning process.

\# of TrueNegative: It is the number of negative examples that were correctly categorized as negative examples during the learning process.

\# of FalsePositive: It is the number of negative examples that were wrongly categorized as positive examples during the learning process.

\# of FalseNegative: It is the number of positive examples that were wrongly categorized as negative examples during the learning process.

5.5 Summary

This chapter introduced our example application used to test the concept learning module of our system. In this example, a learner tries to learn new concepts from several teachers. It then calculates learning accuracies without and with using social networks in communicating between the learner and the teachers. Our example consists of two case studies. In the first case study, only three teachers are used to teach a learner a new concept. Figure 5.6 illustrate a summary of stages of case study I. In the second case study, the number of teachers increases, one by one. Figure 5.7 is a summary of the stages of case study II. Learning accuracy is calculated in each stage using confusion matrix. In the next chapter, the results of all test scenarios explained in this chapter are illustrated.
Figure 5.6: Summary of test scenarios in case study I

Figure 5.7: Summary of test scenarios in case study II
Chapter 6

Test Results

To evaluate our concept learning module, we set up our learners to learn/update concepts from different numbers of teachers. As we mentioned in Chapter 5, each concept represents a unit or program in the taxonomy of a university. Therefore, by learning a concept, the learner provides the teachers with suggestions for how a unit concerned with that concept may be characterized. These suggestions may be keywords only or keywords accompanied with some conceptual information as illustrated in the steps of our case studies described in Chapter 5. In this chapter, we illustrate results of our case studies.

6.1 Case study I

In this case study, we test the effect of using social networks in learning accuracies. We use three teachers: \( Ag_C, Ag_M \) and \( Ag_W \), and two learners: \( Ag_L \) and \( Ag_G \). Three learning techniques are used: K-NN, Naïve Bayes and SVM in learning new concepts. This case study consists of four test scenarios.

6.1.1 Test scenario I

In this scenario, one learner only \( Ag_L \) is involved. \( Ag_L \) wants to learn two concepts: ”Computer Science” and ”Programming Language” from the three teachers. No social networks are defined for the communication between \( Ag_L \) and the teachers. This test scenario consists of three stages.
Stage 1

At the beginning, $A g_L$ does not have the concept ”Computer Science” in its ontology hierarchy as its ontology is empty. $A g_L$ uses only keywords to describe this concept. The teachers use these keywords to find the best matching concepts, $C_{best}$. Each teacher calculates $sim(q_{spec}, C_{best})$ (ratio between number of documents that satisfy the search keywords and the total number of documents for a concept) for each concept in its ontology. From Table 6.1 we can see that at the University of Michigan teacher, $A g_M$, both ”Chinese Studies” and ”Complex System” concepts have higher values of $sim(q_{spec}, C_{best})$ than ”Electrical Engineering and Computer Science” concept, but ”Electrical Engineering and Computer Science” concept is chosen to be $C_{best}$, because the number of documents returned from the search in both ”Chinese Studies” and ”Complex System” concepts are very low (one document each). The same case is with the University of Washington teacher, $A g_W$, where the concept ”Vietnamese” has higher value of $sim(q_{spec}, C_{best})$ Than ”Computer Science and Engineering” concept, but it is not chosen as $C_{best}$ because only three documents are returned from the search. Instead, the concept ”Computer Science and Engineering” is chosen as $C_{best}$ of $A g_W$. In our case study, we will not consider any concept that returns less than 10 documents from the search to make sure that this concept is well defined in a teacher’s ontology.
Table 6.1: Number of returned documents and similarity values of the search results for some concepts (The search keywords are ("computer science" OR "program language"))

<table>
<thead>
<tr>
<th>University/department</th>
<th># of docs matched</th>
<th>Total # of docs</th>
<th>( \text{Sim}(q_{\text{spec}}, C_{\text{best}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cornell University (Ag\text{C})</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer Science</td>
<td>31</td>
<td>87</td>
<td>0.36</td>
</tr>
<tr>
<td><strong>University of Michigan (Ag\text{M})</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical engineering and computer science</td>
<td>21</td>
<td>180</td>
<td>0.12</td>
</tr>
<tr>
<td>Chinese studies</td>
<td>1</td>
<td>4</td>
<td>0.25</td>
</tr>
<tr>
<td>Complex systems</td>
<td>1</td>
<td>6</td>
<td>0.17</td>
</tr>
<tr>
<td><strong>University of Washington (Ag\text{W})</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vietnamese</td>
<td>3</td>
<td>11</td>
<td>0.27</td>
</tr>
<tr>
<td>Computer science and engineering</td>
<td>23</td>
<td>92</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Based on Table 6.1, we chose the following concepts to be \( C_{\text{best}} \) from each teacher:

1. "Computer Science" from \( \text{Ag}\text{C} \).
2. "Electrical Engineering and Computer Science" from \( \text{Ag}\text{M} \).
3. "Computer Science and Engineering" from \( \text{Ag}\text{W} \).

According to the positive and negative example selection technique described in Chapter 4 (Yang, 2010), we set the threshold to 0.5. If \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) is higher than 0.5, then negative examples are chosen externally from the sibling concepts of \( C_{\text{best}} \). If
\( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) is less than 0.5, then negative examples are chosen internally from \( C_{\text{best}} \) from documents that are not returned by the search.

In this stage, for all teachers, \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) is less than 0.5, i.e., we select negative examples internally from the documents of the chosen concepts (\( C_{\text{best}} \)) that do not satisfy the search keywords. In this test scenario, no social networks are defined. All teachers have the same effect on \( A_{gL} \). From Table 6.1, the maximum number of positive examples can be chosen to represent the concept ”Electrical Engineering and Computer Science” from \( A_{gM} \) is 21. The numbers of positive and negative examples from each teacher should be the same. We got 21 positive examples and 21 negative examples from each teacher. Teachers send those positive and negative examples to \( A_{gL} \) to learn the new concept ”Computer Science”. We use three machine learning techniques: K-NN, Naïve Bayes, SVM. Confusion matrices and overall learning accuracies for the three learning techniques are presented in Table 6.2.

<table>
<thead>
<tr>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>True CS</td>
<td>True n-CS</td>
<td>True CS</td>
</tr>
<tr>
<td>Pred. CS</td>
<td>52</td>
<td>23</td>
</tr>
<tr>
<td>Pred. n-CS</td>
<td>11</td>
<td>40</td>
</tr>
<tr>
<td>Accuracy</td>
<td>73.02%</td>
<td>66.71%</td>
</tr>
</tbody>
</table>

After defining the newly learnt concept ”Computer Science” in \( A_{gL} \)’s ontology, we extract the feature vector of this concept. \( TF \times IDF \) is chosen as the metric to get the feature vector of concepts. Part of the extracted feature vector of the learnt con-
cept "Computer Science" is: \{program, langag, comput, scienc, system, design, logic, algorithm, compil, type, learn, analysi, function, object, cover\}.

Stage 2

The second stage in this test scenario is to use both keywords and conceptual information in finding $C_{best}$ in each teacher’s ontology to refine the definition of the newly learnt concept, "Computer Science", in $Ag_L$’s ontology. The keywords used are the same used in Stage 1. The conceptual information we use in this stage is the extracted feature vector of the learnt concept "Computer Science" from Stage 1. We compare the feature vector of "Computer Science" in $Ag_L$’s ontology with all feature vectors of all concepts in the teachers’ ontologies to get the matching values. We use these values plus the value of $sim(q_{spec}, C_{best})$ obtained by the search to decide on $C_{best}$ for each teacher.
Table 6.3: Similarity values of the search results and the similarity values between feature vectors and average value of both $sim(q_{spec}, C_{best})$ and similarity of feature vector (F.V.: feature vectors; AVG: average)

<table>
<thead>
<tr>
<th>University/department</th>
<th>$sim(q_{spec}, C_{best})$</th>
<th>Sim of F.V.</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cornell University (AgC)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer Science</td>
<td>0.36</td>
<td>0.62</td>
<td>0.49</td>
</tr>
<tr>
<td><strong>University of Michigan (AgM)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical engineering and computer science</td>
<td>0.12</td>
<td>0.49</td>
<td>0.31</td>
</tr>
<tr>
<td>Chinese studies</td>
<td>0.25</td>
<td>0.07</td>
<td>0.16</td>
</tr>
<tr>
<td>Complex systems</td>
<td>0.17</td>
<td>0.11</td>
<td>0.14</td>
</tr>
<tr>
<td><strong>University of Washington (AgW)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vietnamese</td>
<td>0.27</td>
<td>0.1</td>
<td>0.18</td>
</tr>
<tr>
<td>Computer science and engineering</td>
<td>0.25</td>
<td>0.34</td>
<td>0.30</td>
</tr>
</tbody>
</table>

From Table 6.3, we notice the low similarity between the feature vector of concept "Computer Science" sent from AgL and feature vectors of concepts "Chinese Studies", "Complex Systems" and "Vietnamese". That affects the AVG value for those concepts. That excludes these concepts from the selection as $C_{best}$ in AgM and AgW without considering the number of documents. Based on the value of AVG, the chosen best matching concepts from AgC, AgM and AgW are the same as in the previous stage (i.e. "Computer Science" from AgC, "Electrical Engineering and Computer Science" from AgM and "Computer Science and Engineering" from AgW). Positive and negative examples selection is done exactly the same as in Stage 1. In this stage, no social networks are defined.
between $Ag_t$ and the teachers. The same number of positive and negative examples is sent by each teacher. As in first stage, each teacher selects 21 positive examples and 21 negative examples to represent their $C_{best}$. The learning accuracies in this case are exactly the same as those in Stage 1 shown in Table 6.2. No improvement in the learning accuracy has occurred compared to Stage 1.

Stage 3

The third stage of this test scenario is to learn the new concept ”Programming Language” which is a child of the concept ”Computer Science”. In this case, we have the conceptual information of the parent concept ”Computer Science” (the extracted feature vector). We use this conceptual information along with an annotation query (”program language” | ”C++” | ”Java”) when searching the ontologies of each teacher for the best matching concept $C_{best}$. Table 6.4 shows the number of returned documents for some concepts and also $sim(q_{spec}, C_{best})$ of each of them.
Table 6.4: Number of returned documents, total number of documents and similarity values of the search results for some concepts. The search annotation is ("program language" | C++ | JAVA)

<table>
<thead>
<tr>
<th>University/department</th>
<th># of docs matched</th>
<th>Total # of docs</th>
<th>Sim ( (q_{spec}, C_{best}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cornell University (AgC)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer Science</td>
<td>25</td>
<td>87</td>
<td>0.29</td>
</tr>
<tr>
<td><strong>University of Michigan (AgM)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical engineering and computer science</td>
<td>12</td>
<td>180</td>
<td>0.06</td>
</tr>
<tr>
<td>School of Music/ Ensemble</td>
<td>2</td>
<td>36</td>
<td>0.06</td>
</tr>
<tr>
<td><strong>University of Washington (AgW)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer science and engineering</td>
<td>13</td>
<td>92</td>
<td>0.14</td>
</tr>
</tbody>
</table>

From Table 6.4, we can see that the best matching concepts selected are:

1. "Computer Science" from AgC.
2. "Electrical Engineering and Computer Science" from AgM.
3. "Computer Science and Engineering" from AgW.

We use the same strategy used previously in choosing positive and negative examples. Values of \( sim(q_{spec}, C_{best}) \) of all chosen concepts are less than 0.5. All negative examples of each concept are chosen internally from documents that do not satisfy the search query. The number of documents that satisfy the search query for concept "Electrical
Engineering and Computer Science” in $Ag_M$ is 12. That means, the maximum number of positive examples that can be chosen for this concept is 12. No social networks are defined in this stage. All teachers have the same effect on $Ag_L$ and have to use the same number of positive and negative examples. Each teacher selects 12 positive examples and 12 negative examples and sends them to $Ag_L$ to learn this new concept. $Ag_L$ uses the same machine learning techniques to get the learning accuracies of this stage, see Table 6.5.

Table 6.5: Confusion matrices and learning accuracies for learning concept ”Programming Language” to an empty learner and no social networks used (PL: Programming Language, n-PL: non Programming Language)

<table>
<thead>
<tr>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True PL</td>
<td>True n-PL</td>
</tr>
<tr>
<td>Pred. PL</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>Pred. n-PL</td>
<td>15</td>
<td>30</td>
</tr>
<tr>
<td>Accuracy</td>
<td>72.75%</td>
<td>62.2%</td>
</tr>
</tbody>
</table>

6.1.2 Test scenario II

In this test scenario, a social network is used for communication between $Ag_L$ and each teacher. Tie strengths affects the number of selected positive and negative examples from each teacher, which in turns may affect the learning accuracy. This test scenario consists of three stages:

Stage 1

In this stage, $Ag_L$ controls an empty repository (no concepts defined). At the beginning of this stage, all relationships between $Ag_L$ and the teachers have the same tie strengths.
Only keywords are used in this stage to describe the concept "Computer Science". The chosen \( C_{best} \) from all teachers will be exactly the same as those in Stage 1 of Test Scenario I.

In this stage, there is a social network defined between \( Ag_L \) and the teachers, but the strengths of ties between them are the same. For \( Ag_L \), all teachers have equal effect on the learning process and should send the same number of positive and negative examples for learning a new concept. Each teacher selects 21 positive examples and 21 negative examples to represent their \( C_{best} \). Based on these factors (i.e. same chosen concepts from each teacher, same number of positive and negative examples for each concept and same learning techniques), the learning accuracies are exactly the same as in Stage 1 of Test Scenario I. The learning accuracies of this stage are exactly the same as in Table 6.2. After finishing the learning process, \( Ag_L \) has a definition for the new concept "Computer Science". \( Ag_L \) creates a feature vector for this concept.

Stage 2

Now, we need to update the strength of ties between \( Ag_L \) and each teacher. We measure the closeness between the extracted feature vector of the newly learnt concept "Computer Science" in \( Ag_L \) and all concepts used by teachers. We calculate the strength of ties between \( Ag_L \) and each teacher as shown in Table 6.1. From these values we notice that \( Ag_L \) is closer to \( Ag_C \) and the next closest to \( Ag_W \) but far from \( Ag_M \).
Table 6.6: The strength of ties between $Ag_L$ and teachers $Ag_C$, $Ag_M$ and $Ag_W$ after learning the concept "Computer Science"

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th>Tie strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>0.49</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>0.087</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>0.18</td>
</tr>
</tbody>
</table>

In order to enhance the definition of the learnt concept "Computer Science", we use both keywords and conceptual knowledge (feature vector of the previously learnt concept "Computer Science" extracted in stage 1) in searching for $C_{best}$ in the teachers’ ontologies. The selected best concepts are the same as before.

Table 6.7: Number of positive and negative examples selected from each teacher for teaching "Computer Science" concept to $Ag_L$

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th># of +ve/-ve examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>31</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>6</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>11</td>
</tr>
</tbody>
</table>

Number of positive and negative examples selected from each teacher is proportional to the tie strengths between that teacher and $Ag_L$. Based on the values of tie strengths between $Ag_L$ and all teachers in Table 6.6 and the number of documents returned for the selected $C_{best}$ from Table 6.4, the number of positive and negative examples selected by each teacher can be calculated (see Table 6.7). We use 31 positive examples and 31 negative examples for concept "Computer Science" from $Ag_C$, 11 positive examples
and 11 negative examples for concept "Computer Science and Engineering" form $Ag_W$
and 6 positive examples and 6 negative examples for concept "Electrical Engineering and
Computer Science" from $Ag_M$. $Ag_L$ uses those sets of positives and negative examples to
relearn the concept "Computer Science". The learning accuracies are shown in Table 6.8.

Table 6.8: Confusion matrices and learning accuracies for learning concept "Computer
Science" to $Ag_L$ with a social network used (CS: Computer Science, n-CS: non Computer
Science)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True CS</td>
<td>True n-CS</td>
<td>True CS</td>
</tr>
<tr>
<td>Pred. CS</td>
<td>39</td>
<td>9</td>
<td>33</td>
</tr>
<tr>
<td>Pred. n-CS</td>
<td>9</td>
<td>39</td>
<td>15</td>
</tr>
<tr>
<td>Accuracy</td>
<td>81.21%</td>
<td>71.79%</td>
<td>70.83%</td>
</tr>
</tbody>
</table>

We notice that the accuracies of the learning process have improved in all used learning
techniques compared to Test Scenario I. The only difference between Test Scenario I and
Test Scenario II is the use of social networks. According to the above tie strengths used,
$Ag_L$ is closer to $Ag_C$, i.e. $Ag_L$ depends more on $Ag_C$ when it comes to the learning of new
concepts (i.e. the learner trusts more the teacher that controls the Cornell University
knowledge base than the two other teachers). That is why it uses more examples from
this teacher in the learning process. The opposite occurs with $Ag_M$. The strength of
the tie between $Ag_L$ and $Ag_M$ is the weakest. $Ag_L$ knows that the concept definitions in
this ontology are far from its own. $Ag_L$ cannot trust $Ag_M$ much, so the lowest number
of examples is considered from this agent. This helps improve the learning accuracy.
Table 6.9 shows the updated values of tie strengths between $Ag_L$ and all teachers. We
notice that $Ag_L$ gets closer to $Ag_C$ and further away from $Ag_M$.  

162
Table 6.9: Updated tie strength between AgL and teachers: AgC, AgM and AgW after relearning the concept "Computer Science"

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th>Tie strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University (AgC)</td>
<td>0.51</td>
</tr>
<tr>
<td>University of Michigan (AgM)</td>
<td>0.04</td>
</tr>
<tr>
<td>University of Washington (AgW)</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Stage 3
In order to test the efficiency of our social network used in the system, we try to learn a new concept "Programming Language", and compare the learning accuracies with those obtained in Stage 3 of test scenario I to see if there are any improvements or not. Using the same annotation used before, plus the feature vector of the newly learnt concept "Computer Science" from Stage 2, we get the same concepts as the best matching concepts from all teachers.

Table 6.10: Number of positive and negative examples selected from each teacher for teaching the concept "Programming Language" to AgL

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th># of +ve/-ve examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University (AgC)</td>
<td>25</td>
</tr>
<tr>
<td>University of Michigan (AgM)</td>
<td>2</td>
</tr>
<tr>
<td>University of Washington (AgW)</td>
<td>6</td>
</tr>
</tbody>
</table>

According to the tie strengths shown in Table 6.9, positive and negative examples are selected from each teacher. The number of positive and negative examples selected are proportional to the values of tie strengths between AgL and each teacher, see Ta-
Table 6.10. $Ag_L$ uses those sets of positive and negative examples to learn the new concept "Programming Language". The learning accuracies are shown in Table 6.11.

Table 6.11: Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_L$ with a social network used (PL: Programming Language, n-PL: non Programming Language)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True PL</td>
<td>True n-PL</td>
<td>True PL</td>
</tr>
<tr>
<td>Pred. PL</td>
<td>31</td>
<td>11</td>
<td>27</td>
</tr>
<tr>
<td>Pred. n-PL</td>
<td>2</td>
<td>22</td>
<td>6</td>
</tr>
<tr>
<td>Accuracy</td>
<td>80.33%</td>
<td>80.33%</td>
<td>78.90%</td>
</tr>
</tbody>
</table>

We notice that, the learning accuracies of all three learning techniques have improved compared to those obtained from Stage 3 of Test Scenario I where no social networks were defined.

In both Stages (2 and 3), the learning accuracies are improved compared to the correspondings in Test Scenario I. The only difference is the introduction of a social network for the communication between the learner and the teachers. We can conclude that using social networks in defining relationships between a learner and teachers has a positive effect on the learning accuracies even if the concept to be learnt is not a standalone concept in the teachers’ ontologies; as in the case of "Programming Language" concept (El-Sherif et al., 2012b).

In the following two test scenarios, we need to test if this positive effect of using social networks will continue if the learner controls a non-empty repository and has its own defined ontology.
Table 6.12: Part of the feature vector of the concept "Computer Science" in AgG’s ontology

<table>
<thead>
<tr>
<th>Feature word</th>
<th># of occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>comput</td>
<td>301</td>
</tr>
<tr>
<td>scienc</td>
<td>212</td>
</tr>
<tr>
<td>softwar</td>
<td>129</td>
</tr>
<tr>
<td>system</td>
<td>92</td>
</tr>
<tr>
<td>model</td>
<td>57</td>
</tr>
<tr>
<td>algorithm</td>
<td>48</td>
</tr>
<tr>
<td>mathematic</td>
<td>45</td>
</tr>
<tr>
<td>program</td>
<td>45</td>
</tr>
<tr>
<td>design</td>
<td>44</td>
</tr>
<tr>
<td>inform</td>
<td>40</td>
</tr>
<tr>
<td>analyse</td>
<td>37</td>
</tr>
<tr>
<td>secur</td>
<td>34</td>
</tr>
<tr>
<td>prepar</td>
<td>33</td>
</tr>
<tr>
<td>seng</td>
<td>27</td>
</tr>
<tr>
<td>network</td>
<td>26</td>
</tr>
<tr>
<td>develop</td>
<td>23</td>
</tr>
<tr>
<td>applic</td>
<td>21</td>
</tr>
<tr>
<td>communic</td>
<td>19</td>
</tr>
<tr>
<td>known</td>
<td>19</td>
</tr>
</tbody>
</table>

6.1.3 Test Scenario III

In this test scenario, we use same three teachers (AgC, AgM and AgW) and one learner AgG that controls a non-empty repository. AgG has a primary definition of concept "Computer Science" in its ontology. In this test scenario, no social networks are defined between AgG and the teachers. This test scenario consists of two stages.

Stage 1

In this stage, AgG extracts the feature vector of the exist concept "Computer Science". Some words of the feature vector and their occurrence are shown in Table 6.12.

In order to enhance the definition of the existing concept "Computer Science", AgG needs to relearn it from all teachers. AgG sends both keywords ("computer science"
OR "program language") and conceptual information (the extracted feature vector) to all teachers to search their ontologies for \( C_{\text{best}} \). The selected best concepts from each teacher are:

1. "Computer Science" from \( Ag_C \).
2. "Electrical Engineering and Computer Science" from \( Ag_M \).
3. "Computer Science and Engineering" from \( Ag_W \).

No social networks are defined in this test scenario. The number of positive and negative examples selected by each teacher is the same. Each teacher selects 21 positive examples and 21 negative examples. \( Ag_G \) uses these example sets to relearn the concept "Computer Science" using three learning techniques: K-NN, Naïve Bayes and SVM. The learning accuracies are shown in Table 6.13.

Table 6.13: Confusion matrices and learning accuracies for learning concept "Computer Science" to \( Ag_G \) with no social networks used (CS: Computer Science, n-CS: non Computer Science)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True CS</td>
<td>True n-CS</td>
<td>True CS</td>
</tr>
<tr>
<td>Pred. CS</td>
<td>52</td>
<td>23</td>
<td>42</td>
</tr>
<tr>
<td>Pred. n-CS</td>
<td>11</td>
<td>40</td>
<td>21</td>
</tr>
<tr>
<td>Accuracy</td>
<td>73.02%</td>
<td>66.71%</td>
<td>70.48%</td>
</tr>
</tbody>
</table>

Stage 2
In this stage, \( Ag_G \) needs to learn a new concept "Programming Language". This new concept is a child of concept "Computer Science". \( Ag_G \) sends the annotation describing
the concept ("program language" | "C++" | "Java") in addition to the feature vector of the parent concept "Computer Science" extracted in Stage 1 to all teachers. Each teacher uses this information to search its ontology for $C_{\text{best}}$. The selected $C_{\text{best}}$ from each teacher are the same as in Stage 1.

No social networks are defined in this test scenario. The number of positive and negative examples selected by each teacher is the same. Each teacher selects 12 positive examples and 12 negative examples and sends them to $Ag_G$. $Ag_G$ uses those examples to learn the new concept "Programming Language". The learning accuracies are shown in Table 6.14.

Table 6.14: Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_G$ with no social networks used (PL: Programming Language, n-PL: non Programming Language)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True PL</td>
<td>True n-PL</td>
<td>True PL</td>
</tr>
<tr>
<td>Pred. PL</td>
<td>18</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Pred. n-PL</td>
<td>15</td>
<td>30</td>
<td>13</td>
</tr>
<tr>
<td>Accuracy</td>
<td>72.75%</td>
<td>62.2%</td>
<td>63.74%</td>
</tr>
</tbody>
</table>

The next Test Scenario tests if using social networks helps improve the learning accuracies in the case of non-empty learner or not.

6.1.4 Test Scenario IV

In this test scenario, we use the same three teachers ($Ag_C$, $Ag_M$ and $Ag_W$) and same learner: $Ag_G$. In this test scenario, we define a social network between $Ag_G$ and all teachers. This test scenario consists of two stages:
Stage 1

In this stage, we need to set initial values of tie strengths between $Ag_G$ and each teacher. We can use concept definitions in $Ag_G$’s ontology to measure the closeness between $Ag_G$ and each teacher in order to set the initial values of tie strengths between them by calculating $(du)$ as described in Chapter 4 (Olivares-Ceja and Guzmán-Arenas, 2004). The initial values of tie strengths between $Ag_G$ and each teacher are shown in Table 6.15.

Table 6.15: Initial tie strengths values between $Ag_G$ and each teacher: $Ag_C$, $Ag_M$ and $Ag_W$

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th>Tie strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>0.19</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>0.26</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>0.17</td>
</tr>
</tbody>
</table>

In this case, we notice that $Ag_G$ is closer to $Ag_M$ than to the other two teachers. $Ag_G$ trusts $Ag_M$ more than other teachers during the learning process.

$Ag_G$ extracts a feature vector of the existing ”Computer Science” concept. $Ag_G$ uses both keywords and conceptual knowledge (feature vector extracted) in describing the concept ”Computer Science” to all teachers. Each teacher searches its ontology for $C_{best}$. The following concepts are chosen as $C_{best}$ from each teacher:

1. ”Computer Science” from $Ag_C$.
2. ”Electrical Engineering and Computer Science” from $Ag_M$.
3. ”Computer Science and Engineering” from $Ag_W$.

Based on tie strengths between $Ag_G$ and $Ag_C$, $Ag_M$ and $Ag_W$ obtained in Table 6.15 and the number of returned documents for each concept, the number of positive and
negative examples chosen from each teacher is calculated as shown in Table 6.16. The number of positive and negative examples is proportional to the value of tie strengths between $Ag_G$ and each teacher. In this case, we use 21 positive examples and 21 negative examples for the concept "Electrical and Computer Science" from $Ag_M$, we use 14 positive examples and 14 negative examples for the concept "Computer Science and Engineering" from $Ag_W$ and we use 15 positive example and 15 negative examples for the concept "Computer Science" from $Ag_C$.

Table 6.16: Number of positive and negative examples selected by each teacher for teaching the concept "Computer Science" to $Ag_G$

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th># of +ve/-ve examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>15</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>21</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>14</td>
</tr>
</tbody>
</table>

Applying the same learning techniques used before, to learn the concept "Computer Science", we calculate the learning accuracies as shown in Table 6.17.
Table 6.17: Confusion matrices and learning accuracies for learning concept "Computer Science" to $Ag_G$ with a social network used (CS: Computer Science, n-CS: non Computer Science)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True CS</td>
<td>True n-CS</td>
<td>True CS</td>
</tr>
<tr>
<td>Pred. CS</td>
<td>48</td>
<td>24</td>
<td>36</td>
</tr>
<tr>
<td>Pred. n-CS</td>
<td>2</td>
<td>26</td>
<td>14</td>
</tr>
<tr>
<td>Accuracy</td>
<td>74.00%</td>
<td>76.00%</td>
<td>84.00%</td>
</tr>
</tbody>
</table>

Comparing accuracies obtained in Test Scenario III (Table 6.13) and those obtained here in Table 6.17, we notice that the accuracy of all learning algorithms have improved compared to those in Stage 1 of Test Scenario III (where no social networks are used).

The next step is to update tie strengths values by measuring the new closeness values between $Ag_G$ and each teacher. The new tie strength values are shown in Table 6.18.

Table 6.18: New tie strengths values between $Ag_G$ and all teachers after relearning concept "Computer Science"

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th>Tie strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>0.20</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>0.24</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>0.18</td>
</tr>
</tbody>
</table>

We notice that the strength of ties between $Ag_G$ and both $Ag_C$ and $Ag_W$ increase and the strength of the tie between $Ag_G$ and $Ag_M$ decreases. This is because $C_{best}$ used from $Ag_M$ is part of the concept "Electrical and Computer Science", so it contains additional
examples for other courses used for electrical engineering plus those used for the concept 
"Computer Science" which is not the case in the updated concept "Computer Science" for AgG. This is a clear indication that the establishment of strong ties between a learner and some teachers in the beginning of the learning process does not mean that these strong ties would carry on in the future. In our case, AgG starts with a strong tie with AgM and weaker ties with AgC and AgW. After learning a new concept from those teachers and because the used concept from AgM is not accurate enough, the strength of the tie between AgG and AgM decreases. Moreover, the definition of the selected concepts in both AgC and AgW are much better in describing the concept "Computer Science", the strengths of ties between AgG and both AgC and AgW increase. According to these results, we can say that, the learning process will not end up being biased to one teacher or a small group of teachers even if they initially have strong ties between them.

This triggers another interesting property of social networks. This property is the ability of an agent in our social network to lose a friend. In this case, tie strength between AgC and AgM decreased after the learning of concept "Computer Science", because the best concept selected by AgM was not accurate enough. If AgM continues teaching AgC inaccurate concepts, the tie strengths between them will continue degrading until the relation between them ends and AgC loses its friend AgM.

Stage 2

The next stage in this scenario is to learn a new concept "Programming Language" which is a child of the concept "Computer Science". AgG sends the same annotation used in Stage 3 of Test Scenario I plus the feature vector of the parent concept "Computer Science" extracted in Stage 1 to all teachers. Each teacher searches its ontology for Cbest of the new concept "Programming Language". The same concepts are chosen as Cbest from each teacher as in the previous stage.
Table 6.19: Number of positive and negative examples selected from each teacher for teaching the concept "Programming Language" to $Ag_G$

<table>
<thead>
<tr>
<th>Teacher agent</th>
<th># of +ve/-ve examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University ($Ag_C$)</td>
<td>10</td>
</tr>
<tr>
<td>University of Michigan ($Ag_M$)</td>
<td>12</td>
</tr>
<tr>
<td>University of Washington ($Ag_W$)</td>
<td>9</td>
</tr>
</tbody>
</table>

A social network is used in this stage to define relationships between $Ag_G$ and each teacher. The effect of all teachers on $Ag_G$ during the learning process is not the same. The number of positive and negative examples selected by each teacher is different and depends on the new tie strengths calculated in Table 6.18 and the number of documents returned from the search. The number of positive and negative examples chosen from each teacher are shown in Table 6.19. We apply the same learning techniques used before to learn the new concept "Programming Language". The learning accuracies of this stage are shown in Table 6.20.

Table 6.20: Confusion matrices and learning accuracies for learning concept "Programming Language" to $Ag_G$ with a social network used (PL: Programming Language, n-PL: non Programming Language)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True PL</td>
<td>True n-PL</td>
<td>True PL</td>
</tr>
<tr>
<td>Pred. PL</td>
<td>29</td>
<td>7</td>
<td>27</td>
</tr>
<tr>
<td>Pred. n-PL</td>
<td>2</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>Accuracy</td>
<td>85.48%</td>
<td>88.71%</td>
<td>87.10%</td>
</tr>
</tbody>
</table>
Comparing learning accuracies obtained earlier in Test Scenario III (Table 6.14) and those obtained in this stage in Table 6.20, we notice that all learning accuracies are improved using all three learning techniques compared to the accuracies of Stage 2 of test scenario III where no social networks are defined in the system.

In Test Scenarios III and IV, we use the same teachers (AgC, AgM and AgW). The learner is the same (AgG), The concepts to be learnt are described the same way. The best matching concepts selected in learning both concepts ”Computer Science” and ”Programming Language” are the same from all teachers. The only difference is the use of a social network. From the learning accuracies obtained in each stage in both test scenarios, we notice the improvement which occurred either in updating the existing concept ”Computer Science” or in learning a new concept ”Programming Language”, even if it is not exist as a standalone concept in the teachers’ ontologies. We can conclude that this improvement is due to the introduction of social networks in our system (El-Sherif et al., 2012a).

We repeated the same learning process again with more concepts. The three teachers teach the following concepts to the learner: Chemistry, German, Linguistics, Mathematics and Physics. Table 6.21 shows learning accuracies for those concepts without using any social networks in communicating between the learner and the teachers using three learning techniques (K-NN, Naïve Bayes and SVM). Table 6.22 shows learning accuracies for learning the same concepts from the same teachers and using the same learning techniques, but this time with a social network used in communicating between the learner and the teachers.
Table 6.21: Summary of the learning accuracies for learning different concepts without using social networks

<table>
<thead>
<tr>
<th>Concept</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemistry</td>
<td>82.09%</td>
<td>77.08%</td>
<td>80.49%</td>
</tr>
<tr>
<td>German</td>
<td>76.37%</td>
<td>72.62%</td>
<td>78.77%</td>
</tr>
<tr>
<td>Linguistics</td>
<td>80.51%</td>
<td>79.79%</td>
<td>84.08%</td>
</tr>
<tr>
<td>Mathematics</td>
<td>86.27%</td>
<td>82.22%</td>
<td>85.33%</td>
</tr>
<tr>
<td>Physics</td>
<td>87.33%</td>
<td>84.91%</td>
<td>87.09%</td>
</tr>
</tbody>
</table>

Table 6.22: Summary of the learning accuracies for learning different concepts by using social networks

<table>
<thead>
<tr>
<th>Concept</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemistry</td>
<td>85.88%</td>
<td>79.41%</td>
<td>84.71%</td>
</tr>
<tr>
<td>German</td>
<td>87.75%</td>
<td>86.32%</td>
<td>87.03%</td>
</tr>
<tr>
<td>Linguistics</td>
<td>88.90%</td>
<td>84.87%</td>
<td>87.43%</td>
</tr>
<tr>
<td>Mathematics</td>
<td>89.22%</td>
<td>84.54%</td>
<td>87.63%</td>
</tr>
<tr>
<td>Physics</td>
<td>88.42%</td>
<td>87.81%</td>
<td>88.42%</td>
</tr>
</tbody>
</table>

Comparing learning accuracies obtained in learning those new concepts (Chemistry, German, Linguistics, Mathematics and Physics), we notice that using social networks for the communication between the learner and the teachers has a positive effect on improving the learning accuracies in all cases independent of the learning techniques used.

For the first case study, we conclude that using social networks in both learning new
concepts from different teachers or updating existing concepts by the help of different
teachers, have always a positive effect on the learning accuracies either if the learner
controls an empty repository or has a predefined ontology in its repository.

Most of research done in the area of concept learning based on MAS considered
either mapping to a certain ontology or agent-to-agent communication during learning
new concepts not full multi-agent collaborative. The researchers who handle the problem
of learning new concepts from many teachers at the same time deal with all teachers with
the same degree of trust. All teachers involved in the learning process have the same
effect on the learner. No previous research considered differentiating between teachers.
That is the reason why we could not compare our results with other research. We do not
have any results to compare our results with except the data set we built.

6.2 Case study II

The aim of this case study is to:

1. Test the effect of increasing the number of teachers on learning accuracies
   when learning a new concept from several teachers for both empty and
   non-empty learners.

2. Study the effect of using social networks on learning accuracies with a large
   number of teachers with both empty and non-empty learners.

In this case study, we set up twenty six ontologies for course syllabi of twenty six
universities to represent our teachers: Cornell University, the University of Michigan,
the University of Washington, University of Victoria, Simon Fraser University, the
University of Northern British Columbia, Dalhousie University, the University of New
Brunswick, the University of Manitoba, the University of Winnipeg, Memorial Univer-

sity, the University of British Columbia, the University of Alberta, Vancouver Island
University, the University of the Fraser Valley, Mount Allison University, Mount Royal University, Athabasca University, King’s University, University of Lethbridge, Kwantlen Polytechnic University, Thompson Rivers University, Trinity Western University, Acadia University, Mount Saint Vincent University and Saint Mary’s University. In order to perform our case study, the learner needs to learn the concept "Computer Science" from all teachers. The learner sends keywords ("computer science" OR "program language") that describe the concepts to all teachers. Each teacher searches its ontology for the best matching concept with the highest value of $\text{sim}(q_{\text{spec}}, C_{\text{best}})$. Table 6.23 shows the number of documents returned from all teachers for their $C_{\text{best}}$.

From Table 6.23, we notice that the number of documents returned for $C_{\text{best}}$ in nine teachers (Athabasca University, King’s University, University of Lethbridge, Kwantlen Polytechnic University, Thompson Rivers University, Trinity Western University, Acadia University, Mount Saint Vincent University and Saint Mary’s University) is less than ten documents. That means that maximum number positive examples can be used to teach concept "Computer Science" from those teachers is less than ten examples. That means concept "Computer Science" is not well defined in those teachers' ontologies. Using these teachers during the learning will degrade the learning accuracies dramatically. Those teachers are unable to teach this concept to any learner. These teachers are excluded from our teacher list. We only use the remaining seventeen teachers those control ontologies of the following universities: Cornell University, the University of Michigan, the University of Washington, University of Victoria, Simon Fraser University, the University of Northern British Columbia, Dalhousie University, the University of New Brunswick, the University of Manitoba, the University of Winnipeg, Memorial University, the University of British Columbia, the University of Alberta, Mount Allison University, Vancouver Island University, Mount royal University and the University of the Fraser Valley.
Table 6.23: Number of returned documents from the search and value of $sim(q_{spec}, C_{best})$ for the selected $C_{best}$ in all teachers

<table>
<thead>
<tr>
<th>University</th>
<th># of docs matched</th>
<th>Total # of docs</th>
<th>$Sim(q_{spec}, C_{best})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University</td>
<td>31</td>
<td>87</td>
<td>0.36</td>
</tr>
<tr>
<td>University of Michigan</td>
<td>21</td>
<td>180</td>
<td>0.12</td>
</tr>
<tr>
<td>University of Washington</td>
<td>23</td>
<td>92</td>
<td>0.25</td>
</tr>
<tr>
<td>University of Alberta</td>
<td>14</td>
<td>62</td>
<td>0.23</td>
</tr>
<tr>
<td>Athabasca University</td>
<td>9</td>
<td>35</td>
<td>0.25</td>
</tr>
<tr>
<td>King’s University</td>
<td>9</td>
<td>27</td>
<td>0.33</td>
</tr>
<tr>
<td>Mount royal University</td>
<td>11</td>
<td>44</td>
<td>0.25</td>
</tr>
<tr>
<td>University of Lethbridge</td>
<td>9</td>
<td>28</td>
<td>0.32</td>
</tr>
<tr>
<td>Kwantlen Polytechnic University</td>
<td>4</td>
<td>6</td>
<td>0.67</td>
</tr>
<tr>
<td>Simon Fraser University</td>
<td>21</td>
<td>135</td>
<td>0.16</td>
</tr>
<tr>
<td>Thompson Rivers University</td>
<td>6</td>
<td>46</td>
<td>0.13</td>
</tr>
<tr>
<td>Trinity Western University</td>
<td>6</td>
<td>34</td>
<td>0.18</td>
</tr>
<tr>
<td>The University of British Columbia</td>
<td>14</td>
<td>56</td>
<td>0.25</td>
</tr>
<tr>
<td>University of the Fraser Valley</td>
<td>10</td>
<td>22</td>
<td>0.45</td>
</tr>
<tr>
<td>University of Victoria</td>
<td>23</td>
<td>101</td>
<td>0.23</td>
</tr>
<tr>
<td>Vancouver Island University</td>
<td>13</td>
<td>46</td>
<td>0.28</td>
</tr>
<tr>
<td>University of Manitoba</td>
<td>18</td>
<td>61</td>
<td>0.295</td>
</tr>
<tr>
<td>The University of Winnipeg</td>
<td>15</td>
<td>45</td>
<td>0.33</td>
</tr>
<tr>
<td>Mount Allison University</td>
<td>14</td>
<td>30</td>
<td>0.47</td>
</tr>
<tr>
<td>University of New Brunswick</td>
<td>18</td>
<td>53</td>
<td>0.34</td>
</tr>
<tr>
<td>Memorial University</td>
<td>14</td>
<td>51</td>
<td>0.27</td>
</tr>
<tr>
<td>Acadia University</td>
<td>5</td>
<td>34</td>
<td>0.15</td>
</tr>
<tr>
<td>Dalhousie University</td>
<td>18</td>
<td>67</td>
<td>0.27</td>
</tr>
<tr>
<td>Mount Saint Vincent University</td>
<td>8</td>
<td>17</td>
<td>0.47</td>
</tr>
<tr>
<td>Saint Mary’s University</td>
<td>9</td>
<td>30</td>
<td>0.3</td>
</tr>
</tbody>
</table>
In Case Study I, we had three teachers only. In this case study II, we start with four teachers and increase them by gradually increasing the number of teachers and determining the learning accuracy. We have two learners. The first one is $Ag_L$ controls an empty repository. The second one is $Ag_G$ that controls a repository that contains an ontology hierarchy of course syllabi of the University of Calgary. We use the same learning techniques used in Case Study I (K-NN, Naïve Bayes and SVM) to learn the concept "Computer Science". This case study consists of three test scenarios:

6.2.1 Test scenario I

In this test scenario we start with an empty learner, $Ag_L$, and the initial number of teachers is four. No social networks are used in this test scenario. This test scenario consists of two stages:

Stage 1

In this stage, we use only keywords to define the concept need to be learnt. Each teacher searches its ontology to get the best matching concept ($C_{best}$). Based on the number of documents returned from the search and the value of $sim(q_{spec}, C_{best})$, $C_{best}$ can be determined by each teacher. No social networks are defined in this test scenario. All teachers have the same effect on $Ag_L$. The numbers of positive and negative examples sent by each teacher are the same. The teachers send those positive and negative examples for the concept "Computer Science" to $Ag_L$. $Ag_L$ uses three machine learning techniques: K-NN, Naïve Bayes and SVM to learn this concept one at a time. After learning the new concept "Computer Science", the learning accuracies are calculated in each case (see Table 6.24).

After leaning the new concept "Computer Science", $Ag_L$ extracts feature vectors of the learnt concept each time to be used in the next stage of this test scenario.
Table 6.24: Summary of the learning accuracies for learning $Ag_L$ the new concept "Computer Science" from different number of teachers without defining any social networks in communication between learner and teachers

<table>
<thead>
<tr>
<th># of teachers</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>71.43%</td>
<td>69.63%</td>
<td>72.60%</td>
</tr>
<tr>
<td>5</td>
<td>68.57%</td>
<td>72.86%</td>
<td>70.95%</td>
</tr>
<tr>
<td>6</td>
<td>70.00%</td>
<td>69.58%</td>
<td>71.67%</td>
</tr>
<tr>
<td>7</td>
<td>71.44%</td>
<td>66.30%</td>
<td>72.64%</td>
</tr>
<tr>
<td>8</td>
<td>73.92%</td>
<td>68.05%</td>
<td>70.47%</td>
</tr>
<tr>
<td>9</td>
<td>72.22%</td>
<td>67.60%</td>
<td>73.47%</td>
</tr>
<tr>
<td>10</td>
<td>70.00%</td>
<td>67.67%</td>
<td>73.00%</td>
</tr>
<tr>
<td>11</td>
<td>70.15%</td>
<td>65.57%</td>
<td>70.13%</td>
</tr>
<tr>
<td>12</td>
<td>70.54%</td>
<td>66.68%</td>
<td>72.02%</td>
</tr>
<tr>
<td>13</td>
<td>68.68%</td>
<td>65.93%</td>
<td>69.23%</td>
</tr>
<tr>
<td>14</td>
<td>66.84%</td>
<td>63.27%</td>
<td>68.62%</td>
</tr>
<tr>
<td>15</td>
<td>65.13%</td>
<td>62.82%</td>
<td>66.41%</td>
</tr>
<tr>
<td>16</td>
<td>62.78%</td>
<td>58.52%</td>
<td>62.22%</td>
</tr>
<tr>
<td>17</td>
<td>59.71%</td>
<td>54.41%</td>
<td>59.41%</td>
</tr>
</tbody>
</table>

From Table 6.24, the learning accuracy varies when adding new teachers but overall the accuracies are still acceptable until the number of teachers reaches twelve. By adding more teachers, the leaning accuracies, in all learning techniques, steadily decrease.

At the beginning, the learning accuracies decrease after adding one more teacher using one learning technique, but it goes up again after adding one or two more teachers. For example, for five teachers, the learning accuracy using K-NN goes down from 71.43% to
68.57%. It also goes down using SVM from 72.60% to 70.95%. At the same time, the learning accuracy using Naïve Bayes goes up from 69.63% to 72.86%. By adding one more teacher (six teachers), the learning accuracies using both K-NN and SVM start to recover and go up to 70.00% and 71.67% respectively. But the one using Naïve Bayes starts to go down to 69.58%. This decreasing of the learning accuracy using Naïve Bayes continues also with seven teachers and goes down to 66.30%. With eight teachers, the learning accuracy using Naïve Bayes starts to recover and increased to 68.05%. This behaviour of learning accuracies of going up and down continues for all three learning techniques used till the number of teachers reaches twelve teachers. Starting from thirteen teachers, the learning accuracies of all learning techniques start going down rapidly by adding more teachers and never recover back till the number of teachers reaches seventeen teachers (all teachers defined in our case study).

We can conclude that at a certain number of teachers, increasing the number of teachers when teaching an empty learner new concepts negatively affects the learning accuracy (see Figure 6.1).

Figure 6.1: A graph that represent the relationship between increasing the number of teachers and the leaning accuracy for learning a new concept to an empty learner with no social networks used and using three learning techniques

180
Stage 2
In this stage, $Ag_L$ tries to enhance the definition of the newly learnt concept "Computer Science". $Ag_L$ sends the same keywords used in Stage 1 along with the feature vector extracted in the previous stage to all teachers. All teachers use this information to search for the best matching concepts in their ontologies. The best matching concepts returned by teachers in this stage are exactly the same as those in Stage 1.

No social networks are used in this stage. All teachers have the same effect on $Ag_L$. All teachers send the same number of positive and negative examples to $Ag_L$ (the same example sets used in Stage 1). Based on that (i.e. same best matching concepts chosen by teachers and same positive and negative example sets sent by all teachers), the learning accuracies are exactly the same as those in Table 6.24. The learning accuracies do not change and no enhancement occur.

6.2.2 Test scenario II
In this test scenario, we start with four teachers and gradually increase their number to seventeen. The learner in this test scenario is $Ag_L$ that controls an empty repository. In this test scenario, a social network is used to represent relationships between $Ag_L$ and the teachers. We use the same three learning techniques to learn the concept "Computer Science". This test scenario consists of two stages:

Stage 1
As $Ag_L$’s ontology is empty, so strengths of ties between $Ag_L$ and all teachers are the same at the beginning. All teachers equally affect the learner during the learning process.

In this stage, $Ag_L$ only uses same keywords used in test scenario I to describe the concept "Computer Science". $Ag_L$ sends those keywords to all teachers to search their ontologies for $C_{\text{best}}$. The selected concepts are exactly the same as in test scenario I and so are the positive and negative example sets selected by each teacher. The learning
accuracies of this stage are the same as those obtained in Test Scenario I (Table 6.24). The obtained results have the same deficiency (i.e. starting from thirteen teachers, learning accuracies decreases with increasing the number of teachers). AgL extracts the feature vector of the newly learnt concept ”Computer Science”. AgL updates the tie strengths with all teachers,

Stage 2
In this stage, we aim at enhancing the learning accuracies obtained in Stage 1 by relearning the concept ”Computer Science” from all teachers. AgL sends the same keywords that describe the concept ”Computer Science” used in Stage 1 along with feature vectors extracted in the previous stage to all teachers to search their ontologies for the best matching concepts. The chosen concepts by each teacher are the same as in stage 1. Using the tie strengths updated in the previous stage and the number of documents returned from the keyword search by each teacher, we can determine the corresponding number of positive and negative examples sent by each teacher. The number of positive and negative examples sent by each teacher is proportional to the tie strength between AgL and this teacher. All teachers send their example sets to AgL to relearn the concept ”Computer Science”. AgL, then, calculates the learning accuracies each time.

Table 6.25 shows the summary of the learning accuracies using different numbers of teachers for each learning technique used.
Table 6.25: Summary of learning accuracies for concept "Computer Science" from different number of teachers with defining a social network in communication between $A_{gL}$ and the teachers using three learning techniques

<table>
<thead>
<tr>
<th># of teachers</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>80.81%</td>
<td>78.86%</td>
<td>79.81%</td>
</tr>
<tr>
<td>5</td>
<td>81.33%</td>
<td>78.67%</td>
<td>78.67%</td>
</tr>
<tr>
<td>6</td>
<td>80.11%</td>
<td>77.46%</td>
<td>78.57%</td>
</tr>
<tr>
<td>7</td>
<td>81.00%</td>
<td>74.00%</td>
<td>79.00%</td>
</tr>
<tr>
<td>8</td>
<td>79.13%</td>
<td>76.09%</td>
<td>76.52%</td>
</tr>
<tr>
<td>9</td>
<td>77.78%</td>
<td>74.76%</td>
<td>77.35%</td>
</tr>
<tr>
<td>10</td>
<td>76.30%</td>
<td>76.28%</td>
<td>75.53%</td>
</tr>
<tr>
<td>11</td>
<td>76.56%</td>
<td>69.47%</td>
<td>73.12%</td>
</tr>
<tr>
<td>12</td>
<td>77.73%</td>
<td>69.98%</td>
<td>77.05%</td>
</tr>
<tr>
<td>13</td>
<td>74.14%</td>
<td>68.35%</td>
<td>71.24%</td>
</tr>
<tr>
<td>14</td>
<td>73.28%</td>
<td>65.69%</td>
<td>74.37%</td>
</tr>
<tr>
<td>15</td>
<td>74.95%</td>
<td>66.40%</td>
<td>72.62%</td>
</tr>
<tr>
<td>16</td>
<td>74.77%</td>
<td>67.56%</td>
<td>73.28%</td>
</tr>
<tr>
<td>17</td>
<td>75.64%</td>
<td>70.94%</td>
<td>75.64%</td>
</tr>
</tbody>
</table>

From Table 6.25, we notice the following:

- First, all learning accuracies have improved compared to their corresponding values in Table 6.24 in Stage 1 where no social networks are used. This improvement occurred in all learning techniques used and with any number of teachers from four teachers up to seventeen teachers.
• Second, in the first part of Table 6.25, the behaviour of the learning accuracy values are the same as in Stage 1 (up to twelve teachers). The learning accuracies change by increasing the number of teachers. In some cases it goes down then it recovers and goes up again. For example, using six teachers, the learning accuracies for all three learning techniques: K-NN, Naïve Bayes and SVM, goes down to 80.19%, 77.46% and 78.57% respectively. By adding one more teacher (i.e., total of seven teachers), the learning accuracies of K-NN and SVM go up to 81.00% and 79.00% respectively. The learning accuracy of Naïve Bayes keeps going down to 74.00% but by increasing one more teacher (eight teachers), it starts to recover and its learning accuracy increases to 76.09%.

• With more than twelve teachers, by adding more teachers, the same behaviour of the learning accuracies continues. It keeps going down and recovering up again by adding more teachers which is not the case in stage 1 where no social networks are used. Even at seventeen teachers (all teachers defined in our case study), the learning accuracies for all the three learning techniques are higher than those for sixteen teachers.

We conclude that the effect of increasing the number of teachers on the learning accuracy is much lower using social networks than this effect if no social networks are used, as shown in Figure 6.2.
Figure 6.2: A graph that represents the relationship between increasing the number of teachers and the learning accuracy for learning a new concept to an empty learner while using a social network using three learning techniques.

Figures 6.3, 6.4, and 6.5 compare the learning accuracies with and without social networks used in our system in each learning technique used. These figures show clearly the effect of using a social network in our concept learning module in learning a concept while increasing the number of teachers using three learning techniques: K-NN, Naïve Bayes and SVM. In these figures, we notice that using a social network always improves the learning accuracies with any number of teachers. We notice too that without a social network, the curve of the learning accuracy always goes down after a while, although the other curve (representing learning accuracy with social network used) keeps going up and down within the same range of learning accuracy values for all number of teachers.
Figure 6.3: A graph compares between learning accuracies of learning a new concept to $A_gL$ with and without social networks using K-NN learning algorithm (SN: social network)

Figure 6.4: A graph compares between learning accuracies of learning a new concept to $A_gL$ with and without social networks using Naïve Bayes learning algorithm (SN: social network)
We can conclude that using social networks in defining relationships between the learner and the teachers always positively affects the learning accuracies in the three learning techniques used. It also decreases the effect of increasing the number of teachers participating in the learning process.

6.2.3 Test scenario III

In this test scenario, we start with a non-empty learner, $Ag_L$. A preliminary definition of the concept ”Computer Science” already exists in $Ag_G$’s ontology. In this test scenario, we increase the number of teachers one by one and relearn the same concept ”Computer Science” using three learning techniques K-NN, Naïve Bayes and SVM in order to study the effect of increasing the number of teachers on the learning accuracy having the learner with non-empty repository. We calculate the overall learning accuracies each time. This test scenario consists of two stages:
Stage 1

In this stage, we do not consider any social networks when communicating between $Ag_G$ and the teachers. We start with four teachers then increase them to seventeen teachers.

As we said earlier, $Ag_G$ has a non-empty repository, i.e. it has already an initial definition for the concept "Computer Science" in its ontology hierarchy but it needs to update this definition by asking the teachers to relearn it. It uses both keywords, ("computer science" OR "program language"), and conceptual information (i.e. feature vector of existing concept "Computer Science" in $Ag_G$’s ontology) to describe this concept.

Each teacher searches its ontology to get $C_{best}$. Based on the number of documents and the value of $\text{sim}(q_{spec}, C_{best})$, $C_{best}$ of each teacher can be determined. As no social networks are defined in this stage, all teachers have the same effect on $Ag_G$ and numbers of positive and negative examples sent by each teacher are the same. Teachers send those positive and negative examples for the concept "Computer Science" to $Ag_G$. The learning accuracy is calculated in each case. After learning the concept "Computer Science", $Ag_G$ updates the definition of this concept in its ontology. Table 6.26 shows the learning accuracies using different numbers of teachers with each learning technique.
Table 6.26: Learning accuracies for learning concept ”Computer Science” from different number of teachers without defining any social networks in communication between $A_gG$ and the teachers having $A_gG$ controls a non-empty repository using three learning techniques

<table>
<thead>
<tr>
<th># of teachers</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>71.43%</td>
<td>69.63%</td>
<td>72.60%</td>
</tr>
<tr>
<td>5</td>
<td>68.57%</td>
<td>72.86%</td>
<td>70.95%</td>
</tr>
<tr>
<td>6</td>
<td>70.00%</td>
<td>69.58%</td>
<td>71.67%</td>
</tr>
<tr>
<td>7</td>
<td>71.44%</td>
<td>66.30%</td>
<td>72.64%</td>
</tr>
<tr>
<td>8</td>
<td>73.92%</td>
<td>68.05%</td>
<td>70.47%</td>
</tr>
<tr>
<td>9</td>
<td>72.22%</td>
<td>67.60%</td>
<td>73.47%</td>
</tr>
<tr>
<td>10</td>
<td>70.00%</td>
<td>67.67%</td>
<td>73.00%</td>
</tr>
<tr>
<td>11</td>
<td>70.15%</td>
<td>65.57%</td>
<td>70.13%</td>
</tr>
<tr>
<td>12</td>
<td>70.54%</td>
<td>66.68%</td>
<td>72.02%</td>
</tr>
<tr>
<td>13</td>
<td>68.68%</td>
<td>65.93%</td>
<td>69.23%</td>
</tr>
<tr>
<td>14</td>
<td>66.84%</td>
<td>63.27%</td>
<td>68.62%</td>
</tr>
<tr>
<td>15</td>
<td>65.13%</td>
<td>62.82%</td>
<td>66.41%</td>
</tr>
<tr>
<td>16</td>
<td>62.78%</td>
<td>58.52%</td>
<td>62.22%</td>
</tr>
<tr>
<td>17</td>
<td>59.71%</td>
<td>54.41%</td>
<td>59.41%</td>
</tr>
</tbody>
</table>

We notice that, the learning accuracy for learning the concept ”Computer Science” does not vary a lot by increasing the number of teachers. Afterwords, the accuracy of the learning decreases continuously in all learning techniques used by increasing the number of teachers as shown in Figure 6.6. We can say that from a certain number of teachers, increasing the number of teachers negatively affects the learning accuracy (same
behaviour as in Test Scenario I).

Figure 6.6: A graph that represent the relationship between increasing The number of teachers and the leaning accuracy of learning a new concept to $Ag_G$ with no social networks used

Stage 2
Using predefined ontology in the learner’s repository causes it to have initially strong tie strengths with some teachers and weak tie strengths with others. In this stage, we want to test if using social networks between a learner that controls a non-empty repository and teachers will negatively or positively affect the learning accuracy.

In this stage, we define a social network between $Ag_G$ and the teachers. We get the closeness between $Ag_G$’s ontology and each teacher’s ontology and consider these values as initial values of tie strengths between them (see Table 6.27). We start with four teachers and increase them one at a time till seventeen teachers (all teachers available for our case study). $Ag_G$ sends keywords that describe the concept ”Computer Science”, along with the feature vector of the already existing concept ”Computer Science” to all teachers. Teachers use sent information to search their ontologies for $C_{best}$. Depending on values of ties strengths between $Ag_G$ and each teacher, the number of documents returned during
the search for $C_{\text{best}}$ the number of positive and negative examples sent by each teacher is determined. $Ag_G$ uses those positive and negative examples sent by all teachers to relearn the concept ”Computer Science” and calculate the learning accuracies. Afterwards the learner updates its definition of the concept ”Computer Science” in its repository.

Table 6.27: Tie strengths values between $Ag_G$ and all teachers involved in the learning process

<table>
<thead>
<tr>
<th>University</th>
<th>Tie strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell University</td>
<td>0.19</td>
</tr>
<tr>
<td>University of Michigan</td>
<td>0.26</td>
</tr>
<tr>
<td>University of Washington</td>
<td>0.17</td>
</tr>
<tr>
<td>University of Victoria</td>
<td>0.32</td>
</tr>
<tr>
<td>Simon Fraser University</td>
<td>0.29</td>
</tr>
<tr>
<td>University of Northern British Columbia</td>
<td>0.24</td>
</tr>
<tr>
<td>Dalhousie University</td>
<td>0.21</td>
</tr>
<tr>
<td>University of New Brunswick</td>
<td>0.22</td>
</tr>
<tr>
<td>University of Manitoba</td>
<td>0.19</td>
</tr>
<tr>
<td>The University of Winnipeg</td>
<td>0.21</td>
</tr>
<tr>
<td>Memorial University</td>
<td>0.23</td>
</tr>
<tr>
<td>The University of British Columbia</td>
<td>0.17</td>
</tr>
<tr>
<td>University of Alberta</td>
<td>0.29</td>
</tr>
<tr>
<td>Mount Allison University</td>
<td>0.14</td>
</tr>
<tr>
<td>Vancouver Island University</td>
<td>0.13</td>
</tr>
<tr>
<td>Mount royal University</td>
<td>0.19</td>
</tr>
<tr>
<td>University of the Fraser Valley</td>
<td>0.12</td>
</tr>
</tbody>
</table>
Table 6.28 is a summary of the learning accuracies calculated using all three learning techniques with different numbers of teachers.

Table 6.28: Learning accuracies for learning concept "Computer Science" from different number of teachers with defining a social network in communication between $Ag_G$ and the teachers having $Ag_G$ controls a non-empty repository using three learning techniques.

<table>
<thead>
<tr>
<th># of teachers</th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>79.29%</td>
<td>76.96%</td>
<td>82.14%</td>
</tr>
<tr>
<td>5</td>
<td>80.56%</td>
<td>75.00%</td>
<td>80.33%</td>
</tr>
<tr>
<td>6</td>
<td>75.61%</td>
<td>78.06%</td>
<td>76.94%</td>
</tr>
<tr>
<td>7</td>
<td>75.61%</td>
<td>78.06%</td>
<td>76.94%</td>
</tr>
<tr>
<td>8</td>
<td>81.11%</td>
<td>75.56%</td>
<td>77.78%</td>
</tr>
<tr>
<td>9</td>
<td>77.22%</td>
<td>76.22%</td>
<td>75.00%</td>
</tr>
<tr>
<td>10</td>
<td>82.33%</td>
<td>78.44%</td>
<td>78.44%</td>
</tr>
<tr>
<td>11</td>
<td>77.00%</td>
<td>76.00%</td>
<td>75.00%</td>
</tr>
<tr>
<td>12</td>
<td>82.86%</td>
<td>85.71%</td>
<td>82.62%</td>
</tr>
<tr>
<td>13</td>
<td>75.71%</td>
<td>77.14%</td>
<td>75.71%</td>
</tr>
<tr>
<td>14</td>
<td>76.81%</td>
<td>81.81%</td>
<td>71.25%</td>
</tr>
<tr>
<td>15</td>
<td>76.81%</td>
<td>81.81%</td>
<td>71.25%</td>
</tr>
<tr>
<td>16</td>
<td>78.06%</td>
<td>78.61%</td>
<td>73.75%</td>
</tr>
<tr>
<td>17</td>
<td>81.11%</td>
<td>82.22%</td>
<td>78.89%</td>
</tr>
</tbody>
</table>

From Table 6.28, we notice that the learning accuracies are all improved compared to the corresponding accuracies calculated in Table 6.26 in Stage 1 (where no social networks are defined). By increasing the number of teachers, the learning accuracies keep going up and down within the same range of values till the number of teachers reaches seventeen.
(all teachers available in our case study). That means, although the learner has an initial ontology in its repository at the beginning of our test scenario, increasing the number of teachers does not negatively affect the learning accuracies if a social network is used in defining relationships between the learner and the teachers.

We can conclude that, the effect of increasing the number of teachers on the learning accuracy is weak when a social network is used in communicating between the learner and the teachers. Even with using seventeen teachers, the learning accuracy is still good and did not significantly decrease as shown in figure 6.7. We notice that the positive effect of using a social network in our system continues by increasing the number of teachers even when the learner controls a non-empty repository.

![Figure 6.7](image)

Figure 6.7: A graph that represent the relationship between increasing the number of teachers and the leaning accuracy while using a social network for a non-empty learner.

Figures 6.8, 6.9 and 6.10 compare the learning accuracy with and without using social networks in our system in each learning technique used. These figures show clearly the effect of using social networks in our system with each learning technique. For all three learning techniques used, while not using social networks, the learning accuracy curve always goes down after twelve teachers. By using social networks, learning accuracy
curve keeps oscillating around the same range of values even when increasing the number of teachers.

Figure 6.8: A graph compares between the learning accuracies with and without social networks using K-NN learning algorithm for a non-empty learner (SN: social network)

Figure 6.9: A graph compares between the learning accuracies with and without social networks using Naïve Bayes learning algorithm for a non-empty learner (SN: social network)
Finally, after completing our case study, we conclude that, for all learning techniques used: K-NN, Naïve Bayes and SVM, using social networks always increases the learning accuracies if the repository of the learner is either empty or non-empty. When no social networks are used, increasing the number of teachers does not affect the learning accuracies if the number of teachers is relatively low. When the number of teachers increases above a certain number (twelve teachers in our case), the learning accuracies go down continuously by increasing the number of teachers. On the other hand, while using social networks in our system, increasing the number of teachers does not affect the learning accuracies even if the number of teachers is relatively high compared to not using social networks.

In these case studies, we aim at improving the obtained learning accuracies by using social networks in communicating between agents. The results obtained are consistent with what we proposed of using social networks in the case of empty and non-empty learner. If the system is using a powerful learning technique, the learning accuracy will be additionally improved by using social networks. Our case studies did not show a case...
where using social networks negatively affects the learning accuracy. The improvement done may be affected by several factors: concept representation in each ontology, example sets used during the learning, information in the learning request sent by the learner which in turns affects the selection of the best matching concept. On the other hand, the improvement of the learning accuracy is not affected by either the topology of the network or the number of teachers involved in the learning process. The number of teachers that can participate in a learning process without affecting the learning accuracy increases in the case of using social networks. Using a social network with different tie strengths can help find the best number of teachers to learn new concepts.

6.3 Summary

In the first case study, we tested the effect of using social networks for the communicating between the learner and the teachers on the learning accuracies. Two concepts are learnt: ”Computer Science” and ”Programming Language”. We used two different learners: the first one controls an empty repository; the second one controls a non-empty repository with a predefined ontology. Three learning techniques are used each time: K-NN, Naïve Bayes and SVM. In all scenarios in this case study, the learning accuracies have improved by using social networks compared to those without social networks used. This shows that in learning new concepts from different teachers, using social networks for communicating between the learner and the teachers improves the accuracy of the learning. Table 6.29 summarize the learning accuracies of all scenarios in Case Study I.
Table 6.29: Summary of the learning accuracies of Case Study I (SN: Social Networks; CS: Computer Science; PL: Programming Language)

<table>
<thead>
<tr>
<th></th>
<th>K-NN</th>
<th>Naïve Bayes</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no SN</td>
<td>SN</td>
<td>no SN</td>
</tr>
<tr>
<td>AgL</td>
<td>CS</td>
<td>73.02%</td>
<td>81.21%</td>
</tr>
<tr>
<td></td>
<td>PL</td>
<td>72.75%</td>
<td>80.33%</td>
</tr>
<tr>
<td>AgG</td>
<td>CS</td>
<td>73.02%</td>
<td>74.00%</td>
</tr>
<tr>
<td></td>
<td>PL</td>
<td>72.75%</td>
<td>85.48%</td>
</tr>
</tbody>
</table>

In Case Study II, we studied the effect of increasing the number of teachers on the learning accuracy for both empty and non-empty learners and by using the same learning techniques used in Case Study I. The results showed that from a certain number of teachers, the learning accuracy decreases continuously with increasing the number of teachers and never recover. In order to overcome this deficiency, we used social networks in the communicating between the learner and the teachers. We tested our system again and the results were promising in all scenarios. The learning accuracies remained within the same range of values for all numbers of teachers available in our case study. This shows the positive effect of using social networks in learning accuracies when learning from a large number of teachers.
Chapter 7

Conclusions and Recommendations for Future Work

This chapter draws conclusions on the research presented in this dissertation, focusing on our concept learning module as part of a semantic search system. Furthermore, we provide possible suggestions for future works.

Section 7.1 briefly summarizes our proposed system and gives a brief review on research done. The main contributions of this research are introduced in Section 7.2. Finally, in Section 7.3, we provide some suggestions for possible extensions of this work.

7.1 Research Summary

As previously described, we proposed a framework for a semantic search system supported by a concept learning module based on a multi-agent system using social networks for the communicating between agents. In this system, we have two major modules: the semantic search module and the concept learning module. In our research, we discuss the integration between the two modules to get the most benefits of both. The integration between those modules is best described by a spiral-like workflow. Our proposed system consists of several multi-agent systems (MAS). Each MAS controls a repository of knowledge bases. All MASs cooperate with each other to perform a semantic search. In order to cooperate with each other, agents from different MASs need to communicate and understand each other.

We additionally assume that, ontologies used by each MAS do not necessarily need to be the same. In order for agents to understand each other, they need to make sure that they are using the same concepts. We enable agents to learn new concepts from several
other agents. This is done in the concept learning module of our system.

When an agent (learner) discovers that it needs to learn a new concept, it sends a learning request to other agents (teachers) to help it learn this concept. We assume that, the learner has an ontology and knows a set of features that describe concepts in its ontology. Each teacher also has its own ontology, and also has a specific set of features for all concepts known to that teacher. For any concept known to a teacher, it has some positive examples that describe this concept and can be used to teach this concept to the learner. The learner and the teachers are not committed to a common ontology. Agents in our system can perform some actions depending on their roles (learner or teacher). Part of the actions available to the learner is to send a learning request to the teachers regarding a concept. The learning request sent by the learner should contain some discriminating information about the concept in question. In our research, the learner uses two types of information describing the concept. The first type is keywords that describe this concept. The second type is conceptual information about the requested concept. In our example, the conceptual information used was the feature vector of a primary definition of the concept itself (if it already exists in learner’s ontology) or the feature vector of the concept’s parent (if known).

Part of the actions set of the teachers is to search the ontology for the best matching concept. The most important part of the process of concept learning is to make sure that each teacher selects the right concept to teach the learner. The selection of the best matching concepts depends on the type of information about the concept in question sent by the learner in the learning request. In case of keywords sent by the learner, each teacher searches its ontology using these keywords and calculates the value of \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) (i.e., the ratio between documents/examples returned by the search and total number of documents/examples) for each concept in the teacher’s ontology. The selected best matching concept is the one with the highest \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) value. The second case is
when the learner sends conceptual information (e.g. feature vector of the concept itself or its parent). In this case, each teacher calculates the similarity of the feature vector sent and the feature vectors of all concepts in its ontology. The selected best matching concept is the one that has the closest feature vector to the sent feature vector. In the case of using both types of information, each teacher calculates the average value of the two similarity values obtained \( \text{sim}(q_{\text{spec}}, C_{\text{best}}) \) and similarity of feature vectors) and selects the concept with the highest average value.

After deciding on the best matching concept, each teacher selects some positive examples describing this concept to send to the learner. In addition, each teacher selects some other examples as negative examples for this concept. After selecting positive and negative examples, each teacher sends its own set of examples to the learner to learn this new concept. The learner collects all positive and negative examples sent by all teachers and uses them for learning the required concept. In our example, the learner uses three different machine learning techniques: K-NN; Naïve Bayes and SVM for learning new concepts. The learner then calculates the learning accuracies using confusion matrix.

In order to improve the learning accuracy obtained with our system, we use a social network in defining relationships between the learner and the teachers. Social networks have a very attractive property that have a great effect on our system. This property is the ability of actors participating in the social network to communicate with each other with different tie strengths. This property enables the learner to classify the teachers, and depends on some teachers more than on others, based on strengths of ties between the learner and each teacher.

We faced a problem in calculating strengths of ties between the learner and the teachers in our system. The problem was that all suggested methodologies in calculating tie strengths depended on that, the actors of social networks are always human, which not the case is in our system. We developed a methodology to calculate tie strengths
in any social network regardless of the type of actors participating in it using Hidden
Markov Model.

The strengths of ties between the learner and the teachers reflect the degree of trust
between them. To express this degree of trust in our system, we depend on the number of
positive and negative examples sent by each teacher. The number of positive and negative
examples selected by each teacher is proportional to the value of the tie strength between
that teacher and the learner. Introducing social networks in our system has a great effect
on improving the learning accuracy either if the learner controls an empty repository
at the beginning of the learning process or if it controls a non-empty repository (i.e.
repository with initial ontology).

We expected to face problems when using social networks with a learner that controls
a non-empty repository. The learner, that controls a non-empty repository, would initially
have strong ties with some teachers and weak ties with others. This initial situation might
lead the learner to favor one teacher or a small group of teachers. Fortunately, that was
not the case in our example. Our case study proves that the strengths of ties were
adjusted according to the representation of the selected concept from each teacher. If the
concept selected is well represented in the teacher’s ontology, the strength of tie between
the learner and this teacher increases after the learning process. On the other hand, if
the selected concept from one teacher does not well represent the required concept to
be learnt, the strength of tie between that teacher and the learner decreases even if the
initial tie strength between them was high.

In this dissertation we tested the effect of increasing the number of teachers in the
learning accuracies. We showed that, by increasing the number of teachers, the diversity
of definitions of the best matching concepts selected from each teacher affects negatively
the learning accuracy. At the beginning, when the number of teachers was relatively
low, this effect was unnoticeable with any of the machine learning techniques used in our
system. The values of learning accuracies go up and down randomly with increasing the number of teachers, but the accuracies are still acceptable. However, starting at a certain number of teachers (thirteen teachers in our case), the learning accuracies of all three learning techniques used start decreasing continuously. This negative effect appeared independent if the learner controls an empty repository or a non-empty repository. We aimed to solve this problem by using social networks. We defined a social network in communication between the learner and all teachers and started increasing the number of teacher again to test the effect of the number of teachers on the learning accuracies in the case of using a social network. We noticed the following:

1. First, the learning accuracies always improved compared to the corresponding learning accuracies obtained without using social networks.

2. Second, the negative effect of increasing the number of teachers did not appear, in our test, with all number of teachers used (seventeen teachers). The learning accuracies remain around the same range of values all the time using the three learning techniques with a learner that controls either an empty or non-empty repository.

We concluded that using social networks in defining relationships between agents when learning new concepts from several teachers has a positive effect on improving the learning accuracy independent of the number of teachers. It also has a positive effect as the number of teachers increases.

7.2 Contribution

In this section, we consider the areas of our contributions in the light of what was presented in this dissertation.
7.2.1 Enhancing system workflow

The workflow suggested for our system is represented by a spiral like workflow. It was suggested by our team in (Far et al., 2009) where both the concept learning module and the semantic search module are considered independent processes. In this research, we enhanced the workflow suggested to better represent the integration between concept learning and semantic search. In the newly suggested workflow, both concept learning and semantic search modules are treated with the same degree of importance. The processes of these two modules are co-related with each other. Concept learning is triggered by semantic search and the semantic search is paused until the concept learning has finished before making use of its results. That helps improve the overall quality of the search results obtained by the system (El-Sherif et al., 2010a).

7.2.2 Using social networks in our system

The second contribution of our research is the introduction of a social network in communicating between agents. Using social networks was very helpful in the following areas:

Semantic search
By defining a social network between agents in the semantic search module, it helps improve search results by providing a better way of ranking the results obtained from remote agents. This ranking depends on values of ties strengths between the local agent and the remote agents (El-Sherif et al., 2010a).

Concept learning
Defining a social network between the learner and teachers in our concept learning module had a great effect on the learning accuracies (El-Sherif et al., 2010b). It improved learning accuracies obtained when learning new concepts from several teachers using different machine learning algorithms. We tested the effects of using a social network in different
cases:

- Learning a new concept from several teachers, where the concept exists in the teachers’ ontologies.
- Updating a concept that already exists in the learner’s ontology by relearning it from several teachers.
- Learning a new concept from several teachers, while this concept is not a standalone concept in teachers’ ontologies.
- Starting with a learner that controls an empty repository (no concepts are initially defined in its ontology).
- Starting with a learner that controls a non-empty repository (its ontology contains already defined concepts).

In all cases, using social networks improves the learning accuracies obtained and gives very promising results (El-Sherif et al., 2012b) (El-Sherif et al., 2012a).

7.2.3 Increasing number of teachers

In this research, we tested the effect of increasing the number of teachers on the learning accuracies. We tested this effect for two cases. In the first case, we did not use any social networks in our module. In this case, the results showed a negative effect of increasing the number of teacher in the learning accuracies. In the second case, we use a social network for the communicating between the learner and the teachers. In this case, the results were promising. They showed how using social networks positively affected all learning accuracies obtained, and how it reduced the negative effect of increasing the number of teachers on the learning accuracy.
7.2.4 Calculating tie strengths

Tie strength is a dynamic property of social networks. It changes during interactions between actors based on several factors (e.g. closeness between actors participating in the network, interactions between them, etc.). All previous research done in this area dealt only with human actors. In this research, we introduce a new methodology of calculating tie strengths dynamically regardless of the type of actors participating in the network. In this methodology, we used Hidden Markov Model in calculating tie strength between actors in any social network (El-Sherif et al., 2011).

7.3 Future Work

In this section, we focus on future work that can be done to enhance the framework presented in this dissertation. There are several opportunities for further research in this challenging area:

- One of the most interesting parts of ontologies is the relations between concepts. In most research done, fixed types of relations between concepts are considered. In reality, any number of relations can be considered. One possibility are a user-defined relations that open a door to thousands of relations being defined. One suggested area of future work is to enable a learner to also learn relations from several teachers in addition to concepts.

- To see the strength of our methodology and the great effect of using social networks, we suggest a research direction in which our general idea is applied to a real product application. This potential application would be using our approach in big data systems. Big data can grow rapidly through a network of nodes. Each node may have its own data stored in a knowledge base. Each knowledge base can be represented by its own on-
tology and needs a number of functions to manipulate, store and retrieve data and also to be able to communicate with other knowledge bases in the same or a different network. Knowledge bases can be controlled by different multi-agent systems (MAS). In order to integrate big data, MASs controlling knowledge bases, need to correctly understand each other and make sure that concepts used in their communication have the same meaning in their ontologies. Learning ontologies can be used in enabling big data integration.

• Another challenging research area could be the automatic concept integration into the local ontology, where the new concept learnt could be self-organized in the ontology hierarchy by utilizing concept features and examples in addition to appropriate relationships around the concept with other concepts that already exist in the ontology.

• In this research, we dealt only with text documents that represent examples of our concepts in the concept learning module. One possible future research is applying our proposed system in learning new concepts from several teachers using different types of examples at the same time including: pictures; multimedia, log files, etc.

• Applying our concept learning process with different specifications (e.g. using different learning techniques, depending on reliable teachers only, referring friends to new agents etc.).

• Enable the semantic search for other data types (e.g. images, video). Other factors can be also considered during ranking the remote results of the semantic search such as: structure of links in the page, frequency of the page to be selected by the user, etc.
• Currently, we assume that each repository is associated with MAS and its ontology is created. Real deployment requires these components to be there. For future work, distributed revision of MAS and creating ontologies may be required to enable new repository to participate the system.
References


