Research on the Application of Agricultural Big Data Processing with Hadoop and Spark

Yan Cheng
School of Computer Information Engineering
Jiangxi Normal University
Nanchang, China
*Chyan88888@jxnu.edu.cn

Qiang Zhang
School of Computer Information Engineering
Jiangxi Normal University
Nanchang, China
598592221@qq.com

Ziming Ye
School of Computer Information Engineering
Jiangxi Normal University
Nanchang, China
2685487478@qq.com

***Abstract*—Numerous terminal equipment in the agricultural park collect environmental data that affects crop growth every day. Proper analysis of these massive amounts of data can acquire useful information on the status of crop growth. In this paper, two cloud computing frameworks, Apache Hadoop and Apache Spark are used to study agricultural big data analysis. This paper developed applications for real agricultural park big data analysis in both frameworks and implemented a yield prediction model based on multiple linear regression using Spark MLlib. The performance of the two frameworks in agricultural big data processing was studied and compared through various experiments. The experiments show that the comprehensive performance of Spark is higher than Hadoop, and the model can obtain better prediction results.**

Keywords—Agricultural big data, Hadoop, Spark, Yield prediction

I. Introduction

In recent years, big data topics have received much attention, which is a new strategic resource that can tap into greater potential value [1]. The concept of “big data” is not only the innovation of data and technology, but also a change of thinking [2]. The data collected during crop planting is a very important source of information in the agricultural field, and many useful information can be obtained by processing this data. However, the larger amount of data collected, the more difficult to store, process and analyze.

The emergence of cloud computing technology has made it possible to solve the problems faced by big data. Some scholars have proposed the idea of "building an agricultural big data analysis application platform" [3], but the processing and analysis techniques for agricultural big data are still rare. Problems such as data redundancy and data noise are bound to bring about an increase in data storage costs. High-efficiency and low-cost data storage technologies are particularly important [4]. The continuous development of technologies such as massive data mining and parallel data processing has made the integration of agricultural knowledge base, knowledge representation, knowledge transformation and data sharing a prerequisite for data processing analysis [5].

Hadoop is a distributed system infrastructure developed by the Apache Foundation that is ideal for analyzing and processing massive amounts of data. The two cores of the Hadoop framework are MapReduce and Hadoop Distributed File System (HDFS), both from the Google file system [6]. The processing of massive data in Hadoop is divided into several nodes for parallel calculation, and the results obtained by each node are combined to form the final output [7]. Hadoop was designed for batch processing and provides fault tolerance and scalability originally but does not have fast performance. Hadoop is a model for reading and writing data processing based on disk, which results in relatively slow calculation speed and low comprehensive performance [8]. Apache Spark is a fast, versatile computing engine designed for large-scale data processing. By performing in-memory calculations with the resilient distributed datasets (RDDs), Spark uses main memory more efficiently and minimizes disk-to-disk data transfers [9]. Spark also provides new tools such as Spark SQL and Spark MLlib to support SQL queries and implement machine learning algorithms.

The work of this paper is to use Hadoop and Spark for agricultural big data processing, and experiment with several different applications and parameters to research and evaluate the data processing efficiency of the two frameworks. We also implemented a multivariate linear regression model based on Spark MLlib for yield prediction and obtained good prediction results.

II. Model

Hadoop is recognized as a set of industry big data standard open source software that provides massive data processing capabilities in a distributed environment. Spark is a general-purpose big data computing framework, just like the traditional big data technology Hadoop MapReduce, Hive, and Storm streaming real-time computing engine. But Spark can only replace a part of Hadoop, which is Hadoop's computing framework MapReduce, Hive query. Spark itself does not provide big data storage, but also depends on Hadoop's HDFS. The framework used in this paper is to combine Hadoop and Spark to fully exploit the advantages of both to deal with agricultural big data. The framework structure is shown in Fig. 1.

Looking at the framework from the bottom up, the server cluster and HDFS are equivalent to the Iaas layer of cloud computing. Map Reduce, Hive, spark SQL and so on are equivalent to the Paas layer of cloud computing. The Saas layer of cloud computing is served by Spark MLlib and Graph X.

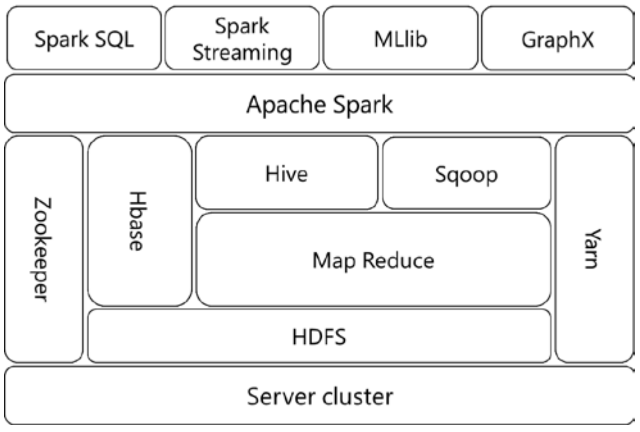


Fig. 1. The overall framework of Hadoop and Spark.

HDFS is a file system that allows files to be shared on multiple hosts over a network, allowing multiple users on multiple machines to share files and storage space. MapReduce is a distributed computing model proposed by Google. It is mainly used in the search field to solve the calculation problem of massive data. This article uses the Hadoop 2.0 version to unify the resource management in the framework. Hadoop Yarn is more functional than before, and its high scalability makes it possible to run MapReduce or Spark applications on Yarn.

Hive, a mechanism for storing, querying, and analyzing large-scale data stored in Hadoop, is a data warehouse infrastructure. Hive includes a SQL parsing engine that translates SQL statements into MapReduce task execution. Sqoop (full name SQL-TO-HADOOP), on the other hand, is an ETL tool for converting between Hadoop data and structured data.

Spark includes various computing frameworks common in big data: Spark Core for offline computing, Spark SQL for interactive queries, Spark Streaming for real-time streaming computing, Spark MLlib for machine learning, and Spark GraphX for graph computing. Spark is based on elastic distributed data sets (RDD) to implement calculations in memory. RDD is a data structure of Spark that is not only fault tolerant and parallel, but also compatible with all objects or classes of Java, Python, etc. This article is mainly applied to Spark Core, Spark SQL and Spark MLlib.

The linear regression method is a classical statistical method. It studies the linear relationship between variables and has many successful application cases in crop yield prediction. It is a representative method for studying yield prediction. The mathematical expression of multiple linear regression equations is defined in Equation (1).

$$Y = a_0 + a_1x_1 + \cdots + a_kx_k + \varepsilon \quad (1)$$

Where Y is the dependent variable, $x_i(i = 1,2, \dots k)$ is the independent variable, ε is the random error term, and $a_0, a_1, \dots a_k$ is the unknown parameter. In the regression model, a_0 is called the regression constant, and $a_1, a_2, \dots a_k$ is called the regression coefficient of the independent variable. The linear equation of the expected value of the dependent variable Y and the independent variable $x_1, x_2, \dots x_k$ is Equation (2).

$$E(Y) = a_0 + a_1x_1 + \cdots + a_kx_k \quad (2)$$

Equation (2) is also referred to as the overall regression equation. For the n sets of sample observations, the equation corresponding to $Y_i, x_{1i}, x_{2i}, \dots, x_{ki}(i = 1,2, \dots k)$ is converted into a matrix expression as Equation (3).

$$Y = \mathrm{XA} + \varepsilon \quad (3)$$

Where $Y = [Y_0\ Y_1 \dots Y_n]^T$ is the n-order column vector and is the observed value of the dependent variable. $X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{k1} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \cdots & x_{kn} \end{bmatrix}$ is an n*(k+1) order matrix, which is an observation of an independent variable. $A = [a_0\ a_1 \dots a_k]^T$ is a k+1 order column vector, which is the population regression parameter vector. $\varepsilon = [\varepsilon_0\ \varepsilon_1 \dots \varepsilon_k]^T$ is an n-order column vector and is a random error term vector. So the mathematical expression of the overall regression equation is Equation (4).

$$E(Y) = \mathrm{XA} \quad (4)$$

According to the existing sample data collected and processed, a linear function relation model of the dependent variable with respect to the independent variable is established, and the above unknown parameter $a_0, a_1, \dots a_k$ is obtained. However, since the regression model parameters are calculated from the sample data, these parameters are only estimates of the true values, expressed as $\hat{a}_0, \hat{a}_1, \dots \hat{a}_k$. Replacing the regression coefficient $a_0, a_1, \dots a_k$ in the population regression function with the parameter estimate, then the multiple linear regression equation is Equation (5).

$$\hat{Y}_i = \hat{a}_0 + \hat{a}_1x_{1i} + \cdots + \hat{a}_kx_{ki} \quad (5)$$

Where $\hat{Y}_i(i = 1,2, \dots k)$ is the estimated value of the sample value. The residual is the deviation between the estimated value $\hat{Y}_i$ of the dependent variable and the actual observed value Yi in the regression equation of the sample. The mathematical expression is Equation (6).

$$e_i = Y_i - \hat{Y}_i \quad (6)$$

From the Least squares' method, $\hat{a}_0, \hat{a}_1, \dots \hat{a}_k$ should make Equation (7) the smallest. Then Equation (8) can be obtained by substituting the above format and deriving.

$$Q(\hat{a}_0, \hat{a}_1, \dots \hat{a}_k) = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \quad (7)$$

$$\hat{A} = (X^TX)^{-1}X^TY \quad (8)$$

In Equation (8), X is the sample observation matrix, X^T is the transposed matrix of the observation matrix, and Y is the observation column vector.

Through the above analysis, we know the parameter solving process of the linear regression model. We mainly use the following API interfaces provided by Spark MLlib to implement multiple linear regression models.

a) LinearRegressionModel(): The class that implements a linear regression model.

b) Gradient.compute(): To call the LeastSquaresGradient-.compute method to calculate the sample gradient.

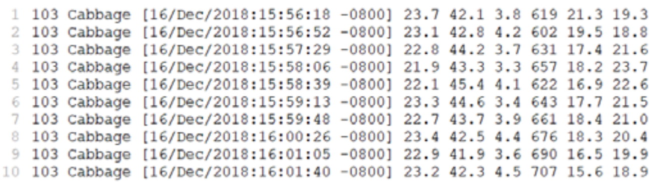
c) Updater.compute(): To call the SimpleUpdater.compute method to update the weight.

d) Predicet(): To calculate the predicted value of the sample based on the linear regression model.

III. Experiment

We virtualized the lab's computer equipment with VMware and created a private cloud cluster of six virtual servers connected to the lab's local area network, all using CentOS 6.4. The built cluster contains one master node and five slave nodes. The primary node is configured for 8GB RAM and 80GB storage. The remaining five slave nodes have 6GB of RAM and 50GB of disk space. Each node has a static IP address set up for configuration and identification. In addition, as required by Hadoop, and Java version 1.8.0 has been installed in each node. Nodes implement secret-free links through ssh, so Hadoop can easily execute applications for each node and start or shut down various processes.

This paper is supported by a provincial key project and takes the Jizhou Modern Agricultural Demonstration Park as an application example. Crop data collected from greenhouses in real agricultural garden cabbages were used in the experiment. Figure 2 shows the top ten data used in our experiments. The data file consists of semi-structured text data.



```
103 Cabbage [16/Dec/2018:15:56:18 -0800] 23.7 42.1 3.8 619 21.3 19.3
103 Cabbage [16/Dec/2018:15:56:52 -0800] 23.1 42.8 4.2 602 19.5 18.8
103 Cabbage [16/Dec/2018:15:57:29 -0800] 22.8 44.2 3.7 631 17.4 21.6
103 Cabbage [16/Dec/2018:15:58:06 -0800] 21.9 43.3 3.3 657 18.2 23.7
103 Cabbage [16/Dec/2018:15:58:39 -0800] 22.1 45.4 4.1 622 16.9 22.6
103 Cabbage [16/Dec/2018:15:59:13 -0800] 23.3 44.6 3.4 643 17.7 21.5
103 Cabbage [16/Dec/2018:15:59:48 -0800] 22.7 43.7 3.9 661 18.4 21.0
103 Cabbage [16/Dec/2018:16:00:26 -0800] 23.4 42.5 4.4 676 18.3 20.4
103 Cabbage [16/Dec/2018:16:01:05 -0800] 22.9 41.9 3.6 690 16.5 19.9
103 Cabbage [16/Dec/2018:16:01:40 -0800] 23.2 42.3 4.5 707 15.6 18.9
```

Fig. 2. The top ten rows of cabbage greenhouse data files.

As shown in Figure 2, the first element in each row is the greenhouse number, and the second element is the name of the plant variety. The fourth element in each line indicates the date and time the terminal device received the data, and in the same parentheses there is a server time zone such as -0800 which refers to the Chinese standard time zone. The data acquisition uses the ZigBee protocol, and the acquisition delay of the greenhouse is about 36 seconds. The elements in the back are temperature (℃), humidity (%), light intensity (Lux), co2 concentration (PPM), ground temperature (℃), and ground moisture (%).

We used the Eclipse IDE and Apache Maven and developed real-world applications in both frameworks through the Java programming language. Hadoop and Spark applications have direct access to crop data we use to experiment with storage in HDFS. Since the growth period of cabbage is about 56 days, we need the maximum and minimum values of temperature and humidity for each cycle, as well as the average of light intensity and co2 concentration.

In the research of crop big data processing, we focus on the performance indicator of execution time. Multiple experiments are performed to check how the input file size, application type, and number of active slave nodes affect the execution time of each application.

In Figure 3, 4, and 5, we observe that as the input file size becomes larger in Hadoop, the time increment increases, and increasing the number of active nodes reduces the execution time. As shown in Figure 6, 7 and 8, this is also true in Spark, and when the input file size is the same, Spark execution time is much smaller than Hadoop. Of course, especially when there is only one active node, increasing the input file size or reducing the active slave will increase the application execution time. In addition, different computing requirements make execution time different for applications. Overall, it's clear that Spark is better at handling big data than Hadoop.

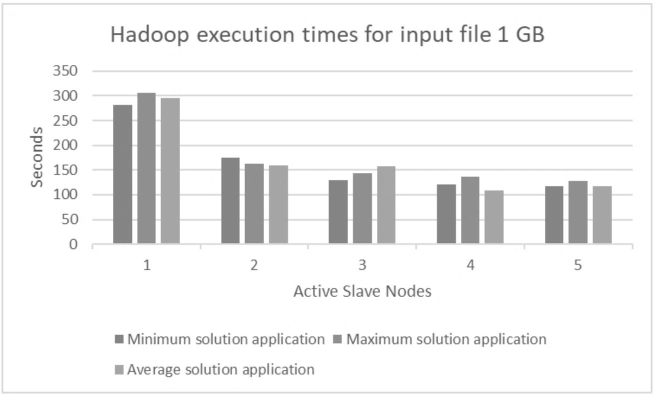


Fig. 3. Execution time of Hadoop applications with about 1 GB input file.

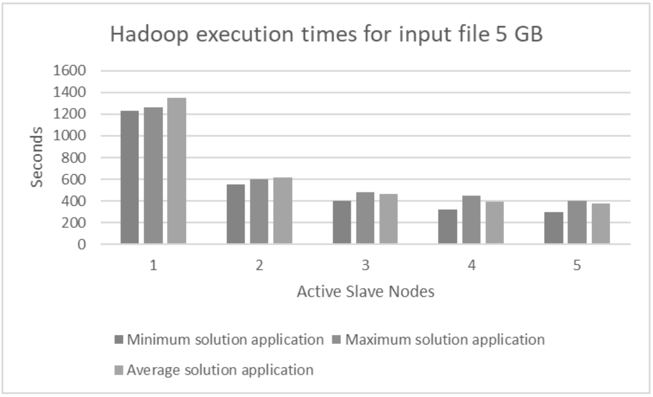


Fig. 4. Execution time of Hadoop applications with about 5 GB input file.

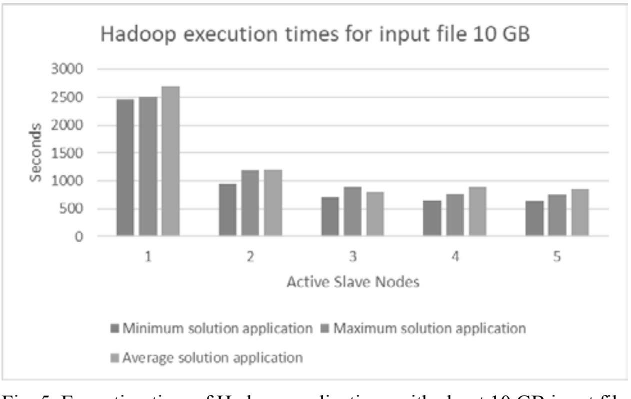


Fig. 5. Execution time of Hadoop applications with about 10 GB input file.

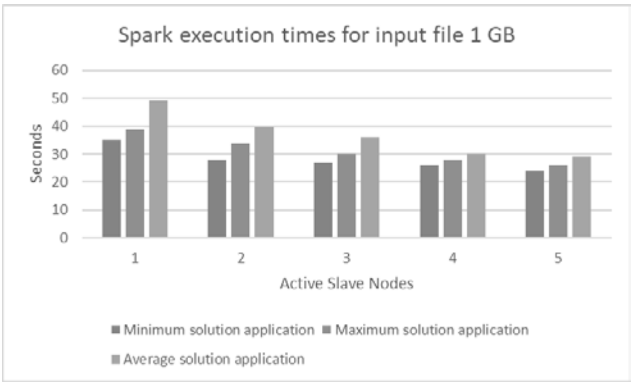


Fig. 6. Execution time of Spark applications with about 1 GB input file.

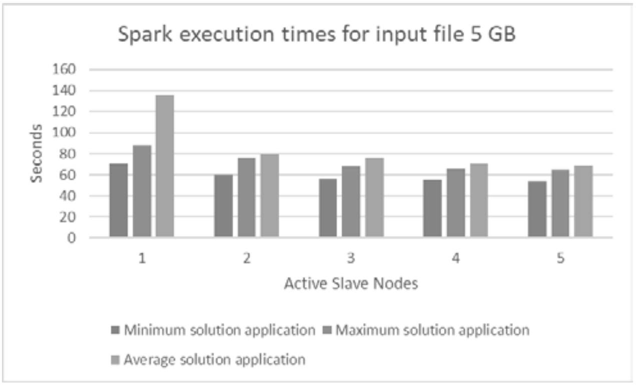


Fig. 7. Execution time of Spark applications with about 5 GB input file.

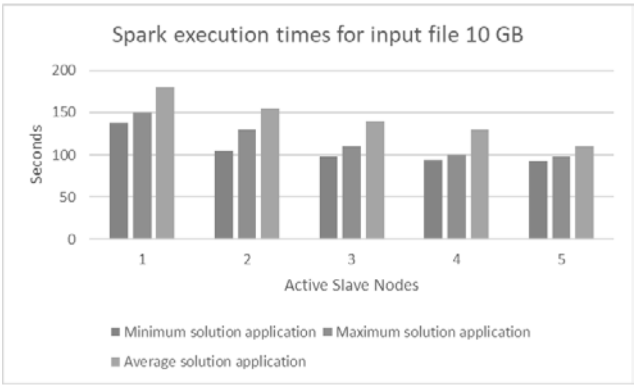


Fig. 8. Execution time of Spark applications with about 10 GB input file.

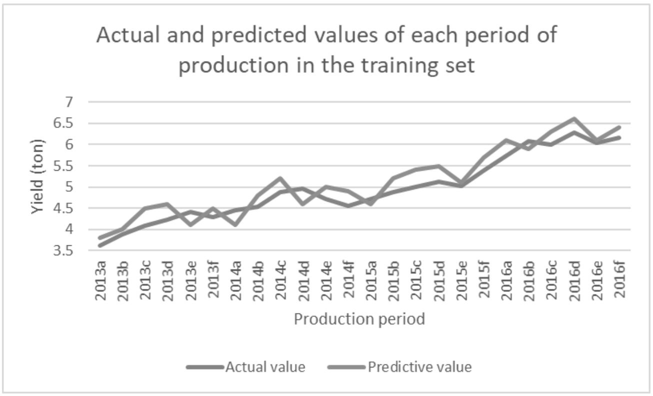


Fig. 9. Actual and predicted values for each production period in training set.

In the experiment of the yield prediction model, we selected the data of the same cabbage greenhouse in the past six years, with six productions per year. The data processed by the three applications of the above Spark framework is divided into a training set and a test set. We use the correlation coefficient, average absolute error, root mean square error, and relative absolute error as the overall evaluation index of the model. The true and predicted values of the training sample data are shown in Fig. 9, and the difference between the predicted value and the true value is shown in Fig. 10. As can be seen, all errors are less than 0.45, and the error is small, which is within an acceptable range. It can say that the established multiple linear regression model is more accurate.

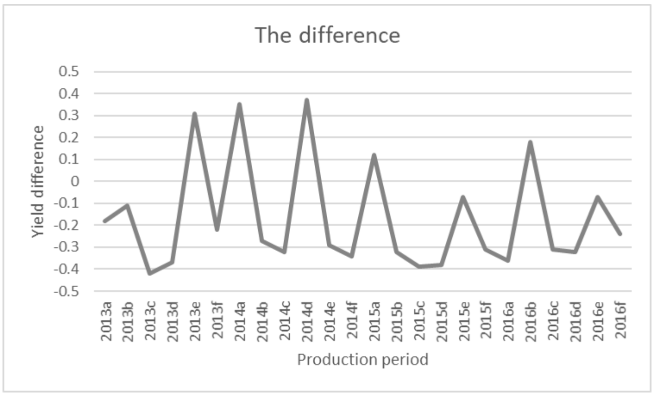


Fig. 10. The difference between the predicted and actual values of annual production in training set.

The actual and predicted values of the test sample data are shown in Fig. 11, and the difference between the predicted value and the actual value is as shown in Fig. 12. As can be seen from Figure 11 and 12, all errors are less than 0.35 with less error and are within acceptable limits. Therefore, the established multiple linear regression model is more accurate.

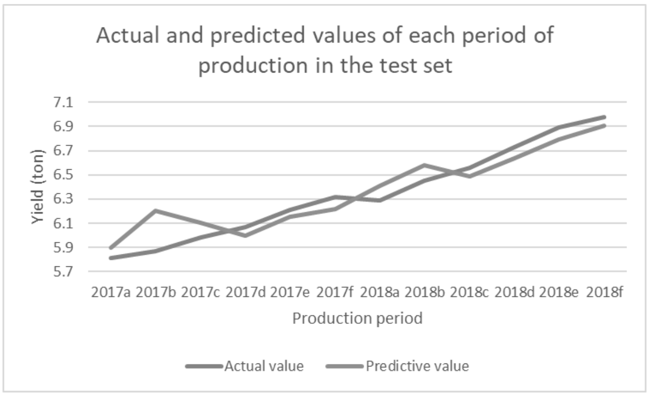


Fig. 11. Actual and predicted values for each production period in test set.

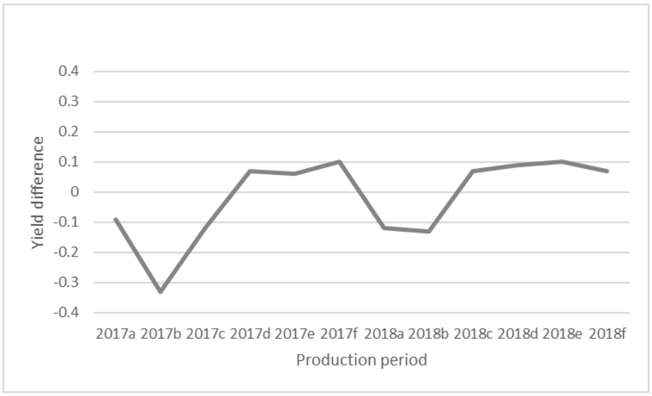


Fig. 12. The difference between the predicted and actual values of annual production in test set.

After the above test, it is considered that the established multiple linear regression model can be applied to the production forecast and the effect is better.

IV. Conclusion

Our work focuses on how to use Hadoop and Spark to achieve efficient processing of agricultural big data. In order to evaluate the performance of the framework and further predict the yield of crops, this paper developed and implemented two frameworks to meet the actual needs of the application. Hadoop was one of the first frameworks for big data processing and has been used by many large companies for many years. Hadoop is constantly evolving and improving to meet the needs of the new era, but also contributed to the creation of Spark. It is obvious from the experimental comparison that Spark processes data faster than Hadoop. This is mainly because Spark can save the processing results of each step of the program in memory, which greatly reduces execution time. It experimented with different sizes of input files, different number of active nodes, and different application types to confirm the best performance of Spark. We found that Spark is faster in every situation due to the efficient use of efficiency optimization techniques and the use of main memory. We also successfully implemented a yield prediction model based on multivariate linear regression through the API interface provided by Spark MLlib to obtain more accurate prediction results.

Of course, it have developed and compared other production forecasting models, and perhaps other models can get better predictions. In addition, in addition to the MapReduce function, Spark also supports a variety of new features, this paper just uses Spark SQL and Spark MLlib. Therefore, in the future work, we can continue to study the combined use of other functions, perhaps to better solve the problems in agricultural big data processing, and even develop new applications that have not been possible so far.

Acknowledgment

This research is supported by the key research and development plan project of Jiangxi Province (grant No. 20161BBE50086). Professor Yan Cheng is the correspondence author of this article.

References

[1] Li G J , Cheng X Q. Research status and scientific thinking of big data[J]. Bull. Chin. Acad. Sci., 2012, 27(6):647－657.

[2] Big Data: A Revolution That Transforms How we Work, Live, and Think" (with Kenneth Cukier), Houghton Mifflin Harcourt,2012, ISBN 978-0544002692.

[3] Sun Z F , Du K M , Zhang F X, et al. Perspectives of research and application of big data on smart agricultur[J]. J.Agric.Sci.Technol. , 2013, 15(6):63－71.

[4] Luo S M , Wang Z K , Wang Z P. Big-data analytics: challenges, key technologies and prospects[J]. ZTE Commun. ,2013,11(2):11－17.

[5] Wei Y Y. Research of ontology-based agricultural knowledge modeling and reasoning[D]. Hefei:University of Science and Technology of China, Doctor Dissertation, 2011.

[6] Yong H , Co P M . GIS Cloud Platform and Application of Agricultural Bank of China Based on Big Data on TB Level[J]. Geomatics World, 2016.

[7] Yun-Chi C , Yun Y E , Bo Z , et al. Design and Implementation of Web Services Platform for Agricultural Professional Town Based on the Cloud Service[J]. Hubei Agricultural Sciences, 2017.

[8] Stavrinides G L , Karatza H D . A Cost-Effective and QoS-Aware Approach to Scheduling Real-Time Workflow Applications in PaaS and SaaS Clouds[C]// International Conference on Future Internet of Things & Cloud. IEEE, 2015.

[9] Zaharia M , Chowdhury M , Franklin M , et al. Spark: cluster computing with working sets[C]// Usenix Conference on Hot Topics in Cloud Computing. USENIX Association, 2010.