Parameter Estimation of Conditional Random Fields Model

Based on Cloud Computing

John Chiang

Lab Department, Newegg Inc.

Abstract

Conditional Random Field (CRF), a type of conditional probability model, has been widely used in Nature Language Processing (NLP), such as sequential data segmentation and labeling. The advantage of CRF model is the ability to express long-distance-dependent and overlapping features. However, the model parameter estimation of CRF is very time-consuming because of the large amount of calculation. This paper describes the method that use of MapReduce model to parallel estimate the model parameters of CRF in open-source and distributed computing framework that provided by Hadoop. Experiments demonstrated that the proposed method can effectively reduce the time complexity of model parameter estimation.

# 1. Introduction

This paper presents the method for parallel estimating the model parameters of CRF in Hadoop. The large scales of product attribute information that gather from various e-commerce sites are used as training data set. Comparison with other methods [1, 3, 7, 8, 11, 13], CRF has been widely used because of its good features, but the disadvantages of CRF are the long training cycle and large amount of calculation, especially when dealing with big data. In order to improve the training efficiency of CRF, this paper improves the existing CRF training algorithm to make it run in Hadoop, and achieved good results.

Current research will be introduced in the Section II, Section III presents the CRF training methods, and Section IV describes the use of MapReducemodel to define CRF training methods, and Section V describes the experimental results, Section VI shows Conclusion and Future Work.

# 2. Related Work

The problem of CRF is that the model parameter estimation cycle is long, the time complexity and space complexity of the whole algorithm showing non-linear growth with the growth of the training data, this is particularly reflected in massive training set. On the optimization of the CRF training there are also some more mature methods, Such as SGD, L-BFGS [6] and stand-alone multi-threaded approach to improve the efficiency of training. These methods are all based on stand-alone environment [4, 5]. The method that runs in distributed environment is rare.

# 3. CRF Training Method

CRF is trained by setting a set of weights to maximum the log-likelihood of a given training data set. Bayes estimation and maximum likelihood estimation are common used methods in CRF model parameter estimation. Maximum likelihood estimation is less complex and easier to understand than Bayes estimation. In this paper, EM algorithm is used to do the maximum likelihood estimation.

## 3.1. EM Algorithm

EM algorithm was explained and given its name in a classic paper by Arthur Dempster, Nan Laird and Donald Rubin in 1997. It is a simple and practical machine learning algorithm, and usually used to iterate the computation of maximum-likelihood estimates when the observations can be viewed as incomplete data[9, 10, 12, 14].

We assume that data X is a collection of observed and incomplete data, and is generated by some distribution. We assume that a complete data set exist Z = (X, Y) and also assume a joint density function:

(1.1)

This joint density function arises from the marginal density function, the assumption of hidden variable Y, parameter value guesses and an assumption of joint relationship between the missing and observed values. Below is the discussion about the specific form of the joint density function:

With this newly density function, we can define a new likelihood function

(1.2)

called the complete-data likelihood. Note that this function is in fact a random variable since the hidden variable Y is unknown, and determined by the hidden variable Y.

**Expectation step (E step)**: Calculate the expected value of the complete-data log-likelihood with respect to the unknown data Y given the observed data X and the current parameter estimates.

That is, we define:

(1.3)

where are the current parameters estimates that we used to evaluate the expectation. In equation (1.3), X and are constants, is a normal variable that we wish to adjust and Y is a random variable governed by the distribution

(1.4)

the right side of equation (1.3) can therefore be re-written as:

(1.5)

note that is the marginal distribution of the unobserved data and is dependent on both the observed data X and on the current parameters, and D is the space of values Y can take on. In some special cases, this marginal distribution is a simple analytical expression of the assumed parameters and the data X, but this might be very hard to obtain. Sometimes, in fact, the density actually used is

(1.6)

but this doesn’t affect the optimization of the likelihood function in equation (1.5) since the extra factor is not dependent on .

We define a binary function:

(1.7)

where y is a random variable governed by some distribution , then

(1.8)

is a function of , then the estimation of parameter can be get through a simple optimization method.

**Maximization step (M step)**: Maximize the expectation. That is, we find

(1.9)

where is the parameter space.

EM algorithm is a iterative method, each iteration is guaranteed to increase the log-likelihood and the algorithm is guaranteed to converge to a local maximum of the likelihood function.

# 4. Training CRF in MapReduce Model

MapReduce is a programming model for processing large data set, and the name of implementation of the model by Google. The framework allows the developers to focus on the application, since it hides the details of parallelization, data-distribution, load balancing and fault-tolerance. Apache Hadoop is an open source project. It implements the MapReduce framework of Google, and also provides a distributed file system Hadoop Distributed File System (HDFS). The user just need to put the data in HDFS, and implement the map interface and reduce interface, then the application will be automatically and concurrently processed in cluster. Hadoop charge the underlying operations, such as data segmentation, computing resource allocation, exception handling and internal communication [2, 15]. This paper discusses how to implement the CRF training algorithm in Hadoop using the MapReduce programming model.

## 4.1. Data Preparation

The size of training data set has a great impact on the output of CRF. To generate the training data set, we gather large scale of product attribute information from various e-commerce sites by crawler engine, save these attribute information in HDFS, and organize the similar products into groups by clustering algorithm, and then each product in one group will be a record of training data set. The attribute name is used as label and attribute value is used as observation. Table 1 shows two labeled records. Those records are separated by blank row and are stored in HDFS with this format.

**Table** 1**:** Labeled training data

|  |
| --- |
| 8GB DDR3 | memory  AMD FX-8120 3.1GHz | processor  1TB 7200RPM | hard drive |
|  |
| AMD Athlon II X4 645 3.1GHz | memory  4GB DDR3 | processor  500GB SATA | hard drive |

## 4.2. MapReduce Job Definition

The long training cycle of CRF is mainly caused by the maximum likelihood algorithm. This paper improved the maximum likelihood algorithm to make it can be processed parallel. The derivation of EM algorithm in section 3 demonstrates that the maximum likelihood algorithm can be implemented in MapReduce framework.

### 4.2.1 EM Algorithm in MapReduce

**E step**: Calculate the expectation, with respect to the initial assumption parameters or model parameters generated by last iteration. In this step, the training data set is divided into many subsets. Each subset is handled by one map job to calculate the expectation and gradient vector.

**M step**: Re-calculate the model parameter in a single reduce job, according to the expectations and gradient vectors generated by map jobs.

Iterate these two steps to specified number of times.

**EM Implementation Steps**:

**Initialization Steps**:

1. Specify the output path for reduce job of each iteration, store path for initial assumption parameters and output path for final result.
2. Generate the initial assumption parameters and save them in specified path.

**Iteration Steps**:

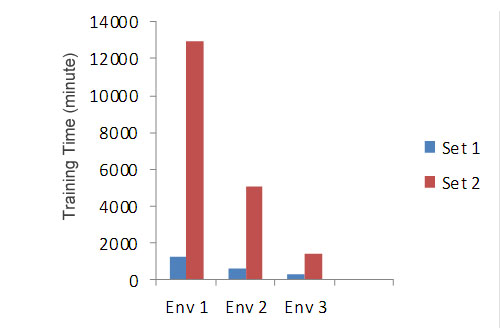
1. Each map job loads the respective subset of training data and the model parameters, and calculates the corresponding expectation and gradient vector. In the first iteration, the model parameters are generated by assumption. In the following iterations, the model parameters are generated by last iteration.
2. The only one reduce job collect the expectation and gradient vector that output by each map job, generate the global expectation and gradient vector, update the model parameters using L-BFGS algorithm and save the updated model parameters to the output path of this iteration.
3. Continue iteration if the specified number of iterations is not reached, otherwise stop.

# 5. Experimental Results

The experiment is composed of two groups of training data and three groups of runtime environment. Training data is the product attribute information from various online e-commerce sites, Set 1 consists of 100,000 records, Set 2 consists of 500,000 records. Env 1 is a single-machine, Env 2 and Env 3 are tow clusters based on Hadoop,

The Env 2 consists of three servers, one as a named node, the others two are both task nodes and data nodes. The Env 3 consists of six servers, one as a named node, the others five are both task nodes and data nodes.

Experimental results:



**Figure** 1: Training time

Figure 1 shows the difference of time-consuming between single-computer environment and distributed environment. In the single-computer environment, the time-consuming increases very fast along with the increase of training data set. In distributed environment, the time-consuming is reduced significantly and can be further reduced by increasing the number of computing node.

# 6. Conclusion and Future Work

In this paper, we describe how to use MapReduce programming model to parallel estimate the CRF model parameters in Hadoop. The test results show that the training of in a distributed environment is feasible. In addition, the computing resources are no longer the bottleneck in distributed environment since resources can be horizontal scaling. So, training the CRF model parameters using massive training data become possible. MapReduce framework suitable for handling the large amount of data calculation and analysis, In practical application, many tasks cannot be handled in a single MapReduce procedure,, such as the Iterative EM algorithm mentioned in this paper. To implement these iterative algorithms in Haddoop, a chain of MapReduce procedure is need. The reduce job of previous MapReduce procedure save the temp result to HDFS, then the map job of next MapReduce procedure load this temp result from HDFS. Such a large number of IO operations can be a waste; also programming has too much trouble. In the future we can introduce pipe to Hadoop to improve performance and change the iterative processing model.

# 7. References

[1] J. Lafferty, A. McCallum and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data”, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001)*, 2001.

[2] J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters”, *Proceedings of the 6th USENIX Symposium on Operating Systems Design and Implementation*, 2004, pp. 137–150.

[3] S. Sarawagi and W. Cohen, “Semi-markov conditional random fields for information extraction”, *NIPS*, 2004.

[4] X.-H. Phan, L.-M. Nguyen, Y. Inoguchi, and S.Horiguchi, “High performance training of conditional random fields for large-scale applications of labeling sequence data”, *IEICE - Transactions on Information and Systems*. 2007, pp. 13-21.

[5] X. LIN, L. ZHAO, D. YU and X. WU, “Distributed Training for Conditional Random Fields”, *Natural Language Processing and Knowledge Engine*e*ring (NLP-KE)*, 2010

[6] D. C. Liu and J. Nocedal. On the limited memary BFGS method for large scale optimization. *Mathematical Programming*, pp. 503-528, 1989.

[7] F. Sha and F. Pereira, “Shallow parsing with conditional random fields”, *Proceedings of Human Language Technology*, *NAACL*, 2003.

[8] T. Kristj ansson, A. Culotta, and P. Viola, “Interactive information extraction with constrained conditional random fields”, *AAAI*, 2004, pp. 412-418.

[9] Dempster A，Laird N，Rubin D, “Maximum likefihood eitimation from incomplete data via EM algorithm”, [J], *J．Royal Stafstical* *Society Series B*, 1977.

[10] Blimes J, “A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov model”, [R], *TR-97-021*, April 1998.

[11] Shai Fine, Yoram Singer, and Naftali Tishby, “The hierarchical hidden Markov model: Analysis and applications”, *Machine Learning*, 1998, pp. 41–62.

[12] J. Darroch and D. Ratcliff, “Generalized iterative scaling for log-linear models”, *The Annals of Mathematical Statistics*, 1972, pp. 1470–1480.

[13] P. Brown, S. Della Pietra, V. Della Pietra, , R. Mercer, A. Nadas and S. Roukos, “Translation models using learned features and a generalized Csiszar algorithm”, IBM Technical Report.

[14] Jeff A. Bilmes, “A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models”, *Technical Report ICSI-TR-97-021*, International Computer Science Institute, Berkeley, 1997.

[15] http://hadoop.apache.org