ASOBEK at SemEval-2016 Task 1: Sentence Representation with Character N-gram Embeddings for Semantic Textual Similarity

Asli Eyecioglu and **Bill Keller**
University of Sussex
Department of Informatics
Brighton, BN19QJ, UK
A.Eyecioglu@sussex.ac.uk,
billk@sussex.ac.uk

Abstract

A growing body of research has recently been conducted on semantic textual similarity using a variety of neural network models. While recent research focuses on word-based representation for phrases, sentences and even paragraphs, this study considers an alternative approach based on character n-grams. We generate embeddings for character n-grams using a continuous-bag-of-n-grams neural network model. Three different sentence representations based on n-gram embeddings are considered. Results are reported for experiments with bigram, trigram and 4-gram embeddings on the STS Core dataset for SemEval-2016 Task 1.

1 Introduction

This paper presents an approach for finding the degree of semantic similarity between sentence pairs. Semantic textual similarity (STS) is of relevance to many NLP applications. Recent tasks in recognizing textual entailment, sentence completion and paraphrase identification are closely related. The approach described here makes use of a neural network (NN) algorithm (word2vec) that is typically used to generate word embeddings (Mikolov et al., 2013a). Rather than generating vector representations for words however, we propose a character n-gram-to-vector approach. A sentence is then represented as a vector generated through a combination of character n-gram embeddings.

The use of character level vectors has been proposed in a number of recent studies. Subword language models that use the combination of characters, syllables and frequent words have been explored by Mikolov et al. (2012). Character-level language modeling has been performed for modeling OOV words, where using words as the atomic units of the model would not be sufficient to assign a probability score. In Ling et al. (2015), word representations are composed of vectors of characters, called character to word (C2W). The C2W vectors are used successfully for language modeling and POS tagging without any handcrafted features. The resulting model is competitive on English POS tagging with the Stanford POS tagger word look-up tables and also produces a notable improvement in results for morphologically rich languages such as Turkish. Kim et al. (2015) apply a simple convolutional neural network model, which uses character level inputs for word representations. Again, this method outperforms the models that use word/morpheme level features in morphologically rich languages, while also having competitive results in English. Huang et al. (2013) introduce a word hashing technique using character n-grams to scale up training of deep NN models for large-scale web search applications.

Our motivation for exploring character n-grams derives in part from previous work we have conducted on paraphrase identification (PI). The PI task is that of deciding whether two sentences have the same meaning. We have shown that sentence representations based on bags or sets of character n-gram features can perform well at this task (Eyecioglu and Keller, 2015). It is hypothesized that n-grams are useful for capturing lexical similarity and perform a role similar to lemmatization whilst preserving differences. Thus, sentence representations based on collections of n-grams as features may offer some advantages over representations based on words as features.

Proceedings of SemEval-2016, pages 1320–1324,
San Diego, California, June 16-17, 2016. ©2016 Association for Computational Linguistics

The current study aims to extend this earlier work by working with n-gram embeddings. (Mikolov et al., 2013a) introduced two new NN architectures that were applied to learning word embeddings: continuous-bag-of-words (CBOW) and skip grams (SG). These NN models have been shown to perform well in many NLP areas such as STS and PI (Zarrella et al., 2015; Yin and Schütze, 2015; He et al., 2015). Recent research has taken steps to extend these word vectors to sentences, paragraphs and documents (Le and Mikolov, 2014).

We introduce an alternative approach to obtaining sentence level embedding vectors that make use of character n-grams rather than words. We adopt a model architecture that is analogous to CBOW, which we call *continuous-bag-of-n-grams* and notate as CBO*n*G throughout the paper. In keeping with our earlier work on paraphrase identification (Eyecioglu and Keller, 2015), preprocessing is kept to a minimum and no use is made of any manually constructed semantic or syntactic processing tools or resources.

Operationally, STS is similar to paraphrase identification. The two tasks differ in that STS sentence pairs are assigned a degree of semantic equivalence instead of a binary paraphrase label. STS shared tasks have produced a sizable amount of research on sentence similarity (Agirre et al., 2012, 2013, 2014, 2015).

2 The Task

SemEval-2016 Task 1: Semantic Textual Similarity (Agirre et al., 2016) requires systems to determine the degree of semantic similarity between pairs of sentences. Similarity scores are on a scale from 0 (completely dissimilar) to 5 (semantically equivalent). We participated in the monolingual STS Core subtask. This subtask includes English language evaluation data from multiple sources organized into 5 distinct evaluation datasets: Plagiarism Detection, Q&A Question-Question, Q&A Answer-Answer, Post-Edited Machine Translations and Headlines. The evaluation data have a gold standard similarity score based on human judgements collected using Crowdsourcing. The Pearson correlation between similarity scores assigned by the systems and the human judgements is used to assess task performance.

3 Approach

The sentences within the STS pairs are split into character n-grams. No preprocessing is applied besides lowercasing of the text and removing punctuation.

Our training procedure was unsupervised, using only a large unlabelled dataset drawn from Wikipedia. These data are used to train a CBO*n*G model. We explore using three different methods for constructing sentence level vector representations from the character embeddings. STS scores for the sentence pairs are computed as the cosine similarity of the resulting sentence level embedding vectors. Three different cosine similarity scores, one from each representation, are obtained.

4 Wikipedia Dataset

We used a dump of English Wikipedia articles[1] that includes 3,831,719 articles and 8,179,596 unique words. Wikipedia provides a large and accessible collection of text consisting of well-formed sentences that is suitable for training purposes. We obtained a plain text representation of the documents by removing the data dump xml tags using the script provided in the Gensim package (Rehurek and Sojka, 2010). Matching the minimal pre-processing performed on the STS pairs, training data are only lowercased and cleared of punctuation. The Wikipedia dataset is split into adjacent n-grams and spaces between words replaced with the "-" symbol. For example, the following sequence of trigrams would be produced from the text *amazon gift*.

```
-am ama maz azo zon on- n-g -gi gif ift ft-
```

4.1 Constructing a CBO*n*G Model

Our CBO*n*G is constructed and trained identically to a CBOW model (Mikolov et al., 2013a) but substituting character n-grams in place of words. The quality of such a model is affected by a number of hyper-parameters such as the size of the character n-gram vectors (embeddings), the size of the training window, and the cut-off point for less frequent n-grams. Although experiments by Mikolov et al. (2013b) describe how to choose the appropriate features for a word similarity task, the same fea-

[1] Downloaded at 07/07/2015 from https://dumps.wikimedia.org/enwiki/

tures might not be ideal here. Moreover, our model brings an additional new modeling parameter that defines the size of the character n-grams.

Embedding models scale linearly with the number of unique unfiltered tokens in the training data. Each token has a corresponding fixed size embedding vector. Mikolov et al. (2013a) explored using embedding vector sizes ranging from 20 to 600. In general, the results showed that increasing the dimensionality of vectors and the size of training data, improved the accuracy on semantic-syntactic word relationship task using otherwise identical features.

For our experiments, we make use of 400 dimension embedding vectors. Our CBO*n*G model was trained using surrounding n-grams within a window size of 5. Character n-grams that occur fewer than 5 times are filtered. Our model is trained using the Gensim package (Rehurek and Sojka, 2010) and its support for CBOW *(word-level)* models but over data tokenized into character n-grams.

4.2 Compositions of Vector Representations

The construction of embedding vectors for phrases and sentences directly from word embeddings is still an active area of research. As noted in Section 2, there have been various efforts to build good embedding representations of text beyond those tied to individual words. We believe the diversity of methods for constructing textual embedding representations beyond the word level is due to the fact the appropriateness of the various representations is very task dependent.

One simple approach is to construct textual embedding representations using either point-wise addition multiplication of the embedding vectors representing individual words and phrases. The resulting representations have been shown to work well for phrase similarity and PI tasks (Blacoe and Lapata, 2012).

We make use of a similar composition algorithm for our sentence level embeddings, but using character n-gram rather than word embeddings. We describe three different methods for combining n-gram embeddings. The first is formed by addition of the embeddings of the n-gram *tokens* in a sentence, effectively weighting the embeddings by their frequency. The second and third are based on n-gram *types*, and formed by concatenation and weighted addition, respectively.

4.3 Sentence Representations

For the STS task, the core experimental unit is a pair of sentences. A target sentence, S, consisting of some sequence of n tokens (i.e. character n-grams) $(t_{S_1}, t_{S_2}, \dots, t_{S_n})$ is paired with another sentence, S' that consists of some sequence m of tokens $(t_{S'_1}, t_{S'_2}, \dots, t_{S'_m})$. Note that the numbers of tokens for each sentence do not necessarily need to be equal. In the following, a vector embedding associated with a token t is notated by v_t. We consider three different vector based sentence representations built from n-gram embeddings.

For a given sentence S, the first representation is produced through element-wise addition of the vectors associated with each token in S. The result of this operation is a vector V_S^1 having the specified vector size of the token embeddings, as is defined in advance to building the models (i.e. 400 in this case).

For the second approach, S is represented as a matrix of size $400 \times d$, where d is the size of the model vocabulary (i.e. the number of unique n-grams using in training). Combining the d vectors in order forms a vector representation:

$$V_S^2 = \left[v^S{}_{t_1}, v^S{}_{t_2}, \dots, v^S{}_{t_d}\right]$$

where $v^S{}_{t_i}$ is the embedding associated with the i^{th} term t_i in the model vocabulary if t_i occurs in S, and is the null vector if the i^{th} term does not occur in S.

Finally, for the third representation, element-wise addition of all of the embeddings in $V^2{}_S$ is computed. We obtain a new vector of $V^3{}_S$ the dimension of the embedding vector size, which is specified as 400:

$$V_S^3 = \left[v^S{}_{t_1} + v^S{}_{t_2} + \cdots + v^S{}_{t_d}\right]$$

We note that the difference between Run1 and Run3 is that for Run1, the contributions of the n-gram embeddings are weighted for frequency.

5 Experiments

To obtain semantic similarity scores for each pair of sentences S and S', the cosines of the corresponding vector representations are computed.

Run1: $Sim1(S, S') = cosine(V_S^1, V_{S'}^1)$
Run2: $Sim2(S, S') = cosine(V_S^2, V_{S'}^2)$
Run3: $Sim3(S, S') = cosine(V_s^3, V_{S'}^3)$

Previous work on the use of character n-grams for PI has shown that trigrams perform well. The three runs chosen for submission to SemEval 2016's STS task use sentence representations constructed from trigram embeddings. Wikipedia articles were pruned using the Gensim package with default parameters. A total of 108,452 articles are used to construct a character-trigram model for the experiments. The statistical properties of the Wikipedia-trained model using character trigrams are shown in Table 1 below.

Properties	Count
Total trigrams	1,068,456,094
Unique trigrams	91,686
Unique trigrams > 5	47,951

Table 1: Statistical properties of Wikipedia dataset used in our experiment.

Further experiments were conducted for sentence representations based on bigrams and 4-grams. Although these were not submitted for the task, the results are also reported in the following section.

6 Results

The Pearson Correlation results from our three different runs obtained using trigram embeddings are presented in Table 2. These results represent the ASOBEK submission to SemEval-2016 Task 1. It is noted that Run2 and Run3 generally appear to outperform Run1, with Run2 performing best overall. The best individual result is obtained on the Post-editing dataset. This result is ranked above the median for results reported on this sub-task. The correlation scores evidence variable performance across the individual datasets. Most notable is that the results from each of the runs applied on the Question-Question dataset are much lower relative to the other categories. In spite of this, the overall performance is 0.6178 with Run2.

Datasets (Trigrams)	**Run1**	**Run2**	**Run3**
Overall	*0.5956*	***0.6178***	*0.6143*
Answer-Answer	0.4269	**0.5228**	0.4792
Headlines	0.6790	0.6374	**0.6865**
Plagiarism	0.7572	**0.7852**	0.7778
Post-editing	0.8195	**0.8425**	0.8409
Question-Question	0.2618	**0.2635**	0.2480

Table 2: Pearson Correlation Results using character trigrams

Experiments were also conducted for bigrams and 4-grams. Results are shown in Table 3. Highlighted scores indicate cases where performance exceeds that obtained using trigram embeddings.

As for trigrams, better results are generally obtained for Run2 and Run3. For 4-grams, the overall performance of Run3 actually exceeds that based on trigrams. Across all of the systems, results on the Question-Question dataset depress the overall performance significantly. This is especially evident for Run2 of the 4-gram dataset, where there is little correlation evident between system scores and human judgements. In contrast, using bigrams Run2 produces our best result for this dataset. Examination of the data indicates that this may be due to the particular form of the questions. For example, character n-grams generated from the following pair will contain many tokens that are not informative in discriminating meaning:

What should I look for in a jump rope?
What should I look for in a running shoe?

Datasets (Bigrams)	**Run1**	**Run2**	**Run3**
Overall	*0.5340*	*0.5996*	*0.5784*
Answer-Answer	0.3869	0.4547	0.4520
Headlines	0.6422	0.6697	0.6668
Plagiarism	0.6204	0.7088	0.6582
Post-edit.	0.7439	0.8168	0.7879
Question-Question	0.2768	**0.3482**	0.3270
Datasets (4-grams)	**Run1**	**Run2**	**Run3**
Overall	*0.6088*	*0.5567*	***0.6183***
Answer-Answer	0.4587	**0.5375**	0.4927
Headlines	0.6757	0.6023	0.6831
Plagiarism	0.7743	0.7507	0.7796
Post-editing	0.8295	0.8356	0.8417
Question-Question	0.3057	0.0574	0.2943

Table 3: Pearson Correlation Results using character bigrams and 4-grams

7 Conclusions

A method for STS based on embeddings of character n-grams generated by a CBOnG model was introduced. This is the only study that we are aware of that utilizes embeddings of character n-grams to build representations of sentences. The study presents preliminary results showing that the approach can successfully help identify semantic similarity of sentence pairs.

Using our method, we observe significant variations in performance across the STS Core datasets. In particular, performance is generally poor for the

Question-Question dataset. This suggests that it may improve performance to weight the contributions of the embeddings according to the informativeness of the associated n-grams. We intend to consider this in future experiments.

Acknowledgements

We are grateful for the very constructive comments from the reviewers on an earlier draft of this paper.

References

Agirre, Eneko et al. (2014) 'SemEval-2014 Task 10: Multilingual Semantic Textual Similarity'. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*.

Agirre, Eneko et al. (2015) 'SemEval-2015 Task 2 : Semantic Textual Similarity , English , Spanish and Pilot on Interpretability'. *SemEval2015*.

Agirre, Eneko et al. (2016) 'SemEval-2016 Task 1: Semantic Textual Similarity - Monolingual and Cross-lingual Evaluation'. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016)*. San Diego, USA.

Agirre, Eneko, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. (2012) 'Semeval-2012 task 6: A pilot on semantic textual similarity'. *Proceedings of the 6th International Workshop on Semantic Evaluation, in conjunction with the First Joint Conference on Lexical and Computational Semantics*.

Agirre, Eneko, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. (2013) '*SEM 2013 shared task : Semantic Textual Similarity'. *The Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)*, vol. 1.

Blacoe, William, and Mirella Lapata. (2012) 'A Comparison of Vector-based Representations for Semantic Composition'. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*.

Eyecioglu, Asli, and Bill Keller. (2015) 'ASOBEK : Twitter Paraphrase Identification with Simple Overlap Features and SVMs'. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado.

He, Hua, Kevin Gimpel, and Jimmy Lin. (2015) 'Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks'. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal.

Huang, Po-sen et al. (2013) 'Learning Deep Structured Semantic Models for Web Search using Clickthrough Data'. *Conference on Information and Knwoledge Management (CIKM-2013)*.

Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M. Rush. (2015) 'Character-Aware Neural Language Models'. *CoRR* 1508.06615. http://arxiv.org/abs/1508.06615.

Le, Qv, and Tomas Mikolov. (2014) 'Distributed Representations of Sentences and Documents'. *International Conference on Machine Learning - ICML 2014* 32:1188–1196.

Ling, Wang et al. (2015) 'Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation'. *CoRR* 1508.02096. http://arxiv.org/abs/1508.02096.

Mikolov, Tomas, Greg Corrado, Kai Chen, and Jeffrey Dean. (2013a) 'Efficient Estimation of Word Representations in Vector Space'. *Proceedings of the International Conference on Learning Representations (ICLR 2013)*.

Mikolov, Tomas, Ilya Sutskever, and Anoop Deoras. 2012. *Subword language modeling with neural networks*.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. (2013b) 'Linguistic regularities in continuous space word representations'. *Proceedings of NAACL-HLT* 746–751.

Rehurek, Radim, and Petr Sojka. (2010) 'Software framework for topic modelling with large corpora'. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* 45–50.

Yin, Wenpeng, and Hinrich Schütze. (2015) 'Convolutional Neural Network for Paraphrase Identification'. *Naacl 2015*.

Zarrella, Guido, John Henderson, Elizabeth M Merkhofer, and Laura Strickhart. (2015) 'MITRE : Seven Systems for Semantic Similarity in Tweets'. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado.