

# Semantic Textual Similarity

Shivam Soni

Supervisor: Prof. K. Narayana Murthy

University of Hyderabad  
(NLE LAB)

*sonishivam131@gmail.com*

July 2, 2020

# Overview I

## 1 Introduction

- Applications
- STS Issue

## 2 Related Works

- Knowledge And Corpus Based Techniques
- Neural Based Technique

## 3 Proposed Hybrid Architecture

- Motivation

## 4 KACB Methods

- N - Gram Overlap
- WordNet - Augmented Word Overlap
- Vector Space Sentence Similarity
- Weighted Word Overlap
- Normalized Diff. and Number Overlap
- Named Entity Feature
- Experiments Using Modified KACB Methods

# Overview II

- 2012 SemEval Dataset
- 2017 SemEval Dataset

## 5 CNNs And FCNN

- Training And Evaluation
- Experiments Using Modifications in CNN

## 6 Experiments Using Hybrid Architecture

- MLP Regressor
- SVR

## 7 Conclusion and Future Work

# Introduction

- The idea behind Semantic Textual Similarity(STS) task is, given two snippets of text, compute upto what degree the two sentences are similar in meaning.
- The degree of similarity is measured on a scale of 0 to 5 (real values), according to the shared tasks competition held by SemEval [Agirre et al., 2012].

# Example

- A score of '5' can be given if two sentences means exactly the same thing, as shown below:

**Bob is diving into the pool.**

**Bob is jumping into the swimming pool.**

- A score of '3' can be given if two sentences roughly means the same thing but some information is missing in one or the other, as shown below:

**Akshay said chapter 5 is very important than chapter 3 from the book.**

**Chapter 3 is not so important. Akshay said.**

- A score of '0' can be given if two sentences are completely independent from each other, as shown below:

**John is going for a walk with his friends.**

**There is bird flying in the sky.**

- The dataset released by SemEval (International Workshop on Semantic Evaluation ) for STS task from the year 2012-2017 contains gold standard labels for every pair of sentences which is annotated by various experienced annotators ( [Agirre et al., 2012] [Agirre et al., 2013] [Agirre et al., 2014] [Agirre et al., 2015] [Brychcín and Svoboda, 2016] [Cer et al., 2017]).
- To measure the performance of any model, i.e, how well a particular model has captured the semantic similarity between two sentences , **Pearson Correlation Coefficient** is used as standard across all datasets.
- Pearson's correlation coefficient for sample data is represented as  $r_{ab}$  and is expressed as follows:

$$r_{ab} = \frac{\sum_{i=1}^m (a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^m (a_i - \bar{a})^2} \sqrt{\sum_{i=1}^m (b_i - \bar{b})^2}} \quad (1)$$

where,  $m$  is the total number of samples,  $a_i$ ,  $b_i$  are the individual points indexed with  $i$  and  $\bar{a}$ ,  $\bar{b}$  are sample mean.

- One of the goals of the STS task is to create a unified framework for combining several semantic components.
- This STS framework itself be used for within various NLP applications such as :  
Machine Translation (MT), Summarization, Generation, Question Answering (QA), etc.

# Major Problem In STS

- The big question is how can we make our model understand these sentences better ?
- So, this is where various research is going on currently, that how and what kinds of different representations can be given to the sentence pair.



- A rise in the research on measuring similarity between two sentences mainly started when SemEval (International Workshop on Semantic Evaluation ) announced Semantic Textual Similarity shared task, which successfully ran from the year 2012 to 2017 [Agirre et al., 2012] [Agirre et al., 2013] [Agirre et al., 2014] [Agirre et al., 2015] [Brychcín and Svoboda, 2016] [Cer et al., 2017].
- Before the SemEval STS task was announced, a substantial amount of work had been done for measuring the similarity between the pair of texts but it was mainly focused on comparison between two long texts or documents [Allen, 1988] [Hatzivassiloglou et al., 1999] [Landauer et al., 1997] [Meadow et al., 1999]
- Therefore, in this field, some of the earliest works can be considered to be given by [Foltz et al., 1998] [Li et al., 2006] [Lee et al., 2005] [Mihalcea et al., 2006]. These proposed work were mainly based on knowledge and corpus based techniques. And lately the STS task is mainly driven by Neural techniques.

# Knowledge And Corpus Based Techniques

- [Bär et al., 2012] in its paper pointed out that these earlier measures were under the assumption that a single measure can itself capture all the required information out of the sentences.
  - So, in their work they combined various *content based*, *style based* and *structure based* similarity measures which resulted in more than 300 score vectors (out of which 20 features were finally used), which when combined together, served as a feature set for a simple log-linear classifier and their results gave one of the best performances in the STS task-2012.
  - The *content based measure* includes character n-grams which are compared using the implementation given by Barron and word n-grams which are compared using Jaccard coefficient. Structured based measure included various *String Similarity Measures* like, *longest common substring* and *longest common subsequent* measure.

# Knowledge And Corpus Based Techniques

- Similar approach of combining various measures was followed by [Šarić et al., 2012] , and by the combination of various knowledge based, corpus based and syntactical techniques which they proposed, a feature set of scores was constructed.
- These extracted feature set was used to train a SVR model, which finally predicted sentence similarity score.
- Some examples of their knowledge based techniques are: *WordNet-Augmented Word Overlap*, *Weighted Word Overlap* etc., all them used *WordNet* as their lexical database. The corpus based techniques like : *Vector Space Sentence Similarity* and *Weighted Vector Space Similarity* used LSA vectors.

# Neural Based Techniques

- [He et al., 2015] developed a *sentence model* using a Multi-perspective Convolutional Neural Network(MPCNN) architecture in which they used two different types of filters.
  - The first one was a holistic filter, which by the help of a particular window size matches the complete word vectors.
  - The second one was a per-dimensional filter which independently match each dimension of word embeddings.
  - The output of the sentence model after passing it through various different types of pooling layer was finally given to a fully-connected layer to get the final result.
- The multi-perspective CNN model (MPCNN) [He et al., 2015] was modified in [He et al., 2016] by adding an interaction layer before passing the input sentences in MPCNN model. This modification was mainly done to make the input sentences more inter-related, because earlier, the model lacked contextual interaction information between the sentences.

# Neural Based Techniques

- [Barrow and Peskov, 2017] in 2017 presented an end-to-end LSTM system which was a version of Siamese LSTM [Mueller and Thyagarajan, 2016].
- In Siamese LSTM [Mueller and Thyagarajan, 2016], an LSTM model is employed for each input sentence, weights are shared between layers and updated parallelly during learning phase.
- Whereas, [Barrow and Peskov, 2017] allowed two sentences to be given as an input into the same LSTM model and shared the weights.
- Another paper in 2017 [Shao, 2017], gave a simple CNN model without adding any dropout or batchnorm layer and passed the output of the CNN model into a Fully-Connected Neural Network (FCNN) to get the final scores.

# Proposed Hybrid Architecture

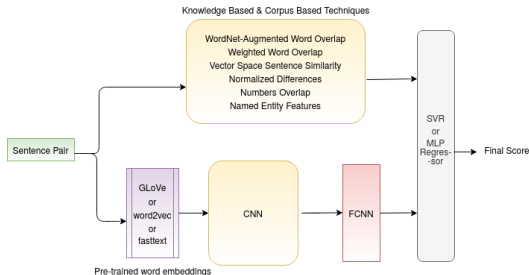


Figure: Overview of the system

- We have employed a hybrid architecture by combining various knowledge based and corpus techniques (KACB) with a CNN architecture, as shown in figure 1.

# Motivation

- The knowledge based and corpus based techniques have been taken from [Šarić et al., 2012] with some modifications in two techniques called *WordNet-Augmented Word Overlap* which is a knowledge based technique and *Vector Space Sentence Similarity* which is a corpus based technique. Whereas the CNN architecture followed by a FCNN has been inspired from [Shao, 2017].
- We have opted for *KACB* technique presented in [Šarić et al., 2012] because they have proposed one the best *KACB* based measures and are still used among benchmark results among all *KACB* techniques.
- The reason for using CNNs for training our model is, generally, RNNs or LSTMs works better on dynamic length of inputs sentences but if the input length is fixed, CNNs have proved to perform better. And in the STS 2017 dataset (for English sentence pairs) which we have considered in our work, the lengths of the sentences among each other does not vary much, i.e., the standard deviation of the lengths of sentences is less so CNNs will work better here.

- To compare both LSTMs and CNNs more precisely, we can compare [Shao, 2017] work with [Barrow and Peskov, 2017]. The whole architecture is exactly the same except that, [Shao, 2017] used CNNs in their work whereas [Barrow and Peskov, 2017] used LSTMs. And we can clearly conclude from their work that CNNs are working much better than LSTMS, where CNNs and LSTMs got a *Pearson Correlation* of **0.78** and **0.47** on English test dataset [Cer et al., 2017] respectively. That is the reason we have also used CNNs here.
- And we chose the CNN architecture presented by [Shao, 2017] among various others like [He et al., 2015], [He et al., 2016], [He and Lin, 2016] etc. because it showed that rather than having a deep complex architecture for computing semantic relation between sentences, a simple architecture can also work very well.



The following methods have been used in this work :

- ① N - Gram Overlap
- ② *WordNet – Augmented Word Overlap\**
- ③ Weighted Word Overlap
- ④ *VectorSpaceSimilarity\**
- ⑤ Normalized difference and Number Overlap
- ⑥ Named Entity Feature

NOTE: *Method with (\*) are modified in this work*

# N - Gram Overlap

- Let  $S_1$  and  $S_2$  be the sets of consecutive ngrams (e.g., unigrams, bigrams) in the first and the second sentence, respectively. The ngram overlap is defined as follows:

$$ngo(S_1, S_2) = 2 \cdot \left( \frac{|S_1|}{|S_1 \cap S_2|} + \frac{|S_2|}{|S_1 \cap S_2|} \right)^{-1} \quad (2)$$

- The ngram overlap is the harmonic mean of the degree to which the second sentence covers the first and the degree to which the first sentence covers the second. The overlap, defined by (1), is computed for unigrams, bigrams, and trigrams.

# Example For N-Gram Overlap

- Let, sent1= **The bird is bathing in the sink.**  
sent2= **Bird is washing itself in the water basin.**
- (unigram set)  $S1 = \{the, bird, is, bathing, in, the, sink\}$   
(unigram set)  $S2 = \{bird, washing, itself, in, the, water, basin\}$
- $|S1| = 6, |S2| = 5$  and  $|S1 \cap S2| = 3$
- Therefore,  $ngo(S1, S2) = 0.5454$

# WordNet - Augmented Word Overlap

- WordNet Augmented coverage is defined as  $P_{WN}(S1, S2)$ :

$$P_{WN}(S1, S2) = \frac{1}{|S2|} \sum_{w_1 \in S_1} score(w_1, S_2) \quad (3)$$

$$score(w, S) = \begin{cases} 1 & \text{if } w \in S \\ \max_{w' \in S} sim_{pl}(w, w') & \text{otherwise} \end{cases}$$

- where  $sim_{pl}(w, w')$  represents the WordNet path length similarity. The WordNet-augmented word overlap feature is defined as a harmonic mean of  $P_{WN}(S1, S2)$  and  $P_{WN}(S2, S1)$ .
- $Pathlen(C1, C2)$  = No. of edges in the shortest path (in hypernym graph) between 2 senses/concepts C1 and C2.
- $sim_{pl}(c1, c2) = 1 / Pathlen(C1 + C2)$

# Example for WAWO

- Let, sent1= **The bird is bathing in the sink.**  
sent2= **Bird is washing itself in the water basin.**
- Suppose we remove *stop words* here:  
 $S1 = \{\text{bird, bathing, sink}\}$   $S2 = \{\text{bird, washing, water, basin}\}$
- For  $P_{WN}(S1, S2) : (\text{bird, bird}), \text{score}(w, S) = 1$
- But, for the word '*bathing*':  
 $\text{score}(\text{bathing}, S) = \max \{ \text{sim}_{pl}(\text{bathing}, \text{bird}), \text{sim}_{pl}(\text{bathing}, \text{washing}), \text{sim}_{pl}(\text{bathing}, \text{water}), \text{sim}_{pl}(\text{bathing}, \text{basin}) \}$
- $\text{score}(\text{bathing}, S) = \max \{ (0.142), (0.5), (0.142), (0.1) \}$   
 $\text{score}(\text{bathing}, S) = 0.5$
- And  $\text{score}(\text{sink}, S) = 0.5$

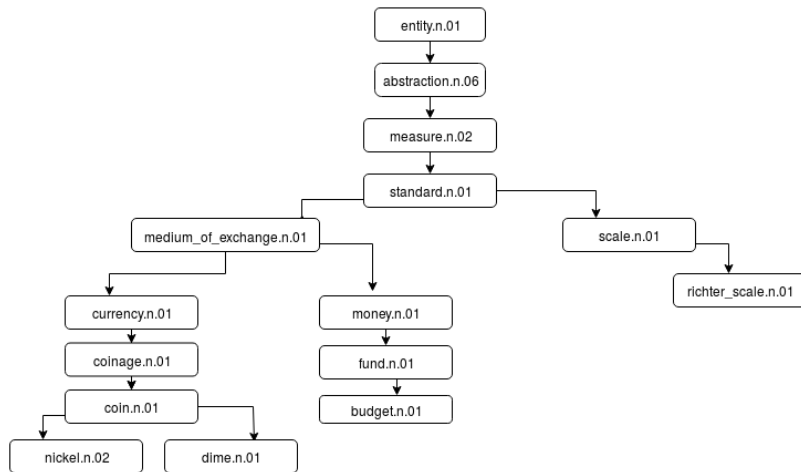
# Example for WAWO

- Therefore,

$$P_{WN}(S1, S2) = \frac{1 + 0.5 + 0.5}{4} = 0.5$$

- Similarly ,  $P_{WN}(S2, S1) = 0.72$
- Harmonic Mean of  $P_{WN}(S1, S2)$  and  $P_{WN}(S2, S1) = 0.5907$

- A hierarchically organized lexical database.
- For a word we can say it as a 'lexeme' or a 'sense' . Different words that have the same meaning are there in same synsets.
- So, we have *Synsets* here, which basically mean "synonym set" which represents a sense.





# Drawback with WAWO

- *pathlen similarity* assigns same similarity score to different pairs of concepts without considering the generality or specificity of the pairs in WordNet heirarchy.
- For example, let us consider two pairs of concepts (*nickel.n.02*, *dime.n.01*) and (*medium\_of\_exchange.n.01*, *scale.n.01*) from the wordnet figure(previous slide). Both the pairs have a pathlen of 2 (i.e, the count of the edges in the shortest path between them), therefore *pathlen similarity* between them is 0.5 given by the equation 4.

$$sim_{pl} = \frac{1}{pathlen(C1, C2)} \quad (4)$$

- But in actual, if we look intuitively, (*nickel.n.02*, *dime.n.01*) are more similar to each other than (*medium\_of\_exchange.n.01*, *scale.n.01*), also (*nickel.n.02*, *dime.n.01*) are more specific in hierarchy than (*medium\_of\_exchange.n.01*, *scale.n.01*), therefore the similarity score for (*nickel.n.02*, *dime.n.01*) should have been greater. So, we can observe that the similarities given by *pathlen similarity* are not accurate.

# Modification in WAWO

- To overcome these drawbacks we have used *Jiang-Conrath Similarity (JC similarity)* and *Lin Similarity* proposed by [Jiang and Conrath, 1997] and [Lin, 1998] respectively, replacing *pathlen similarity*, which we discuss in the next sections.
- JC and LS similarities rely on WordNet structure and adds probabilistic information derived from a corpus.

- *Jiang- Conrath similarity* [Jiang and Conrath, 1997] is given as:

$$sim_{JC}(c1, c2) = \frac{1}{IC(c1) + IC(c2) - 2 * IC(LCS(c1, c2))} \quad (5)$$

where  $IC(c)$  is the *Information Content* of the concept ' $c$ ' and  $LCS(c1, c2)$  is the least common subsumer between the concepts  $c1$  and  $c2$ .

- Where *Information Content*( $IC$ ) as given by *Resnik(1998)* in [Resnik, 1995] is used in this work, which is given as follows:

$$IC(c) = -\log P(c) \quad (6)$$

we have used *Brown Corpus* and *treebank Corpus* from NLTK to calculate  $IC$  of each concept in WordNet.

- $P(c)$  is the probability that a randomly selected word in a corpus is an instance of concept  $c$  (i.e., a separate random variable, ranging over words, associated with each concept). This implies that  $P(\text{root}) = 1$  since any word is subsumed by the root concept. Therefore, probability is given as:

$$P(c) = \frac{\sum_{w \in \text{words}(c)} \text{count}(w)}{N} \quad (7)$$

- where  $\text{words}(c)$  is the set of words subsumed by concept  $c$ , and  $N$  is the total number of words in the corpus that are also present in the thesaurus.

# JC Similarity Derivation

- IC is used to assign lengths to graph edges:

$$\text{dist}_{JC}(c, \text{hypernym}(c)) = IC(c) - IC(\text{hypernym}(c))$$

- $\text{dist}_{JC}(c1, c2) = \text{dist}_{JC}(c1, \text{LCS}(c1, c2)) + \text{dist}_{JC}(c2, \text{LCS}(c1, c2))$

$$= IC(c1) - IC(\text{LCS}(c1, c2)) + IC(c2) - IC(\text{LCS}(c1, c2))$$

$$= IC(c1) + IC(c2) - 2 * IC(\text{LCS}(c1, c2))$$

- Therefore,

$$\text{sim}_{JC}(c1, c2) = \frac{1}{IC(c1) + IC(c2) - 2 * IC(\text{LCS}(c1, c2))}$$

# Lin- Similarity

- Lin - similarity is the information content common to  $c1$  and  $c2$ , normalized by their average information content, which is given as:

$$sim_{Lin}(c1, c2) = \frac{2.IC(LCS(c1, c2))}{IC(c1) + IC(c2)}$$

- It basically tells the more information they don't share , the less similar they are.
- In LS also we have used the same technique to calculate *Information Content*, as used in JCS.

# Vector Space Sentence Similarity

- In this method each sentence is represented as a single distributional vector  $\vec{u}$  by summing the distributional (i.e., LSA) vector of each word  $w$  in the sentence  $S$  :

$$\vec{u}(S) = \sum_{w \in S} x_w$$

where  $x_w$  is the vector representation of the word  $w$ .

- Another similar representation  $u_{\vec{W}}$  uses the *information content*  $ic(w)$  to weigh the LSA vector of each word before summation:

$$u_{\vec{W}}(S) = \sum_{w \in S} ic(w)x_w$$

- Then  $|\cos(u(S_1), u(S_2))|$  and  $|\cos(u_W(S_1), u_W(S_2))|$  is used for the vector space sentence similarity features between the two sentences.

# Modification in VSS Similarity

- Rather than using *LSA vectors*, vectors provided by Google's pre-trained *word2vec model* [Mikolov et al., 2013b] has been used.
- This word2vec model makes use of *skip-gram architecture* and is trained on various news dataset, which contributed approximately *100 billion word*.
- Also, same modification has been done with the weighted word vectors, where word vectors are weighted by their Information Content.
- These modifications have been done because *word2vec algorithm* has shown to capture better similarity between the words from a given corpus ([Mikolov et al., 2013a]). But it gives better performance when the training data is in large amount, that is the reason a pre-trained model is preferred here.



# Weighted Word Overlap

- In this measure importance is given to the words which have more information content:

$$ic(w) = \ln \frac{\sum_{w' \in C} freq(w')}{freq(w)}$$

- where  $C$  is the set of words in the corpus and  $freq(w)$  is the frequency of the word in the corpus.
- The corpus which they have used is Google Books Ngrams [?] to obtain the frequencies because of its excellent word coverage for English.
- The weighted word coverage of the second sentence by the first sentence is given by :

$$wwc(S1, S2) = \frac{\sum_{w \in S1 \cap S2} ic(w)}{\sum_{w' \in S2} ic(w')} \quad (8)$$

- The weighted word overlap between two sentences is calculated as the harmonic mean of the  $wwc(S1, S2)$  and  $wwc(S2, S1)$ .

# Normalized Diff. and Number Overlap

- Normalized Diff: They measured the normalized differences in a pair of sentences using: A) sentence length , B) the noun chunk, verb chunk, and predicate counts, and (C) the aggregate word information content.
- Numbers Overlap:
  - The annotators gave low similarity scores to many sentence pairs that contained different sets of numbers, even though their sentence structure was very similar.
  - So, in number overlap three features are used:

$$(N1 \subseteq N2 \vee N2 \subseteq N1), (\log(1 + |N1| + |N2|)) \text{ and } (|N1 \cap N2|)/(|N1| + |N2|)$$

- where  $N1$  and  $N2$  are the sets of numbers in  $S1$  and  $S2$ .

# Named Entity Feature

- In this feature, capitalized words are considered as named entities and an overlap is computed between them as done equation 2.
- An overlap between the stock index symbols is also computed. Stock index symbols are the tokens which are in all caps and begin with a period.

# Feature Extraction Using Methods

- The sentences are represented as shown in the following table:

sentence pairs	feat1	feat2	feat3	feat4	feat5	label
sp1	0-1	0-1	0-1	0-1	0-1	0-5
sp2	0-1	0-1	0-1	0-1	0-1	0-5
sp3	0-1	0-1	0-1	0-1	0-1	0-5
.	0-1	0-1	0-1	0-1	0-1	0-5
.	0-1	0-1	0-1	0-1	0-1	0-5

- These feat1 or feat2 are different methods(shown in previous slide) or techniques for measuring similarity between these sentences.
- (0-1) represents the float value between 0-1 and similarly (0-5) represents the float value between 0-5.

# Experiments

Five different types of experiments have been done which are as follows:

- **jc**: It contains results for the changes made in *WordNet-Augmented Word Overlap*, where *pathlen similarity* is replaced by *JC similarity* keeping rest of the techniques same.
- **lin**: It contains results for the changes made in *WordNet-Augmented Word Overlap*, where *pathlen similarity* is replaced by *Lin similarity* keeping rest of the techniques same.
- **jc\_lin**: It contains results for the changes made in *WordNet-Augmented Word Overlap*, where *pathlen similarity* is replaced by *Lin similarity* in one feature and by *JC similarity* in another, keeping rest of features same.
- **ww\_wwv**: It contains results for the changes made in *Vector Space Sentence Similarity*, where LSA vectors are replaced by Word2Vec vectors and also an additional feature is used where LSA vectors is replaced by weighted Word2Vec vectors, rest all the features are the same.
- **jc\_lin\_ww\_wwv**: In this technique all the proposed changes are merged together keeping rest of the features same.

# Dataset [Agirre et al., 2012]

- The dataset is taken from the SemEval STS task (2012)[Agirre et al., 2012] which contains in total 5342 sentence pairs in English language and corresponding to each sentence pair a real value score is provided in the range from 0 to 5. This data is collected from various different sources as shown in the *Table 1*.

Dataset	Pairs	Source
MSRpar	1500	News
MSRvid	1500	Video
SMTeuroparl	750	MT eval
SMTnews	399	MT eval
OnWn	1193	Glosses

**Table:** Dataset Description

- The MSRvid and MSRpar dataset is collected by Microsoft Research(MSR).
- SMTeuroparl dataset was created from the translation shared task of the 2007 and 2008 ACL Workshop on Statistical Machine Translation (SMT) [Callison-Burch et al., 2007] [Callison-Burch et al., 2008]. In addition, two surprise datasets SMTnews and OnWn were also released which used here.

# KACB Results on [Agirre et al., 2012]

- The table below shows the **Pearson Correlation** value between the gold-standard human annotated scores and the scores given by the model on the test dataset.

Technique	MSRvid	MSRpar	SMTeuroparl	SMTnews	OnWn
takelab	0.8803	0.7343	0.4771	0.3989	0.6797
jc	0.8747	<b>0.7381</b>	0.5364	0.4237	0.6914
lin	0.8705	0.7371	0.5362	0.4319	0.6904
jc_lin	0.8713	0.7377	0.5354	0.4068	0.6899
wv_wwv	<b>0.8837</b>	0.7378	<b>0.5376</b>	<b>0.5014</b>	<b>0.7067</b>
jc_lin_wv_wwv	0.8791	0.7367	0.5323	0.4720	0.7027

**Table:** Pearson Correlation Values for each dataset corresponding to each experiments.

# KACB Results on [Cer et al., 2017]

In the table below we have compared our results with averaged word embedding baseline [Salle et al., 2016] on monolingual(English) test dataset [Cer et al., 2017]. We notice that all our results are above the baseline scores.

Experiments	Scores
word_vec_baseline [Salle et al., 2016]	55.8
jc	73.0
lin	72.4
jc_lin	72.7
wv_www	71.4
jc_lin_wv_www	72.7

**Table:** Pearson Correlation  $\times 100$  Scores On Test Dataset 2017

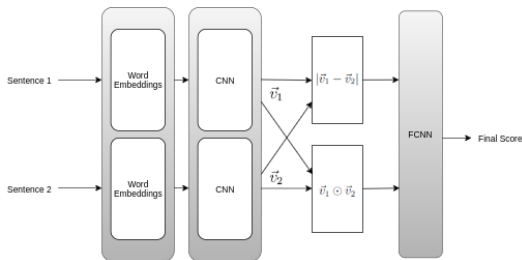


In the table below we have shown benchmark results of STS 2017 [Cer et al., 2017] on test dataset (with the techniques which they have adopted) and compared with our top performing experiment. We notice that except [Duma and Menzel, 2017] any of our results are not above rest of the benchmark results.

Experiments	Scores
jc	73.0
Ensemble [Tian et al., 2017]	81.0
Wordnet + Embeddinglin [Wu et al., 2017]	80.9
Ensemble [Maharjan et al., 2017]	79.2
CNN [Shao, 2017]	78.4
Doc2Vec [Duma and Menzel, 2017]	59.2

Table: STS 2017 benchmark

# CNNs and FCNN



**Figure:** CNN Architecture [Shao, 2017]. Note that there is only one CNN in the shaded region onto which both sentences are passed.

Each sentence is first represented in a matrix form, by passing it into word embedding layer. Then each sentence matrix is given as an input to the convolution layer and we get two vectors  $v_1$  and  $v_2$ , after the *max-pooling operation*, for both the sentences respectively. At last, by the concatenation of element-wise absolute difference and the element-wise multiplication between the two representations of input sentences  $v_1$  and  $v_2$ , a *Semantic Difference Vector* ( $S\vec{D}V$ ) is created. This  $S\vec{D}V$  is finally passed through the Full-Connected Layer (FCNN) to get the final output.

- **During training**, we considered the task of semantic relatedness as a classification problem instead of a regression problem, by considering the outputs from the last layer as probability distribution over 6 classes. And the gold labels (referred as  $a$  in 9) which were real values in range 0-5 were converted into a probability distribution as follows [Tai et al., 2015]:

$$p_t = \begin{cases} a - \lfloor a \rfloor, & i = \lfloor a \rfloor + 1 \\ \lfloor a \rfloor - a + 1, & i = \lfloor a \rfloor \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

- And then used the loss functions to compute the errors, which were then backpropagated for training the weights.
- Additionally we have also used KL-Divergence as loss function in our work. KL-Divergence between two probability distribution  $A(i)$  and  $B(i)$  (where  $B(i)$  is the approximation and  $A(i)$  is the true distribution is expressed as follows:

$$D_{KL}(A||B) = \sum_{i \in I} A(i) \log \left( \frac{A(i)}{B(i)} \right) \quad (10)$$

**During evalution** a weighted sum or expectation of the scores was calculated, which was considered as the final score.

# Experiments Using CNNs

[Shao, 2017] have used 300 dimensional GloVe word embeddings for the representation of each word in a sentence and Pearson Correlation Coefficient as the loss function. We have tried five different modifications in their work by using different loss function and pre-trained word embeddings. The abbreviations and the interpretation of them are as follows:

- **glove\_kl**: In this experiment we have replaced loss function correlation coefficient with KL-Divergence loss function.
- **word2vec\_coef**: In this experiment we have replaced pretrained GloVe word embeddings with pretrained word2vec embeddings.
- **word2vec\_kl**: Here both embeddings and loss functions have been replaced with *word2vec* and *KL-Divergence* respectively.
- **fasttext\_kl**: Here also, both embeddings and loss functions have been replaced with *word2vec* and *KL-Divergence* respectively.
- **fasttext\_coef**: In this experiment embeddings are replaced with fasttext embeddings, keeping the loss function same.

# Results Using CNNs Modification

Each experiment discussed above produces a score between the two sentences, expressing the semantic relatedness between them. These scores, corresponding to every English sentence pair from the SemEval 2017 test dataset [Cer et al., 2017] are compared with Gold Standard scores using person correlation coefficient<sup>1</sup>, which are shown in table 5.

Experiments	Scores
(shao) [Shao, 2017]	78.4
glove_kl	70.26
word2vec_coef	76.9
word2vec_kl	70.10
fasttext_kl	72.5
fasttext_coef	73.4

**Table:** (Pearson Correlation Scores  $\times 100$ ) On Test Dataset(2017) on English Pairs

We observe that none of our sytems were able to beat the scores of [Shao, 2017], which proves that GloVe word embeddings with correlation coefficient as loss function was certainly a good combination for this architecture.

# Results Using Whole Architecture(MLP regressor)

The following table shows the *Pearson Correlation Coefficient* between the output scores from our proposed architecture (figure 1) and the gold standard labels. The table below shows the results when the combined outputs of *CNN* and *Knowledge Based and Corpus Based (KACB)* is passed into an MLP regressor and a *Pearson Correlation Coefficient* is computed between output of the scores from MLP regressor and the gold standard labels.

MLP Regressor	jc	lin	jc_lin	wv_wwv	jc_lin_wv_wwv	[Šarić et al., 2012]
(glove_coef)[Shao, 2017]	<b>0.787</b>	0.742	0.731	0.742	0.752	0.786
glove_kl	0.761	0.752	0.732	0.733	0.745	0.734
word2vec_coef	0.731	0.725	0.713	0.713	0.716	0.715
word2vec_kl	0.721	0.724	0.712	0.715	0.70	0.730
fasttext_kl	0.732	0.721	0.716	0.712	0.708	0.730
fasttext_coef	0.736	0.727	0.713	0.712	0.715	0.732

**Table:** Pearson Correlation Values b/w MLP regreessor Scores and Gold labels

The best performance is given when CNN architecture as proposed by [Shao, 2017] is combined with our proposed KACB technique (*jc*).

# Results Using Whole Architecture(SVR)

The table below shows the results when the combined outputs of *CNN* and *Knowledge Based and Corpus Based (KACB)* is passed into an MLP regressor and a *Pearson Correlation Coefficient* is computed between output of the scores from SVR and the gold standard labels.

SVR	jc	lin	jc_lin	wv_www	jc_lin_wv_www	[Šarić et al., 2012]
(shao-2017)[Shao, 2017]	0.771	0.762	0.761	0.752	0.753	0.770
glove_kl	0.731	0.721	0.745	0.752	0.742	0.751
word2vec_coef	0.733	0.723	0.751	0.743	0.742	0.751
word2vec_kl	0.761	0.752	0.761	0.752	0.754	0.753
fasttext_kl	0.751	0.752	0.730	0.731	0.732	0.752
fasttext_coef	0.763	0.752	0.752	0.741	0.731	0.762

**Table:** Pearson Correlation Values b/w SVR Scores and Gold labels

# Comparison with Benchmark Results

In the table below we have compared our top performing architecture with the benchmark results as per SemEval 2017 test dataset on English sentence pairs [Cer et al., 2017] and noticed that we are getting slightly better results than [Shao, 2017].

Experiments	Scores
Ensemble [Tian et al., 2017]	81.0
Wordnet + Embeddinglin [Wu et al., 2017]	80.9
Ensemble[Maharjan et al., 2017]	79.2
glove_coef & $j_c$	<b>78.7</b>
CNN[Shao, 2017]	78.4
Doc2Vec[Duma and Menzel, 2017]	59.2

**Table:** STS 2017 benchmark. The bold score is our score on the benchmark database.



# Conclusion and Future Work

- With some modifications in *Knowledge and Corpus Based* techniques we were able to improve scores for measuring semantic similarity between sentences ( as per STS 2012 test dataset for English pairs [Agirre et al., 2012]). But for the 2017 STS dataset [Cer et al., 2017], except [Duma and Menzel, 2017] our scores were not able to beat other benchmark results.
- Some modifications in CNN architecture given by [Shao, 2017] did not really bring any improvement in measuring similarity between sentences. Which therefore proved that the combination of word embeddings and the loss function as suggested in [Shao, 2017] (i.e. *pretrained GLoVe vectors* as word embedding [Pennington et al., 2014] and *Pearson Correlation Coefficient* was actually the best among all the experiments we carried out.
- Our final hybrid architecture was able to perform much better than [Duma and Menzel, 2017] and slightly better than [Shao, 2017] which are part of the benchmark results 2017.
- So, this shows that by combining the traditional *Knowledge and Corpus Based* techniques with the recent techniques with DNNs, we can extract more relevant feature set out of the sentence pairs, to compute the similarity.
- Therefore, in the future work, we will include some more traditional techniques, like using word alignment [Sultan et al., 2014] to improve the capability of our system to measure similarity between the two sentences.

- [1] “Combining Knowledge And Corpus Based Techniques with Convolutional Neural Nets for Semantic Textual Similarity.” Shivam Soni, Kavi Narayana Murthy.

# References I



Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Guo, W., Lopez-Gazpio, I., Maritxalar, M., Mihalcea, R., et al. (2015). Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability.

*In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.



Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Guo, W., Mihalcea, R., Rigau, G., and Wiebe, J. (2014).

Semeval-2014 task 10: Multilingual semantic textual similarity.

*In Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 81–91.



Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., and Guo, W. (2013).

\* sem 2013 shared task: Semantic textual similarity.

*In Second Joint Conference on Lexical and Computational Semantics (\*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43.

# References II



Agirre, E., Diab, M., Cer, D., and Gonzalez-Agirre, A. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics, SemEval '12*, pages 385–393, Stroudsburg, PA, USA. Association for Computational Linguistics.



Allen, J. (1988). *Natural Language Understanding*. Benjamin-Cummings Publishing Co., Inc., USA.



Bär, D., Biemann, C., Gurevych, I., and Zesch, T. (2012). Ukp: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics, SemEval '12*, pages 435–440, Stroudsburg, PA, USA. Association for Computational Linguistics.

# References III



Barrow, J. and Peskov, D. (2017).

UMDeep at SemEval-2017 task 1: End-to-end shared weight LSTM model for semantic textual similarity.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 180–184, Vancouver, Canada. Association for Computational Linguistics.



Brychcín, T. and Svoboda, L. (2016).

Uwb at semeval-2016 task 1: Semantic textual similarity using lexical, syntactic, and semantic information.

In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 588–594.



Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2007).

(meta-) evaluation of machine translation.

In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158. Association for Computational Linguistics.

# References IV



Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008).

Further meta-evaluation of machine translation.

In *Proceedings of the third workshop on statistical machine translation*, pages 70–106. Association for Computational Linguistics.



Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. (2017).

SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.



Duma, M.-S. and Menzel, W. (2017).

SEF@UHH at SemEval-2017 task 1: Unsupervised knowledge-free semantic textual similarity via paragraph vector.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 170–174, Vancouver, Canada. Association for Computational Linguistics.

# References V



Foltz, P. W., Kintsch, W., and Landauer, T. K. (1998).  
The measurement of textual coherence with latent semantic analysis.  
*Discourse Processes*, 25(2-3):285–307.



Hatzivassiloglou, V., Klavans, J. L., and Eskin, E. (1999).  
Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning.  
In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.



He, H., Gimpel, K., and Lin, J. (2015).  
Multi-perspective sentence similarity modeling with convolutional neural networks.  
In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586, Lisbon, Portugal. Association for Computational Linguistics.

# References VI



He, H. and Lin, J. (2016).

Pairwise word interaction modeling with deep neural networks for semantic similarity measurement.

In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 937–948, San Diego, California. Association for Computational Linguistics.



He, H., Wieting, J., Gimpel, K., Rao, J., and Lin, J. (2016).

UMD-TTIC-UW at SemEval-2016 task 1: Attention-based multi-perspective convolutional neural networks for textual similarity measurement.

In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1103–1108, San Diego, California. Association for Computational Linguistics.







Jiang, J. J. and Conrath, D. W. (1997).

Semantic similarity based on corpus statistics and lexical taxonomy.





In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 19–33, Taipei, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).







# References VII

-  Landauer, T. K., Laham, D., Rehder, B., and Schreiner, M. E. (1997). How well can passage meaning be derived without using word order? a comparison of latent semantic analysis and humans.
-  Lee, M. D., Pincombe, B., and Welsh, M. (2005). An empirical evaluation of models of text document similarity.
-  Li, Y., McLean, D., Bandar, Z. A., O'Shea, J. D., and Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1138–1150.
-  Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

# References VIII

-  Maharjan, N., Banjade, R., Gautam, D., Tamang, L. J., and Rus, V. (2017). 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 Evaluation (SemEval-2017)*, pages 120–124, Vancouver, Canada. Association for Computational Linguistics.
-  Meadow, C. T., Kraft, D. H., and Boyce, B. R. (1999). *Text information retrieval systems*. Academic Press, Inc.
-  Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1, AAAI06*, page 775780. AAAI Press.
-  Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

# References IX

-  Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.
-  Mueller, J. and Thyagarajan, A. (2016). Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI16, page 27862792. AAAI Press.
-  Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
-  Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

# References X



Salle, A., Villavicencio, A., and Idiart, M. (2016).

Matrix factorization using window sampling and negative sampling for improved word representations.

In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 419–424, Berlin, Germany. Association for Computational Linguistics.



Šarić, F., Glavaš, G., Karan, M., Šnajder, J., and Dalbelo Bašić, B. (2012).

TakeLab: Systems for measuring semantic text similarity.

In *\*SEM 2012: The First Joint Conference on Lexical and Computational Semantics –(SemEval 2012)*, pages 441–448, Montréal, Canada. Association for Computational Linguistics.



Shao, Y. (2017).

HCTI at SemEval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 130–133, Vancouver, Canada. Association for Computational Linguistics.

# References XI



Sultan, M. A., Bethard, S., and Sumner, T. (2014).

Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence.

*Transactions of the Association for Computational Linguistics*, 2:219–230.



Tai, K. S., Socher, R., and Manning, C. D. (2015).

Improved semantic representations from tree-structured long short-term memory networks.

In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.



Tian, J., Zhou, Z., Lan, M., and Wu, Y. (2017).

ECNU at SemEval-2017 task 1: Leverage kernel-based traditional NLP features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 191–197, Vancouver, Canada. Association for Computational Linguistics.



Wu, H., Huang, H., Jian, P., Guo, Y., and Su, C. (2017).

BIT at SemEval-2017 task 1: Using semantic information space to evaluate semantic textual similarity.

In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 77–84, Vancouver, Canada. Association for Computational Linguistics.

# Thank You