Croujaction
A Novel Approach to Text-based Job Name Clustering with Correlation Analysis

Zunhe Liu, Yan Liu, Xiao Yang, Shengyu Guo and Buyang Cao
Department of Software Engineering, Tongji University, Shanghai, China

Keywords: Correlation Analysis, TF-IDF, Jobname Clustering, Hadoop Anomaly Detection.

Abstract: Job name clustering gradually becomes more and more important in terms of numerous anomaly detections and analysis of cloud performance nowadays. Unlike crude texts, job name is a kind of sequential characters or tokens. This made it a challenge for clustering based on job name text. In this paper we analysis the correlation between columns and use user-job correlation to improve classic algorithm TF-IDF. We optimize words tokenizing and feature sets generating. We use hierarchical clustering methods to implement experience. Finally we develop a module and evaluate the performance of optimized algorithm, delivering it as a product to a prestige e-commerce company.

1 INTRODUCTION

Detection of execution anomalies is important for the maintenance, development, and performance of large scale distributed systems. Anomalies detection (Chandola et al., 2009)focuses on both work flow errors and low performance problems. Now software often uses system logs produced by distributed systems for troubleshooting and diagnosis. However, manually inspecting system logs to detection is unfeasible due to the increasing scale and complexity of logs(Lou et al., 2010). Thus there is a great demand for automatic anomaly detection techniques based on log analysis.

The cloud computing continues to grow at an amazing speed. At the same time, there is also a quickly growing requirement of anomaly detection and cloud computing performance analysis. This leads to challenges that logs and information exists in the text or hypertext documents managed in an organized format.

Compared to classic text clustering methods, 3 major challenges must be addressed for clustering text-based job name clustering.

- Large size of job name: this requires algorithm to deal with large size of keywords
- Multiple tokens of job name: this require algorithm to split and tokenize job name by a sufficient regulation.
- Multiple columns of information: can be used as context to analyze relations between columns.

A lot of different text clustering algorithms have been proposed in the literature, including bisecting k-means(Huang, 1998), Scatter/Gather(Cutting et al., 1992), Apriori(Perego et al., 2001). These algorithms are efficient but not sufficient in the circumstances above.

Another challenge is how to define the evaluation of each keyword. The log consists of multiple columns of information, including job name, user name and other attributes of job(Fu et al., 2009). Based on TF-IDF algorithm(Ramos, 2003), we need documents and its keyword sets. In the log information data, there is no column or entity of document(Beil et al., 2002).

In order to solve the challenges above, we design an approach "Croujaction". This approach uses correlation between text for clustering. It describes how to cluster text contents which are stored in different columns in file. By using the correlation between columns, the approach avoids the disadvantage of TF-IDF algorithm which is needed a document entity when building vector-space. It allows us reduce the dimensionality of representative word and optimize the time consumption performance.

The rest of the paper is organized as follows. Section 2 briefly introduces the related work and theories. In section 3, we analyze the correlation between username and jobname. Conducting statistical knowl-

Liu Z., Liu Y., Yang X., Guo S. and Cao B..
Croujaction - A Novel Approach to Text-based Job Name Clustering with Correlation Analysis.
DOI: 10.5220/0005271601990204
In *Proceedings of the International Conference on Operations Research and Enterprise Systems* (ICORES-2015), pages 199-204
ISBN: 978-989-758-075-8

edge, we prove the relation correctness. In section 4, we introduce our novel approach of text based clustering and present the approach Croujaction. Section 5 reports its evaluation and improvements. Section 6 summarizes the paper and outlines some directions in future work.

2 RELATED WORK

Hadoop-based large scale distributed systems are becoming key engines of IT industry. However, most systems generate and collect logs and developers detect anomalies by manually checking system printed logs(Tan et al., 2009). It is very time consuming to diagnose through manually examine a great amount of log messages produced by a large scaled distributed system.

All methods of text clustering require several steps of preprocessing of the data(Neto et al., 2000). First, any non-textual information is removed from the documents. Then a term is a sequence of characters separated from other terms by some delimiters.

Most text clustering algorithms rely on the so-called vector-space model. In this model, each text document d is represented by a vector of frequencies of the containing m terms:

$$d = (tf_1, \ldots, tf_m). \tag{1}$$

Often, the vectors are normalized to same length to allow comparison between documents of different lengths. Even though the vector-space has a very high dimensionality after preprocessing.

To measure the similarity between two document d_1 and d_2 represented in the vector space model, typically the cosine measure is used which is defined by the cosine of the angle between two vectors:

$$similarity(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\|\|d_2\|} \tag{2}$$

where $d_1 \cdot d_2$ denotes the vector dot product and denotes the length of a vector.

3 CORRELATION ANALYSIS

Frequent patterns(Beil et al., 2002) are patterns (such as itemsets, subsequences, or substructures) that appear in a data set frequently.

Let A be a set of items. An association rule is an implication of the form A $\Rightarrow$ B. The rule A $\Rightarrow$ B holds in the transaction set D with support s, where s is the percentage of transactions in D that contain A $\cup$ B. This is taken to be the probability P(A $\cup$ B). The rule A $\Rightarrow$ B has confidence c, where c is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability P(B | A). That is,

$$support(A \Rightarrow B) = P(A \cup B) \tag{3}$$

$$confidence(A \Rightarrow B) = P(B|A) \tag{4}$$

$$\begin{aligned} confidence(A \Rightarrow B) &= P(B|A) \\ &= support(A \cup B)/support(A) \end{aligned} \tag{5}$$

After analysis, we dig out that column username and jobname having a strong correlation. Respective user in system always submits a bunch of specific jobs. On the other hand, a series type of jobs are always can be categorized by a specific user. Hence, we can assume that for all jobs named j in log file. There always exists a high confidence and support that it belongs to the user u.

4 CROUJACTION CLUSTERING APPROACH OVERVIEW

- Workflow

In the anomaly detection process, the log analyzer is presented with log files generated by system with columns of data. the system scatters the data into a number of system-designed document groups by the value of column username. Based on these documents, the algorithm select each for further study. Each document is processed based on TF-IDF algorithm and generate a vector-space for keywords in it. With all jobname texts in documents generating the keyword vector-space, the system could calculate the similarity of each two jobname and cluster job names by similarity.

4.1 Data Preprocessing

One of the major problems in text clustering is that a document can contain a very large number of words. It requires crucially an approach to apply preprocessing procedure that could greatly reduce the number of dimensions. Our system applies several preprocessing methods to the original job names, also namely documents, including numeric digits replacement, removal of stop words. Each of these methods will be briefly discussed next.

We now describe numeric digits replacement, where the job name is partially replaced by punctuation symbols. Replacement is applied to normalize

Figure 1: ID replacement result.

the job name because job name consist of some timestamps or meaningless numeric character sequences. Based on the timestamp format in the cloud system, Regex is designed and used to substitute timestamp with punctuation symbols.

Figure 1 shows job names before and after ID replacement.

4.2 Terms Representation

Data representation is usually straightforward. In general, data is represented as a set of records, where each record is a set of attribute values.

The term frequency of a word w in a document d, denoted TF (w, d), is the number of times that the word w occurs in document d. the higher the TF (w, d), the more the word w is representative of document d.

The document frequency of a word w, denoted DF (w), is the number of documents in which w occurs. The inverse document frequency of a word w, denoted IDF (w) is given by the formula:

$$IDF(w) = 1 + log(|D|/DF(w)) \quad (6)$$

Hence, the IDF (w) of a word w is low if this word occurs in many documents, indicating that the word has little representative power in documents. Oppositely, the IDF (w) of a word w is high if this word occurs in few documents, indicating the word has a great representative power.

In practice, we want words that have a high TF and a high IDF. We indicate the words importance of representative in the following formula:

$$TF - IDF(w,d) = TF(w,d) \times IDF(w) \quad (7)$$

4.3 Job Name Delimitating and Building Vector-Space

This section is intended to delimiter each job name into terms and build its vector-space. Based on the data representation and preprocessing, we need to delimitate each job name and calculate the TF-IDF value for each word.

First System applies splitting method to each job name. By using the replacement rules in the data preprocessing procedure, system defines a series of symbols as split tokenizes.

Next is the most important part in approach, system groups job names by user name. Each group of data is regarded as a document. After grouping, system splits each job name into terms in each group, building vector-space for each. Then system collects all the terms in each job name vector-space and calculates the occurrence of every term in it. Based on TF-IDF algorithm and our correlation analysis, we calculate TF-IDF value and assign it to each term. After all the procedure above, system builds vector-space for every job name with a representative value for each term within. Then system could calculate jobname similarity by using vectors.

4.4 Job Name Similarity Calculation

After building vector-space for each job name in data set, system can calculate cosine value between any two vectors. Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any other angle. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a Cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1].

4.5 Clustering Algorithm

Two different types of document clusters methodology can be constructed. One is a flat partition of the documents into a collection of subsets. The other is a hierarchical cluster, which can be defined recursively as either an individual document or a partition into sets, each of which is hierarchically clustered.

- Partitioning Clustering

Seed-base partitioning clustering algorithm has three phases:

1. Find k centers.
2. Assign each document in the collection to a center.
3. Refine the partition so constructed.

The result is a set of P of k disjoints document groups that each element in the data set belongs to one particular cluster.

```
while jobname in dataset {
      normalize(jobname);
}

for each row in dataset do
      documentDataStrucutre := groupDataByUsername
( jobname, username);

for each item in documentDataStructure do
      delimiter Item By Tokens;
      vectorForDocument := buildVector-spaceForKeywords;

generateVectorSpaceForJobname ( jobname,
vectorForDocument) ;

while data not in any group {
      initial center;
      value := evaluateSimilarity(jobname, center);
      clusterStructure := clusterJobnameByValue(jobname,
center);
}

return clusterStructure;
```

Figure 2: Pseudo Code of Croujaction.

Cluster initial procedure is intended to find centers in the data set. Particular in our system, the algorithm is applied to find one single center by randomly in Step 1. We implement Step 2 by assign each jobname to the selected center. Group the jobname to the selected center by a similarity comparison. Then system finds another center from the rest of job names as cluster initial procedure does and executes the previous steps again until all job names in data set have been clustered to one group. In Step 3, system squeezes each group clustered by running previous 2 steps. System calculates average similarity of each group and compares it to a threshold of average similarity. If below average threshold, system would assign the whole procedure to this group iteratively until every sub-group satisfies average threshold.

- Pseudo Code

Algorithm Crroujaction works with a hadoop log file, starting with a procedure scanning every jobname and applying preprocessing to it. It continues selecting each jobname and username pair as input. The algorithm groups jobname into document by username in this procedure by. Then it delimiters every jobname in each document and build vector-space with calculating coordinates. Next, each jobname builds its own vector-space by fetching its keyword corresponding coordinate from which in document. In the last step, system initials a jobname as center and calculates similarity of center and every jobname in log file. It continues calculating and clustering with jobname as step above until every jobname in file having been clustered in a cluster. Figure 2 presents algorithm croujaction in pseudo-code.

Resource	Job decrease ration	decrease ration of job occurs once
Data 1	80%	94%
Data 2	78%	93%
Data 3	77%	90%
Data 4	69%	83%

Figure 3: Decrease Ratio.

5 EVALUATION AND IMPROVEMENT

Croujaction has been evaluated on real hadoop log file for anomaly detection. It provides the service for detection analyzer for jobname clustering. In this part, we present a hadoop log file of a days history data. We apply croujaction algorithm to the data. In section 5.1, we describe the history data and conduct correlation analysis. Section 5.2 reports the main experimental result by illustrating clustering diagram.

5.1 History Data

To test and verify cluster algorithms, we collected 4 days' history data from a hadoop cloud computing node.

Data Set : data set contains 13 columns and nearly 15,000 rows items. It contains lots of job execution attributes including states, time, MapReduce numbers and so on. In our experiment, we focus on jobname and username these two columns.

First in order to evaluate system performance of data preprocessing procedure, we apply our replacement method to 4 different data sets. According to the replacement rule designed in section 4.1, we calculate the amount of job variety before and after the replacement procedure. Then, we calculate decrease ratio of job name. Figure 3 shows the result of experiment.

After the experiment, it shows that data preprocessing has an efficient effect on decreasing job name variety. Especially on that kind of job occurs only once with a particular timestamp or ID. Preprocessing erases the original effect because of meaningless character sequences in the job name.

5.2 Evaluation of Croujaction

To evaluate the clustering quality of algorithm, we design three experiments with different average similarity for each cluster. Based on the data processing above, we apply the clustering algorithm to the data set. First average similarity of experiment is 50%, while second one is 75% and third one is 90%. Due to the large size of data set, we extract typical jobnames

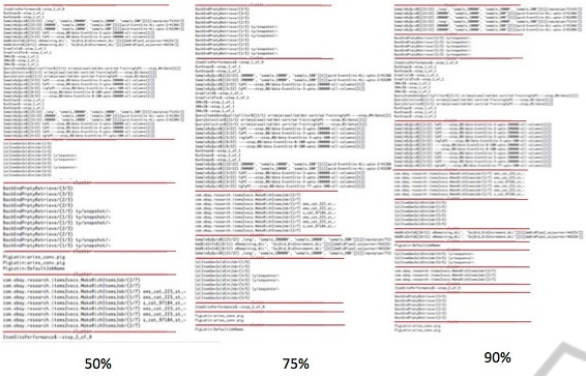

Figure 4: Croujaction Cluster Results.

and compress it to a proper size for a better showcase.

Figure 4 illustrates the results and drawing clustering in snapshots. Red line is used to separate different jobname clusters.

We observe that as the average similarity threshold increase more groups are clustered. A higher similarity threshold yields a better clustering performance. Note that in 90% diagram, compared with 50%, it provides a more accurate jobname clustering.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented a novel approach for text clustering. We introduced the algorithm Croujaction for hadoop log file analysis. It helps solve text clustering limitation caused by data storing in different columns in log file when using TF-IDF algorithm. In our experimental evaluation on the data set, we find correlation between different columns and group job names by user name as one document. This provides efficient foundations for text clustering. It presents a methodology for analyzing and clustering text contents in log file. It details the approach which could be used for correlation refine in contexts with columns format.

6.1 Limitation and Advantage

In our algorithm, we reference TF-IDF and we have seen that TF-IDF is efficient and simple for calculating similarity between texts. TF-IDF has its limitations. In terms of synonyms, it does make the relationship between words. In our system, we could avoid this limitation because we dont need to worry about semantic synonyms. We regard every word as string object and just compare them by characters.

6.2 Future Work

Finally, we would like to outline a few directions for future research. We already noticed that the most important parts in text base job name clustering are 1) data preprocessing 2) building vector-space and 3) clustering algorithm.

We could find out an improvement in data preprocessing especially a better replacement rule to meaningless characters. This could significantly speed-up in term delimitating process and help building vector-space more precisely. When system calculate vector ordinate for word, in job name, system could find more properties of word and apply some weight value to each word. This process may make the similarity calculation more accurate. In our approach, system just uses the simplest hierarchical clustering method in the last step of clustering. We plan to assign some other algorithms in data mining to our system. Thus, we improve the time efficiency and memory space in clustering process. In other perspective, we could deepen correlation analysis between more columns, complementing the space-vector building in TF-IDF algorithm.

ACKNOWLEDGEMENT

This work was financially supported by China Intelligent Urbanization Co-Creation Center for High Density Region (CIUC2014004).

REFERENCES

Beil, F., Ester, M., and Xu, X. (2002). Frequent term-based text clustering. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 436–442. ACM.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58.

Cutting, D. R., Karger, D. R., Pedersen, J. O., and Tukey, J. W. (1992). Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 318–329. ACM.

Fu, Q., Lou, J.-G., Wang, Y., and Li, J. (2009). Execution anomaly detection in distributed systems through unstructured log analysis. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 149–158. IEEE.

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery*, 2(3):283–304.

Lou, J.-G., Fu, Q., Yang, S., Xu, Y., and Li, J. (2010). Mining invariants from console logs for system problem detection. In *USENIX Annual Technical Conference*.

Neto, J. L., Santos, A. D., Kaestner, C. A., Alexandre, N., Santos, D., et al. (2000). Document clustering and text summarization.

Perego, R., Orlando, S., and Palmerini, P. (2001). Enhancing the apriori algorithm for frequent set counting. In *Data Warehousing and Knowledge Discovery*, pages 71–82. Springer.

Ramos, J. (2003). Using tf-idf to determine word relevance in document queries. In *Proceedings of the First Instructional Conference on Machine Learning*.

Tan, J., Pan, X., Kavulya, S., Gandhi, R., and Narasimhan, P. (2009). Mochi: visual log-analysis based tools for debugging hadoop. In *USENIX Workshop on Hot Topics in Cloud Computing (HotCloud), San Diego, CA*, volume 6.