**Data Mining a Knowledge Corpus to enrich ROS Graphs**

**John Meehan**

**13702625**

Final Year Project – **2017**

**MSc in Computer Science (Applied)**



Department of Computer Science

Maynooth University

Maynooth, Co. Kildare

Ireland

A thesis submitted in partial fulfilment of the requirements for the

**MSc in Computer Science (Applied)**

Supervisor: **Dr. Diarmuid O’ Donoghue**

**Table of Contents**

[Abstract: 3](#_Toc501702189)

[Section 1: Introduction 3](#_Toc501702190)

[Section 2: Background 4](#_Toc501702191)

[2.1 The Usefulness of Analogy 4](#_Toc501702192)

[2.2 The Dr Inventor System 5](#_Toc501702193)

[2.3 Problems associated with the addition of background information 6](#_Toc501702194)

[2.4 Dr Inventor Results 7](#_Toc501702195)

[2.5 Other systems that could benefit from incorporating background information 7](#_Toc501702196)

[Section 3: Theory 8](#_Toc501702197)

[3.1 Choosing a Knowledge Resource 8](#_Toc501702198)

[3.2 The ConceptNet Knowledge Resource 10](#_Toc501702199)

[3.3 Problem Data 10](#_Toc501702200)

[3.4 Pre-Processing 11](#_Toc501702201)

[Section 4: The Solution 11](#_Toc501702202)

[4.1 A General Solution 12](#_Toc501702203)

[4.2 Additional Methods Employed in the System 13](#_Toc501702204)

[Section 5: Results 14](#_Toc501702205)

[5.1 Run-Time 14](#_Toc501702206)

[5.2 Initial Testing 16](#_Toc501702207)

[5.3 Results from the sets of problem data 17](#_Toc501702208)

[5.4 Comparisons Between Problem Data sets 20](#_Toc501702209)

[Section 6: Conclusion 21](#_Toc501702210)

[6.1 General Conclusions 21](#_Toc501702211)

[6.2 Future Works 21](#_Toc501702212)

[References: 22](#_Toc501702213)

# 

# Abstract:

Many Natural Language Processing systems represent an input texts semantic and syntactic information using graphs, where nodes are used to represent nouns while edges represented the links between nouns – these types of graphs are known as *knowledge graphs*. While these systems focus on representing the input texts semantic and/or syntactic information, background information is generally forgotten about. This background information represents implicit information that is generally understood by a human reader, but which necessitates explicit addition by a computational means. This thesis aims to analyse the addition of background information into these types of NLP systems by firstly generating the background data between graph nodes and then having participants of a study rate the quality of background link produced. With these tests, we will be able to conclude whether adding this information would be a worthwhile endeavour to various NLP systems.

# Section 1: Introduction

Analogies are ubiquitous in computer science. There are constantly used by students and lecturers alike to make the unfamiliar familiar. However, as Bunge describes, analogies may “give birth to as many monsters as healthy babies” [2]. With this in mind, we will look the Dr Inventor, a system that provides inspiration for scientific creativity through the use of computationally created analogies between different research publications from the area of computer graphics [3]. Dr Inventor uses a set of highly-reputed, highly-cited web-based research papers which are then used to inform scientists of relevant research concepts and approaches, assessing the novelty of research ideas and highlighting unexpected features for new scientific discovery [4]. In short, the system does this my taking in text-based documents and returning a graphical representation of the semantic knowledge and relations for each input text in the form (relation (subject, object)) - this type of graph is known as a ROS graph (See *Figure Three*). Using these ROS graphs, the system can then compare any two texts and find hidden structural similarities between the two via a process of analogical matching. However, the system can only draw links between explicitly linked nouns in the input text. For example, if a graph showed that “Dog: HasA: Tail” and “Labrador: HasA: Coat”, the system will not match “Dog” and “Labrador”, as this is considered background information. This information represents implicit information that is typically understood by a human reader, but it necessitates explicit addition by a computational means. With this in mind, the goal of this thesis is to propose data mining techniques that aim to enrich the different input ROS graphs by drawing associations between non-explicitly linked nouns and verbs in the input graph. If we achieve this, in the above example, we will be able to draw a link between “Dog” and “Labrador” with the hope that we will refute Bunge’s statement and in turn produce more healthy babies than monsters for use by Dr Inventors analogical matching phase.

In the following thesis, *Section 2* presents the background of the paper where we will firstly discus the power of analogy as a learning mechanism and why it is so important in fields of science for both learning and creativity. Following that we will look at the Dr Inventor system, a system which uses analogy as a core driving mechanism. We will then look at some of the problems associated with the addition of background information and why this addition can sometimes lead to unwanted results. Finally in *Section 2* we will look at IBM’s Watson, a similar NLP system that could utilise the process of data mining to improve the output quality of the system. *Section 3* presents the theory of the paper where we will discuss the various knowledge resources proposed to enrich the knowledge graphs. The eventual choice was ConceptNet [1] and so we will look at the resource in more detail. We will then look at some of the problem ROS graphs obtained from Dr Inventor which we will be finding background relations for and discuss the relevance of this in relation to the remained of the thesis. Finally in this section, we will look at some pre-processing that had to be performed on the ConceptNet corpus before it was ready for use in our application. In *Section 4* we will take a look at my solution to the problem of adding this additional background information. Two versions of the application were developed and so we will take a brief look at the underlying structure of each. Finally in this chapter we will discuss some additional methods developed to increase the quality of output for use by the Dr Inventor system. *Section 5* presents the results of the system. Firstly we will discuss the initial testing of the system, where 2 participants rated the various outputs using a Likert Scale [5]. We will then move on to discuss a more though test of the system, where 5 participants were asked to rate the various outputs from the system. Finally in this section we will compare and contrast the expected results of the system vs the actual results obtained. In Section 6, we will discuss the conclusions that can be drawn from the results of the system and look at future works that could be performed to enhance the process of adding additional background information to ROS graphs.

# Section 2: Background

*In this section we will look at the different background aspects related to this thesis. Firstly, we will discuss the power of analogy, and why it is so useful to both students and lecturers alike. We will then focus on the Dr Inventor system, a system that uses analogy as a core driving mechanism. As stated in Section One, the background information of different ROS nodes are currently not considered by the system however, we will weigh the pros and cons of adding this additional information and discuss why it’s integration should be a worthwhile addition to the system. We will then take a brief look at the results produced by the Dr Inventor system, so that we have some context as to the methods employed in my results section. Finally, we will look at a different system that could integrate a similar data mining process - IBM’s Watson, and discuss why this system should also integrate the mining of background data from a knowledge corpus.*

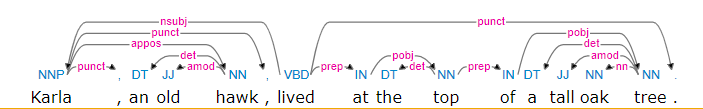
## 2.1 The Usefulness of Analogy

As stated in the Introduction of this paper, analogy is ubiquitous in computer science. Analogy is defined as the drawing of similarities between different subjects [6] where this similarity is used to form a relationship between an unknown subject and a known one. Most who have taken a class on Data Structures for example, will be familiar with the analogy where a Queue data structure is compared to a group of people standing in a line. Similarly, the structure of a family tree will be compared to a Tree data structure. These comparisons can be seen as helpful in the learning process of computer science and similar analogies are created throughout every stage of the learning process, from the learning of addition by children to the learning of in depth concepts at a college level [7]. In the 1980s Mary Gick and Keith Holyoak [8] conducted a study to test the power of analogy whereby they gave participants a difficult insight problem. Half of the participants were given an analogy to help with the solution of the problem while half were not. Out of the those given no analogy, only ten percent solved the problem while out of those given an analogy, thirty percent solved the problem. Those that were given an analogy to solve the problem but didn’t solve the question were then reminded of the analogy and over ninety percent of them solved the problem. Thus, the solution rate tripled from people who weren’t given an analogy and this figure was further tripled when people were reminded of the analogy. Although the question was considered “difficult” by the researchers, analogy made the new information easier to imagine [9], and so more concrete for the participants. With the power of analogy in mind, it is clear to see why systems such as Dr Inventor use it as the driving mechanism behind them.

## 2.2 The Dr Inventor System

Systems such as Dr Inventor observed the power of analogy and instead of using it as part of the learning process, employed it for use in the creative process. The computational modelling of analogy has relied mainly on human constructed data [10], however the Dr Inventor system aims to produce this computational model based on raw data, sourced directly from different computer graphics research publications. Dr Inventor focuses on identifying the most creative comparisons that can be made for a given target text so that the user only has to explore the most creative hypotheses [11]. The system aims to simulate the creative thinking process of humans, to identify comparisons that might enhance the researcher’s creativity. It achieves this through a process of deep natural language processing retrieved directly from research publications, from which it then derives an attributed relation graph called a Research Object Skeleton (ROS) graph. This is outputted in the form (relation (subject, object), which was the format originally proposed by Gentner [12] for ROS graphs. In these graphs, a concept node describes a subject or object (represented with a blue node in *Figure Three)* whereas a relation node represents the relation between concept nodes (represented with a green node in *Figure Three).*

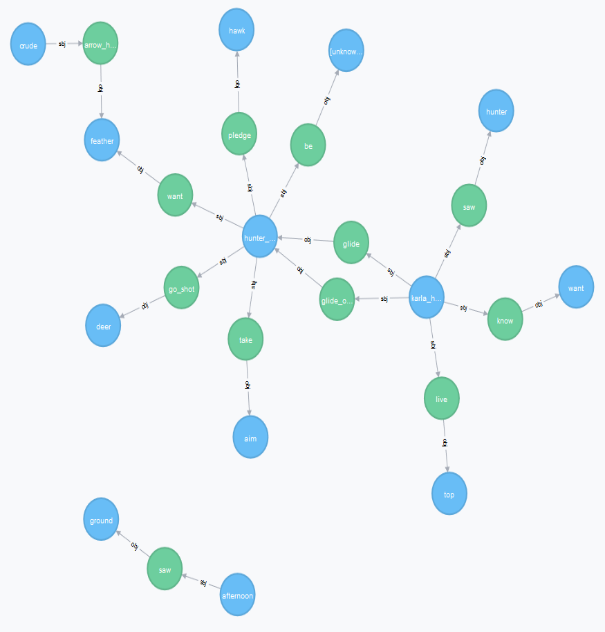
The Dr Inventor system performs many different operations to produce output to the end user. Firstly, the input research paper must be converted to a PDF document to work with the system. This PDF file is then parsed using the well-known GATE dependency parser [13], which performs a semantic extraction to find the important language and logic in the input paper.

  
***Figure One:*** Shows the process performed by the GATE dependency parser [13] where a semantic extraction is performed on the input text.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 3710 | 5 | karla\_hawk\_she\_her | 16 | live | 4 | top |
| 3710 | 10 | hunter\_him\_he | 20 | take | 5 | aim |
| 3710 | 0 | hunter\_him\_he | 13 | want | 1 | feather |

***Figure Two:*** This is the output produced from Figure One. The GATE dependency parser will output data based on the data obtained from *Figure One.*

This information is then used to generate the ROS graph that represents each publication through the remainder of the system.

  
***Figure Three:*** ROS graph generated using the data obtained in *Figure 2*. Note: Dr Inventor produced the content of the graphs. This information is then graphically represented using Neo4j [14]. (See *Appendix A* for a larger version of this graph).

The mapping between two publications is then generated from the ROS graphs using a version of the VF2 subgraph matching algorithm [15]. This algorithm ensures an appropriate balance of 1) the mapping process of matching semantically similar concepts and 2) the different topologies of the input ROS graphs. It restricts the space of possible mappings, allowing concepts (nodes) to only be mapped with other concepts while allowing relation nodes (verbs) to be mapped to other relations (see *Figure Three*). Dr Inventor then blends the new information into the pre-existing target text and presents it to the user by placing the inferences in the context of the target paper. The analogy in the system is then a structure-based comparison between the two collections of information, centred on a one to one mapping between the two [11].

## 2.3 Problems associated with the addition of background information

As stated in the introduction of this paper, Dr Inventor does not check for background links between nodes. In *Figure Three*, the ROS graph contains unlinked concept nodes that could be linked if background information was considered – for example, in the graph we see a concept node for *hawk* and a concept node for *feather*. They are not linked in the output graph however we know that hawks have feathers. Generally speaking, most natural language processing systems focus on representing the input texts semantic and/or syntactic information while background information is forgotten about. This is due to the low coverage of available knowledge resources and the difficulty of matching text and its ontological elements [16]. There are also problems when it comes to adding contradictory information due to the integration of background knowledge. For example, if a ROS graph contained information such as (isA (Labrador, Cat)), which makes sense in the context of the input text and another concept node for *Dog*, adding background information would produce a link (isA (Labrador, Dog)), which may not make sense in the context of the story. However, as Abgaz et al [11] states in systems such as Dr Inventor, “the value of analogy depends on the strength of the mapping between two analogues” and so it must be presumed that drawing a link between different concepts like “View Transform” and “Rendering Pipeline” (in the context of computer graphics) would be invaluable to increasing the reliability of a graph and hence increase the strength of the analogy it produces. This is the main argument so that Dr Inventor and similar systems should consider the addition of background information.

## 2.4 Dr Inventor Results

When compiling results for the Dr Inventor system, the team recruited 15 participants from the area of computer graphics, from professors with SIGGRAPH publications to PHD students. The team selected 10 random target publications and asked each participant to evaluate each publication. They were asked to rate the output of the system based on 1) the output being novel or containing an unexpected comparison 2) potentially useful and recognizes gaps in the research or 3) this comparison challenges the norms of the discipline of computer graphics. The agreement between participants was then calculated using a Krippendorff alpha [17], which calculates the inter-rater agreement and return values between 0.0 and 1.0, where 1.0 indicates maximum agreement and 0.0 indicates no agreement. The rates of agreement returned quite low however as the number of categories to be rated increases, the alpha level tends to decrease. Also, rating the level of creativity is quite a subjective process as some users may be experts in certain areas of computer graphics while some may know little to nothing about certain areas [11]. As the results for Dr Inventor were presented using a Krippendorff alpha and a form of Likert scale, we will rate our addition to the system in a similar fashion (see *Section 5*). Finally, although Dr Inventor was rated on a basis of the quality of the analogue produced, our proposed addition to the system will be rated on the quality of the additional background links produced.

## 2.5 Other systems that could benefit from incorporating background information

Another system that could utilise the process of data mining background information is IBM’s Watson. Although today, more data is available than ever before, only a fraction of it is being integrated, understood and analysed [18]. This is due to the fact that much of the data is unstructured, such as text books, how to manuals, Wikipedia articles, tweets etc. It is estimated that eighty percent of the data in the world today is unstructured [19] and so IBM developed Watson as a way to combat this. Watson uses techniques from deep natural language processing (similar to Dr Inventor) to help understand the complexities of unstructured data. For example, in English how is it that we can fill in a form by filling it out? As humans, we understand the complexities of language and can see through the gaps and inconsistencies produced by it, however current computer systems find it hard to read through the lines of language [19]. However, the Watson system tries to achieve accuracy when reading unstructured text. It does this by attempting to assess as much context as possible. Firstly, Watson will take in and parse the input question to extract the major features, for example it extracts the objects, subjects etc. Then it will generate a set of hypotheses by looking across passages that have some potential for containing a valuable response. Finally, it performs a comparison of the language of the question with the language of each potential response, using a variety of reasoning algorithms. However, despite the success of Watson (for example on the US TV show JEPORADY! [20]) it is quite a simplistic system. In essence, it simply parses the input question on a basis of each words type and then finds articles and texts on the web which have a similar structure and contain similar words. In the various papers about the internal workings of Watson, none reference any type of background data mining process to search for similar words to that found in the input question. For example, if asked a question about Labradors, the system will only search for text’s containing the word Labrador, but perhaps some background data mining could produce a link to the word Dog. This may in turn create unexpected links to help find the desired answer more consistently.

# Section 3: Theory

*In this section we will look at the various aspects of research and testing that had to be conducted to reach the end goal of enriching the various ROS graphs with background information. We will mainly focus on the different knowledge resources available to enrich the various ROS graphs. This is one of the most important aspects of this thesis as it will ultimately dictate the quality of information the rest of the project has at its disposal. Considering the Dr Inventor system creates analogies between research publications from the area of computer graphics, if a knowledge resource is too small, it will likely not contain any terms related to this field and so we may find it difficult to draw background links between any nodes. On the other hand, if the knowledge resource chosen is too large, it could in turn oversaturate our graph with background links, overwhelming the original data produced by the Dr Inventor System.*

*In this section, firstly we will look at a variety of knowledge resources as potential corpuses to be used to enrich our ROS graphs. We will then look at ConceptNet, the knowledge resource that was ultimately chosen as the most suited for this project. We will then move on to look at the problem data provided by the Dr Inventor system, talk about the number of relations per data set and discuss the relevance of this. Finally, we will discuss some pre-processing that was conducted so that the ConceptNet corpus would be ready for use in the remainder of the project.*

## 3.1 Choosing a Knowledge Resource

The amount of information obtainable from a knowledge resource for even a single named entity can be very large. For example, ConceptNet alone contains 5024 *verb: noun: noun* triples containing the word *foot*. Not only is most of this information irrelevant to the task at hand, only some of the triples can be useful to resolve the problem of adding additional background information (e.g. linking the word *foot* to the word *leg when they are not explicitly linked in a ROS graph*). With this in mind, one of the first tasks to overcome for this project was picking a reliable knowledge resource. This type of problem can be seen throughout the field of natural language processing as a whole – the difficulty in finding a balance between having too much information as opposed to having too little. For example, if an input ROS graph has 10 nodes, we want to pick a resource that will find meaningful links between some nodes, without linking every node in the graph and oversaturating it. When choosing potential knowledge resources for use in this project, five test cases were devised with expected results as a means of choosing the best resources – this is a form of ad hoc, interactive manual testing. For example, an input checked on every knowledge resource was *foot* where it was expected that *foot* would have some link to *leg.* If the resource did not link these terms, it would be considered less viable for use in the application.

The first knowledge resource tested was BabbleNet, a resource which contains an “encyclopaedic coverage of different terms” [21]. BabbleNet stores its knowledge corpus in a csv file which is over 16GB in size compressed. Before even running the test cases, having a file of this size was a problem. Searching through a file of this size for every node in our input graph would likely take too long, in addition to the fact that it would likely fall victim to the problem described above of oversaturating our ROS graph with information. As expected (due to the size of the file), BabbleNet passed all our input test cases however, it also contained a lot of information that could be deemed unnecessary and overly specific for use in the project. For example, when running the file on the input of *foot*, it was found that the word had over 1000 links, relating with several bones, muscles, arteries etc, associated with the *foot*. It also had over links with several units of measurement and the conversion rates between them, which would in turn add too much information to the input graph. This resource was ultimately decided against, due to the volume of data it contained.

The next resource tested was ProjectNell – a system which “reads the web” [22] for different facts, by parsing sites such as Wikipedia as a source of different candidate beliefs. The corpus from ProjectNell was only 4GB in size uncompressed, however on closer inspection, the format of the file was very ridged and many of the links in the file were too specific for use in our system. For Example, a randomly chosen candidate belief present in the corpus is

*concept:economicsector:hotels concept:atdate concept:dateliteral:2006…*

This candidate is not only ridged in the sense that one must firstly find that we are talking about hotels in the context of the economic sector, but also in the fact that it contains a date. It also holds no real meaning in the sense that we don’t know what happened in the economic sector to hotels in 2006 or how they are even linked. However, ProjectNell did pass our ad hoc testing inputs, finding connections between *leg* and *foot* for example. The ProjectNell corpus was ultimately chosen against due to the specificity of the content within the file.

WordNet [23], a different knowledge resource which consists of a lexical database of English words also found a relation between *foot* and *leg* however the output was in the form

*(n) foot, human foot (the part of the leg of a human begin below the ankle joint*

As you can see, this is just a sentence, which is not desirable for use in the project. There is no specific way to differentiate the nouns in text from the verbs linking them, meaning we would have to implement a parser into the system to differentiate between these. This would add additional, unnecessary work, considering the other knowledge resources have clear distinctions made between the nouns and the verbs that link them (for example ProjectNell has the String *“:concept”* preceding each). The WordNet system also failed on the input test case of *algorithm* and *software*, and considering that many ROS graphs we will be working with consist of publications from the area of computer graphics, this would be an important link to make. WordNet was decided against due to the factors mentioned above.

The final corpus tested as a potential knowledge resource was ConceptNet [1] – a system which contains a “semantic network, designed to help computers understand the meanings of words”. The file provided by ConceptNet which contains their knowledge corpus is under 8GB in size uncompressed and contains different facts (or as they call them, “*assertions*”) in the form

/a/[/r/RelatedTo/,/c/en/foot/,/c/en/leg/] /r/RelatedTo /c/en/foot /c/en/leg {"dataset": "/d/verbosity", "license": "cc:by/4.0", "sources": [{"contributor": "/s/resource/verbosity"}], "surfaceEnd": "leg", "surfaceStart": "foot", "surfaceText": "[[foot]] is related to [[leg]]", "weight": 1.0}

This system passed all 5 ad hoc test cases and due to the fact that it is both nicely structured and generally not overly specific in content, it was chosen as the knowledge resource for use in this project. What follows in *Section 3.2* is a closer look at this knowledge resource.

## 3.2 The ConceptNet Knowledge Resource

As stated above, ConceptNet contains a “semantic network, designed to help computers understand the meanings of words”. It obtains much of its data from DBPedia (which extracts knowledge from the info-boxes of Wikipedia articles) and Wiktionary (a multilingual dictionary). Interestingly, ConceptNet even obtains some of its data from WordNet which was a candidate knowledge resource mentioned above. The content present in ConceptNet’s corpus is both nicely formatted and not overly specific in content. As you can see in the above example, verbs in the file are preceded by “/r/” while nouns in the file are preceded by “c/../” where “..” can be any acronym of a specific language. This format is nice as we can simply extract all text in the file preceded by these indicators. The fact that knowledge contained within the file is simplistic is a bonus. For example, in the ProjectNell file, a link can be seen between 2006 and hotels in the context of the economic sector. This information is too specific considering our input ROS graphs don’t contain the context of the different nodes present within. With this in mind, ConceptNet was chosen as the best knowledge resource for use in this project, due to the simplicity of its formatting. The fact that the file this corpus is contained in is quite small (in comparison to a corpus like BabbleNet) is also an advantage.

## 3.3 Problem Data

The problem data studied for this project are ROS graphs, derived from the Dr Inventor system. As stated in *Section 2.2*, ROS graphs are Research Object Skeleton graphs, where data is presented in the form relation(subject, object). We are working with various sets of problem data, which will be an important distinction to make when looking at *Section 5 Results.* The first set of data derived from the system were ROS graphs based on the abstracts of different SIGGRAPH publications. In this corpus, there are currently 1180 distinct graphs. The excel file this corpus is stored in has 16369 rows and so on average there are 13.87 rows (or edges) per graph (Note: by row I am referring to a relation(subject, object) pair). The next data set that will be tested are ROS graphs based on the full papers of SIGGRAPH publications. Again, there are 1180 distinct graph in this corpus however there are 174152 rows in this file meaning there are on average 147.59 rows per full paper graph. This will be relevant when looking at our results as the greater a graph is in size, the more links we would expect it to produce. Different sets of ROS graphs based on computer graphics Patents were then derived by the system which have 1243 unique graphs in total with the corpus containing them consisting of 30564 lines. On average there are 24.59 lines per Patent graph. Finally, Psychology texts were derived by the system where there are 27 unique ROS graphs and 533 rows in this corpus. This means there are on average 19.74 rows per Psychology ROS graph. This is relevant in the context of this project as it will be interesting to see which problem data receives more new background information and which data set will receive less. It is to be expected that SIGGRAPH abstracts will receive the least amount of addition background links due to the average row size per graph. In a similar fashion the full paper SIGGRAPH data should receive the greatest number of new links, considering there are on average 147.59 rows per full paper. It will also be interesting to observe if specific fields receive more background links than others. For example, are more background links produced when looking at Psychology or Computer Graphics ROS graphs.

## 3.4 Pre-Processing

Before the ConceptNet corpus was ready for use in our project, some pre-processing had to be performed on the file. As the goal of this project is to enrich knowledge graphs by adding additional background information between nodes, the actual background information is all that is required from the ConceptNet file. As you can see from the sample data in *Section 3.1,* information in the ConceptNet corpus comes in the form of *verb: noun: noun* triples, followed by different elements related to said triple. For example, we don’t care about the sources or weights of the different triples in the file, we care about the content of the triples themselves. Also, as stated in *Section 3.2*, ConceptNet takes much of its information from Wiktionary, which is a multi-lingual dictionary. As Dr Inventor is built on the GATE parser which operates in English, for the purposes of this project, it was satisfactory to modify the ConceptNet corpus so that it only contained English and it only contained the desired *verb: noun: noun* triples. The method used to remove the useless information was a Java Class whereby we took in the original corpus from ConceptNet which was contained in a 28,841,278 line csv file. We extracted the English *verb: noun: noun* Triples and outputted these into a new file, which would be used throughout the remainder of this project. This means that the data contained within the ConceptNet corpus would now only consist of information like

belgium,country,IsA  
belarusians,ethnic\_group,IsA  
belgian\_sheepdog,dog,IsA

This reduced the file down to just 2,809,254 lines which is a reduction in length of around 10%, helping not only improve the content within the file but also to improve the search times in the main part of the system. Also note that in the code used to make these changes, we extracted the English terms using the search string of “c/en/”. *En* can easily be changed to the acronym of many other languages, leaving the data mining process extendible if the Dr Inventor system chooses to branch into other languages, or if some other similar system which operates in a different language wanted to avail of the data mining process described in this thesis.

# Section 4: The Solution

*In this section, we will look at my solution to adding the additional background information to ROS graphs using the ConceptNet knowledge resource. We note that it is important to include some missing information that would generally be understood by a human reader but which is not explicitly stated in the input documents, or which was not identified by the existing ROS generation process. We also note again the importance of not overwhelming the graph with information, rendering it useless for subsequent tasks in the Dr Inventor system.*

*We will firstly take a look at Version One of the system to add this additional background information. This version of the system was very slow and so another version had to be created. Version Two of the system uses hashing in Java and so is several times faster than the first which simply brute force searched for matches between background nodes. We will also briefly look at some additional methods that was written to enhance the quality of the output for use by the Dr Inventor system.*

## 

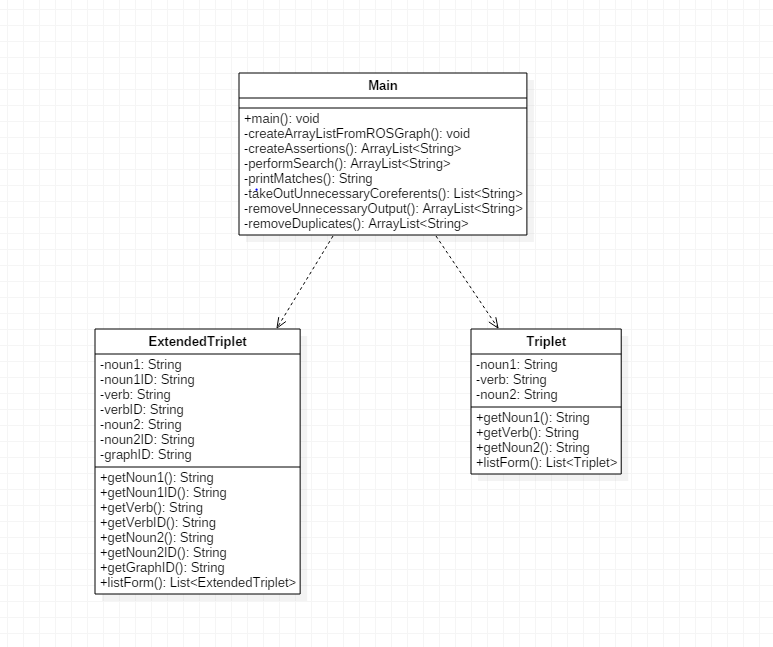
## 4.1 A General Solution

An input ROS graph, which contains the information from a specific input is presented in the form

*{"Prpnoun":false,"IntNodeId":27,"GVLabel":"c27","Word":"ribbon","Graphid":3702,"nodeOccurrence":"21,426;23,445;29,560;31,643;31,658;37,780;37,812"}**{"IntNodeId":74,"SectionType":1,"GVLabel":"r22","Word":"be","Graphid":3702,"RhetCategory":1,"nodeOccurrence":"23,450"}**{"Prpnoun":false,"IntNodeId":29,"GVLabel":"c29","Word":"way\_it","Graphid":3702,"nodeOccurrence":"23,470;25,473"}*

where the above is one line of the input file which represents one link formed by the Dr Inventor system (i.e. using the conventional ROS structure, this would be written as *be(ribbon, way\_it) – looking at the “Word” tag*) . The first thing that had to be done was to extract the relevant information from the above data. As the ROS graph data was stored in an Excel file, we needed to incorporate external libraries to read in the correct ROS graph. With this in mind, the Apache POI [24] library was added to the system to make it possible Excel files. For this project, we don’t care about attributes such as the *GVLabel* or *nodeOccurrence* however we are interested in the *IntNodeId* (this allows Dr Inventor to keep track of different nodes in a graph), the *Graphid* (many graphs can be stored in one file by the Dr Inventor system and as we want to treat every graph as a separate entity, this id is important) and most importantly the *Word* which is the actual word we will be finding background relations for. Before we start searching for background links we must firstly remove coreferents from the input. As you can see in above example, in the third group of information presented, *Word* has *way\_it* attached to it. This means that *way\_it* is the link we’ll be finding background information for (or more specifically finding information between this Word and other Words in the input file). In this case, the word *it* is considered a coreferent and as ConceptNet does not contain these types of coreferents in its corpus, we want to remove such words. The coreferents removed include it, they, the, be and you, meaning these will be extracted before the search process is conducted. With the specific information we want extracted and the coreferents removed, it was time to start searching for links between the input ROS graph and the ConceptNet corpus. Information in the input ROS graph now consisted of *“noun1: noun1ID: verb: verbID: noun2: noun2ID: Graphid”* (where noun1 is the subject in the above sample of ROS input and noun2 is the object in the above sample of ROS input) and as stated in *Section 3.4*, information from the ConceptNet corpus now comes in the form *“Cat, Mammal, IsA”*, after pre-processing has been performed on the corpus. With this taken into account, it was a matter of searching through each *Word* in the input ROS graph and if any two *Word*(s) were present consecutively in the ConceptNet corpus, we could draw a link between the two. For example, if we had a node in our input ROS graph for *Dog* and an input node for *Labrador*, and these terms were present consecutively in the ConceptNet corpus (i.e. IsA Dog Labrador), we could draw a link between the two. If we found a link, this links information (*“noun1: noun1ID: verb: verbID: noun2: noun2ID: Graphid”*) would be sent forward as output to the end system. A similar process was also run on the relational nodes (verbs) in the input ROS graph, to see if we could find relations between different verbs. The above process produced the expected output, however the run time of the system was extremely slow. I used a nested for loop of depth three to perform this search process which means the complexity of the search process was *O(n3)*. For some context, when tested using an input ROS graph of length 503 (503 lines similar to the above example), the programme took over 9 hours to complete running. As the Dr inventor system couldn’t add an additional 9 hours run time to the system, other methods for doing this search had to be employed*.*

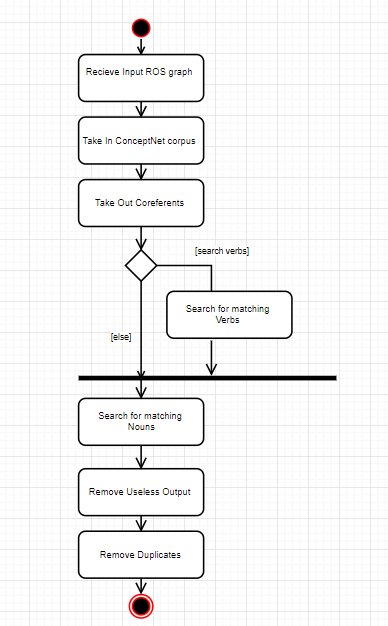
With this in mind, I began experimenting with hashing in Java. After some research, I found that it was possible to partition a list into the buckets of a HashMap based on a certain property. I decided to transform each *“verb: noun: noun”* triplet from the ConceptNet file into one “Triplet” object and all *“noun1: noun1ID: verb: verbID: noun2: noun2ID: Graphid”* from the input ROS graph into one “ExtendedTriplet” object. I then partitioned the *ExtendedTriplets* list into Map buckets, where each key was a specific noun1 (more specifically called a subject in relation to a ROS graph) and each value was an *ExtendedTriplet* which contained an occurrence of a noun1. A similar process was employed to partition the noun2’s (or more specifically called an object in relation to a ROS graph) into buckets, where this time the key was any noun2 and the corresponding value was every *ExtendedTriplet* which contained a noun2. I then searched these lists using another nested for loop of depth three however this time I used the *.getOrDefault* java method. A similar method was also employed to find links between verbs in the input ROS graph. With the HashMap partitioned correctly and the .getOrDefault employed we never search for non-matching entries and so the complexity of this system gets reduced from cubic (the time complexity of the previous version of the system) to linear plus the number of matches found. The run time of the system went from taking 9 hours for a graph of size 503 in length to just 15.893 seconds meaning the search speed increased drastically.

  
***Figure Four:*** Class Diagram of Version 2 of the system

## 4.2 Additional Methods Employed in the System

After the matching relations were returned, some final methods had to be employed on the output data before it was ready to be sent to the Dr Inventor system. Firstly, some outputs consisted of relations between the same word. For example, we might see relations such as “foot IsA foot”. This information is not useful to the end user of the Dr Inventor system and it would likely overpopulate the graph to a point where every node could hypothetically link back to itself. This is unwanted and so a method was developed to remove this form of output before it is sent to the Dr Inventor system. This code was run after the searching processes is finished.

The final method added was to remove duplicates from the output. We can’t just skip over duplicates in our code (by adding break statements to our loops for example) in case we skip over some useful relation and so duplicates must be removed after the search process has been completed. The method employed to do this simply parses through the output relations and if it finds two identical outputs it will remove the latter version. The Dr Inventor doesn’t need any more than one link to make a relation and so it is useless to return this extra output to the system.



***Figure 5:*** Activity diagram of the final Version of the system.

# Section 5: Results

*In this section, we will look at the results produced by the system. As stated in Section 4, originally the system ran in O(n3) time however with the use of hashing the speed of the system increased drastically going from a cubic complexity to linear. However, despite how fast the program runs, this does not matter if the content the system produces is not of value. With this in mind, we will look at the background links produced by the system with the help of 5 participants. Participants have been chosen from people with no prior knowledge of computer science to a lecture in the discipline to see how this affects the rating of the system. Participants will be given a 5-point Likert Scale [5] which can be seen in Figure 8, where they will be asked to rate the different outputs from the different problem sets based on this scale. A Krippendorff alpha [17] will then be calculated for each problem set, to test the overall user agreement throughout the different data sets. As 3 out of the 4 problem data sets are related to the field of computer graphics, it is believed that these categories will receive lower alpha values considering the rating will be depended upon the user’s expertise. Finally, in this section we will look at comparisons that can be drawn between the various problem sets in relation to the average Likert scale value obtained from participants, the Krippendorff alpha values obtained and the analogical rating received form the Dr Inventor system once these relations have been added.*

## 5.1 Run-Time

As stated in *Section Four*, the run time of the program was very slow. The system ran in *O(n3)* time plus the additional operations of reading in the ROS graph, reading in the ConceptNet corpus etc. This is obviously going to lead to problems considering some of our input ROS graphs have hundreds of nodes and the ConceptNet corpus has 2808254 different facts. For example, input Ros graphs from our corpus of SIGGRAPH abstracts have on average 13.87 lines, meaning it would take 540243199 (2808254 \* 13.87 \* 13.87) operations to find the relevant background information for one graph, not including the other operations that must be performed in one run of the system such as reading in the ConceptNet corpus, searching for specific GraphIDs etc. Also, the SIGGRAPH abstract corpus is the smallest data set we have, when looking at the full SIGGRPAH papers, there are on average 148 lines meaning it would take on average 61511995616 operations to find links for one of these graphs (plus the additional operations).

***Figure Six:*** Shows the time complexity of Version One of our program. If the number of rows in a graph is low, the number of operations needed is relatively low, however, the number of operations needed increases drastically as the number of graph rows go up.

This version of the system was tested on a ROS graph which contained 503 lines. This took over nine hours to return the results for this single graph. At the time of this testing process, the system did not take separate GraphIDs into account or did not search to find unknown internal node id values and so if these were also incorporated it would likely take a lot longer to run.

As it would not be feasible for Dr Inventor to add an extra nine hours run time to the system, a newer version for adding the additional background information was developed. This system ran in *O(n)* time complexity, meaning the speed of the system will have increased drastically. This was achieved using hashing in Java whereby version two of the system was set up to eliminate looking at cases where equals would return false in our for loop (i.e. no non-matching entries ever get compared). This means that the complexity of the searching process gets reduced form cubic to linear plus the number of operations needed to add values to our output.

***Figure Seven:*** A graphical representation of the time complexity of Version Two of our program. As the number of rows in a graph increases, the number of operations increase in a linear fashion.

Version 2 of the system was also tested on the same ROS graph with 503 lines, which took just 15.893 seconds. This means that the same output was produced over 2000 times faster. Although the speed of the system increased drastically, that doesn’t matter if the quality of results produced are not useful to the Dr Inventor system. With this in mind, in the following sections we will look at the quality of results produced when additional background information is added.

## 5.2 Initial Testing

As part of the initial test of the system, two participants were asked to rate the outputs from 5 randomly selected input ROS graphs based on the abstracts of SIGGRAPH papers. The information was rated on how informative or potentially useful the participant felt the new relation is at connecting the other two terms. For example, if a participant was given “cat isA mammal”, they would be asked to rate how useful they think *isA* is at relating cat and mammal. The results were based on a Likert Scale [5] which is a scale used to represent a participant’s attitude towards different results. Participants were given options from 1 to 5, where they would choose 1 if they considered the additional information very un-informative while they would choose 5 if they found the additional information very informative (See Figure Two).

|  |  |
| --- | --- |
| 1 | Very Un-informative |
| 2 | Somewhat un-informative |
| 3 | Neither uninformative nor informative |
| 4 | Somewhat informative |
| 5 | Very Informative |

***Figure Eight:*** Shows the Likert scale given to participants as a rating mechanism.

Surprisingly, there were on average 124.3 new relations formed between nodes in this corpus, and considering there were on average 13.87 lines per graph to begin with, this is a very large increase in relations. Adding an additional 124.3 relations per graph is far too high and shows that the system fell victim to the problem described in *Section 3* of oversaturating a graph with information. Also, considering the average rating obtained from the Likert scale was 1.62 per graph, the knowledge contained within the output was not considered particularly informative by the participants. However, when observing the results obtained in conjunction with the output of the system, it was noticed that specific relational verbs had very low ratings. For example, in the outputs presented to participants, the most frequent relation in all outputs was “RelatedTo”, which made up over half of the new relations for each output. It also received the lowest average results with most of these relations given a rating of 1. Example links which scored a 1 were *“RelatedTo, shape, distortion”* which can be read as “shapes are related to distortion” and *“RelatedTo, following, time”* which can be read “following is related to time”. These sentences make no sense to an English reader and so rightfully received a rating of 1. Other relations that generally produced values on the lower end of the Likert scale were “Antonym”, “Synonym”, “DerivedFrom”, “DistinctFrom” and “EtymologicallyRelatedTo”. The reason these relations generally scored lower than others is due to the fact that many of them draw relations between nodes based on them having an opposite meaning or give a different derivation of the same root word. For example, an *Antonym* is defined as “a word that means the opposite of another word” [25] and it is generally not useful to include relations between nodes based on them having an opposite meaning. Another example is *EtymologicallyRelatedTo* where etymology is a study of the origin of different words. In the output, we can find links such as *“EtymologicallyRelatedTo: today: day”*, which is not a useful relation to make for an output graph. As these relations generally received such a low value from participants, we can assume they will generally just clutter the graph with more useless information that good. With this in mind, it was decided to remove the relations mentioned above from the ConceptNet corpus and test the results again using the same procedure described above. As a note, removing these relations from the ConceptNet corpus reduced the file from 2808254 lines to just 576714, bringing the data to around 20% of its original size.

Another test was conducted in a similar fashion to the one described above with the same participants. The same procedure was conducted whereby participants were given the results obtained from 5 random input ROS graphs which consisted of information from the abstracts of SIGGRAPH publications. This rating was based on the same Likert scale shown in Figure Two. This time there were on average 6.92 new relations made between nodes in the input ROS graphs and considering there were 13.87 lines to begin with, this number of relations seems more fitting than the previously obtained 124.3 additions per graph. Considering the number of new relations added are quite low, it can be said that this time, the system did not fall victim to the problem described in *Section 3* of oversaturating a graph with useless information. This time, the average results given by each participant was 2.96 which is almost double the value (1.62) obtained from the previous test. Like with the previous test, this time particular words stood out as making more bad relations than good. For example, if a graph contained nodes such as *it*, *we* or *be,* the graph tended to have several more relations than others (because these are used by many relations in the ConceptNet corpus) and hence these words had a disproportionate frequency in the outputs. For example, out of the 5 graphs rated by participants, over 14% of outputs consisted of the words *it, we* and *be*. Links made between these words were generally given a rating of 1 except in one case where the relation “We isA Us” was given a rating of 2 by one participant. However, out of the 14% of output graphs, one occurrence given a rating of 2 (by one participant) doesn’t justify keeping all these relations in. Due to the fact that these words generally hinder the output of the system it was decided to remove these relations from the ConceptNet corpus also.

## 5.3 Results from the sets of problem data

After the words and relations that generally hindered the output were removed, it was decided to increase the number of participants rating the output data from 2 to 5. When reviewing the first two sets of data, I expected that the scores returned from the Likert scale would be quite low due to the number of relations and the quality of the data within. However, when run for the third time, the results seemed quite good on inspection and so it seemed fitting to add an additional 3 participants to get a better overview of results. For some context, the original two participants were two MSc in Computer Science students however for the third test, a lecturer of Computer Science and two people with no background in the field were asked to participate. As the results presented henceforth are from different fields such as Computer Science and Psychology, I believed it would be interesting to see how much a participant’s background affected their results. Each participant was given 80 random relations created by my system to rate on the Likert scale presented in Figure Eight. 20 of the relations were formed from the corpus of abstracts based on SIGGRAPH publications. 20 were formed from the corpus based on the full papers of these SIGGRAPH publications. 20 were from a corpus based on technology patents (related to the SIGGRAPH corpus) and finally 20 were based on papers from the field of Psychology. What follows is an in depth look at the different ratings for the different corpuses. In all results given to participants, 10 relations were between concept nodes (nouns) and 10 relations were between relational nodes (verbs). Participants were not told this fact before rating as I wanted to see if generally relations between concept nodes returned a greater average Likert scale than that of relations between verb nodes. This was due to the fact that finding relations between verb nodes can be tricky (i.e. finding a verb to link two other verbs) and I wanted to see if adding links between these produced a good enough output to garner inclusion in the final project.

The first input corpus we will look at is ROS graphs produced by the abstracts of SIGGRAPH publications. As stated in *Section 3.4*, this corpus originally had 13.87 rows (or edges) per graph and our system produced an extra 3.45 relations per graph. Again, this number is much more desirable than the original average of 113.2 additional relations per graph as it shows the graph isn’t being oversaturated with information. Overall the average Likert scale value obtained by each tester was 3.92 which is much higher than the original 1.62 and the next value of 2.96, meaning generally the information was considered “somewhat informative”. As expected, participants with a background in computer science tended to give better ratings as opposed to participants with no knowledge of the discipline. For example, one person with a computer science background rated the abstracts with an average value of 4.33 while one participant without the knowledge rated the paper 3.28. This was due to the computer specific language contained in some graphs. For example, one relation created was “R isA software”. This relation was given a score of 5 by the 3 participants with a background in the field while people without a background in computer science scored this relation with a 1. As computer scientists know, R is in fact a software however a lay person certainly would not know this information. The links between relational nodes in the output received an average Likert value of 3.26 so generally they dragged down the quality of the output of the system to a certain extent. The Krippendorff alpha value obtained was 0.3702455 which is higher than expected due to the number of possible options in our Likert scale, however it also has the lowest level of agreement obtained from any corpus. As stated at the start of *Section 5*, the Krippendorff alpha is used to test the user agreement achieved from our Likert scale and considering we have a large number of categories, this value is quite high.

The next corpus of problem data tested were 20 random relations based on the full papers of SIGGRAPH publications. This corpus originally had 147.59 relations per graph and our system produced an additional 56.3 relations per graph. This number is quite high however in such a large graph where there are numerous different words, there are going to be certain words that produce many links (similar to the problem with words like *it, be, we* described above). For example, in two of our graphs we can see the word *object*. This is generally going to be an important word to find links for in the context of computer graphics publications, so we could not remove it, however the word object appears 779 times in the ConceptNet corpus and so it’s generally going to create a lot of relations in our output graph. Having said this, the overall average Likert scale value given by each tester was 3.97 which is the highest value achieved from any of the problem corpuses. As with the information outputted by the abstracts of SIGGRAPH publications, this information is considered “somewhat informative” more often than not. Again, participants with a knowledge of computer science appeared to generally mark outputs with a higher Likert scale value. For example, “method TakesIn parameter” which can be read as a method takes in a parameter, was a relation formed in one graph. While the 3 people with a knowledge of computer science rated this relation with a 5, the two without a knowledge of the field rated this relation with a 1. The relations between verbs in this corpus were marked relatively evenly in comparison to the relations made between nouns, receiving an average result of 3.89. For example, one relation “generate isA make” scored on average 3.89 and a similar link “generate isA produce” scored 4.26 on average out of the 5 participants. This is relatively close to the overall average of 3.97 and so it can be said these relations between verb nodes don’t hinder the results. The overall agreement between participants was again calculated using a Krippendorff alpha which retuned a value of 0.548, which means it is the problem data with the highest rate of agreement between participants. Also, it must be noted that being the category with the highest rate of agreement, it is also the category with the highest average results, which was unexpected due to the volume of data produced when adding the additional background information to graphs in this corpus.

The next data set given to participants for rating were based ROS graphs derived from technology Patents related to the SIGGRAPH corpus. This corpus originally had on average 24.59 relations per graph. On average our system introduced an additional 4.68 links per graph and so it can be said that this information will not overwhelm the ROS graph. Overall, the average Likert scale value obtained by each tester was 3.58 which is the lowest average obtained from any problem set. The additional information in this corpus found the highest frequency of 3’s from the Likert scale, meaning generally information in this corpus was considered “neither uninformative nor informative”. For example, a link can be seen between “signal IsA communication” and “data isA information”. Both of these relations scored 3, meaning although they were correct, participants didn’t find the information particularly useful. In this data set relations between verbs scored higher than relations between nouns on average, the only data set where this happened. Several verb relations in this data set received a score of 5 among all participants, such as “validate isA confirm” and “recruit isA enlist”. Again, the Krippendorff alpha value obtained from this data set was higher than expected (due to the number of categories) at 0.428843. Finally, due to the fact that this corpus was based on patents related to the SIGGRAPH corpus, it was expected that the results would vary based on the participants expertise, however, this aspect did not seem to come into effect here, with the averages remaining consistent among participants.

The final data set tested were the outputs based on ROS graphs derived from different Psychology texts. The main reason this corpus was tested was because all other sets of problem data were loosely based in the domain of computer graphics and I thought it would be interesting to step outside this to see how results from a different domain perform. Also, as some participants had no knowledge of computer graphics, this would give them an equal opportunity in ratings and so we would believe the Krippendorff alpha value obtained would be higher than that of any other corpus. Originally this corpus had on average 19.74 relations per graph. On average, our system added an additional 3.12 relations per graph. Interestingly in this data set, there were more graphs than in any other data set that found no background relations. Overall the average Likert scale value obtained by each user was 3.81 which is lower than that obtained from the corpus of abstracts or the corpus of full papers, which was unexpected. Also, the relations between verbs in the output were not rated very highly for example, two relations made were “tug isA pull” and “think hasPrerequisite do”. Rightfully neither of these relations were rated highly as neither are particularly informative. Having said this, the average result obtained from relations between verbs was 2.98. As expected (due to the fact that participants from the field of computer science had no greater knowledge from the field of Psychology than the other participants), the Krippendorff value achieved was quite high at 0.5175596 but again this is higher than expected due to the fact that there are so many categories in our Likert scale.

## 

## 5.4 Comparisons Between Problem Data sets

The average results obtained from all 80 outputs including relations between nouns and verbs was 3.82, meaning that on average participants felt the information was somewhat informative. The corpus of technology patents received the lowest rating, receiving an average of 3.58. Surprisingly, the full papers based on SIGGRAPH publications scored the highest average of 3.97. This was a surprising result due to the number of relations created per graph. As stated in Section 5.4, certain words tend to produce several links and words such as object, which appeared in this data set frequently, tend to clutter the graph with several unwanted relations. When looking at this data set, the fact that on average an additional 56 relations per graph were added, it is surprising that on average this data was considered more informative than smaller sets of relations. Interestingly, the full papers also had the highest level of agreement between users, obtaining a value of 0.5481558 using the Krippendorff alpha measurement. Again, this was unexpected not only due to the amount of data produced from the corpus but also due to the fact that some of the participants have no knowledge of the field of computer science. Although the abstract corpus had the second highest average result from the Likert scale, the Krippendorff value was the lowest at 0.3702455, which is closer to what we would have expected to receive from the full papers.

Throughout all data sets the Krippendorff alpha value obtained was much higher than expected. On average, the alpha value received was 0.467950975 which is extremely high not only considering there are 5 categories in the Likert scale but also due to the fact that participants had different areas of expertise. As the number of categories in a Likert scale increases, the alpha value tends to decrease [11] and so the value obtained was very high. This was partly due to the fact that background information considered bad was generally given a result of “very un-informative” by all participants and additional information considered good was generally considered “very informative” by all participants. For example, if a relation such as “Dog isA cat” is presented to participants, all participants will give this a rating of 1 considering the relation is untrue. On the other hand, if a relation such as “set CreatedBy elements” is presented to participants, generally all will give this a rating of 5. Although there were some exceptions to this problem, the majority of links tended towards 5 or 1 and so this made the Krippendorff value obtained higher than expected.

Although the results received were in most cases higher than expected, the most import test to partake in was to pass this newly created data into the Dr Inventor system and see how this effects the quality of analogy produced by the system. When looking at the psychology data set, two texts were chosen randomly which ended up being “Wide-Canal-Carocci” and “Tumour-Carocci” (Appendix B and C respectively). These two texts were then fed into the Dr Inventor system where the papers received an “analogical similarity” of 0.0764 and an overall similarity of 0.0000. However, during the phase of the Dr Inventor system where the GATE parser [13] returns information such as that present in Figure Two, we took this information and found the additional background information using my program. This information was then fed back into the Dr Inventor system where an “analogical similarity” of 0.0786 was given and an “overall similarity” of 0.0289 was given. An additional 3 relations were added to the “Wide-Canal-Carocci” text (those being “Canal HasA Water” and “Lake HasA Water” and “Water CapableOf Flow”). Originally this file had 16 relations and so an addition of 3 background links should not be considered to be oversaturating the graph. No additional relations were added to the “Tumour-Carocci” text likely due to the fact that only 8 relations were present originally (graphs with a lower number of nodes to begin with tend to get less links). As both the analogic similarity and overall similarity increased, the addition of background information is successful. A general improvement to these factors is what was expected while a dramatic change would be worrying as it is not intended that the additional background information will overwhelm the existing data.

# Section 6: Conclusion

*In this section, we will look at the various conclusions that can be drawn from the above results. We will look at factors such as how many relations we can generally expect from an input graph. We’ll also look at some future works which could be added to my portion of the project to enhance the results obtainable.*

## 6.1 General Conclusions

The results obtained from the various participants was higher than expected. Generally speaking, this type of rating system is quite subjective and based on the participants expertise and so obtaining an average Likert scale value of 3.82 and an average Krippendorff alpha of 0.5481558 was very good. I thought it would be interesting to then see how many relations can be expected on average from any given input graph. With this in mind, 25 full graphs were outputted from each set of problem data and the number of relations created by each were counted. The average lines per original graph were also counted. I then divided the average number of original relations per file by the average number of additional relations our system created and achieved a value of 2.2486. This means that on average for every 2.2486 lines in an input graph, my program would be expected to draw one new relation. However, this number does not generally work into practice. Certain data sets skew this number highly. For example, in larger input graphs such as those consisting of the full papers from SIGGRAPH publications, there were on average 147.59 relations to begin with, while there were on average 56.3 relations added by our system. This means that on average the number of relations found by an input graph are roughly one third the size of the input graph. On the other hand, the abstracts generally had 13.87 rows or edges per graph to begin with while our system produced an extra 3.45 relations per graph. In this case, average number of relations found are roughly 4 times the smaller than that of the input graph. As stated throughout Section 5, bigger graphs generally receive disproportionately more relations than a smaller graph, due to the occurrence of certain words that over populate the graph (for example, the word object can overpopulate a graph as the occurrence of such a word is so high in the ConceptNet corpus). However, with this aside, one can generally assume they will receive one relation for every 2.2486 concept nodes.

## 6.2 Future Works

Although I am very happy with the output of the system, if time permitted, I would change one major aspect of the system. ConceptNet was chosen as the best knowledge resource for use in this project and I believe that decision was justified, however, if time permitted, I would choose to add an additional knowledge resource alongside ConceptNet. I believe results would receive an even higher rating if both ConceptNet and BabbleNet were integrated alongside each other to mine the background data. Perhaps the system could run as normal and return the relevant background links and then we could look in the BabbleNet corpus to see if the additional links produced are also present in this corpus. If the link is present in both resources, then it would give us a stronger indication that this relation is valid. Again, this is important as one of the main objectives of this thesis was to add useful background information while keeping the frequency of useless information low.

# References:

[1] ConceptNet.io. *ConceptNet.* 2017. [web] Available at: <http://conceptnet.io/> [Accessed 12 October 2017].   
[2] Bunge, M. *Method, model and matter.* Dordrecht, Boston: D. Reidel Publishing Company, 1973 [3] O’ Donoghue, Diarmuid et al. “Proceedings of the Sixth International Conference on Computational Creativity”. (2015): 220-227  
[4] Dr Inventor, a personal research computer assistant for scientific creativity. *Eu.europa.eu*. [web] Available at: <https://ec.europa.eu/digital-single-market/en/news/dr-inventor-personal-research-computer-assistant-scientific-creativity/> [Accessed 28 October 2017].  
[5] Joshi, Ankur. “Likert Scale: Explored and Explained”. *British Journal of Applied Science & Technology.* 7.4 (2015): 397.  
[6] Gentner, Deidre. “Analogical Reasoning”. *Encyclopaedia of Human Behaviour.* 2nd Edition (2012): 130-136.  
[7] Duit, Reinders. “On the Significance of Conceptual Metaphors in Teaching and Learning Science”. *International Journal of Science Education*. Volume 37 (2015): 958-965.  
[8] Gick, Mary and Holyoak, Keith. “Analogical Problem Solving”. *Cognitive Psychology*. 12 (1980): 306-355  
[9] Shapiro, M. “Analogies, Visualization and Mental Processing of Science Stores”. *Paper presented to the information Systems Division of the International Communication Association*. (1985) 15 – 40.  
[10] O’ Donoghue, Diarmuid and Keane, P. “A Creative Analogy Machine” Results and Challenges. Proc Intl, Conf. on Computational Creativity (ICCC), 17-24.  
[11] Abgaz et al. “Characteristics of Pro-c Analogies and Blends between Research Publications”. *International Conference on Computational Creativity* (ICCC), Atlanta, GA, USA, (2017).  
[12] Gentner, Deidre. “Structure-Mapping: A Theoretical Framework for Analogy”. *Cognitive Science – A multidisciplinary Journal*. Volume 7, Issue 2. (1983). 155-170.  
[13] Ronzano, F and Saggion, H. “Dr Inventor Framework”. *Extracting Structured information from Scientific publications.* (2015): 209 – 220.   
[14] Neo4j.com. [web] Available at <https://neo4j.com/> [Accessed 15 November 2017].  
[15] Cordella, L., Foggia, P., Sansone, C., and Vento, M. A subgraph isomorphism algorithm for matching large graphs. IEE Transactions on Pattern Analysis and Machine Intelligence. 26 (2004): 1367 – 1372.  
[16] Bryl, V., Giuliano, C., Serafini, L and Tymoshenko, K. “Supporting Natural Language Processing with Background Knowledge”. *ISWC*. (2010): 80 – 95.  
[17] Krippendorff, Klaus. “Computing Krippendorff’s Alpha-Reliability”. (2011).  
[18] Chen, Ying., Argentinis, Elenee., and Weber., Griff. “IBM Watson: How Cognitive Computing Can Be Applied to Big Data Challenges in Life Sciences Research”. *Clinical Therapeutics*. Vol 38. (2016): 688 – 701.  
[19] High, Rob. “The Era of Cognitive Systems: An Inside look at IBM Watson and How it Works”. (2012).  
[20] “Watson and the Jeopardy! Challenge” YouTube, uploaded by IBM Research, 6 November 2013, https://www.youtube.com/watch?v=P18EdAKuC1U.  
[21] Babblenet.org [web] Available at: http://live.babelnet.org/ [Accessed 14 October 2017].  
[22] ProjectNell. Carnegie Mellon University. [web] Available at: <http://rtw.ml.cmu.edu/rtw/home> [Accessed 14 October 2017].  
[23] Wordnet. Princeton.edu. [web] Available at: <https://wordnet.princeton.edu/> [Accessed 18 October 2017].  
[24] Apache. POI Java Library (Version 3.17) [Source Code]. (2017): Available at <https://poi.apache.org/>   
[25] Collins English Dictionary. (2006). Anderson Sandra.