Add Name Entity Recognizer to Convolutional Neural Network

Guangyu Lin
UTEID: gl8429
glin@utexas.edu

Ge Gao
UTEID: gg24984
gegao1118@utexas.edu

Huihuang Zheng
UTEID: hz4674
huihuang@utexas.edu

1. Introduction

Traditionally, *word2vec* [5] will map same words to same vectors. Some information may be lose. For example, word "Apple" will be mapped by *word2vec* ignoring the entity "Apple" means a kind of fruits or the company. **We are going to propose a framework of using word2vec + NER as input of CNN**. We think our model can get better performance in NLP tasks involving name entity. Our framework will be evaluated in TREC [4] question dataset, which involves classifying a question into 6 question types.

2. Proposed Method

2.1. Traditional Method

Traditional method maps words w to vectors:

$$word2vec : w \rightarrow v$$

Then the $v \in \mathbb{R}^k$ is the k-dimensional word vector corresponding to the i-th word in the sentence. A sentence of length n (padded where necessary is represented as)

$$v_{1:n} = v_1 \oplus v_2 \oplus v_3 ... \oplus v_n$$

In general, let $v_{i:i+j}$ refers to the concatenation of word vectors $v_i, v_{i+1}, ..., v_{i+j}$. A convolutional filter of Convolutional Neural Network (CNN) [3] takes fixed size of input of $\mathbb{R}^{kh}$. The filter is applied to each possible window of words of the sentence $\{x_{1:h}, x_{2:h+1}, ..., x_{n_h+1:n}\}$ and extract features, do max pooling, and full connected layer to get output of CNN.

2.2. Our Approach

We are going to add *Name Entity Recognizor* (NER) to the word expression v. Let our word vector be $v' \in \mathbb{R}^{k+1}$. We are going to add one dimension to *word2vec* expression. Let

$$v' = [v, e]$$

where v is the output vector of *word2vec* and e is the label number for name entity. Then we will modify CNN architectures in Kim's paper [3]. Their CNN convolutional filters take input of $\mathbb{R}^{kh}$ and we will change input size into $\mathbb{R}^{(k+1)h}$.

3. Evaluation

3.1. Software Detail

We will use *word2vec* trained model from Mikolov [5] on 100 billion words of Google News and are publicly available. We will build our framework using NER from Stanford NER software [2] and deep learning tool from Theano [1]. Since Kim's work[3] has Github published code using Theano [1], it will be easy for us to build baseline to compare and modify their CNN architecture to ours. We need to train a new CNN for our input and compare the performance to Kim's CNN.

3.2. Dataset

We are going to compare accuracy of our method and Kim's CNN in TREC [4]. The task involves classifying a question into 6 question types (whether the question is about person, location, numeric information, etc.). We choose this dataset because of two reasons. The fact that Name Entity Recognizer can classify person, location may help us in this task. Second, in Kim's paper [3], he also used this database and reported his accuracy so it will be easy for us to compare fairly.

References

[1] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[2] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[3] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[4] X. Li and D. Roth. Learning question classifiers: the role of semantic information. *Natural Language Engineering*, 12(3):229–249, 2006.

[5] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.