Chinese Named Entity Extraction System Based On Word2vec Under Spark Platform

Jialu Yuan[1, a], Yongping Xiong[1, b]

[1](Beijing University of Posts and Telecommunications, Institute of Network Technology, Beijing 100000, China)

[a]yuanjialu114@163.com, [b]xiongyp@163.com

Keywords: Spark, word2vec, NER, neural network, machine learning

Abstract. This paper proposes a real-time system that support the Chinese named entity extractions, which through word2vec algorithm training language mode to obtain word vector, and by calculating the Euclidean distance between word vectors to extract Chinese named entity, and transplant algorithm to Spark platform, using the Spark distributed computing ability improve training efficiency. First the system cut corpus into words with the help of existing word segmentation and get the rough corpus, then trains the rough corpus by word2vec algorithm to obtain word vectors and extracts the first layer of named entity according clustering algorithm, finally, the system uses the Named Entity Extraction(NEE) algorithm to extract the named entities and realize it on the spark platform.

Introduction

Named Entity Recognition (NER) refers to identification of entities with specific meaning in the text, mainly includes the person names recognition, place names recognition, organization names recognition, time recognition and date recognition etc. As a basic function of word segmentation program, NER is often used to improve the accuracy of word segmentation and is a basic work in the Natural Language Processing(NLP). NER has a wide range of applications in the field of Machine Translation, Information Retrieval, Automatic Question Answering System etc. And because of the particularity of Chinese, the recognition of the Chinese named entity is very important for Chinese Natural Language Processing(CNLP). In this paper, the author studies on Named Entity Extraction(NEE), its main purpose is to improve the accuracy of Named Entity Recognition, and to extract the new named entity according to the similarity degree between the named entities.

The commonly method used of Chinese named entity recognition is based on sequence labeling and feature matching. The most commonly sequence labeling algorithm is Hidden Markov Model (HMM), among them, Yu Hongkui and Zhang Huaping proposed the idea of named entity recognition based on cascade HMM model which has a good effect on Chinese named entity recognition [1]. On this basis, some scholars have put forward a method of sequence labeling based on Random Fields Conditional (CRFs) model [2]. CRFs is a probabilistic model without phase diagram based on the Maximum Entropy Model and Hidden Markov Model, and is a conditional probability model for labeling and segmenting ordered data. The model also uses the idea of sequence labeling to recognize the named entities, and its experimental results are improved in terms of accuracy and recall rate than the HMM model's results.

With the development of the big data technology in recent years, people began to use machine learning algorithm and neural network algorithm to solve the problem in the Natural Language Processing field and get a good result. This is a new idea for Natural Language Processing research, the most representative is that proposing a method that using the neural network algorithm to fit the language model, and training the corpus to get distributed representations of word [3][4]. Among them, the word2vec algorithm proposed by Google is most concerned by people, and has an excellent result [6][7][8]. At the same time, UC Berkeley AMP lab open source a distributed parallel computing framework based on memory called Spark in 2009[5], and promote the development of big data technology process. Spark is a kind of cluster computing model which is similar to Hadoop, but it is more excellent in computing performance. Spark uses abstract data type RDD (Distributed

Dataset Resilient) as the unified interface for computing jobs and saves all the results in memory rather than in the disk. At the same time, the Spark provides a rich operation for RDD, and the intermediate results of these operations is also RDD, so the execution of the program actually forms a Directed Acyclic Graph(DAG) consists of multiple RDD. Spark schedule and optimize the DAG job, calculate the final result, and return the results in the form of RDD.

In this paper, we train the target corpus to get the language model based on the efficient and improve computing ability provided by Spark platform, then get the word vector that store in the Spark memory, and calculate the named entity according to the similarity of word vector, finally we extract the named entity categories set by taking intersection of named entity sets.

Related algorithm and key technology

Language Model. Language Model is used to model the natural language, the traditional Language Model is a Statistical Language Model (SLM), which is a probability distribution function represent language fragments， the mathematical expression is as follows:

(1)

In which is represented by the T words in order to constitute the language fragment, the context of each word was taken as uniformly the Context, we can get the expression (1) according to the Bayesian formula.

According to the Context of different partitioning strategy, it can form different language model, such as n-gram model, the decision tree model, the maximum entropy model and Maximum Entropy Markov models, conditional random field model and neural network model [4], etc. word2vec algorithm is based on the Neural Network Language Model (NNLM). In the understanding of NNLM before, we first need to understand a concept of word vector, which has a very important significance in the NNLM model.

Word Vector. Word vector is using to represent words into mathematical vectors form natural language, its advantage is, therefore, the word can be calculated by mathematical formula so that computer can understand the natural language [6], commonly, the division of words vector has one - hot representation that it is a long vector with only one items for 1 other items for 0 to represent a word, the vector dimension N is the size of dictionary, the item in the vector of 1 correspond to the index of the word in the dictionary, for example, the word "China" can be represented as the following term vectors:

$$[0, 1, 0, \ldots, 0]^T$$

Another kind of word vector division is Distributed Representation [6],the basic idea is: natural language is mapped into a fixed length vector through the training that its dimension is generally short ,so all the word vector can constitute a vector space, at the same time because the vector length is shorter, but also conducive to the corresponding mathematical operation so as to study the relationship between them, so all the word vector can constitute a vector space, at the same time as vector length is shorter, also benefit to the corresponding mathematics so as to study the relationship between them. Word2vec is used in this type of word vector.

The principle of the algorithm in word2vec.NNLM model is mentioned in the section before, this section mainly elaborates NNLM model used in the word2vec that mainly use the CBOW Model (Continuous Bag - of - Words Model) [7] as a language Model, CBOW model predict the probability of the current word W_t under the W_{t-2}, W_{t-1}, W_{t+1}, W_{t+2}, in the context of the current word W_t. The language model of natural language can be known as (1) is the type of natural language, and the language model based on neural network usually takes the log likelihood function as the objective function:

Word2vec constructs a three-layer neural network: input layer, projection layer and output layer, the meaning of these three layers are:

(1) input layer: Contain 2c words vector V(), …, V(), V(), V(), V(), …, V() in the Context()。

(2) projection layer: the input layer of the vector sum up, namely:

(3) output layer: the output layer corresponds to a Huffman tree, Huffman tree is constructed of weights that word frequency of each word in the corpus, and its leaf node as for the words in the corpus.

After neural network model is constructed, the main problem is how to solve the fitting condition , through the above probabilistic neural network model,，word2vec will it as a binary classification problems, the method of using logistic regression to calculate the probability of each node which is from the Huffman root node to a leaf node, left nodes can be divided into negative class, right nodes are divided into positive class, using logistic regression and show that the probability of a node is divided into positive class is:

Obviously, the probability that it is divided into a negative class is:.

Among them, as the sigmoid function which is the activation functions of the neural network, is the word vector and is the parameter vector, which represents the leaf node vector in the Huffman tree.

According to the above definition, we can get the calculation algorithm: for any words w in the dictionary D, there is a path p^w from the root node to the word w in the Huffman tree. There are l^w-1 branches on the path p^w, each branch is seen as a binary classification, each time the classification will produce a probability, these probabilities multiply up, that is, [8]。

Thus, the formula for conditional probability is:

(4)

Among them:

In the expression of , represent the Huffman encoding corresponding to the first j node in the path p^w, represent the vector corresponding to the J-1 non leaf node in path p^w.

The above formula is into the logarithm likelihood function (2), that:

Among them, let L(w, j) is:

The above formula is the objective function of the CBOW model which makes the maximum of the function, when more than a threshold, it can be used to determine whether the language model is a natural language. Word2vec calculate the objective function by using the stochastic gradient rise method, because the calculation process does not involve algorithm, will not repeat here.

Spark framework overview. Spark is a big data processing framework over speed, usability and complexity. Originally developed in 2009 by the University of California at Berkeley's Lab AMP, Spark became one of the Apache's open source projects in 2010. As a memory distributed computing framework of the most popular today, Spark’s advantages are far more than just as the computing center of big data so simple, it has already formed a set of complete big data processing ecosystem, including Spark Streaming: used to process real-time data, the Spark SQL: using class SQL statements to query Spark data, Spark MLIB: extensible machine learning library, including the commonly used machine learning algorithms, Spark Graphx: support the Spark library of figure computing and parallel computing.

Spark big data computing framework is a distributed computing framework based on MapReduce, but compared with simple Hadoop Map and Reduce operations, Spark provides a distributed memory abstraction data sets for RDD, and also provides more rich operator operation specific to the RDD, these operations are divided into conversion (Transformations) and Action (Action). Transformation operation is used to return a new RDD operator specific to the RDD, including the map, reduce, the union, the filter, reduceByKey, partitionBy, join, count, sample and so on, the action is returned results after a calculation on the data set. All Spark applications are composed by the driver, this drivers used to execute in parallel the user submits the computing tasks, computing tasks are composed of various one-way transformation operations, in the Spark, all RDD transformation are lazy evaluation, old RDD data after conversion to generate a new set of RDD, each RDD can have multiple partitions, so every time the RDD conversion is equivalent to generating a node to another node of the directed acyclic graph (DAG), at the same time, the Spark optimize the processing , determine the phase, partition, pipelining, tasks and the cache, the data flow according to the direction of DAG, and when encountered Action type of operator after the actual operation, so the transfer of data is carried out in memory and calculation efficiency is improved greatly improving the calculation efficiency. The following figure shows the calculation model of the Spark.

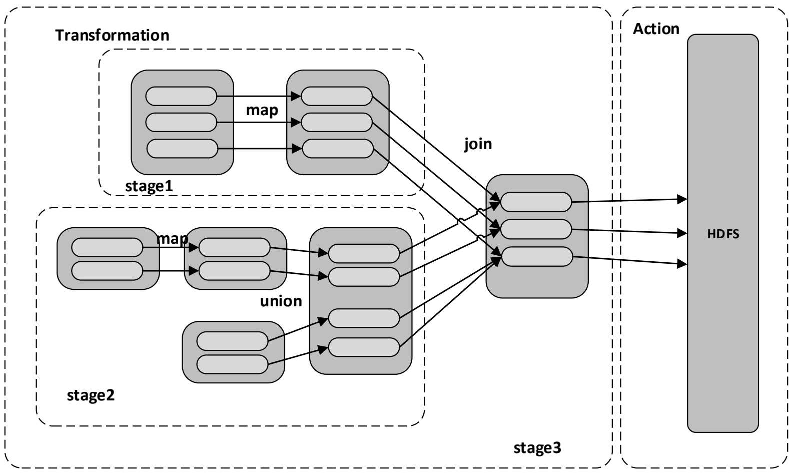

Fig.1 Calculation model of Spark

Named Entity Extraction algorithm based on word2vec. Word2vec algorithm use the neural network model to fit the natural language probability model, so as to obtain a highly efficient method of calculating language probability model. In the process of training, Word2vec can get a distributed representations of word in a vector space, and the dimension of each vector is determined by the input parameters, usually in the 100-500 dimensional. According to the distributed representations of word we can get a vector space, the word vector space contains the features and characteristics between words, we can use the space distance between the vectors to calculate the similarity degree of the words. This feature can be applied to Natural Language Processing to extract the Named Entity, the main idea is: first, using existing program to cut the corpus into words, and get the Named Entities which is labeled by the program. Usually the Named Entities exists in the dictionary used by program, we use set N to represent these Named Entities, and classify it into several subset {A,B,C…} according to the categories of Named Entity. Then we drop the stop words in the sliced corpus, and training the corpus by word2vec algorithm to get the distributed representations of word in it. Finally, we chose a classified Named Entity subset as the process set, for example we chose set A as the process set, and for each named entities in the set A, calculating the similarity between the words by vectors using Euclidean distance, after calculating, we can get a series of similarity words sets A1,A2,A3…,An (n is the number of elements in set A), by intersecting the sets Ai(i=1,2,3…,n) to obtain the new named entity set A^* which has the same categories compare with set A, and then take A and A^* to get a new set of Named Entities. We can complete Named Entity Extraction task by performing the above operations on all categories of Named Entities.

According to the description above, we can get the following steps of Named Entity Extraction algorithm based on word2vec:

```
(1). C cut the corpus into labeled word sequence and get set C
(2). N get the Named Entity set N from word sequence set C
(3). $C^*$ filter the stop word of set C and get set $C^*$
(4). W training the $C^*$ by word2vec model and get the word vector space W
(5). For $N_i$ in N:
          For $n_j$ in $N_i$:
               Cos($n_j$,w)
(6). Get the set $N^*$ by taking the intersection of N1, N2, N3, …, Nm.
```

(7). Get the set M by taking the union of N^* and N
(8). end

Named Entity Recognition based on the traditional HMM model or CRFs model need train corpus which have been labeled, and need to make rules to complete text sequence tagging, at the same time, it also needs a special dictionary to match the special named entity to optimize the recognition effect, thus Named Entity Recognition is depend on the labeled corpus and special dictionary. Once training model parameters in the traditional way, it is difficult to modify the parameters. With the data update, the semantic environment has changed, the old parameters are difficult to meet the new requirements, and the training cost is very high. If you need to modify the parameters, you must also collect a lot of new materials for manual labeling and training costs are high. While, the language training based on the neural network model does not need to manually labeled, only needs to cut the corpus into word sequences, the word sequences of the corpus can be trained to get the language probability model, greatly saving the cost of training. And the neural network language model uses the sliding window to retain the relationship of context between the words, it is very important for semantic analysis.

System implementation based on Spark framework

Initial corpus processing. Firstly, cutting the word into word sequence and extracting the named entities, the idea is importing the initial corpus from HDFS to Spark system, and constructing the initial corpus RDD, then mapping the corpus RDD into segmentation sequence of words and saving to the new RDD, recorded as RDD1. The map transformation can map the RDD to a new RDD through the mapping function, here is segmentation function. The RDD1 is actually a list, where each element is a token object, the object contains three attributes: ‘word’, ‘offset’, ‘nature’. Where ‘word’ is the word that include in the corpus, ‘offset’ is the position of the word in the corpus, and ‘nature’ is the nature of the word. According to ‘nature’ attributes we can get the initial Named Entity set RDD2 by filter operator, it is not a complete set of Named Entities set and the accuracy is not satisfied, then we will filter corpus segmentation using stop word dictionary, eliminate useless stop words, and get a new Named Entities set RDD3 which used for training word2vec language model.

So, we get three sets of RDD (RDD1, RDD2, RDD3), which correspond to the C, N, C^* mentioned in section 2, and then we need to train the corpus samples to get word vector. The transformation flow chart of RDD in Spark is shown in Figure 2:

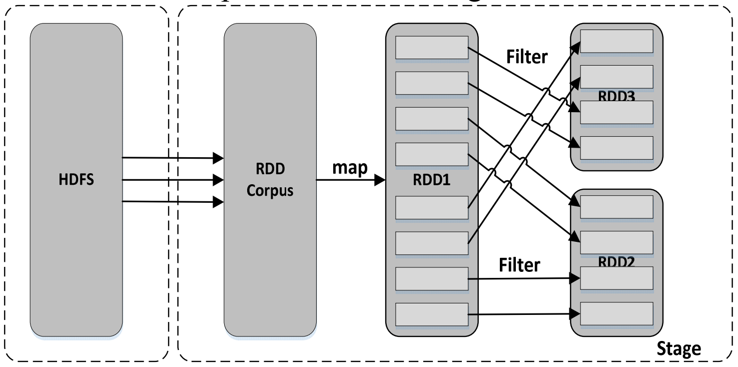

Fig.2 Transformation of initial corpus in Spark

Using word2vec training corpus samples on Spark. The corpus samples used for training word2vec model have filtered the stop words, the benefits of doing so is that it can make the named entity relation more compacting, and the word vectors can contain more features of training samples, Spark contains a scalable Mlib which have achieved the word2vec algorithm, we can use the algorithm to generate the word vectors, the calling procedure is as follows:

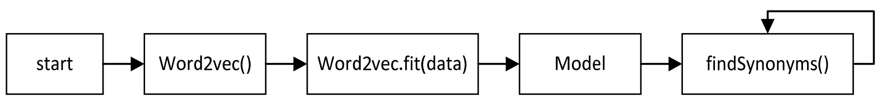

Fig.3 Process of training word2vec model

In the process of calling the algorithm, it needs to be processed in parallel, since we need to train a large set of samples. Training samples need to be cut into small samples set that is distributed in training node. It can be regarded as a vector space that contains the word vector trained by training node, in the process of similarity iterative, in fact, the word vector space is calculated iteratively. the calculation process as shown below:

```
For model in models:
    For w in model:
        If w is NE[i]:
            synonyms = model.findSynonyms(w,n)
        For word,cosineSimilarity in synonyms:
        If cosineSimilarity > threshold:
            List.insert(word)
```

Among them, ‘models’ is the word vector space, which contains the corpus training results, ‘model’ is the word vector, W is the word vector elements, NE is named entity set. The calculated ‘synonyms’ contains the word array that similar to W, and it will be filtered and added to a similar set of entities (List) according to the ‘threshold’, the n and threshold parameters can be changed according to the training results.

According to the word vector model mentioned in section 3.2 and Named Entity set RDD2 mentioned in section 2, we can extract Named Entity according to Named Entity Extraction algorithm discusses in section 2, first mapping RDD2 to different sub RDD using Map Operator, the sub RDD contains the same type Named Entities, for example, the name of person. Then traversing the word in the sub RDD through the fliterMap operator. For each word, calling the findSynonyms function to obtain the 50 words closest to its similarity and there will be forming a new RDD. Repeat the above operation, we can obtain a list of RDD, and a new Name Entity RDD through computing intersection of RDDs in the list. The following figure is Spark program flow chart which implementing the progress of Named Entity Extraction:

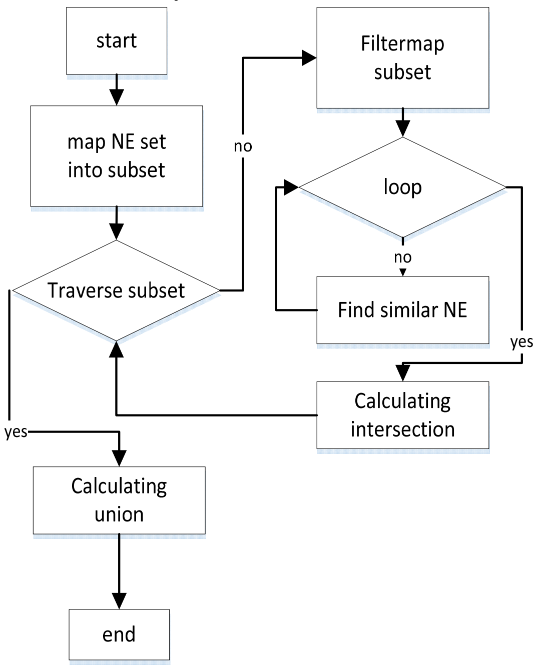

Fig.4 Program flow chart of Named Entity Extraction

The result of the experiment and analysis

Experimental environment. In this experiment, we use 4 servers to set up the experimental environment of Spark cluster, one of which is master, and the remaining three sets as slave node, the experimental environment hardware configuration table is as follows:

Table1. The hardware of experimental environment

SERVER	MEM	CPU	THREAD
Master	16G	4core	8thread
Node1-3	3x16G	3x4core	3x8thread

The software configuration table is shown below:

Table2. The software of experimental environment

Software	Version
OS	Centos6.7
Spark	1.6.1
FS	HDFS2.7.2

Results and analysis. This experiment adopts the sogou tech-oriented news data (SogouCA) as the training and test corpus, the three important indexes of named entity recognition: accuracy rate, recall rate and F value as a analysis standard, their definitions are as follows:

The Comparison result of NEE using two different methods as follows:

Table3. NEE based on word sequence labeling model

NE	P	R	F-1
Person Name	82.17	89.21	85.54
Place Name	82.56	90.03	86.13
Organization Name	74.32	72.87	73.58
Proper Noun	85.30	88.19	86.72

Table4. NEE based on word2vec model

NE	P	R	F-1
Person Name	85.27	93.34	89.12
Place Name	87.45	94.27	90.73
Organization name	78.41	81.02	79.69

According to the experimental results, we can get the conclusion that recognition effect has certain increases on the accuracy and recall rate by using word2vec model, the average recognition value increased from 82.93% to 91.1%. we can filter and extract the Named Entity by calculating the similarity of the word vector, and obtain accurate results in the multiple iterations, make the name entity recognition F-1 value increased.

Summary

In this paper, we first analyze the traditional way of Named Entity Extraction, points out the method that the traditional way can be improved, and put forward the way of using neural network language model to extract named entity, in the choice of model, we choose the word2vec algorithm as the training model because its efficiency and the effect is excellent, and we improve the quality of the training model by improving the training sample. Finally we extract the named entity through multiple iterations intersection. Using the distributed computing capability of Spark platform, the algorithm runs on the cluster in parallel, and further improves the training efficiency. After experiments we can conclude that for the massive text data samples, using the neural network language model to extract the named entity, not only can improve the accuracy and the coverage rate of extraction but also can ensure extraction efficiency, and it is the future trend of development that applying the parallel computing ability and machine learning algorithm to Natural Language Processing through the big data processing platform.

Acknowledgments

The authors would like to thank Mr. Xu for excellent technical support and Assistant professor Yongping Xiong for critically reviewing the manuscript.

References

[1] YU Hong-kui,ZHANG Hua-ping,LIU Qun. Chinese named entity identification using cascaded hidden Markov model[J]. Journal on Communications,2006,27(2):88-93.

[2] ZHANG Zhu-yu,REN Fei-liang,ZHU Jing-bo.A Comparative Study of Features on CRF-based Chinese Name Entity Recognition[D].Shengyang:Natural Language Processing Lab Northeastern University,2010

[3] David E Rumelhart,Geoffrey E Hintont,and Ronald J Williams.Learning representations by backpropagating errors.Nature,323(6088):533-536,1986.

[4] Yoshua Bengio,Rejean Ducharme,Pascal Vincent, and Christian Jauvin.A neural probabilistic language model.Journal of Machine Learning Research(JMLR),3:1137-1155,2003.

[5] Zaharia M, Chowdhury M, Franklin M, Shenker S, Stoica I. Spark: Cluster computing with working sets. HotCloud 2010. 2010.

[6] Tomas Mikolov,Kai Chen,Greg Corrado,Jeffrey Dean.Efficient Estimation of Word Representations in Vector Space.arXiv:1301.3781,2013.

[7] Tomas Mikolov,Quoc V.Le,Ilya Sutskever.Exploiting Similarities among Languages for Machine Translation.arXiv:1309.4168v1,2013.

[8] Quoc V.Le, Tomas Mikolov.Distributed Representations of Sentences and Documents.arXiv:1405.4053,2014.

[9] Xiaoqing Zheng,Hanyang Chen,Tianyu Xu.Deep Learning for Chinese Word Segmentation and POS tagging.Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,pages 647-657.

[10] Ronan Collobert,Jason Weston,Leon Bottou,Michael Karlen,Koray Kavukcuoglu and Pavel Kuksa.Natural Language Processing (Almost) from Scratch.Journal of Machine Learning Research(JMLR),12:2493-2537,2011.

[11] Michael U Gutmann and Aapo Hyvarinen.Noise-contrastive estimation of unnormalized statistical models,with applications to natural image statistics.The Journal of Machine Learning Research,13:307-361,2012.