Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification

Motivation

- Introduce semantic knowledge and expand short texts;
- This is achieved by using word embeddings in the form of a model pre-trained over a large-scale external corpus;
- The semantics of the original short texts which are being classified is preserved by integrating text understanding and vectorization into a joint framework

Main tasks and objectives

- The main problem/task being resolved in the paper is **classification of short texts**:
  - micro-blogs
  - product reviews (including the Yelp dataset used for our Capstone class!)
  - search snippets
  - short messages
- This is important because, as we all have realized during this class, it enables **intelligent information retrieval** including intent understanding, development of question answering systems, etc.
  - This is difficult (=interesting) since we deal with the **data sparsity** problem - short texts do not have sufficient contextual information
  - Bag-of-words (**BoW) models cannot be applied directly** because they ignore the order and semantic ties between words

Basic idea and contributions

- To overcome the data sparcity weakness, the authors propose a unified framework which expands short texts by using word embedding clustering and CNN.
- Summary of main contributions of the article:
  - ➢ Via fast clustering based on density peaks, discover **semantic cliques** which are then used for supervised learning to extract fine-tuned semantics
  - ➢ Define **multi-scale semantic units** (the main novelty of this work) and compute their representations by using a one-dimensional convolution-like operation (using word embeddings from context)
  - ➢ Select the restricted **nearest word embeddings (NWEs)** in embedding spaces and construct **expanded matrices**
  - ➢ Feed the projected matrix combined with the expanded matrices into a **CNN** in parallel to extract high-level features and receive classifier output

Background

- Many popular solutions have been proposed for the data sparcity problem. The most recent neural network language models include:
  - **Skip-gram model** - efficiently learns high-quality word embeddings from large-scale unstructured text data [1, 2]
  - **Paragraph vector model** - learns fixed-size features for documents with variable length [3]
  - **Dynamic convolutional neural network** (DCNN) - used for modeling sentences, most related to this article [4]
  - **Deep convolutional neural network** (DNN) - extracts lexical and sentence level features, used for relation classification [5]
  - **Recursive neural network** (RNN) - effective in sentiment prediction [6]
  - **Long Short-Term Memory** (LSTM) algorithm - improves recurrent neural network language models; currently widely used in spoken language understanding and sequence prediction [7]
- The above methods capture high-order n-grams and word order, but the small length of short texts still strongly affects the classification performance. That is why a novel method to expand the short text representation is needed

Theory: semantic composition (1)

Word embeddings are trained to predict the surrounding words (e.g. Skip-gram model) => they encode an implicit distribution of the context.
Co-occurrence information can effectively describe each word
Word embeddings capture various syntactical and semantic relationships: INSERT capital, football player
Words often appearing in similar contexts have similar vector representations and form clusters or semantic cliques in embedding spaces (see Fig. 1)

Theory: semantic composition (2)

Methods based on semantic composition can be used to discover latent semantics and obtain vector representations of phrases or sentences (see Fig. 2.)

Semantic composition based on co-occurrences is very useful for language understanding, e.g. to analyze phrase similarities or as input features for classifiers

Theory: word embedding clustering

- Semantic cliques can be effectively found in embedding spaces using clustering
- BUT, the vocabulary size of word embeddings is usually large (e.g. the public Word2Vec dataset contains 3 million words)
- THEREFORE, a fast algorithm based on searching for **density peaks** is used for these purposes
- Two properties of data point i are computed, include: local density ρi and distance δi from points of higher density
- INSERT equations 3, 4, 5
- A simple clustering example is shown in Fig. 1 above: the decision graph shows the two properties ρ and δ of each word embedding. The word embeddings with large ρ and δ are considered cluster centers and labeled using the corresponding words from the decision graph

Implementation: projected matrix

- PROJECTED MATRIX (PM) = **EXISTING DATA** IN THE FORM OF A VECTOR REPRESENTATION OF THE SHORT TEXT ITSELF

Short text S = {w1, w2, … wN}, D=finite vocabulary of size v (pretrained model), d=dimension of each word embedding

S is transformed into PM=LT x index(S) (PM c RdxN)

Where:

LT is a lookup table LT c Rdxv containing pretrained word embedding that encode word-level information (each word=word embedding has a dimension d, therefore LT contains embeddings for every word in the pretrained vocabulary D)

and index() is a function transforming each word in S into a one-hot representation that corresponds to the vocabulary D in LT

Implementation: expanded matrices

- EXPANDED MATRICES = **NEW DATA** ADDED TO THE SHORT TEXT REPRESENTATION TO EXPAND IT AND IMPROVE CLASSIFICATION PERFORMANCE
- The method proposed by the authors aims to introduce external knowledge by using high-quality pretrained word embeddings to obtain more contextual information for short texts in order to get better classification results (see Fig. 3)
- First, a set of sentence-wide semantic units is discovered by using the inner product of the projected matrix and a special window matrix of varying width m.
- Select only meaningful semantic units - each meaningful semantic unit is assumed to have at least one close embedding neighbor.
- In the associated semantic cliques, find the restricted nearest word embeddings (NWE); restricted - because the distance to them should be smaller than a preset Euclidean distance.
- Restricted NWEs for one value of the window matrix width constitute one expanded matrix (see Fig. 4)
- Multiple expanded matrices can be computed in parallel by increasing the window matrices - this multi-scale contextual information is used to expand the input short text S MAKE TWO SLIDES, INSERT FIGURES 3 AND 4

Implementation: convolution layer

- Once the short text has been expanded, a convolutional layer is used to extract local features.
- The projected matrix PM and expanded matrices EMs are fed into it in parallel.
- Kernel matrices of weights k c R2xn with certain widths n are utilized to calculate the convolution with the input matrices (Fig. 3)
- In order to obtain a feature map, the convolutional operation is defined as the inner product of the kernel matrices k with pair-wise rows of each input matrix denoted by X:
- INSERT EQUATIONS 9 AND 10

Implementation: K-max pooling

- K-max pooling is used to obtain feature representations of input short texts
- The feature map from the previous step is down-sampled to capture the most relevant global features of a fixed-length and enable the output features that can be adapted to various classifiers
- A non-linear subsampling K-max pooling function returns a subsequence of K maximum values, where K is a hyper-parameter optimized during training
- Then, a tangent function is applied to perform non-linear
- and element-wise transformations

Implementation: output layer

- Once the short text xi passes through the above sequence of layers, the following linear transformation is performed:
- EQUATION 13
- Where f^ are semantic representations and Wz are weights with which the output layer is fully connected
- The resulting output vector contains possible scores for corresponding classes (its length equals the number of classes)
- Then, the following softmax function is used to transform the score
- vector into a probability distribution:
- EQUATION 14
- The class cj with maximum p(cj | xi, Wz) is selected as the predicted label for xi

Implementation: network training

- The network is trained with the aim to minimize the cross-entropy of the predicted distributions and the actual distributions for all samples
- It is learned with mini-batches of samples by back-propagation
- During training the
- The set of parameters θ = {k, Wz} needs to be optimized in the process of training of the neural network, where k is the kernel weights from the convolutional layer and Wz is the connective weights from the output layer
- The objective function is based on the cross-entropy loss function and an L2 regularization parameter that is introduced to prevent over-fitting

Experiments: datasets

- Google Snippets: consists of 10,060 training snippets and 2280 test snippets from 8 categories, as shown in the table. On average, each snippet has 18.07 words.
- TREC: contains 6 different question types, including LOC., NUM., ENTY., and so on. The training dataset consists of 5452 labeled questions, and the test dataset consists of 500 questions
- The experiments were conducted with the lookup table initialized with three different pretrained word embeddings: Senna, Glove, and Word2Vec
- During the experiments, the words from the short texts that are not present in the corresponding pretrained vocabulary were simply discarded, since they are often low-frequency tokens
- INSERT 3 TABLES (one here and 2 on the next standalone slide?)

Experiments: comparison with state-of-the-art methods

- INSERT TABLE 4
- Conclusion: as a whole, the proposed method CCNN achieves the best performance which validates its effectiveness.
- Future improvement can be obtained by supervised feature down-sampling, task-specific embeddings learning, and embedding affinity measurement in vector spaces

Summary

- This paper proposes a novel semantic hierarchy to model and classify short texts
- A lookup table is initialized using pretrained words embeddings
- This introduces additional knowledge and enables the estimation of word affinities by computing the Euclidean distance between vector representations.
- Multi-scale semantic units are computed for short texts expansion.
- Similar words are grouped together in the embedding spaces which helps learning algorithms to achieve better performance.
- Experimental results on open benchmarks validated the effectiveness of the proposed method.
- Possible future improvements have been mentioned on the previous slide

Opinion

1) Convolutional neural networks offer significant flexibility and power based on their sophisticated architectures. Although the proposed CNN is well thought of, and the entire framework provides superior results compared to other state-of-the-art, I would still think about how the power of CNN can be utilized better to achieve the required objectives because the current CNN architecture seems to be somewhat simple

2)

3) The paper does not mention anything about optimizing m = the window matrix width. Having a reasonable cutoff value may save some computing time when you know that any further increase of m will not provide a significant number of useful features

4)

5) As the authors mentioned, learning task-specific embeddings may be an interesting way to improve the performance even further.

Key References

- [1] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space arxiv:hepth/1301.3781.
- [2] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, Adv. Neural Inf. Process. Syst. (2013) 3111–3119.
- [3] Q.V. Le, T. Mikolov, Distributed representations of sentences and documents, arxiv:hepth/1405.4053.
- [4] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network
- for modelling sentences, arxiv:hepth/1404.2188.
- [5] D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao, Relation classification via convolutional deep neural network, in: Proceedings of the 25th International Conference on Computational Linguistics, 2014, pp. 2335–2344
- [6] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment tree-bank, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, vol. 1631, 2013, p. 1642.
- [7] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.