Mini-Project: Title/Brand Entity Recognition

Hao Zhang

zhanghao.py@foxmail.com

1 Introduction

A big challenge for Shopee is to extract the product name entity from product title. For example, from a product title **Apple iPhone 6 (A1586) 16GB golden 4G**, we shall extract the core term **Phone 6**, in order to serve the purpose of search, recommendation and keywords extraction etc.

Task: Given a title s of a product p, the output should be a core term t and a brand b can fully represent the product p.

2 Methods

2.1 Word vectors and context windows

Assume you have the following sequence of words in your training set.

E.g.: **Apple iPhone 6 (A1586) 16GB golden 4G.**

Each word is associated with a label y defining that word as either CoreTerm or not any named entity 0. We define CoreTerm to be class 1 ($y = 1$) and 0 to be class 0. Each word is also associated with an index into a vocabulary, so the word **Apple** might have index 241.

All word vectors are n-dimensional and saved in a large matrix $L \in \mathbb{R}^{n \times V}$, where V is the size of the vocabulary. We have used the one-hot encoding and word2vec embedding to extract the word feature respectively. The vectors are $n = 2391$ dimensional for one-hot encoding and $n = 128$ dimensional for word2vec embedding.

Assume that at the first location of the sentence we have vector x_1, which was retrieved via the index of the word **Apple**. Similarly, we can represent the whole sentence as a list of vectors $(x_1, x_2, ..., x_7)$. When we want to classify the first and last word and include their context, we will need special beginning x_s and end tokens x_e, which we define as <s> and </s> respectively. If we use a context size of C, we will have to pad each input of the training corpus with C man of these special tokens. More specifically, $C = 1$ stands for one-gram and $C = 3$ represents tri-gram. We define the vector corresponding to the begin padding as x_s and for the end padding as x_e. Hence, we will get the following sequence of vectors:

$$(x_s, x_1, x_2, ..., x_7, x_e) \tag{1}$$

For this training corpus, we have the following windows that we will give as input to the neural network:

$$\{[x_s, x_1, x_2], [x_1, x_2, x_3], [x_2, x_3, x_4], ..., [x_6, x_7, x_e]\} \tag{2}$$

Each window is associated with the label of center word, So for this sequence, we get the labels $(0, 1, 1, 0, 0, 0, 0)$. Note that the model just sees each window as a separate training instance. Thus, we can extract training data matrix $X \in \mathbb{R}^{7\times 3m}$ and training label $y \in \mathbb{R}^7$ for this sentence. Also, training data need to be a tensor $X \in \mathbb{R}^{7\times 3\times m\times 1}$ if we use convolutional neural network for text classification, where 1 represents channel-last.

2.2 Models and Metrics

Model Overview The idea of the model is that in order to classify each word, you take as input that word's vector representation and the vector representations of the words in its context. These vectors constitute our features. These features are the input to a neural network which is a function that will first transform them into two more than 'hidden' vector and then use that vector to predict a probability of how likely the window's center word is of a certain class. Besides, we will carry out a series of models to recognise entity for ER task, including Logistical Regression, Deep Neural Network and Convolutional Neural Network, which are widely used for machine learning tasks.

Metrics Precision Rate, Recall Rate and F1-Score are the best metrics, especially in NLP task, for binary classification task.

3 Experiments

3.1 Dataset

Shopee provides us with a sample of product titles and their corresponding core terms. The sample can be used as case studies to further understand the problem. The sample data format is as follows:

The train corpus is comprised of 4554 product titles with its core terms and brand for all kinds of category from Shopee. Also, the test corpus will be given in English. It contains 200 product titles, including 100 mobile, gadget and 100 Toys, Kids & Babies.

Next, the train corpus is not used for model training directly. Thus, we need to convert them into feature data set with corresponding label according to Section 2.1. And, the feature data set is able to be split into training set and test set, where test set rate is 30%. Note that prediction model will be trained without extra dataset.

In addition, the CoreTerm model and Brand model are independent of each other, and we build feature data and its label respectively.

Table 1. The Sample Data Format by Shopee.

Category	Product Name	Core Terms	Brand
Mobile & Gadget	9.7 inch Onda V975i Tablet PC Screen Protector Film	Screen Protector	Onda
Mobile & Gadget	BLUBOO Picasso Smartphone Silicone Protective Back Cover	Cover	BlUBOO Picasso
Mobile & Gadget	DoogeeX5/DoogeeX5 PRO Flip Cover Protective +Tempered Glass	Cover	DoogeeX5
Mobile & Gadget	YI 4K Action Camera 2 Ambarella A9SE75 Sony IMX377 12MP	Camera	YI
Mobile & Gadget	Makibes Unisex Red LED Digital Band Wrist Watch	Watch	BlUBOO Makibes

3.2 Settings

Vocabulary Table Vocabulary table can be built through train corpus after tokenization process and removing the words with low-frequency. More important, vocabulary table must be merged with the CoreTerm set and Brand set so that the label word must be contained. Besides, we also provide an initial set of word vectors that have been trained with an unsupervised method [1].

Model Parameters For Logistic Regression model, we have used the ℓ_2-norm regularization and training solver with liblinear. What's more, n-512-256-128-2 network structure with dropout regularization has also been set for DNN model. At last, CNN structure is Conv(2×6)-MaxPool(2×2)-Flatten-FC(128, dropout=0.15)-2. Their optimization solver is stochastic gradient descent with learning rate = 0.01 and momentum = 0.9 for both DNN and CNN models.

3.3 Results

We have carried out a series of experiments with different features and models combination, which are context-based BoW/word2vec features with LR/DNN/CNN models. As shown in Tab. 2 and Tab. 3, we can more clearly observe that 3-gram is better than 1-gram for feature selection and DNN model is also preferable to Logistic Regression for model selection throughout F1-score .

Apparently, BoW/word2vec + CNN model is with a low F1-score. However, it's hard to say that DNN must be better than CNN because of bad model parameters or word-embedding feature building without a large corpus.

Table 2. Experimental Results of CoreTerm Model.

Method	Precision (%)	Recall (%)	F1-Score (%)
1-gram BoW + LR	88.704/88.269	68.194/66.148	77.108/75.624
1-gram BoW + DNN	85.791/83.556	77.066/73.045	81.195/77.948
3-gram BoW + LR	93.965/90.503	81.125/75.911	87.075/82.567
3-gram BoW + DNN	94.205/88.235	89.120/80.135	**91.591/83.991**
3-gram BoW + CNN	85.528/81.183	79.766/73.968	82.547/77.408
3-gram word2vec + CNN	81.322/79.334	76.322/73.491	78.367/75.307

Table 3. Experimental Results of Brand Model.

Method	Precision (%)	Recall (%)	F1-Score (%)
1-gram BoW + LR	93.723/93.142	35.868/33.747	51.881/49.544
1-gram BoW + DNN	80.149/79.358	77.262/71.636	78.679/75.299
3-gram BoW + LR	96.671/94.656	58.127/51.346	72.600/66.577
3-gram BoW + DNN	90.597/85.779	88.711/77.433	**89.644/81.393**
3-gram BoW + CNN	81.347/79.688	79.467/72.067	79.458/76.231
3-gram word2vec + CNN	79.473/77.218	76.124/73.054	78.349/74.312

4 Conclusion and Limitations

The context-based BoW with DNN model enables to be used to forecast the test corpus for both CoreTerm model and Brand model because of its high f1-score. However, it's hard for us to discover new CoreTerm and new Brand which don't appear in training corpus because of corpus data insufficiently.

References

1. T Mikolov, K Chen, G Corrado, J Dean: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
2. http://nlp.stanford.edu/ socherr/pa4_ner.pdf