Igevorse at SemEval-2018 Task 10: Exploring an Impact of Word Embeddings Concatenation for Capturing Discriminative Attributes

Maxim Grishin
Innopois University
Innopolis, Russia
m.grishin@innopolis.ru
igevorse@gmail.com

Abstract

Semantic differences extraction is a challenging problem in Natural Language Processing and its solution is necessary for a realistic semantic representation as similarity information is not sufficient to capture individual aspects of meaning. This paper presents a comparison of several approaches for capturing discriminative attributes and considers an impact of concatenation of several word embeddings of different nature on the classification performance. A similarity-based method is proposed and compared with machine learning approaches. It is shown that this method outperforms others on all the considered word vector models and there is a performance increase when concatenated datasets are used.

1 Introduction

Detecting semantic similarity is done well by state-of-the-art models. However, if the model is only good at similarity detection, it would have a limited practical usage (Krebs et al., 2018), since the task of *understanding* the semantics of words cannot be done without capturing semantic differences.

Semantic difference is a ternary relation between two concepts $(apple, banana)$ and a discriminative feature (red) that characterizes the first concept but not the other. Semantic difference detection is a binary classification task: given a triple $(apple, banana, red)$, the task is to determine whether it exemplifies a semantic difference or not (Krebs et al., 2018). In this paper two concepts and a discriminative attribute $attr$ are represented as $(word_1, word_2, attr)$.

This research was done during the participation in ”Capturing Discriminative Attributes” task of SemEval 2018 competition.

The paper is organized as follows. Section 2 describes the methods used. Section 3 shows the results and analyzes them. Section 4 mentions future directions. Section 5 concludes the paper.

2 Methods

There were several approaches considered: classical machine learning algorithms and a similarity-based model.

2.1 Data Preparation

Dataset is provided by SemEval 2018 challenge organizers.

- The train set consists of 17501 automatically generated samples of the form $(word_1, word_2, attr, y)$, where y is a binary target variable indicating whether $attr$ is a discriminative attribute for $word_1$ and $word_2$. Classes are imbalanced: there are 63.83% samples for class 0 (not a discriminative attribute) and 36.16% for class 1 (is a discriminative attribute).
- The validation set contains 2722 manually curated samples of the same form. Classes are almost balanced: 50.1% of samples have class 0, 49.9% - class 1.
- There are 2340 samples in the test set, 55.3% for class 0, 44.7% for class 1.

Each triple $(word_1, word_2, attr)$ was converted to a numeric vector using pre-trained word embeddings. These vectors form a vector space, such that words that share common contexts in the corpus are located in close proximity to one another in space.

Three word embedding models were used:

1. Google News (Mihltz, 2017) corpus word vector model (3 million 300-dimension English word vectors). The disadvantage of this

model is that despite having a lot of words, it includes misspellings and multiple cases of the same word (McCormick, 2017). Its vocabulary does not contain all words from the dataset, so 45 samples (1.27%) are missing.

2. Wikipedia 2014 + Gigaword 5 (Pennington et al., 2014) dataset has 400 thousand 300-dimension word vectors. It contains almost all words from the dataset: only 4 samples (0.11 %) are missing.

3. Concatenated word embeddings of different nature, i.e. collected on different corporas using different methods. It consists of concatenated word vectors from both Google News and Wikipedia models. Thus, each word is represented as a 600-dimensional vector.

The problem of missing words was solved by replacing them with another spelling or a synonym. For each triple $(word_1, word_2, attr)$ corresponding word vectors were concatenated, so each triple is converted to a 900-dimensional vector when using Google News or Wikipedia word vectors, and 1800-dimensional for concatenated word embeddings.

2.2 Machine Learning Approaches

There were several machine learning classification approaches chosen for comparison: logistic regression, stochastic gradient descent (SGD) classifier, k-nearest neighbors classifier and artificial neural network.

The best parameters that maximize F_1 score on the validation set:

- **Logistic regression** with L_2 regularization with regularization strength set to 10.

- **Stochastic gradient descent classifier** with perceptron loss and regularization term set to 1e-05.

- **K-nearest neighbors** with k = 1 using Manhattan distance metric and weighting points by the inverse of their distance.

- The multilayer perceptron **neural network** model was built using Keras with TensorFlow as a backend. Structure of this network is described in Table 1.

Layer	# of inputs	Activation function
Input	900	-
Dense	128	tanh
Dense	64	relu
Dense	1	sigmoid

Table 1: Structure of the neural network.

2.3 Similarity-based Approach

Another approach is to derive an interpretable algorithm based on knowledge about word semantics. The intention is to use word similarities for distinguishing discriminative attributes while keeping the model as simple as possible. Cosine similarity between words a and b is represented as the cosine between corresponding word vectors A and B.

$$sim(a, b) = cos(A, B) = \frac{A \cdot B}{\|A\|\|B\|}.$$

For each given triple $(word_1, word_2, attr)$ similarities of $attr$ with $word_1$ and $attr$ with $word_2$ are computed. Then obtained similarities are compared using a threshold t. If the gap between them is big enough, i.e.

$$sim(word_1, attr) > sim(word_2, attr) + t,$$

$attr$ is treated as a discriminative attribute of $word_1$. It means that the $attr$ word vector is much closer to $word_1$ than to $word_2$ in vector space. Thus, this model has only one tunable hyperparameter: t. Dependency of F_1 score on the threshold is shown in Figure 1.

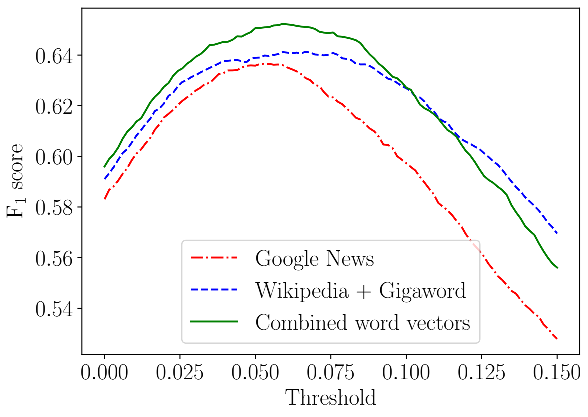

Figure 1: Dependency of F_1 score on the threshold in similarity-based model.

Model \ Embeddings	Google News	Wikipedia	Concatenated
K-nearest neighbors	0.4778	**0.4832**	0.4809
SGD classifier	**0.5211**	0.5036	0.5169
Logistic regression	0.4843	0.5036	**0.5120**
Neural network	0.5169	**0.5625**	0.5440
Similarity-based	0.6066	0.6126	**0.6131**

Table 2: Experimental results on the validation set (F_1 score).

3 Results and Discussions

3.1 Experimental Results

Models are evaluated on F_1 measure.

Figure 1 shows that the behavior of the similarity-based model is highly dependent on the selected word vectors model. Thresholds, learned from the train set for Google News, Wikipedia and concatenated word vectors are 0.053030, 0.066667 and 0.060606 correspondingly.

Experimental results on the validation set are presented in Table 2. K-nearest neighbors showed the worst F_1 score, while neural network is the best among machine learning methods. The proposed similarity-based method outperforms all other models on all considered word embeddings.

Word embeddings with the highest F_1 score on the validation set were chosen for the final comparison: Google News for SGD classifier, Wikipedia for k-nearest neighbors and neural network, concatenated embeddings for logistic regression and similarity-based model. The results on the test set are presented in Table 3. K-nearest neighbors has the smallest F_1 score. In contrast with the performance on the validation set, the result of neural network is noticeably worse, which means that overfitting took place. Logistic regression performed better than SGD classifier, while similarity-based method showed the highest score.

Model	Best F_1 score
K-nearest neighbors	0.502
SGD classifier	0.515
Logistic regression	0.527
Neural network	0.503
Similarity-based	**0.646**

Table 3: Experimental results on the test set.

3.2 Error Analysis

In this section error analysis of similarity-based model is provided. During the evaluation on the test set 65.5% of samples were classified correctly, while 34.5% were not. Predicted classes are imbalanced: 1436 (61.4%) samples were classified as 0 and 904 (38.6%) samples as 1.

Considering misclassified samples, 332 of them were assigned class 1 while it should have been 0, whereas 475 of them got label 0, when it should have got 1. As we can see, the model is more likely to consider attributes as non-discriminative.

108 distinct attributes (130 samples) were misclassified completely. 270 distinct attributes (480 samples) were classified 100% correctly. 70 attributes (180 samples) were classified 50% correctly.

Attributes could be divided in several categories. For example, there are attributes representing colors: ‘black’, ‘brown’, ‘red’, ‘blue’ and ‘yellow’. It worth mentioning that 43.36% of samples with color attributes were misclassified, which is more than 34.5% of misclassified samples for the whole test set.

As can be seen from Table 4, other categories of attributes have 33.4% of misclassified samples.

Attribute	Misclassified	Total occurences
Color	111 (43.36%)	256
Other	696 (33.4%)	2084
Total	807 (34.5%)	2340

Table 4: Error analysis of attribute categories.

4 Future Work

It was shown that there is a performance increase of the similarity-based model when concatenated word vectors are used. Training a Word2Vec model specifically for the task instead of using pre-trained models can solve the mentioned problem of missing words and multiple cases of the same word in word embeddings.

According to SemEval Task 10 organizers, training set contains noisy data, which was not verified by humans. Another potential improvement is training models only on validation dataset, since it was created manually and should not have noise.

It was discovered that samples with color attributes have higher misclassification rate than other samples. There are proposed solutions for learning discriminative properties of images (Lazaridou et al., 2016), which could be combined with a text-based approach to derive a multimodal classifier.

It also worth analyzing other categories of attributes and their misclassification rate.

5 Conclusion

This paper presents several approaches for capturing discriminative attributes. The main contribution is the proposed similarity-based method, which is interpretable and takes into account the semantic similarity of words. This method is compared with machine learning methods, such as Logistic regression, SGD classifier, KNN and Multilayer perceptron neural network. Experiments on three pre-trained word vector models show that similarity-based method outperforms others. It was discovered that concatenation of word embeddings of different nature leads to a quality improvement for several methods.

References

Alicia Krebs, Alessandro Lenci, and Denis Paperno. 2018. Semeval-2018 task 10: Capturing discriminative attributes. In *Proceedings of the 12th international workshop on semantic evaluation (SemEval 2018)*.

Angeliki Lazaridou, Nghia The Pham, and Marco Baroni. 2016. The red one!: On learning to refer to things based on their discriminative properties. *CoRR*, abs/1603.02618.

Chris McCormick. 2017. Google's pre-trained word2vec model in Matlab. https://github.com/chrisjmccormick/word2vec_matlab. [Online; accessed 15-April-2018].

Mrton Mihltz. 2017. Pre-trained Word2vec Google News corpus. https://github.com/mmihaltz/word2vec-GoogleNews-vectors. [Online; accessed 15-April-2018].

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.