Learning Sentiment-Specific Word Representations from Tweets

[Data Warehousing and Data Mining - Team 8]

ABSTRACT

This paper provides a method that makes use of deep learning to learn word representations for Twitter sentiment classification. Most existing algorithms for learning continuous word representations typically only model the syntactic context of words, while ignoring the sentiment of text. This is problematic for sentiment analysis as words with similar syntactic context but opposite sentiment polarity, such as good and bad, are mapped to neighboring word vectors. We address this issue by learning *Sentiment Specific Word Embedding (SSWE)*, which encodes sentiment information in the continuous representation of words. Specifically, we develop neural networks that effectively incorporate the supervision from sentiment polarity of text (tweets) in their loss functions. Following this, we make use of these word embeddings to train an SVM to predict the sentiment polarity of tweets. We observe that the performance of only using SSWE as features is comparable to the stateof-the-art hand-crafted features which verifies the effectiveness of the sentiment-specific word embedding.

Keywords

SSWE; Deep Learning; Sentiment Analysis

1. INTRODUCTION

The main objective is to classify the sentiment polarity of a tweet as positive, negative or neutral. The majority of existing approaches employ machine learning algorithms to build classifiers from tweets with manually annotated sentiment polarity. Under this direction, most studies focus on designing effective features to obtain better classification performance. Feature engineering is important but laborintensive. It is therefore desirable to discover explanatory factors from the data and make the learning algorithms less dependent on extensive feature engineering. We do this by using neural networks to learn the word representations, which can then be used as features for SVM training. The proposed neural network for learning sentiment-specific word embedding is an extension of the traditional C&W model. These word representations have been obtained by incorporating both the syntactic as well as the sentiment polarity. This is then used as the feature set for training a SVM classifier.

2. SENTIMENT-SPECIFIC WORD EMBEDDING

The proposed neural network for learning sentiment-specific word embedding is an extension of the traditional C&W model. Unlike C&W model that learns word embedding by only modeling syntactic contexts of words, the proposed SSWE captures the sentiment information of sentences as well as the syntactic contexts of words. Given an original (or corrupted) ngram and the sentiment polarity of a sentence as the input, SSWE predicts a two-dimensional vector for each input ngram. In this section, we present the details of the algorithm in learning sentiment-specific word embedding (SSWE) of tweets for sentiment classification.

Given an original (or corrupted) ngram and the sentiment polarity of a sentence as the input, SSWE predicts a two-dimensional vector for each input ngram. The two scalars (f_0^u, f_1^u) stand for language model score and sentiment score of the input ngram, respectively. The training objectives of SSWE are that (1) the original ngram should obtain a higher language model score $f_0^u(t)$ than the corrupted ngram $f_0^u(t^r)$, and (2) the sentiment score of original ngram $f_1^u(t)$ should be more consistent with the gold polarity annotation of sentence than corrupted ngram $f_0^u(t^r)$. The loss function of SSWE is the linear combination of two hinge losses,

$$loss_u(t, t^r) = (alpha)loss_cw(t, t^r) + (1-(alpha))loss_us(t, t^r)$$

where $loss_cw(t, t^r)$ is the syntactic loss , and $loss_us(t, t^r)$ is the sentiment loss as described by the following equations

$$loss_cw(t, t^r) = max(0, 1 - f^cw(t) + f^cw(t^r))$$

where t is the original ngram, t^r is the corrupted ngram, $f^cw()$ is a one-dimensional scalar representing the language model score of the input ngram, and

$$loss_us(t, t^r) = max(0, 1-(delta)_s(t)f_1^u(t)+(delta)_s(t)f_1^u(t^r))$$

2.1 Model Training

We learn sentiment-specific word embedding from a given dataset of 1 million tweets. We tokenize each tweet, remove the @user and URLs of each tweet, and filter the tweets that are too short (< 7 words). We train SSWE by taking the derivative of the loss through backpropagation with respect to the whole set of parameters, and use AdaGrad to update the parameters. We empirically set the window size as 3, the embedding length as 50, the length of hidden layer as 20 and the learning rate of AdaGrad as 0.1 for all baseline and our models. The contexts of unigram are the surrounding unigrams. The loss function is the linear combination of hinge losses : syntactic and sentiment loss.

$$loss_u(t, t^r) = (alpha)loss_cw(t, t^r) + (1-(alpha))loss_us(t, t^r)$$

Here (alpha) is the weighting score of syntactic loss of SSWE and trades-off the syntactic and sentiment losses. SSWE performs better when (alpha) is in the range of [0.5, 0.6], which balances the syntactic context and sentiment information. The model with (alpha)=1 stands for C&W model, which only encodes the syntactic contexts of words. The sharp decline at (alpha)=1 reflects the importance of sentiment information in learning word embedding for Twitter sentiment classification.

We used skip gram model to retrieve the context of words. For each word we have a window size of 3 and we retrieve all the positive samples of the word in that window. For negative samples we take any five words not present in its window from the vocabulary. We then send the word, its context and the labels to the neural network to obtain the word embeddings. This gives us syntactic context. For sentiment polarity we again send the context for each word, but now instead of label we send polarity. The polarity of positive samples is that of tweet and of negative samples is !polrity. We use the same Word Lookup and this now gives us word embeddings which incorporates both syntactic context and sentiment polarity.

The whole process is repeated then for bigrams and trigrams and we observe that the bigram and trigram embeddings consistently improve the performance of Twitter sentiment classification. The underlying reason is that a phrase, which cannot be accurately represented by unigram embedding, is directly encoded into the ngram embedding as an idiomatic unit. A typical case in sentiment analysis is that the composed phrase and multiword expression may have a different sentiment polarity than the individual words it contains, such as not [bad] and [great] deal of (the word in the bracket has different sentiment polarity with the ngram).

2.2 SVM Training

Using the embedding obtained from the neural network, we find the embedding of each tweet in the training and the test dataset by taking the average of the embedding for each word in the tweet. Using these embeddings as features, we train a SVM, and then test it on the test data to predict the polarity of each tweet.

3. RESULTS

The performance of only using SSWE as features is comparable to the stateof-the-art hand-crafted features which verifies the effectiveness of the sentiment-specific word embedding. Experimental results further demonstrate that sentiment-specific word embeddings are able to capture the sentiment information of texts and distinguish words with opposite sentiment polarity, which are not well solved in traditional neural models like C&W and word2vec. SSWE outperforms MVSA and ReEmb by exploiting more context information of words and sentiment information of sentences, respectively. The *macro-f1* score for the neural network implemented is 72.83

4. CONCLUSION

In this report, we learn continuous word representations as features for Twitter sentiment classification under a supervised learning framework. We show that the word embedding learned by traditional neural networks are not effective enough for Twitter sentiment classification. These methods typically only model the context information of words so that they cannot distinguish words with similar context but opposite sentiment polarity (e.g. good and bad). We learn sentiment-specific word embedding (SSWE) by integrating the sentiment information into the loss functions of the neural network.

This paper that have been implemented demonstrates a nice proof that learnt high level features produced by deep learning leads to lower classification error compared to the-state-of-the-art of two-level classifiers. One important motivation for using the high level features is domain adoption problem that can be spesifically addressed by Deep Learning.

To improve our results, we can also use additional features for training the SVM , such as

- polarity score of word/ngram from a lexicon
- number of positive/negative emoticons
- number of all capital words (WOW)
- number of negation words (no, none, nobody)
- number of elongated words
- number of elongated punctuations (!!, ??)
- number of each class of POS to improve the accuracy.

We can also obtain sentiment for a specific target from a tweet. Sometimes, people may mention several entities (or targets) in one tweet, which affects the availabilities for most of existing methods. For example, the tweet "@ballmer: windows phone is better than ios!" has three targets (@ballmer, windows phone, and ios). The user expresses neutral, positive, and negative sentiments for them, respectively. If target information is ignored, it is diffi- cult to obtain the correct sentiment for a specified target.

5. REFERENCES

Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification