A Comparison Analysis of Sentence-to-Sentence Similarity

Jaakko Tervonen & Malo de Rivals-Mazéres

Abstract

We perform a comparison analysis of WordNet, YAGO, DBPedia and Word2Vec concepts used to measure sentence-to-sentence analysis. We compare the concepts and how they comply with human judgement with our proposed dataset. Lastly we compare the concepts vs. human judgement with a benchmark dataset.

1. Introduction

Words are semantically similar if they share a relationship, for example coffee and tea are semantically similar because they are both a kind of hot beverage. Sentence similarity is based on word similarity, since sentences can be considered as sets of words; sentence similarity measures the similarity of group of terms. Its computation is an important subtask of many Natural Language Processing tasks, for example text summarization and categorization, machine translation and Web page retrieval [1]. Several different ways of calculating sentence similarity have been proposed and in our project for Natural Language Processing course we looked at some of these ways and how they correlate with human judgement of sentence similarity.

To begin with, we created a database of sentence pairs in decreasing order of similarity. Then we used WordNet [2], YAGO and DBPedia concepts [3] and Word2Vec embeddings through the *gensim*-package for Python [4] to calculate the sentence pair similarities. Through visual inspection we compared the produced sentence pair similarities to human judgement and to each other. To extend on the project description and for further comparison we also analysed another dataset of sentence pairs, conducted by O'Shea et al [1, 5].

We decided to focus mostly on the following three questions:

1. Is there any observable difference in sentence similarity between human judgement and theory-based similarities and if so, what kind of difference is it?
2. Do different sentence similarity metrics produce different results?
3. Do different metrics produce better compatibility with human judgement if human judgement is based on several individuals?

2. Description of data used

2.1. Creation of Own Database of Sentences

According to project description, the database was supposed to contain sentence pairs in decreasing order of similarity. The first pair should consist of a pair of exactly the same sentence, the second with almost the same sentence etc., ending up with completely unrelated sentences.

Our proposed dataset contains fifty sentence pairs ordered in decreasing order of similarity under our judgement and organized in five clusters with both inter-cluster and within-cluster decreasing pattern. The cluster descriptions are as follows:

i. Sentences with very similar meaning but with slight changes such as different wording, synonym usage or different adjectives.
ii. Sentences with somewhat similar words or synonyms used but with slightly different meaning, with end-of-cluster sentences having more different meaning.
iii. Sentences about the same subject but with different words and meanings.
iv. Sentences on different subjects but with some same or similar words.
v. Completely unrelated sentences.

The proposed dataset is included in the appendix of this report.

2.2. The Dataset Conducted by O'Shea et al.

To extend the project and for further comparison we experimented on a pilot benchmark dataset for evaluating algorithms designed to measure short text semantic similarity [1,5]. This data consists of 65 sentence pairs with similarity evaluations by 32 native speakers. The data contains the average and standard deviation of evaluations in scale from 0 to 4 thus permitting the calculation of Pearson correlation coefficient in addition to visual inspection.

In order to make this data comply with ours, we reorganized the sentences into decreasing order of similarity and transformed the similarity value to the interval from 0 to 1 by dividing each similarity by 4.

3. Methodology

3.1. WordNet

WordNet is a large lexical database of English, in which words are grouped into synonym sets (synsets), each expressing a distinct concept [2]. Synsets are linked into a conceptual-semantic and lexical relational hierarchy. There are several WordNet based word-similarity measures, of which we implemented two: the *path similarity* and *Wu & Palmer similarity*.

Path similarity is defined as shortest path across the hierarchy between two concepts, i.e.

$$\mathrm{sim}(\mathrm{w}_1, \mathrm{w}_2) = \frac{1}{\mathrm{length}\big(\mathrm{shortest\,path}(\mathrm{w}_1, \mathrm{w}_2)\big)}$$

The Wu & Palmer similarity is defined as

$$sim(w_1, w_2) = \frac{2 \cdot depth(LCS)}{depth(w_1) + depth(w_2)},$$

where LCS is the least common subsumer of w_1 and w_2 which is the most specific concept that is an ancestor of both words and *depth* is the number of nodes from concept to the root node of the hierarchy.

Obviously, a word may belong to more than one synsets. If this is the case, NLTK's function for Wordnet word similarity chooses the first synset found for both words and uses that for similarity calculus. We, however, decided to calculate the similarity for all the synsets and take the maximum as measure of word-to-word similarity.

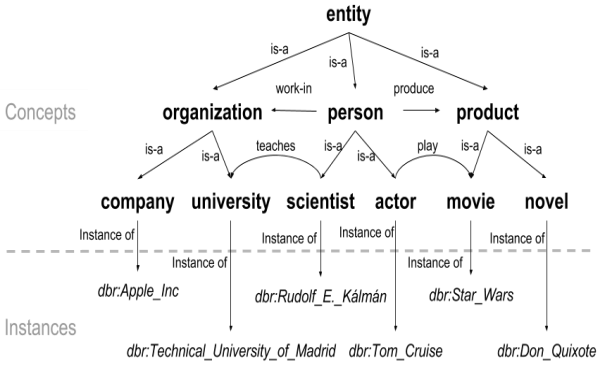
To transform word-to-word similarities into sentence-to-sentence similarities, we used the following formula given in project description. The similarity of sentence *S* and *T* is defined as

$$Sim(S,T) = \frac{1}{|S|}\sum_{s_i \in S} \underset{t_j \in T}{max}\, sim(s_i, t_j),$$

where s_i and t_j are the tokens of *S* and *T* and each individual similarity stands from some word-to-word similarity.

3.2. YAGO

The Yago concepts are originally created based on WordNet synsets. Therefore, the Yago concepts similarity methods extends the WordNet methodology by mapping Yago concepts to WordNet synsets.
Yago allows several ways of computing semantic similarity such as res, lin, jcn and wpath. Moreover, it is also possible to compute the information using a corpus-based or graph-based Information Content.



3.3. DBPedia

DBpedia has two different categories to identify the words: the concepts and the entities. Compared to WordNet it provides a larger set of terms, continuously updated by Wikipedia.

There are two methods to measure entity similarity. **Entity relatedness,** which is based on DBpedia link association, mainly measures entity link overlap between entities. **Entity similarity** is based on YAGO concept similarity: the Yago concepts of the entity are extracted from DBpedia and then the top 5 concepts are selected and returned as a list.

In opposition to Yago, Dbpedia has its own concept similarity class but since Yago concepts have corresponding mappings to WordNet synsets, Sematch reuses existing codes for computing semantic similarity of Yago concepts.

3.4. Word2Vec

In the fourth part of the project we used Word2Vec embeddings to determine the feature vector of each word (token) of each sentence in database; in other words, to express each word as vector of real numbers. The determined vectors form a vector space and thus word-to-word similarity can be taken to be the cosine similarity. Cosine similarity is defined as

$$sim(w_1, w_2) = \frac{\overline{w_1} \cdot \overline{w_2}}{\left|\left|\overline{w_1}\right|\right| \cdot \left|\left|\overline{w_2}\right|\right|},$$

where $\overline{w_1}, \overline{w_2}$ are the vector representations of words w_1 and w_2. Sentence-to-sentence similarity was defined as the average similarity of all the word-to-word similarities, as project description stated.

4. State of the Art and Implementation

For processing the sentences and making the required calculations we used Python version 2.7.13 with NLTK 3.2.5 [6]. The pre-processing tasks done with NLTK's predefined functions include dividing our database of sentence pairs into a two-dimensional list of lists, tokenization of sentences and removing stopwords.

NLTK contains a WordNet module which provides functions to determine word-to-word similarity under both path and Wu & Palmer similarity. To implement sentence-to-sentence similarity we wrote two functions, first of which takes care of word-to-word similarity calculation under any individual similarity measure and the second taking care of sentence-to-sentence similarity calculation by calling the first one. Both take the used individual similarity metric as input.

NLTK's functions for word-to-word similarity return "None" when no path between two words is found. If this is the case for all pairs$(s_i, t_{1,\dots,k})$ where s_i is a fixed token of sentence S and t_j goes through all the tokens of T, the word-to-word (or token-to-token) similarity value is taken to be 0.

For the Yago and DBpedia methods we took them from the sematch [3] package on github. This is a well implemented package with a lot of methods to help implement Natural Language Processing code and compute semantic similarity scores of entities, words and concepts.

Sematch contains a Yago module with functions extended from WordNet such as `YagoTypeSimilarity` which maps Yago concepts to WordNet synsets and computes their semantic similarity using WordNet taxonomy and either corpus-based or graph-based information content. As for the WordNet implementation, we wrote two functions with the same purpose, the main difference being the function of the module called inside of the function computing the semantic similarity between two concepts. Yago's function for word-to-word similarity also returns "None" when no path between two words is found. Similarity is then set as 0.

DBpedia is the second module from Sematch we used. We created four functions in order to compute the semantic similarities of all the pairs. The function `sent_dbpedia` first checks the entities in the first sentence and then the second one by verifying every word (after stop word removal). If it exists in the DBpedia database, then it calls `entity_similarity` that returns the semantic similarity between two given entities. As entry parameters it just enters twice the same word, if the result is "1", then we add this word to the entity list else we just disregard it.

Then the function checks for the concepts in the two sentences. This is way faster to do since it just has to check if a link exists in the database for the current concept. Sematch proposes us a function that returns a list of information for a given concept. If it returns a non-empty list of information, then it is added to the concept list. The concept list is then filtered through the `concept_link` function which picks up the link among the list of information that we will use as a parameter to compare this concept with one from the other sentence.

Once every word is checked as entities or concepts we then compute their semantic similarity with the function `dbpedia_similarity` which is basically the same as the one for WordNet and the one for Yago.

To create Word2Vec embeddings for sentence representation we took use of the gensim (version 3.1.0) Python package. Firstly, we used our own sentence pairs to train a Word2Vec model with one hundred features (or vector dimensions) and then used the determined model for sentence similarity calculus.

However, this model may produce biased results because of small corpus and the use of same corpus for both training and testing. It is also hard to compare between our sentence pair database and the

O'Shea one because both contain some distinct words. So, to extend the project and to be able to compare Word2Vec results between our database and the O'Shea dataset, we also used Google's pretrained Word2Vec model [7]. The model contains 300-dimensional vectors for three million words and phrases trained on part of Google News dataset. To be able to use this pretrained model for the O'Shea data, we were required to do some further preprocessing tasks to the sentences. These include removing a number 12 appearing in data, removing the genitive identifiers ('s) and changing the spelling of 'jewellery' to 'jewelry'.

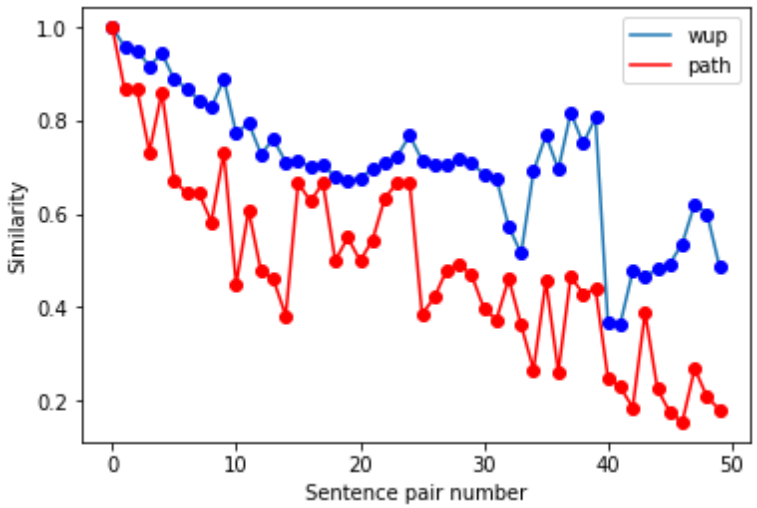
For our sentence pairs we didn't give any numerical value of similarity but only organized them in what we thought was decreasing order. Thus we couldn't use any numerical measure of correlation between different concepts and human judgement and we settled for visual inspection of behaviour of sentence similarity vectors. However, in the O'Shea data they gave the average value of similarity evaluated by all the individuals and so for those sentences we used both visual inspection and the Pearson correlation coefficient which is the instructed measure of correlation to be used [5].

For all the plots produced we used the Python plotting library Matplotlib (version 2.1.0).

5. Results and Discussion

5.1. WordNet

The WordNet based sentence-to-sentence similarities were calculated with two individual word-to-word metrics, the path similarity and Wu & Palmer similarity. Picture 1 shows the sentence-to-sentence similarity curve produced by both metrics, blue line meaning Wu & Palmer similarity and the red one for path similarity.



Picture 1. WordNet similarities of our sentence pairs database.

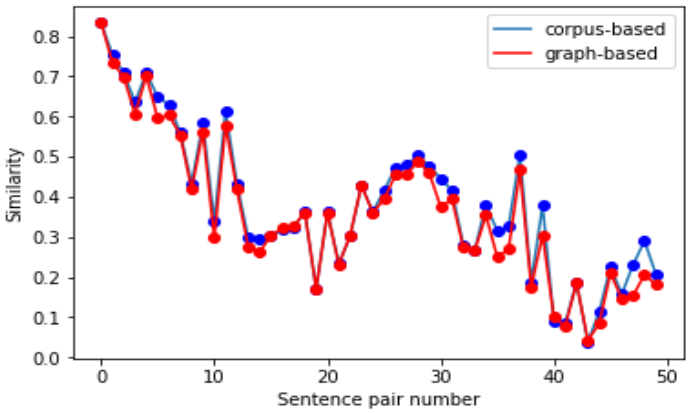
Overall both versions show a decreasing pattern. The decay is not monotonous, and Wu & Palmer similarity seems to produce a much smoother curve. Wu & Palmer also seems to produce higher values of similarity than path similarity, similarity ending up at around 40% at lowest.

Both metrics show an irregularity in similarity around sentence pairs number 35-39. This is around the fourth cluster of sentence pairs containing sentences with some same words but dissimilar meaning. The path similarity also shows big deviation around sentence pairs 10-17 which is not the case for Wu & Palmer similarity.

This irregularity may rise from the fact that both metrics use word-to-word similarities as base for sentence-to-sentence similarity calculation. As the fourth cluster sentence pairs contain same words, it is to be expected that metrics produce higher scores. However, sentence pairs do not share a common subject, so we think that they are more dissimilar than sentences in cluster three, which are from the same subject even if there are no same words used.

5.2. Yago similarities

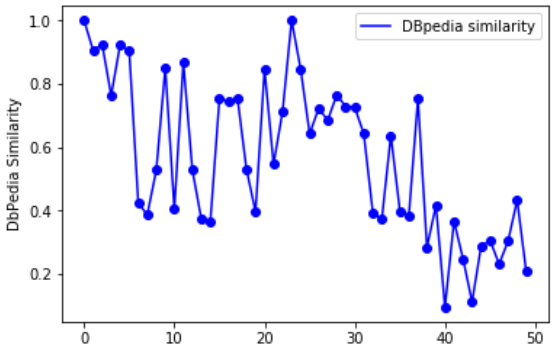
We calculated the Yago similarities with both the corpus-based and the graph-based information content model. As we can see on the graphs both the methods returned almost the same results. Results that are, as in WordNet, globally decreasing but pair-by-pair not really consistent. The main difference with WordNet is that from the pairs 20 to 29 and the last 7 sentences all are increasing. Both versions also show some irregularity around the fourth cluster.



Picture 2. Yago similarities of our sentences pairs database

5.3. DBpedia similarities

The DBpedia model is without a doubt the one with the worst results. As we can see on the picture 3, on average it slowly decreases but all the values are presented in a very spiky way. The 25th pair even has a similarity of 1 where it should be about 0.5. However, these results doesn't come as a shock since we already explained in the state of the art part how it is implemented. Indeed, given that we weren't able to choose between entity or concept for each word we just tried to see if they exist as a concept and an entity and then analyze the similarity. So some words appear as both entity and concept, some other as concept only when they should be entity, etc.

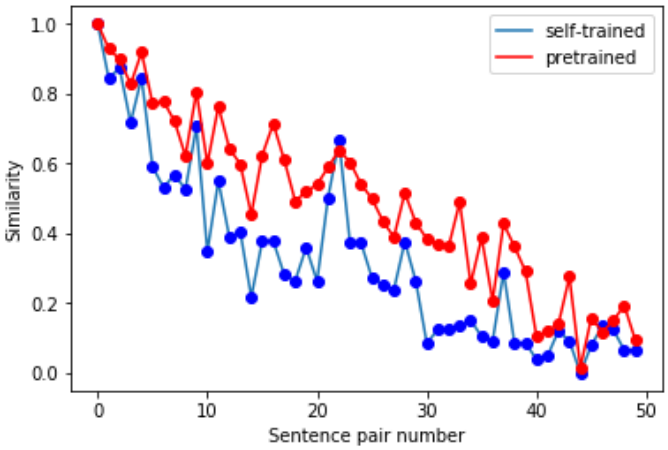


Picture 3. DBpedia similarities of our sentences pairs database

5.4. Word2Vec

We calculated the Word2Vec similarities with both the self-trained model and Google's pretrained one. As discussed before, using the self-trained model for calculating similarities of the sentence pairs used to train the model may produce biased results which is why the pretrained model was used already at this stage.

In Picture 4 we see the produced similarity curves, with blue line for self-trained model and red line for the pretrained one. Globally, both models manage to produce a decreasing curve. The self-trained model produces lower similarities than the pretrained model. Both graphs are quite spiky (a spike occurs when the curve changes up or down direction) but the spikes in the self-trained model are bigger. Both show some turbulence around sentence pairs 10-17 which we saw with WordNet path similarity and both versions of Yago too. As seen before,



Picture 4. Word2Vec similarities of our sentence pairs database.

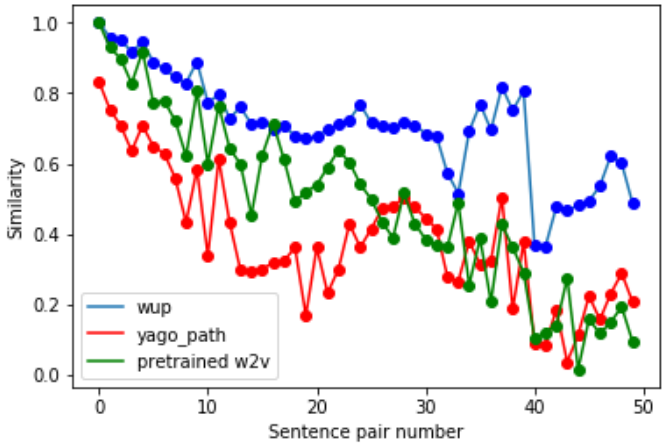
the pretrained model also shows the turbulence around sentence pairs 35-39 which the self-trained model manages to avoid – this may, however, be due to too small a training set of words.

Both curves show a slightly increasing pattern in the end on the fifth cluster. The similarities of the fifth cluster pairs seem to deviate a little but that's what's to be expected since the fifth cluster contains unrelated sentences. However, both models manage to show small similarities for the fifth cluster sentences.

5.5. Comparison of Concepts

To compare between all the concepts used we chose one metric/model for each concept. In WordNet based sentence similarity the Wu & Palmer word-to-word similarity produced a smoother curve so we chose that. The two YAGO similarities used had no big difference so we chose the corpus-based similarity. For the Word2Vec similarity we decided on using the pretrained model, for better comparability later with the O'Shea dataset.

In Picture 5 on the right we have all the used concepts in the same picture, with blue line for Wu & Palmer similarity, red line for YAGO corpus-based similarity and green line for the Google's pretrained Word2Vec model similarities. Overall, all the metrics used were able to capture the decreasing pattern.



Picture 5. Comparing the used concepts with our sentence pairs.

However, the actual values have some differences. Wu and Palmer similarity seems to give higher similarity scores than the other two and YAGO gives mostly higher values than Word2Vec model. All the metrics show the previously mentioned turbulence around the fourth cluster of sentences and all but Wu & Palmer show the mentioned turbulence around sentence pairs 10-17. This kind of behaviour can be seen throughout the curve: Wu & Palmer shows a much smoother curve than the other two but it also shows much higher similarity scores even with totally unrelated sentences.

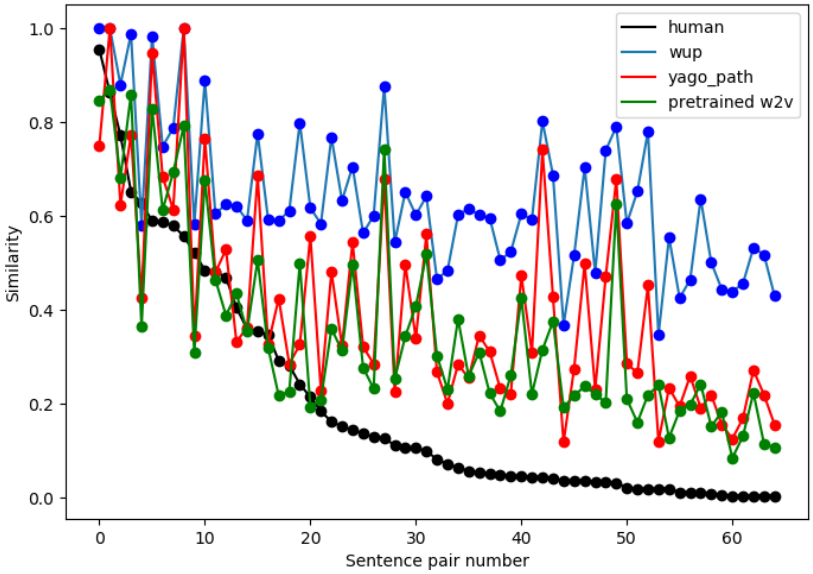
We think that Wu & Palmer giving high similarity scores (which we saw here and in subchapter 5.1.) may be related to the way the individual word-to-word similarities were combined. As project description stated and was explained before in chapter 3.1., we averaged the sentence similarity on maximums of token-by-token similarities. Better results might be obtained if sentence similarity was taken to be the mean of means of token-by-token similarities. We also chose to take the maximum of similarities when tokens belonged to several synsets. This could be improved for example with part-of-speech tagging or word-sense disambiguation, to select the correct synset for each token. However, in a meeting with project supervisor we were told that this is not necessary for this project and is therefore something we didn't do; this is something that can be looked at in future work on this subject.

5.6. Comparison with O'Shea Data

In this subchapter we redo the calculations done in subchapter 5.5. and compare the results. Since the O'Shea data contained numerical values for human judgement of sentence similarity, we also use the Pearson correlation coefficient to measure how well different metrics comply with human judgement. Results are shown in Table 1. Picture 6 on the next page shows the similarity curves produced by

different concepts together with human judgement. The black line is for human judgement and other lines as in Picture 5.

Overall, it seems that human judgement tends to give lower similarities than any of the sentence-to-sentence similarities used. Out of 65 sentence pairs, human judgement always gives the lowest score of similarity starting from pair 21 and from around halfway human judgement gives lower scores for all the sentences than any of the metrics used give for any sentence. Again, we see that Wu & Palmer similarity gives higher scores than other metrics and it also produces slightly smoother graph than the other two metrics. All the metrics, however, show much more turbulence than they did with our dataset: the graphs never seem to even out at all.



Picture 6. Comparing concepts with the O'Shea dataset.

Pearson correlation shows quite the same value for first two metrics, Wu & Palmer having a little lower score than corpus-based YAGO. The pretrained Word2Vec model has clearly the highest score of 0.774, which suggest that the Word2Vec model correlates better with human judgement than other two.

Concept/metric	cor(human judgement, metric)
WordNet Wu & Palmer	0.667
Corpus-Based YAGO	0.690
Google's pretrained Word2Vec model	0.774

Table 1. Pearson correlation for each concept.

6. Conclusions

All the used sentence-to-sentence similarity calculation methods managed to capture the decreasing pattern which we had under human judgement. Some performed better than others, but they all produced somewhat similar results. All the metrics used seemed to deviate a lot: we think that they may find connections that either do not really exist or are not seen by humans.

Used similarity measures had much more deviation with the O'Shea dataset compared to ours. This may be due to almost exponential decrease in similarity under human judgement in the O'Shea data: the sentence pairs similarity lowers down very early and the differences in similarity under human judgement are very small. Human judgement also depends strongly on how much though evaluators give.

No metric proved to be the clearly the best. They all showed good correlation with human judgement, but none complied fully. Having more evaluators did not enhance the results. Since human judgement is not universal, it is to be expected that no sentence-to-sentence similarity measure works in a way that it always predicts what people on average think of sentence similarity. However, all the implemented methods managed to do this to some extent.

7. References

1. Li, Y., et al., Sentence Similarity Based on Semantic Nets and Corpus Statistics. IEEE Transactions on Knowledge and Data Engineering, 2006. 18(8): p. 1138-1150.
2. Princeton University "About WordNet." WordNet. Princeton University. 2010. http://wordnet.princeton.edu
3. YAGO and DBPedia concepts: https://github.com/gsi-upm/sematch - cited 11/16/2017
4. *gensim* Python package: https://radimrehurek.com/gensim/ - cited 11/16/2017
5. O'Shea, James & Bandar, Zuhair & Crockett, Keeley & McLean, David. (2009). Pilot Short Text Semantic Similarity Benchmark Data Set: Full Listing and Description.
6. The NLTK Python package: http://www.nltk.org/ - cited 16/11/2017
7. Google's w2v model: https://code.google.com/archive/p/word2vec/ - cited 16/11/2017

APPENDIX – proposed dataset of sentence pairs

1. A young boy with brown hair walked to the school.A young boy with brown hair walked to the school
2. A young boy with brown hair walked to the school.A young boy with black hair walked to the school
3. A young boy with brown hair walked to the school.A tall boy with brown hair walked to the school
4. A young boy with brown hair walked to the school.A tall boy with black hair walked to the school
5. A young boy with brown hair walked to the school.A young brown haired boy walked to the school
6. A young boy with brown hair walked to the school.A short male kid with black hair walked to the school
7. A young boy with brown hair walked to the school.A young black haired boy went to the school on foot
8. A young boy with brown hair walked to the school.A young freckled boy went to the school on foot
9. A young boy with brown hair walked to the school.A boy walked to the school
10. A young boy with brown hair walked to the school.A young brown haired boy walked home
11. A young boy with brown hair walked to the school.A girl walked to the school
12. A young boy with brown hair walked to the school.A girl with brown hair biked to the school
13. A young boy with brown hair walked to the school.A bald girl ran to the school with a boy
14. A young boy with brown hair walked to the school.A long haired girl held hands with a boy as they walked
15. A young boy with brown hair walked to the school.An old man walked home
16. John said that they will begin evaluating the exams next week.John said that test evaluation will be started by Wednesday
17. John said that they will begin evaluating the exams next week.John told that evaluation of examinations will begin tomorrow
18. John said that they will begin evaluating the exams next week.John told that grading the examinations will start today
19. John said that they will begin evaluating the exams next week.John claimed that test assessment is done during the weekend

20. John said that they will begin evaluating the exams next week.The exams will be evaluated by next lecture
21. John said that they will begin evaluating the exams next week.According to John the exam scores are available online after the weekend
22. John said that they will begin evaluating the exams next week.It is hard to begin evaluating exams on time
23. John said that they will begin evaluating the exams next week.They said that the person evaluating the exams is called John
24. John said that they will begin evaluating the exams next week.John read that exams must be evaluated within three weeks of exam day
25. John said that they will begin evaluating the exams next week.John told that he has evaluated exams since high school
26. The weather forecast promised sunshine but all we got was clouds and rain.The meteorologist was mistaken in telling it would rain
27. The weather forecast promised sunshine but all we got was clouds and rain.The weather was hot yesterday
28. The weather forecast promised sunshine but all we got was clouds and rain.The clouds may stop the Sun from shining
29. The weather forecast promised sunshine but all we got was clouds and rain.Stormy weather prevents sunshine
30. The weather forecast promised sunshine but all we got was clouds and rain.Climate change makes the weather change more rapidly
31. The weather forecast promised sunshine but all we got was clouds and rain.People usually prefer the Sun to drizzle
32. The weather forecast promised sunshine but all we got was clouds and rain.In England it usually rains more often than not
33. The weather forecast promised sunshine but all we got was clouds and rain.In Finland we do not get much Sun in winter
34. The weather forecast promised sunshine but all we got was clouds and rain.In Finland it is always cold and cloudy in winter no matter the forecasts
35. The weather forecast promised sunshine but all we got was clouds and rain.The sea currents around the globe have significant effect on air temperature
36. The young like playing computer games.John has played football since he was a child
37. The young like playing computer games.Mary enjoys drawing pictures of flowers
38. The young like playing computer games.It is important to know how to use a computer
39. The young like playing computer games.Mary has played the violin for years
40. The young like playing computer games.I bought a new laptop last summer
41. I have some work to do this evening.We grow wheat here
42. I made up my mind to study harder. Did you hear the news on the radio
43. They advised me to go the police station.It is time to sleep
44. I can understand why you do not want to eat there.Take a deep breath please
45. He repaired his watch by himself.She has the same bag as you do
46. John could not answer the last question.I find this book boring
47. John lived in Japan for ten years.Deal us the cards
48. John went horse back riding at dawn.Sundown is a magnificent view
49. Picasso is a great painter.Mother cooks all the time
50. Programming is essential skill to computer scientists.Mary has a little lamb