Semantic Textual Similarity

MINOR PROJECT I

Submitted by:

Mrinal Shukla - 9917103040

Prashant Sahrawat - 9917103053

Lovelish Jain - 9917103057

Under the supervision of
Mrs. Akanksha Mehndiratta

विद्या तत्व ज्योतिसम:

Department of CSE/IT
Jaypee Institute of Information Technology University, Noida

October 2019

ACKNOWLEDGEMENT

We would like to state our deep sense of gratitude to Mrs. AKANKSHA MEHNDIRATTA, Asst. professor CSE and IT department, Jaypee Institute of Information Technology, Noida for her generous guidance, help and useful suggestions.

She was extremely supportive and helped us in every way possible. Her insightful comments and constructive suggestions helped us improve the quality of this project work.

Signature(s) of Students

Mrinal Shukla (9917103040)

Prashant Sahrawat (9917103053)

Lovelish Jain (9917103057)

ABSTRACT

Semantic textual similarity measures the similarity of meaning between two sentences. STS is used in computational linguistics, it has important application in NLP, due to wide application range of STS in many fields there is a constant demand for new methods as well as improvement in current methods. This project provides an unsupervised learning-based approach for measuring the semantic similarity of texts. There have been a large body of work focussed on finding semantic similarity using word (embedding) based approaches We introduced two approaches based on Canonical correlation analysis (CCA), one of which uses cosine similarity as calculation metric and other uses Word Mover's Distance (WMD), these models have the potential to outperform the traditional unsupervised learning methods.

Table of Contents

Page No.

Chapter 1 - INTRODUCTION

1.1 Overview-

Semantic Textual Similarity (STS) measures the degree of semantic equivalence between two snippets of text. Measuring text similarity have been used for a long time in applications in natural language processing (NLP) and related areas. Text similarity has been used for machine translation, text summarization, semantic search, word sense disambiguation and many more. While making such an assessment is trivial for humans, making algorithms and computational models that mimic human level performance poses a challenge.

The degree of semantic similarity between two text snippets is graded on a scale from 0 to 5 with 5 being highly similar and 0 being highly dissimilar.

Our project is based on the SemEval-2017 Shared Task for Semantic Textual [1], SemEval is a competition held annually, to bring diverse approaches and improvements to state-of-the-art methods for semantic analysis.

Our objective is to calculate similarity scores as close as possible to the given values using unsupervised learning techniques.

Example 1:

English: Birdie is washing itself in the water basin.

English Paraphrase: The bird is bathing in the sink.

Similarity Score: 5

(The two sentences are completely equivalent, as they mean the same thing.)

Example 2:

English: The young lady enjoys listening to the guitar.

English Paraphrase: The woman is playing the violin.

Similarity Score: 1 (The two sentences are not equivalent, but are on the same topic.)

Chapter 2 – BACKGROUND STUDY

2.1 Word embeddings - Word embedding is a technique where words are mapped to real-valued vectors in a predefined vector space. Each word is mapped to one vector, often with tens or hundreds of dimensions. word vectors are positioned in vector space such that words that share common contexts are located close to one another in the space.

Various methods have been used for word embeddings such as –

2.1.1 Count Vectorizer -As the name suggests the word vector depends on the count of the word, first we Identify unique words in the complete text data, then for each sentence, we'll create an list of zeros with the same length, to find the vector of first word we will replace the count of the word at its position in the above list.

2.1.2 word2vec - Google's word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located close to one another in the space. This method was used by Mikolov et al., 2013a [2].

2.1.3 Glove: Global Vectors for Word Representation – This approach was proposed by Pennington et al., 2014 [3]. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Glove learns by constructing a co-occurrence matrix and the resulting representations showcase linear substructures of the word vector space.

2.2 Sentence embedding – Sentence embedding is a technique where sentences are mapped to real valued vector. Different techniques exist to compute sentence embedding by composing word embeddings using operation on vectors and matrices.

2.2.1 Bag of words – This method represents text as the bag of its words, each dimension represents the frequency of the word in the sentence.

2.2.2 word2vec aggregate – In this approach we find word embedding for each word in the sentence using Google's word2vec word embeddings and then their summation gives us the sentence vector.

2.2.3 Skip-Thought Vectors - This technique was proposed by Kiros et al., 2015 [4]. It is an unsupervised learning method for sentence embeddings. Its functioning is analogous to skip-gram model but it works on sentences. So just like skip-gram will predict surrounding words of a word, this model will predict the nearby sentences or phrases for a given sentence i.e. it predicts the neighbouring phrases using recurrent neural networks (RNN).

2.2.4 FastSent - Skip-thought vectors are slow to train. So FastSent method was developed to overcome this while keeping the core advantage i.e. better distributed representations are achieved by predicting the neighbouring sentences. FastSent makes the training more efficient by representing the sentences as the sum of word vectors of its words. This method was used by Hill et al., 2016a [5].

Chapter 3 – REQIUREMENT ANALYSIS

3.1 Software requirements –

- Anaconda navigator / Jupyter notebook
- Python version 3 or more
- Google's word embeddings

3.2 Hardware requirements –

- Computer with processor 2Ghz or more
- 4 GB RAM or more
- Minimum 5 GB disk space
- 2 GB GPU or more

3.3 Functional requirements –

- The project must provide the semantic similarity scores for every input provided to it. Pearson correlation coefficient of the model used to determine the semantic similarity

- The Project will generate Pearson correlation coefficient for every model, which will be useful for comparing different models.

- The result produced by the project will be usable for future study on STS.

3.4 Non-Functional requirements –

- The project will provide accurate Pearson correlation coefficient of the model employed.

- The code that is used to compute CCA embeddings should be accurate to give correct outcome.

- Dataset size should be kept to its best suitable for outcome analysis.

- Project should be error free and should be able to complete in the stipulated time for the project.

Chapter 4 -DETAILED DESIGN

4.1 Data Preparation –

4.1.1 - Dataset 1

We obtained the data from SemEval -2017 Task 1, **SemEval** (**Sem**antic **Eval**uation) is an ongoing series of evaluations of computational semantic analysis systems.

No of pairs: 250

4.1.2 - Dataset 2

Name – "OnWN", SemEval textual similarity dataset 2012

Description – Pair of sentences where the first comes from the Ontonotes and the second from wordnet definition, contains 750 sentence pairs with a rating between 0 – 5 with 0 indicating highly dissimilar and 5 being highly similar.

4.1.3 - Dataset 3

Name – "headline", SemEval textual similarity dataset 2014

Description – contains sentences taken from news headlines, contains 750 sentence pairs with a rating between 0 – 5 with 0 indicating highly dissimilar and 5 being highly similar.

4.2 Data Pre-processing –

We cleaned the data before we used it for calculating semantic similarity. We tokenized the sentences, removed punctuations from the them, replaced numbers to words and removed stop words, as removing them do not take away any semantic information.

4.3 Evaluation metric –

For the final score, we used Pearson correlation between predicted similarity and human annotated similarity. It has a value between +1 and −1, higher the score, the better the similarity prediction result.

4.4 Methodology –

4.4.1 Canonical Correlation Analysis (CCA) – Canonical correlation analysis is used to identify and measure the associations among two sets of variables. If we have two set of vectors x and y, then CCA will find linear combinations of x and y such that they have maximum correlation with each other. We will generate CCA embeddings for both input sentence and use them to calculate semantic similarity between sentences

- **Working of CCA –**

Consider two random variables x and y with zero mean.

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{bmatrix} = E\left[\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}^T \right]$$

The total covariance matrix is a block matrix where Cxx and Cyy are the within-sets covariance matrices of x and y respectively and Cxy = Transpose of Cyx is the between-sets covariance matrix. The canonical correlations between x and y can be found by solving the eigen-value equations

$$\begin{cases} \mathbf{C}_{xx}^{-1}\mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\hat{\mathbf{w}}_x = \rho^2\hat{\mathbf{w}}_x \\ \mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\mathbf{C}_{xx}^{-1}\mathbf{C}_{xy}\hat{\mathbf{w}}_y = \rho^2\hat{\mathbf{w}}_y \end{cases}$$

where the eigenvalues are the squared canonical correlations and the eigen-vectors wx and wy are the normalized canonical correlation basis vectors. The number of non-zero solutions to these equations are limited to the smallest dimensionality of x and y. E.g. if the dimensionality of x and y is 8 and 5 respectively, the maximum number of canonical correlations is 5. Only one of the eigenvalue equations needs to be solved since the solutions are related by

$$\begin{cases} \mathbf{C}_{xy}\hat{\mathbf{w}}_y = \rho\lambda_x\mathbf{C}_{xx}\hat{\mathbf{w}}_x \\ \mathbf{C}_{yx}\hat{\mathbf{w}}_x = \rho\lambda_y\mathbf{C}_{yy}\hat{\mathbf{w}}_y, \end{cases}$$

$$\lambda_x = \lambda_y^{-1} = \sqrt{\frac{\hat{\mathbf{w}}_y^T\mathbf{C}_{yy}\hat{\mathbf{w}}_y}{\hat{\mathbf{w}}_x^T\mathbf{C}_{xx}\hat{\mathbf{w}}_x}}.$$

4.4.2 Word Mover's Distance (WMD) - WMD is a method that allows us to assess the "distance" between two documents in a meaningful way, use word embeddings to calculate the distance so that it can calculate even though there is no common word. The assumption is that similar words should have similar vectors.

Chapter 5- IMPLEMENTATION

5.1 Data Pre-processing –

Data pre-processing is an important step in leaning process. To improve the overall performance, we performed these tasks.

- Tokenization – We broke the given sentences into list of words which were essential for creating word embeddings.
- Removing punctuations - We used regular expression to remove all the punctuations in the sentence and replaced them with empty strings because we can’t convert punctuation to vectors.
- Replacing numbers - We converted numerical values to their corresponding words, which can then be represented as vectors.
- Removing stop words - A stop word is a most commonly used word (such as “the”, “a”, “an”, “in”) that do not add any valuable semantic information to our sentence. we removed stop words from the sentence before we turn the data in to our models.

5.2 Methodology –

5.2.1 Canonical Correlation Analysis (CCA) Approach using cosine similarity–

First, we pre-processed the data then using Google’s word2vec word embeddings, then we created a matrix for each sentence where each row represented a word of the sentence. then applying CCA, first we fit the model on the two matrices then we transform the matrices using the model and we get vectors from each sentence such that correlation between them is maximum. The Number of vector pairs ranges from two to five, depending on the number of tokens in the sentence. Now we calculated cosine similarity for each of these vector pairs and took mean of their similarity. Finally, we scaled the output similarity to 5 using min-max normalization.

Cosine similarity –

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

min – max normalization –

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

ILLUSTARTION –

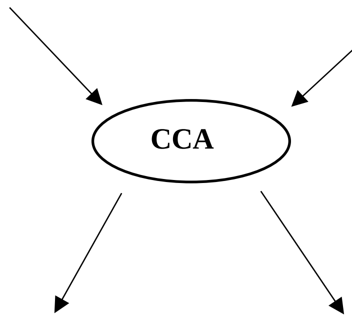
Sentence 1: the group is eating while taking in a breath-taking view.

pre-processed tokens: [‘group’, ‘eating’, ‘taking’, ‘breathtaking’, ‘view’]

Sentence 2: a group of people take a look at an unusual tree.

pre-processed tokens: [‘group’, ‘people’, ‘take’, ‘look’, ‘unusual’, ‘tree’]

[‘group’, ‘people’, ‘take’, ‘look’, ‘unusual’, ‘tree’**],** [**‘**group’, ‘eating’, ‘taking’, ‘breathtaking’, ‘view’**]**



			Cosine Similarity
1st iteration	group	group	S1
2nd iteration	taking	take	S2
3rd iteration	view	look	S3
4th iteration	breathtaking	unusual	S4
5th iteration	people	people	S5
		Overall Similarity -	$\sum S / 5$

5.2.2 Canonical Correlation Analysis (CCA) Approach using Word Mover's Distance (WMD)–

This approach is similar to the above mentioned one. First, we pre-processed the data then using Google's word2vec word embeddings, then we created a matrix for each sentence where each row represented a word of the sentence. then applying CCA, first we fit the model on the two matrices then we transform the matrices using the model and we get vectors from each sentence such that correlation between them is maximum. The Number of vector pairs ranges from two to five, depending on the number of tokens in the sentence. We mapped these vectors to their closest words in the word2vec plane. Now we calculated WMD distance between these words of both the sentences. Final similarity is calculated by subtracting the output by five.

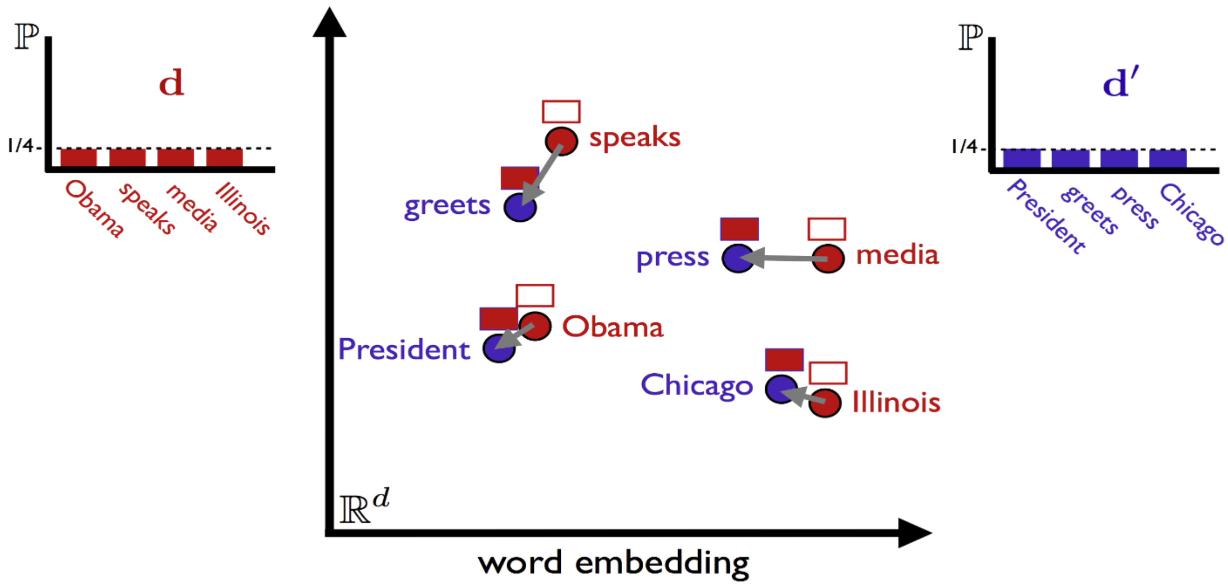
ILLUSTARTION –

Sentence 1: Obama speaks to the media in Illinois.

pre-processed tokens: ['Obama', 'speaks', 'media', 'Illinois']

Sentence 2: The president greets the press in Chicago.

pre-processed tokens: ['President', 'greets', 'Press', 'Chicago']



Chapter 6 – TESTING RESULTS

DATASET 1-

Model	**Pearson**
Simple Baseline	0.633
Published Baseline	0.698
CNN	0.646
LSTM	**0.810**

Our Results –

Model	**Pearson**
CCA using cosine similarity	0.737
CCA using WMD	0.769

DATASET 2- "OnWN", SemEval textual similarity dataset 2012

DATASET 3- "headline", SemEval textual similarity dataset 2014

Results collected from (Wieting et al., 2016 [9])

Dataset	50%	75%	Max	PP	proj.	DAN	RNN	iRNN	LSTM (no o.g.)	LSTM (o.g.)	ST	GloVe	PSL
MSRpar	51.5	57.6	73.4	42.6	43.7	40.3	18.6	43.4	16.1	9.3	16.8	**47.7**	41.6
MSRvid	75.5	80.3	88.0	**74.5**	74.0	70.0	66.5	73.4	71.3	71.3	41.7	63.9	60.0
SMT-eur	44.4	48.1	56.7	47.3	**49.4**	43.8	40.9	47.1	41.8	44.3	35.2	46.0	42.4
OnWN	608	65.9	72.7	**70.6**	70.1	65.9	63.1	70.1	65.2	56.4	29.7	55.1	63.0
SMT-news	40.1	45.4	60.9	58.4	**62.8**	60.0	51.3	58.1	60.8	51.0	30.8	49.6	57.0
STS 2012 Average	54.5	59.5	70.3	58.7	**60.0**	56.0	48.1	58.4	51.0	46.4	30.8	52.5	52.8
headline	64.0	68.3	78.4	72.4	72.6	71.2	59.5	**72.8**	57.4	48.5	34.6	63.8	68.8
OnWN	52.8	64.8	84.3	67.7	68.0	64.1	54.6	**69.4**	68.5	50.4	10.0	49.0	48.0
FNWN	32.7	38.1	58.2	43.9	**46.8**	43.1	30.9	45.3	24.7	38.4	30.4	34.2	37.9
SMT	31.8	34.6	40.4	39.2	**39.8**	38.3	33.8	39.4	30.1	28.8	24.3	22.3	31.0
STS 2013 Average	45.3	51.4	65.3	55.8	**56.8**	54.2	44.7	56.7	45.2	41.5	24.8	42.3	46.4
deft forum	36.6	46.8	53.1	48.7	**51.1**	49.0	41.5	49.0	44.2	46.1	12.9	27.1	37.2
deft news	66.2	74.0	78.5	**73.1**	72.2	71.7	53.7	72.4	52.8	39.1	23.5	68.0	67.0
headline	67.1	75.4	78.4	69.7	**70.8**	69.2	57.5	70.2	57.5	50.9	37.8	59.5	65.3
images	75.6	79.0	83.4	**78.5**	78.1	76.9	67.6	78.2	68.5	62.9	51.2	61.0	62.0
OnWN	78.0	81.1	87.5	78.8	**79.5**	75.7	67.7	78.8	76.9	61.7	23.3	58.4	61.1
tweet news	64.7	72.2	79.2	76.4	75.8	74.2	58.0	**76.9**	58.7	48.2	39.9	51.2	64.7
STS 2014 Average	64.7	71.4	76.7	70.9	**71.3**	69.5	57.7	70.9	59.8	51.5	31.4	54.2	59.5
answers-forums	61.3	68.2	73.9	**68.3**	65.1	62.6	32.8	67.4	51.9	50.7	36.1	30.5	38.8
answers-students	67.6	73.6	78.8	**78.2**	77.8	78.1	64.7	78.2	71.5	55.7	33.0	63.0	69.2
belief	67.7	72.2	77.2	**76.2**	75.4	72.0	51.9	75.9	61.7	52.6	24.6	40.5	53.2
headline	74.2	80.8	84.2	74.8	**75.2**	73.5	65.3	75.1	64.0	56.6	43.6	61.8	69.0
images	80.4	84.3	87.1	**81.4**	80.3	77.5	71.4	81.1	70.4	64.2	17.7	67.5	69.9
STS 2015 Average	70.2	75.8	80.2	**75.8**	74.8	72.7	57.2	75.6	63.9	56.0	31.0	52.7	60.0
2014 SICK	71.4	79.9	82.8	71.6	**71.6**	70.7	61.2	71.2	63.9	59.0	49.8	65.9	66.4
2015 Twitter	49.9	52.5	61.9	52.9	52.8	**53.7**	45.1	52.9	47.6	36.1	24.7	30.3	36.3

Our Results -

Dataset	CCA using cosine similarity	CCA using WMD
OnWN	60.5	37.1
headline	62.5	55.8

Chapter 7 – CONCLUSION AND FUTURE SCOPE

We proposed two unsupervised learning models namely CCA using cosine Similarity and CCA using WMD, we compared our models on three different datasets with various other models. Even though our model couldn't give best results it still performed better than some models and gave competitive results for others, which shows that there is a great scope for improvement. On further improvement the model will be helpful in various ways and can be used in applications such as document summarization, word sense disambiguation, short answer grading, information retrieval and extraction, etc.

References

[1] SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation.

[2] Mikolov, et. al**,** Distributed Representations of Words and Phrases and their Compositionality**.** *In Advances in Neural Information Processing Systems*, 2013a.

[3] Pennington et. al, GloVE: Global Vectors for Word Representation. *Proceedings of the Empirical Methods in Natural Language Processing,* 2014.

[4] Kiros et. al, Skip-Thought Vectors. *In Advances in neural information processing systems*, 2015.

[5] Hill et. al, Learning Distributed Representations of Sentences from Unlabelled Data. *In Proceedings of NAACL-HLT.*

[6] Cer et. al, SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation. *In Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017)*

*[*7] Maharjan et. al, DT Team at SemEval-2017 Task 1: Semantic Similarity Using Alignments, Sentence-Level Embeddings and Gaussian Mixture Model Output. *In Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017)*

[8] Dhillon et. al, *Eigenwords: Spectral Word Embedding*. *In journal of Machine Learning Research 2016*

[9] Wieting et. al, *TOWARDS UNIVERSAL PARAPHRASTIC SENTENCE EMBEDDINGS*. *Published as a conference paper at ICLR 2016*

POWERPOINT PRESENTATION

Semantic Textual Similarity

विद्या तत्व ज्योतिसम:

Under the supervision of
Mrs. Akanksha Mehndiratta

Submitted by:

Mrinal Shukla - 9917103040

Prashant Sahrawat - 9917103053

Lovelish Jain - 9917103057

- Inspired from **SemEval (Semantic Evaluation).**
- The evaluations are intended to explore the nature of meaning in language.
- ongoing series of evaluations of computational semantic analysis systems.
- Semantic analysis refers to a formal analysis of meaning, and computational refers to approaches that in principle support effective implementation in digital computers.
- Conducting bodies and sponsors
  - University of Southern California
  - University of Cambridge
  - Microsoft
- Tasks are declared, teams apply ,the dataset is released by the organizers, participating teams can then use these and implement their models ,teams submit the code and paper, the best performing models are highlighted in the paper and prize money sponsored for some tasks.

- Some of the challenges of the competition-
  - sentiment Analysis
  - word sense disambiguation
  - Semantic Textual similarity.
  - Twitter Analysis

- The extensive work in the field is due to the wide variety of applications it has.
  - document summarization
  - question answering - question-and-answer sites such as Quora or Stack overflow need to determine whether a question has already been asked before.
  - short answer grading
  - machine translation etc.

Problem Statement

- **Semantic Textual Similarity** -
- Given two sentences, it tells us how much they are similar in terms of meaning.
- Example
- Highliy similar -
- there is a cook preparing food, a cook is making food. → 5.0

- Moderately similar -
- the man is catching a ball, a man is kicking a ball. → 3.0

- Highly dissimilar -
- a kid is talking in class, dog and sheep run together. → 0.0

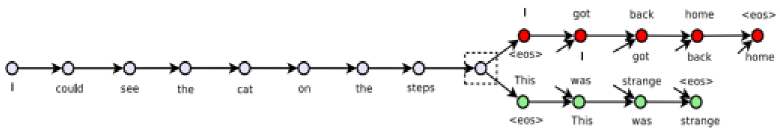
- There have been a large body of work focused on finding semantic similarity, constructing algorithms and computational models that mimic human level performance poses a challenge.

State of the art

Skip –Thought Vectors-

- It is an unsupervised learning method for sentence embeddings. Its functioning is analogous to skip-gram model but it works on sentences. So just like skip-gram will predict surrounding words of a word, this model will predict the nearby sentences or phrases for a given sentence using recurrent neural networks (RNN).

- Skip-thought vectors use the encoder-decoder model to first encode a sentence into a vector, then decode that representation into the surrounding sentences.



Limitations -

- Unlike the other methods, skip-thought vectors require the sentences to be ordered in a semantically meaningful way. This makes this method difficult to use for domains such as social media text, where each snippet of text exists in isolation.

- By virtue of it being a deep neural network model, it is also much slower to train than the other methods.

Objective

- Our objective is to calculate semantic similarity scores as close as possible to the human assigned values resulting in high using proposed unsupervised learning technique.

Work distribution

- Implementation by everyone
- Report done by Mrinal Shukla and Prashant Sahrawat
- Presentation work by Lovelish Jain and Mrinal Shukla
- Research paper by everyone

Proposed Solution

- We introduced two approaches based on Canonical Correlation Analysis (CCA)
  - CCA using cosine similarity
  - CCA using Word Mover's Distance(WMD)

Canonical Correlation Analysis(CCA)

- Canonical correlation analysis is used to identify and measure the associations among two sets of variables. If we have two set of vectors X = (X1, ..., Xn) and Y = (Y1, ..., Ym), then CCA will find linear combinations of X and Y such that they have maximum correlation with each other.

$X = \{X1 + X2 + X3 + . \ . + Xn\}$ $\qquad$ $Y = \{Y1 + Y2 + Y3 + . . + Ym\}$

CCA

a , b $\qquad$ (such that $\rho = \text{corr}(a^T X, b^T Y)$. is maximum)

Now, $U = a^T X$ and $V = b^T Y$ are the are first pair of canonical variables.

and correlation between U and V is maximum.

CCA using cosine similarity

- **Illustration** –
- **Sentence 1**: the group is eating while taking in a breath-taking view.
- **pre-processed tokens**: [‘group’, ‘eating’, ‘taking’, ‘breathtaking’, ‘view’]
- **Sentence 2**: a group of people take a look at an unusual tree.
- **pre-processed tokens**: [‘group’, ‘people’, ‘take’, ‘look’, ‘unusual’, ‘tree’]

- ['group', 'people', 'take', 'look', 'unusual', 'tree'], ['group', 'eating', 'taking', 'breathtaking', 'view']

CCA

-

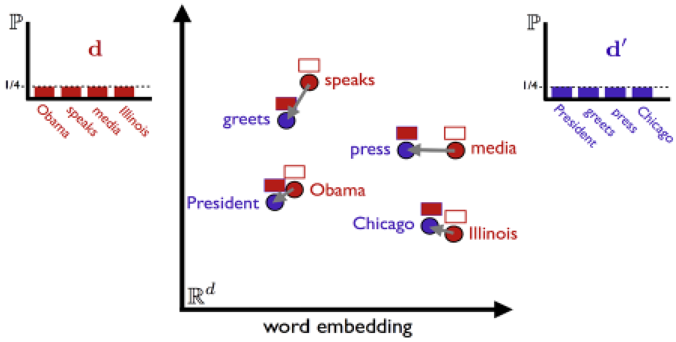
			Cosine Similarity
1st iteration	group	group	S1
2nd iteration	taking	take	S2
3rd iteration	view	look	S3
4th iteration	breathtaking	unusual	S4
5th iteration	people	people	S5

Overall Similarity - $\sum S/5$

- Predicted - **2.17**
- Actual - **2.2**

CCA using Word Mover's Distance(WMD)

- **Illustration** –
- **Sentence 1**: Obama speaks to the media in Illinois.
- **pre-processed tokens**: ['Obama', 'speaks', 'media', 'Illinois']
- **Sentence 2**: The president greets the press in Chicago
- **pre-processed tokens**: ['President', 'greets', 'Press', 'Chicago']



- Overall Similarity score= 5 -WMD
- Sentences dissimilar -> WMD ↑ -> Similarity score ↓
- Sentences similar -> WMD ↓ -> Similarity score ↑

Result and Analysis

- We implemented our models, on three datasets and compared with various other models the results are as follows-
- DATASET 1-

  no of pairs – 250 pairs

Model	Pearson correlation
Simple Baseline	0.633
Published Baseline	0.698
CNN	0.646
LSTM	0.810

- Our Results-

Model	Pearson correlation
CCA using cosine similarity	0.737
CCA using WMD	0.769

- DATASET 2
- Name – “OnWN”, SemEval textual similarity dataset 2012
- Description – Pair of sentences where the first comes from the Ontonotes and the second from wordnet definition, contains 750 sentence pairs with a rating between 0 – 5 with 0 indicating highly dissimilar and 5 being highly similar.

- DATASET 3-
- Name – “headline”, SemEval textual similarity dataset 2014
- Description – contains sentences taken from news headlines , contains 750 sentence pairs with a rating between 0 – 5 with 0 indicating highly dissimilar and 5 being highly similar.

Results collected from (Wieting et al., 2016)

Dataset	50%	75%	Max	PP	proj.	DAN	RNN	iRNN	LSTM (no o.g.)	LSTM (o.g.)	ST	GloVe	PSL
MSRpar	51.5	57.6	73.4	42.6	43.7	40.3	18.6	43.4	16.1	9.3	16.8	**47.7**	41.6
MSRvid	75.5	80.3	88.0	**74.5**	74.0	70.0	66.5	73.4	71.3	71.3	41.7	63.9	60.0
SMT-eur	44.4	48.1	56.7	47.3	**49.4**	43.8	40.9	47.1	41.8	44.3	35.2	46.0	42.4
OnWN	608	65.9	72.7	**70.6**	70.1	65.9	63.1	70.1	65.2	56.4	29.7	55.1	63.0
SMT-news	40.1	45.4	60.9	58.4	**62.8**	60.0	51.3	58.1	60.8	51.0	30.8	49.6	57.0
STS 2012 Average	54.5	59.5	70.3	58.7	**60.0**	56.0	48.1	58.4	51.0	46.4	30.8	52.5	52.8
headline	64.0	68.3	78.4	72.4	72.6	71.2	59.5	**72.8**	57.4	48.5	34.6	63.8	68.8
OnWN	52.8	64.8	84.3	67.7	68.0	64.1	54.6	**69.4**	68.5	50.4	10.0	49.0	48.0
FNWN	32.7	38.1	58.2	43.9	**46.8**	43.1	30.9	45.3	24.7	38.4	30.4	34.2	37.9
SMT	31.8	34.6	40.4	39.2	**39.8**	38.3	33.8	39.4	30.1	28.8	24.3	22.3	31.0
STS 2013 Average	45.3	51.4	65.3	55.8	**56.8**	54.2	44.7	56.7	45.2	41.5	24.8	42.3	46.4
deft forum	36.6	46.8	53.1	48.7	**51.1**	49.0	41.5	49.0	44.2	46.1	12.9	27.1	37.2
deft news	66.2	74.0	78.5	**73.1**	72.2	71.7	53.7	72.4	52.8	39.1	23.5	68.0	67.0
headline	67.1	75.4	78.4	69.7	**70.8**	69.2	57.5	70.2	57.5	50.9	37.8	59.5	65.3
images	75.6	79.0	83.4	**78.5**	78.1	76.9	67.6	78.2	68.5	62.9	51.2	61.0	62.0
OnWN	78.0	81.1	87.5	78.8	**79.5**	75.7	67.7	78.8	76.9	61.7	23.3	58.4	61.1
tweet news	64.7	72.2	79.2	76.4	75.8	74.2	58.0	**76.9**	58.7	48.2	39.9	51.2	64.7
STS 2014 Average	64.7	71.4	76.7	70.9	**71.3**	69.5	57.7	70.9	59.8	51.5	31.4	54.2	59.5
answers-forums	61.3	68.2	73.9	**68.3**	65.1	62.6	32.8	67.4	51.9	50.7	36.1	30.5	38.8
answers-students	67.6	73.6	78.8	**78.2**	77.8	78.1	64.7	78.2	71.5	55.7	33.0	63.0	69.2
belief	67.7	72.2	77.2	**76.2**	75.4	72.0	51.9	75.9	61.7	52.6	24.6	40.5	53.2
headline	74.2	80.8	84.2	74.8	**75.2**	73.5	65.3	75.1	64.0	56.6	43.6	61.8	69.0
images	80.4	84.3	87.1	**81.4**	80.3	77.5	71.4	81.1	70.4	64.2	17.7	67.5	69.9
STS 2015 Average	70.2	75.8	80.2	**75.8**	74.8	72.7	57.2	75.6	63.9	56.0	31.0	52.7	60.0
2014 SICK	71.4	79.9	82.8	71.6	**71.6**	70.7	61.2	71.2	63.9	59.0	49.8	65.9	66.4
2015 Twitter	49.9	52.5	61.9	52.9	52.8	**53.7**	45.1	52.9	47.6	36.1	24.7	30.3	36.3

Our Results -

Dataset	CCA using cosine similarity	CCA using WMD
OnWN	60.5	37.1
headline	62.5	55.8

Future scope

- Semantic Textual Similarity (STS) has application in not only in Computer Science but also in computational linguistic, Biomedical Informatics and Geoinformation.

- Applying STS we can solve many problems related to these areas.

- On further improvement the model will be helpful in various ways and can be used in applications such as document summarization, word sense disambiguation, short answer grading, information retrieval and extraction, etc.