Creating a Phrase Similarity Graph From Wikipedia

Lubomir Stanchev
Computer Science Department
Indiana University - Purdue University Fort Wayne
Fort Wayne, IN, USA
Email: stanchel@ipfw.edu

***Abstract*—The paper addresses the problem of modeling the relationship between phrases in English using a similarity graph. The mathematical model stores data about the strength of the relationship between phrases expressed as a decimal number. Both structured data from Wikipedia, such as that the Wikipedia page with title "Dog" belongs to the Wikipedia category "Domesticated animals", and textual descriptions, such as that the Wikipedia page with title "Dog" contains the word "wolf" thirty one times are used in creating the graph. The quality of the graph data is validated by comparing the similarity of pairs of phrases using our software that uses the graph with results of studies that were performed with human subjects. To the best of our knowledge, our software produces better correlation with the results of both the Miller and Charles study and the WordSimilarity-353 study than any other published research.**

I. Introduction

The main goal of the paper is to describe how to create a similarity graph that can be used to calculate the degree of semantic similarity between phrases. For example, the graph can be used to tell us that the similarity between the phrases "New York" and "big apple" is around 0.5 because the two phrases can represent the same concept. In the same way, the graph can be used to tell us that the similarity between the phrases "National Hockey League" and "United Nations" is around 0.01 because there is little correlation between the two phrases. The graph can also be used to compute the strength of the asymmetric relationship between phrases. For example, the graph can tell us that someone who is interested in documents about the "Guggenheim Museum" is also interested in documents about "New York" with relatively high probability. The reason is that the Guggenheim museum is located in New York. However, just because someone is interested in documents about the phrase "New York" does not give us confidence that they are also interested in documents about the "Guggenheim Museum" phrase. The reason is that the Guggenheim museum is only one of thousands of famous landmarks that are located in New York city.

If we type "big apple" in our favorite Internet search engine, for example Google or Bing, then all top results will contain the phrase "big apple". Most search engines will not return web pages that contain "New York" but do not contain "big apple" as one of the top results. The reason is that most Internet search engines rely on keyword matching to compute the query result and do not posses the knowledge that the phrases "New York" and "big apple" are semantically similar and the degree of this semantic similarity. The similarity graph captures this semantic similarity. For example, the graph can be used to find phrases that are semantically similarly to the phrase "vitamin C", such as "ascorbic acid", and rewrite a query using these phrases. In this way, the similarity graph will allow us to not only perform semantic search (i.e., search based on the meaning of phrases), but it will also help us rank the result. For example, results that contain the phrase"ascorbic acid" may be ranked higher than results that contain the phrase "antioxidant" because, according to the graph, the phrase "ascorbic acid" is semantically closer to the phrase "vitamin C" than the phrase "antioxidant". Another interesting software application is using the similarity graph to partition a set of documents based on the meaning of the phrases in them. The similarity graph can be used to measure the semantic similarity between any pair of documents. Then a clustering algorithm, such as K-Means clustering ([17]), can be applied. The similarity graph can also be used as part of a query-answering system, such as the IBM Watson Computer that competed on the Jeopardy game show and the Siri system for the iPhone. For example, suppose that the phrase "statue of liberty" is part of the user query. Then the similarity graph can be used to rewrite the query using semantically similar phrases, such as "France" and "New York". Such a rewrite can help the system find more information that is related to the user query.

The problem of evaluating the strength of the semantic relationship between phrases is intrinsically hard because computers are not as proficient as humans in understanding natural language text. However, natural language descriptions can provide important evidence about the similarity between phrases. For example, the Wikipedia document with title "Hockey" contains the word "Canada" nine times. This fact can serve as evidence about the strength of the semantic relationship between the words "hockey" and "Canada". Although significant effort has been put forward in automated natural language processing (e.g., [6], [7], [18]), current approaches fall short of understanding the precise meaning of human text. In fact, the question of whether computers will ever become as fluent as humans in understanding natural language text is an open problem. In this paper, unlike most natural language processing applications, we do not parse text and breakdown sentences into the primitive elements of the language (e.g., nouns, verbs, etc.). Instead, we only examine the words in the text. Our algorithm is based on our previous work ([28]) that also considers the order of the words in a sentence.

Current approaches that extract information about word and phrase similarity from freely accessible sources focus on the structured information. In particular, most papers that deal with WordNet (e.g., [15], [31]) adapt the approach taken in [23] that semantic similarity can be measured solely based on the inheritance (a.k.a. kind-of) links and possibly data about the specificity of the words (i.e., their information content – see [22], [16], [11]). Note that WordNet is a lexical database that describes the words in the English language, their meaning, and the relationship between the words. More recent papers, such as [32], explore additional relationship between words, such as the holonym (a.k.a. part-of) relationship. Although these approaches work well in practice and produce similarity data that closely correlates to data from human studies, such as [19], we show that there is room for improvement. In particular, unstructured information, such as that a Wikipedia web page contains a word multiple times, is not considered. In our previously published algorithm ([28]), we also extracted unstructured information from WordNet. For example, the definition of one of the senses of "New York" is that it is a city that is located on the Hudson river. This close relationship between "New York" and "Hudson river" is not considered by the other papers that are cited in this paragraph because these algorithms do not process textual information.

In this paper, we propose a novel mechanism for measuring the semantic similarity between phrases based on information from Wikipedia and we extend our previously published research ([28]) that extracts the degree of semantic similarity between words based on information from WordNet. We show how information from Wikipedia can be used to extend the *similarity graph* that was constructed based on information from WordNet. The graph is created using probability theory and corresponds to a simplified version of a Bayesian network ([21]). The weight of an edge represents the probability that someone is interested in the content of the destination node given that they are interested in the content of the source node. Note that the weight function is asymmetric. We experimentally validate the quality of our algorithm on two independent benchmarks: Miller and Charles ([19]) and WordSimilarity-353 ([5]). Our approach outperforms existing algorithms that we are familiar with on both benchmarks because we process more information as input, including natural language descriptions, and we are able to apply this information to build a better model of the semantic relationships between words and phrases.

In what follows, in Section 2 we review related research. The major contribution of the paper is an algorithm that adds information from Wikipedia to a similarity graph – see Section 3. Section 4 presents two algorithms for measuring the semantic similarity between phrases that use the similarity graph. Section 5 shows how our system compares to existing systems that measure the semantic similarity between words and phrases of the English language, while concluding remarks and areas for future research are outlined in Section 6.

II. Related Research

This paper extends a previous workshop paper that creates a similarity graph from WordNet ([28]). WordNet contains information about the words in the English language. For example, WordNet contains the information that one of the senses of the word "chair" is "a seat for one person". Alternatively, Wikipedia contains information about phrases from the world that we live in, such as "United Nations" and "Olympic Games". This paper extends the similarity graph that was created in [28] by adding information from Wikipedia.

Existing research that applies Bayesian networks to represent knowledge deals with the uncertain or probabilistic information in the knowledgebase (e.g., [24], [20]). In this paper, we will take a different approach and we will not use Bayesian networks to model uncertain information. In contrast, we will create a probabilistic graph that stores information about the similarity of phrases. Unlike Bayesian networks, we store only the probability that a phrase is relevant given that an adjacent (in the graph) phrase is also relevant (e.g., unlike Bayesian networks, we do not store the probability that a phrase is unrelated given that an adjacent in the graph phrase is unrelated).

The idea of creating a graph that stores the degree of semantic similarity between words or phrases is not new. For example, [13], [25] show how to create a graph that only represents inheritance of words, while [10] approximates the similarity of words based on information about the structure of the graph in which they appear. These papers, however, differ from our approach because we suggest representing available evidence from all type of sources, including natural language descriptions and Wikipedia. Our approach is also different from the use of a semantic network ([29]) because the latter does not consider the strength of the relationship between the nodes in the graph.

There are alternative methods to measure the semantic similarity between words. The most notable approach is the Google approach ([4]) in which the similarity between two phrases is measured as a function of the number of Google results that are returned by each phrase individually and the two phrases combined. Other approaches that rely on data from the Internet include [2] and [14]. Although these approaches produce good measurement of semantic similarity, they have their limitations. First, they do not make use of structured information, such as the hyponym (i.e., kind-of) relationship in WordNet and the category-subcategory relationship in Wikpedia. Second, they do not provide evidence about the semantic similarity score that is returned. In contrast, our approach can show the paths in the similarity graph between the two input phrases, which serves as evidence that supports the similarity score.

Research from information retrieval is also relevant to creating and using the similarity graph. For example, if the word "ice" appears multiple times in the Wikipedia page "Hockey", then this provides evidence about the relationship between the two words. Our approach will use a model that is similar to TF-IDF (stands for term frequency, inverse document frequency – see [12]) to compute the strength of the semantic relationship between phrases. In the TF-IDF model, if the word "ice" appears two times in the Wikipedia page for "Hockey", then the term frequency can be computed as 2. This number is multiplied by a number that is inversely proportional to how often the word "ice" appears in other Wikipedia pages. For example, if most Wikipedia pages contain the word "ice", then the fact that the Wikipedia page "Hockey" contains this word is not consequential. Conversely, if the word "ice" appears only in few Wikipedia pages, then the fact that the Wikipedia page "Hockey" contains the word "ice" is statically meaningful.

Note that a lot of research effort has recently focused on using a description language, such as OWL (stands for Web Ontology Language – [1]), to describe document resources. A semantic query language, such as SPARQL (a recursive acronym that stands for SPARQL Protocol and RDF Query Language – [26]), can be used to search for relevant documents. This approach differs from our approach because it does not provide ranking of the query result. At the same time, a SPARQL query returns exactly the resources that fulfil the query description. Alternatively, our system can return resources that are related to the input query in ranked order. Using a similarity graph has some added advantages: there is no need to describe the resources using a mathematical language, there is no need to phrase the query using a mathematical language, and the system is much more scalable (OWL knowledgebases are usually applied only to a limited knowledge domain because query answering over them is intrinsically computationally expensive.)

III. Creating the Similarity Graph

While WordNet provides information about the words in the English language, Wikipedia gives us information about the world that we live in. It contains information about people, events, organizations, sports, and history, to name a few topics. A Wikipedia dump was downloaded from the Wikipedia web site. This download contains a snapshot of Wikipedia as an XML file. We transformed the file into a relational database file using the MediaWiki software (www.mediawiki.org). From the database, information about the categories and the Wikipedia pages was extracted. For example, there are relational tables that describe which category contains what pages and what subcategories. There is also a relational table that contains information about the hyperlinks in every Wikipedia page. We also extracted the text from every Wikipedia page and created links between Wikipedia pages, Wikipedia categories, and word forms from WordNet. Note that while most of the entries in WordNet are single words, it also contains word forms, such as "sports utility vehicle".

Before we present our algorithm, it is is worth mentioning that the algorithm depends on a plethora of parameters, which are represented as constants through this section. Experimental results have shown that increasing the values of these parameters can adversely affect the correlation results that are presented in Chapter 5. These parameters represent our confidence in the Wikipedia data. Since the data in Wikipedia is not as precise as the data in WordNet, the values for the parameters are relatively low. For example, we believe that there is a 10% chance that someone who is interested in the title of a Wikipedia page will be also interested in one of the word sequences that appear in the title. Conversely, we believe that there is 60% chance that someone who is interested in a sense from WordNet will be also interested in one of the word sequences in the definition of the sense.

It is also worth briefly describing how WordNet is used to create the input graph. A node is created for every word form and every sense. The label of a word form node is the word, while the label of a sense node is the definition of the sense. Two-way edges are drawn between every word form and its senses. Next, two-way edges are drawn between senses to represent the *hyponym* (a.k.a. kind-of) and *meronym* (a.k.a. part-of) relationships between senses that represent nouns. Similarly, edges that represent the *troponym* relationship for verbs are drawn, where the verb Y is a troponym of the verb X if the activity Y is doing X in some manner. We also draw edges that represent the *related to* and *similar to* relationships between adjectives. Note that text descriptions also result in new edges. For example, we draw edges between a sense node and the nodes for the words that appear in its definition and its example use.

The first step of the algorithm is to add nodes to the similarity graph that represent titles of Wikipedia pages, categories, and redirections. A redirection is a Wikipedia page that points to a different Wikipedia page with different title. The label of each new node is the title of the Wikipedia page, category, or redirection in all lowercase letters. Note that if a node with that label already exists (e.g., from WordNet), then a new node is not created.

We next process the redirection information in Wikipedia. For example the Wikipedia page with title "Accessible computing" has a redirection to the Wikipedia page with title "Computer accessibility". We assume that there is a 20% change that someone who is interested in the title of the initial Wikipedia page will also be interested in the title of the page that the redirection link points to. Therefore, we will draw an edge from the node "accessible computing" to the node "computer accessibility" with weight 0.2 (see Figure 1). We will also draw a reverse edge from "computer accessibility" to "accessible computing". The weight of this edge will be equal to 0.2 divided by the number of redirections to the node "computer accessibility". For example, if there are three redirections to the Wikipedia page "Computer accessibility", then this will result in the partial graph that is shown in Figure 1.

Next, we will draw edges between nodes for Wikipedia pages, categories, and redirections and nodes that represent word forms in WordNet. Note that a Wikipedia page can have both a title and subtitle, where the subtitle is written in parentheses. Given a title, we will tokenize it and extract all words, pairs of consecutive words, and triplets of consecutive

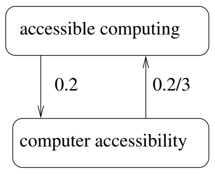

Fig. 1. Representing redirection from Wikipedia.

words from it. We will then draw edges between the Wikipedia node and each word form node from WordNet that has label that is one of the extracted tokens. We consider a sequence of at most three words because this is the longest word form in WordNet. The weight of the edge is computed using the formula *computeMinMax*(0,0.1,*ratio*). The variable *ratio* is equal to the number of times the word form appears in the title divided by the total number of words in the title. For example, Figure 2 shows how the title "National Hockey League" will be processed. The number 0.1 represents the fact that we assume that there is a 10% probability that someone who is interested in the title of a Wikipedia page will be also interested in one of the word sequences that appear in the title.

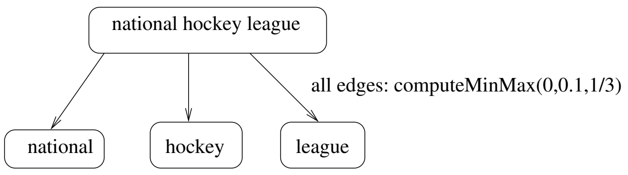

Fig. 2. Wikipedia pages to word form edges.

The *computeMinMax* function returns a number that is almost always between the first two arguments, where the magnitude of the number is determined by the third argument. Since the appearance of a phrase in the title of a Wikipedia page is not a reliable source of evidence about the relationship between the two, the value of the second argument is set to 0.1 in the above example. The *computeMinMax* function smooths the value of the *ratio* parameter. For example, a word that appears as one of 20 words in the title of a Wikipedia page is not 10 times less important than a word that appears as one of the two words in the title of a Wikipedia page. The function makes the difference between the two cases less extreme. Using this function, the weight of the edge in the second case will be only roughly four times smaller than the weight of the edge in the first case. This is a common approach when processing text. The importance of a word in a text decreases as the size of the text increases, but the importance of the word decreases at a slower rate than the rate of the growth of the text. Formally, the function *computeMinMax* is defined as follows.

$$computeMinMax(minValue, maxValue, ratio) = \\ minValue + (maxValue - minValue) * \frac{-1}{log_2(ratio)}$$

Note that we use the above formula only when the value of the *ratio* variable is smaller or equal than $1/2$. For example, if `ratio`=1, then the value of *computeMinMax* is not well defined because of division by zero. Therefore, when *ratio* $> 1/2$ we set the value of the function as *maxValue**1.2. Note that this can happen only in rare circumstances (e.g., a Wikipedia title that consists of a single word).

We use the formula *computeMinMax*(0,0.05,*ratio*) to compute the weight of an edge between a word form in the subtitle of a Wikipedia document and a word form node. In other words, we consider the information in the subtitle twice less important than the information in the title of a Wikipedia document. Therefore, we assume that there is a 5% chance that someone who is interested in the title of a Wikipedia document will be also interested in one of the word sequences that appear in the subtitle of the document.

We also examine the text of each Wikipedia document and identify word forms that repeat five times of more. We believe that this signals a relationship between the title of the Wikipedia document and the word forms that appear multiple times. We will compute the weight of such an edge as *computeMinMax*(0,0.05,*ratio*), where *ratio* here is equal to the number of times the word form appears in the text multiplied by the number of words in the word form and divided by the size of the text that consists of word forms that appear five times or more. In other words, we do not penalize for the size of the document. Instead, we only consider how many word forms appear five times or more. The number 0.05 is the probability that someone who is interested in the title of a Wikipedia document will be also interested in one of the word forms in the document that appears five times or more. For example, the word form "Canada" appears 89 times in the Wikipedia page with title "Ice hockey at the the Olympic Game". If the word forms in the document that repeat five times or more make up 300 total words, then we will draw the edge that is shown in Figure 3.

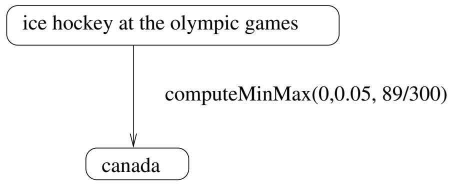
ice hockey at the olympic games
computeMinMax(0,0.05, 89/300)
canada

Fig. 3. Edges for text in Wikipedia pages.

We will also add backward edges between the word forms and the Wikipedia documents. For example, suppose that the word form "coat" appears in four Wikipedia page titles. Then we will draw an edge between the word form and each of the four Wikipedia pages. The weight of each edge will be equal to *computeMinMax*(0,0.05,1/4). We chose the number 0.05 because we estimated the probability that a user is interested in one of the Wikipedia pages that contain a specific word form in their title given that they are interested in the word form as 5%. Figure 4 shows an example of how the graph is built.

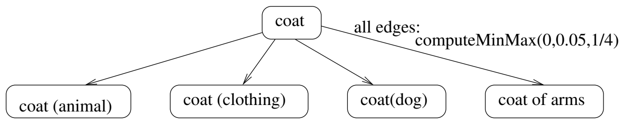

Fig. 4. Edges between word form nodes and Wikipedia page nodes.

Similarly, we will draw edges between a word form and nodes for Wikipedia categories that contain the word form. The value of the edge will be computed the same way. Finally, we will draw an edge between a word form and a node for a Wikipedia document that contains the word form in its subtitle. Here, the weight of each edge will be computed as *computeMinMax*(0,0.025,*ratio*), where *ratio* is equal to 1 divided by the number of Wikipedia pages that contain the word form in their subtitle. We chose the number 0.025 because we estimated the probability that a user is interested in one of the Wikipedia pages that contain a specific word form in their subtitle given that they are interested in the word form as 2.5%.

Next, we will examine the see-also links. For example, consider the Wikipedia page for “Hospital”. It has five “see also” links, including “Burn center”, and “Trauma center”. The see-also links provide evidence about the relationship between the concepts (e.g., hospital is related to trauma center). We will draw edges between the Wikipedia page node and each of the see-also page nodes. The weight of each edge will be equal to 0.05 divided by the number of see-also links – See Figure 5. We chose the number 0.05 because we believe that there is a 5% probability that someone who is interested in the title of a Wikipedia page will also be interested in the title of one of the see-also pages.

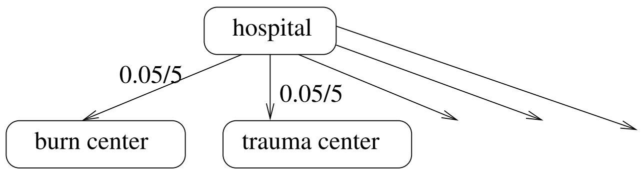

Fig. 5. Edges for see-also links.

We will also add backward edges for the see-also links. For example, if there are 20 see-also links that point to the node for the Wikipedia page with title “Hospital”, then we will draw an edge for each link and give it a weight that is equal to 0.025/20. In general, the weight of each edge will be equal to 0.025/*count*, where *count* is the number of incoming see also-links. We chose the number 0.025 because we believe that there is a 2.5% probability that someone who is interested in the title of a Wikipedia page is also interested in one of the titles of the Wikipedia pages that points to it using a see-also link.

We will also add edges for the hyperlinks in the Wikipedia documents. For example, consider the Wikipedia page with title ”Canada”. It has a single hyperlink to the Wikipedia page with title “Maple Leaf Flag”. At the same time, it has 530 hyperlinks to Wikipedia pages. We will draw the edge between the two nodes that is shown in Figure 6. In general, the weight of an edge is equal to 0.05 times the number of hyperlinks to the destination Wikipedia page divided by the total number of hyperlinks in the original Wikipedia page. We have chosen the number 0.05 because we believe that there is a 5% chance that someone who is interested in the title of a Wikipedia page will also be interested in one of the titles of Wikipedia pages that can be reached using one of the hyperlinks in the original Wikipedia page.

Consider again the Wikipedia page for “Maple leaf flag”. If, for example, there are 10 hyperlinks pointing to it, then we will draw backward edges for each hyperlink. The weight of each edge will be equal to 0.025 divided by the total number of hyperlinks towards the page – see Figure 6. Here we assume that there is a 2.5% chance that someone who is interested in the title of a Wikipedia page will be also interested in the title of one of the Wikipedia pages that points to it using a hyperlink.

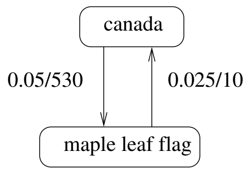

Fig. 6. Edges for hyperlinks.

Next, consider the “Furniture” Wikipedia category. “Beds” is one of 24 subcategories. Therefore, we will draw an edge between the nodes for the two pages with weight that is equal to 0.1*(sub category size)/(size of all subcategories). This is the probability that someone who is interested in furniture is also interested in beds. The number 0.1 represents that we assume that there is a 10% change that someone who is interested in the title of a category will also be interested in the title of one of the subcategories. We estimate the “size” of a category as the total number of Wikipedia pages that it contains. For example, the category “Beds” contains 41 pages, while all 24 subcategories of the “Furniture” category contain a total of 917 Wikipedia pages. Therefore, we will draw the edge that is shown in Figure 7. Note that “Beds” is one of the bigger subcategories of the “Furniture” category. Therefore, the edge between the two nodes will have bigger weight than the edge between the nodes for “Furniture” and “Kitchen countertops”, for example. The reason is that the “Kitchen countertops” category contains only 5 pages.

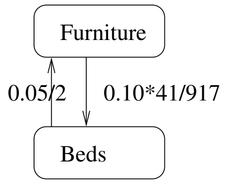

Fig. 7. Edges for subcategories.

We will also draw a backward edge from "Beds" to "Furniture". The weight of this edge will be computed as 0.05/*count* – see Figure 7. The variable *count* represents the number of super-categories of the category page. For example, the "Beds" category has two super-categories: "Furniture" and "Sleep". Therefore, the variable *count* in this case is equal to 2. We have chosen the number 0.05 because we estimate that there is a 5% chance that someone who is interested in the title of a Wikipedia category will also be interested in the title of one of the super-categories.

Lastly, consider the "Beds" category and the "Adjustable bed" Wikipedia page that belongs to the category. Recall that there are 41 pages in the "Beds" category. We will draw the edges that are shown in Figure 8. The forward edge is calculated as 0.1 divided by the number of pages in the category. In other words, we estimate that there is a 10% chance that if someone is interested in the title of a Wikipedia category, then they are also interested in the title of one of the Wikipedia pages in the category. The backward edge is calculated as the 0.05/*count*, where *count* is the number of categories that the Wikipedia page appears in. This means that there is a 5% probability that someone who is interested in the title of a Wikipedia page is also interested in the title of one of the categories that the page appear in. Since the "Adjustable bed" Wikipedia page appears only in the category "Beds", we will draw the backward edge that is shown in Figure 8.

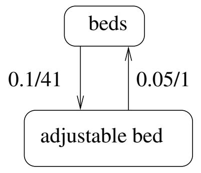

Fig. 8. Page-category edges.

IV. Measuring Semantic Similarity Between Phrases

The similarity graph is used to represent the conditional probability that a user is interested in a phrase given that they are interested in an adjacent phrase in the graph. We compute the directional similarity between two nodes using the following formula.

$$A \rightarrow_s C = 1 - \prod_{Pt \text{ is a cycleless path from A to C}} (1 - P_{Pt}(C|A)) \quad (1)$$

$$P_{Pt}(C|A) = \prod_{(n_1,n_2) \text{ is an edge in the path } Pt} P(n_2|n_1) \quad (2)$$

Informally, we compute the directional similarity between two nodes as a function of all the paths between the two nodes, where we eliminate cycles from the paths. Each path provides evidence about the similarity between the phrases that are represented by the two nodes. For example, suppose that there are two paths between "car" and "auto". The first path has weight of 0.6 and the second path has a weight of 0.5. In other words, we have evidence that someone who is interested in "car" is also interested in "auto" with probability 60% and 50%. If we combine the available evidence, then we get directional similarity of $1-(1-0.6)\cdot(1-0.5)=0.8$. This is the probability that we succeed in two independent tries, where the probability in the first try is 50% and the probability of the second try is 60%. In other words, every path brings new evidence that can increases the value of the directional similarity, but the value can never become more than one.

We compute the directional similarity between two nodes on a path as the product of the weights of the edges along the path, which follows the Markov chain model. Since the weight of an edge along the path is almost always smaller than one (i.e., equal to one only in rear circumstances), the value of the conditional probability will decrease as the length of the path increases. This is a desirable behavior because a longer path provides less evidence about the similarity between the two end nodes.

Next, we present two functions for measuring similarity. The linear function for computing the similarity between two phrases is shown in Equation 3.

$$|w_1, w_2|_{lin} = min(\alpha, \frac{w_1 \rightarrow_s w_2 + w_2 \rightarrow_s w_1}{2}) * \frac{1}{\alpha} \quad (3)$$

The minimum function is used to cap the value of the similarity function at 1. α is a coefficient that amplifies the available evidence. The experimental section of the paper shows how the value of α affects the correlation between the results of the system and that of human judgement.

The second similarity function is inverse logarithmic, that is, it amplifies the smaller values. It is shown in Equation 4. The *norm* function simply multiplies the result by a constant (i.e., $-log_2(\alpha)$) in order to move the value of the result in the range [0,1]. Note that the *norm* function does not affect the correlation score.

$$|w_1, w_2|_{log} = norm(\frac{-1}{log_2(min(\alpha, \frac{w_1 \rightarrow_s w_2 + w_2 \rightarrow_s w_1}{2}))}) \quad (4)$$

Given two nodes, the similarity between them is computed by performing a depth-first traversal of the graph from each node in parallel. Common nodes between the two traversals identify paths between the two nodes. When the weight of a path becomes under 0.0001, we prune the path. We do this in order to make the algorithm more efficient. Paths with weight under 0.001 will have little effect on the semantic similarity score. In our experimental results we only consider path of lengths 100 edges or less. In other words, we cap the depth-first traversal algorithm on both ends to 50 edges. A path with length of more than 100 edges will provide little evidence about the relationship between two phrases.

Note that we take the average of the two directional similarity distances in order to determine the similarity score. Empirical observations have shown that multiplying the two numbers is an inferior approach because often one of the two numbers is very small. For example, consider trying to compute the similarity distance between the words “ostrich” and ”animal”. One should hope this score to be high because the two words are clearly related. However, the directional similarity between the words “animal” and “ostrich” is low because there is very little evidence that someone who is interested in learning about an animal is interested in exactly the ostrich.

V. Experimental Results

The system consists of the two programs: one that creates the similarity graph and one that queries the similarity graph. We used the Java API for WordNet Searching (JAWS) to connect to WordNet. The interface was developed by Brett Spell ([27]). All experiments were performed on a Silicon Graphics UV10 Linux machine. It takes about 16 hours to build the similarity graph and save it to the hard disk. The similarity graph is saved in several hash tables and its total size is about 10GB. It takes about 40 minutes to load the similarity graph back into main memory. The average time for computing the similarity distance between two phrases once the graph is loaded into main memory is about 45 seconds. Of course, this time can be made faster if we decrease the maximum length of a path to be below 100 edges. However, this may result in less accurate results.

We used the similarity graph to compute the similarity between 28 pairs of words from the Miller and Charles study ([19]). The study presented the words to humans and computed the mean score of the human ranking. As Table I suggests, the correlation drops as the value of α increases.

Table I shows the result of the correlation with different values for α. Table II show how our results compare with other proposals for extracting semantic similarity between phrases. The results are for $\alpha = 0.1$. As the table suggests, both our algorithms produce better results (i.e., closer correlation with the results from the human judgement experiment in [19]) than existing algorithms. We also outperform our previous algorithm that uses information only from WordNet ([28]).

α	$\lvert\cdot\rvert_{lin}$	$\lvert\cdot\rvert_{log}$
0.1	0.93	0.93
0.2	0.88	0.90
0.3	0.85	0.85
0.4	0.80	0.80
0.5	0.75	0.75
0.6	0.70	0.68
0.7	0.68	0.59
0.8	0.65	0.49
0.9	0.63	0.34
1.0	0.62	0.18

TABLE I
Correlation results for different values of α on the Millers and Charles benchmark.

We explore how the coefficient α affects the quality of the result. We get the highest correlation with the results from the Miller and Charles study ([19]) when α is equal to 0.1. The correlation score is 0.93 for both the linear and logarithmic algorithms. A correlation score of 0.93 shows very close correlation between the results that were produced by our system and the data from the human judgement in the Miller and Charles study. To the best of our knowledge, this is the highest correlation with the study ever achieved in published research.

algorithm	*correlation*
Hirst and St-Onge ([8])	0.74
Leacock and Chodorow ([15])	0.82
Resnik ([22])	0.77
Jiang and Conrath ([11])	0.85
Lin ([16])	0.83
Stanchev ([28])	0.87
$\lvert\cdot\rvert_{lin}$	0.93
$\lvert\cdot\rvert_{log}$	0.93

TABLE II
Correlation results with the Millers and Charles benchmark.

In order to avoid overfitting, we decided to check if similar results hold for a different benchmark. In particular, we used the WordSimilarity-353 dataset ([5]). It contains 353 phrase pairs. Thirteen humans were used to rate the similarity between each pair of phrases and give a score between 1 and 10 (10 meaning that the phrases have the same meaning and 1 meaning that the phrases are unrelated). The average similarity rating for each word pair was recorded. Table III shows the correlation of our linear and logarithmic algorithms and different values of α with the results from the WordSimilarity-353 benchmark.

α	$\lvert\cdot\rvert_{lin}$	$\lvert\cdot\rvert_{log}$
0.1	0.54	0.53
0.2	0.52	0.53
0.3	0.52	0.52
0.4	0.51	0.52
0.5	0.50	0.49
0.6	0.49	0.44
0.7	0.46	0.40
0.8	0.45	0.35
0.9	0.43	0.31
1.0	0.42	0.18

TABLE III
Correlation results for different values of α on the WordSimilarity-353 benchmark.

Table IV shows how our system compares with eight existing systems that have documented their performance on the WordSimilarity-353 benchmark. The results of our system are for $\alpha = 0.1$. As the table shows, our system produces better results then all other systems. We also outperformed our previous algorithm that uses information only from WordNet ([28]). Note that some algorithm (e.g., [2]) use additional information from the web, while our algorithm only uses information from WordNet, Wikipedia, and data from University of Oxford’s British National Corpus ([3]) that contains the frequency of use of each word in the English language. Although computing the degree of similarity between phrases is not the main application of the similarity graph, the experimental results

algorithm	*correlation*
Jarmasz ([9])	0.27
Hirst and St-Onge ([8])	0.34
Jiang and Conrath ([11])	0.34
Strube and Ponzetto ([30])	0.19-0.48
Leacock and Chodrow ([15])	0.36
Lin ([16])	0.36
Resnik ([22]	0.37
Stanchev ([28])	0.49
Bollegala et al. ([2])	0.50
$\lvert\cdot\rvert_{lin}$	0.54
$\lvert\cdot\rvert_{log}$	0.53

TABLE IV
CORRELATION RESULTS WITH [5]

give us confidence about the quality of the data in the graph.

VI. CONCLUSION AND FUTURE RESEARCH

In previous work, we have created a similarity graph from WordNet data. In this work, we extended this similarity graph to include data from Wikipedia. As a result, we can now process phrases, such as "United Nations" and "National Hockey League", and not just simple word forms that are found in WordNet.

There are numerous applications of the similarity graph, where the most obvious application is semantic search. We can present to the user the documents that they are interested in based on phrase similarity. Similarity, the similarity graph can be used to find documents that are semantically similar to an input document and for document clustering. We verified the data quality of the similarity graph by showing that it can be used to compute the semantic similarity between phrases and we experimentally verified that our algorithms produce results of better quality than existing algorithms on the Charles and Miller and WordSimilarity-353 benchmarks. We believe that we outperformed existing algorithms because our algorithms processes not only structured data, but also natural language.

Our plan for future research is to use the similarity graph to create a suit of semantic applications. We believe that the similarity graph can be used to not only find data that cannot be found by performing keyword search, but it can also help us achieve good ranking of the query result based on semantic relevance.

REFERENCES

[1] OWL Web Ontology Language Guide. *http://www.w3.org/TR/owl-guide/*.

[2] D. Bollegala, Y. Matsuo, and M. Ishizuka. A Relational Model of Semantic Similarity Between Words Using Automatically Extracted Lexical Pattern Clusters from Web. *Conference on Empirical Methods in Natural Language Processing*, 2009.

[3] L. Burnard. Reference Guide for the British National Corpus (XML Edition). *http://www.natcorp.ox.ac.uk*, 2007.

[4] R. L. Cilibrasi and P. M. Vitanyi. The Google Similarity Distance. *IEEE ITSOC Inforamtion Theory Workshop*, 2005.

[5] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing Search in Context: The Concept Revisited. *ACM Transactions on Information Systems*, 20(1):116–131, January 2002.

[6] C. Fox. Lexical Analysis and Stoplists. *Information Retrieval: Data Structures and Algorithms*, pages 102–130, 1992.

[7] W. Frakes. Stemming Algorithms. *Information Retrieval: Data Structures and Algorithms*, pages 131–160, 1992.

[8] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *Fellbaum*, pages 305–332, 1998.

[9] M. Jarmasz. Roget's Thesaurus as a Lexical Resource for Natural Language Processing. *Master's thesis, University of Ottawa*, 1993.

[10] G. Jeh and J. Widom. SimRank: A Measure of Structural-context Similarity. *Proceedings of the Eight ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 538–543, 2002.

[11] J. Jiang and D. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Proceedings on International Conference on Research in Computational Linguistics*, pages 19–33, 1997.

[12] K. Jones. "a statistical interpretation of term specificity and its application in retrieval". *Journal of Documentation*, 28(1):11–21, 1972.

[13] R. Knappe, H. Bulskov, and T. Andreasen. Similarity Graphs. *Fourteenth International Symposium on Foundations of Intelligent Systems*, 2003.

[14] S. Kulkami and D. Caragea. Computation of the Semantic Relatedness Between Words Using Concept Clouds. *International Conference of Knowledge Discovery and Information Retrieval*, 2009.

[15] C. Leacock and M. Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. *WordNet: An electronic lexical database*, pages 265–283, 1998.

[16] D. Lin. An Information-theoretic Definition of Similarity. *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, 1998.

[17] J. B. MacQueen. Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, page 281297, 1967.

[18] M.F.Porter. An Algorithm for Suffix Stripping. *Readings in Information Retrieval*, pages 313–316, 1997.

[19] G. Miller and W. Charles. Contextual Correlates of Semantic Similarity. *Language and Congnitive Processing*, 6(1):1–28, 1991.

[20] R. Pan, Z. Ding, Y. Yu, and Y. Peng. A Bayesian Network Approach to Ontology Mapping. *Proceedings of the Fourth International Semantic Web Conference*, 2005.

[21] J. Pearl. Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning. *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine, CA.*, page 329334, 1985.

[22] P.Resnik. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. *International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

[23] R. Rada, H. Mili, E. Bickness, and M. Blettner. Development and Application of a Metric on Semantic Nets. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(1):17–30, 1989.

[24] Q. Rajput and S. Haider. Use of Bayesian Networks in Information Extraction from Unstructured Data Sources. *Proceedings of International Conference on Ontological and Semantic Engineering*, pages 325–331, 2009.

[25] Simone Paolo Ponzetto and Michael Strube. Deriving a Large Scale Taxonomy from Wikipedia. *22nd International conference on Artificial intelligence*, 2007.

[26] E. Sirin and B. Parsia. SPARQL-DL: SPARQL Query for OWL-DL. *3rd OWL: Experiences and Directions Workshop (OWLED)*, 2007.

[27] B. Spell. Java API for WordNet Searching (JAWS). *http://lyle.smu.edu/ tspell/jaws/index.html*, 2009.

[28] L. Stanchev. Building Semantic Corpus from WordNet. *The First International Workshop on the role of Semantic Web in Literature-Based Discovery*, 2012.

[29] M. Steyvers and J. Tenenbaum. The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. *Cognitive Science*, 29(1):41–78, 2005.

[30] M. Strube and S.P.Ponzetto. Wikirelate! Computing Semantic Relatedness using Wikipedia. *Association for the Advancement of Artificial Intelligence Conference*, 2006.

[31] Z. Wu and M. Palmer. Verb semantics and lexcial selection. *Annual Meeting of the Association for Computational Linguistics*, pages 133–138, 1994.

[32] D. Yang and D. M. Powers. Measureing Semantic Similarity in the Taxonomy of WordNet. *Australian Computer Science Conference*, pages 315–322, 2005.