Machine Learning Log File Analysis

Research Proposal

Kieran Matherson
ID: 1154908
Supervisor: Richard Nelson

13 March, 2015

Abstract

The need for analysis of systems log files is increasing as systems grow larger and more complicated the quantity and complexity of log files grow. This project will take an exploratory look into how machine learning analysis performs on log files by using textual classification tools to explore these types of documents and observe whether events and failures can be identified.

1 Introduction

Can current log file analysis methods be relied upon to correctly identify all events and failures? The current solution for this is most commonly done with regular expressions which the programmer must manually create to identify the events they want to find[14]. This method could be prone to missing events that were left out of the regular expression and it also requires constant tuning and adaption to correctly function.

This is where machine learning analysis comes into the picture. The goal of this project is to test different text based classification schemes on log file data and see whether any of the algorithms can identify events and failures thus eliminating the need for constant tuning; there is also the possibility of discovering events that the were not thought to exist in that data.

2 Background

There are many popular solutions to automatic log file analysis already available. One of the more popular pieces of software is Splunk and then there is a wide variety of alternatives being aimed at both the enterprise and the personal demographics, many being free and open source like Graylog2, Logstash and Fluentd with a multitude of other alternatives existing[10, 5, 6, 4].

There has been some previous success at using machine learning techniques on log data for examples include clustering[16], principal component analysis[18], support vector machines and random indexing[11], genetic algorithms[17] and inductive learning[13]. Numerous blogs have discussed trialling machine learning algorithms on log files and there is a reference of someone developing a toolkit but no definite conclusions on their usefulness was found which is why the results of this project will help us discover the usefulness of unsupervised document classification machine learning algorithms in log file analysis[14].

2.1 Text Classification

The type of machine learning that will be explored is unsupervised learning. This is when a label is not put to the training data; in this case the learning algorithm is not told what events and failures it is looking for, it is left purely up to the algorithm to cluster data together and find these events by itself.

On the other hand if supervised learning were to be used it would be told what are events and what are not events so that it can learn what to look for; this however is not what is aimed to be achieved with this project, the aim is not to tell it which events to find because this could be accomplished with the means that are already available. Though that is not to say supervised learning would not be effective but the goal of this research is to see if the learning algorithm can correctly identify these events by finding patterns in the log file.

There are many available machine learning algorithms to test, so popular unsupervised document classification algorithms will be selected. Since in this field there is no general do it all algorithm to accomplish this analysis, it will be required to select from suitable algorithms to see which work best with this particular form of data.[12, 15]

2.2 Syslog

The first type of log files that will be examined are those handled by the Syslog protocol which separates the message logging system into a smaller process, separate from the process that generated them. These logs are generated from any process that transmits a message to the Syslog server so their type can range anywhere from mail to kernel and numerous others. Any of these logs could be analysed by itself to find events in that log or even as a possibility a combination of different logs that have a relationship could be analyse together, but this will need to be tested to find the most effective method. Each entry is made up of a timestamp, hostname or IP, tag or process ID and finally a text message; these elements are what the learning algorithm will use to identify events.[19]

2.3 Big Data

Once past the initial proof of concept phase the next step is to move onto testing real world data most likely from an interested company with a large system. Once access to a log file database has been obtained there will be the challenge of getting the data out to process it which would require the knowledge to query that database. The database will most likely be an implementation of a NoSQL database like Apache Cassandra, Couchdb, and Mongodb[1, 3, 8].

The reason a large system will have a database to store their logs is because in systems that large it is the most practical way of storing the multitudes of logs that are output which will consist of thousands of lines of text. Once a method of extracting this data has been learnt the next step is to get the data into some format compatible with the current tool that is being used, then run it through the tool to perform an analysis.

3 Planned Approach

1. Learn a text based classification tool to run algorithms on log files. The first tool to be trialled will be Mallet, but there are also other alternatives that could be used like NLTK(Natural Language Toolkit) and Apache UIMA[7, 9, 2].

2. Once familiar and confident with using Mallet(or another tool), Syslog log files will need to be obtained to start testing different algorithms. The parameters of these algorithms will be fine tuned on the data to best identify events.

3. Now that results are available a decision point is reached; here it may decided that current methods are not working and another approach will need to be tested; for example changing tools or a further investigation into an interesting point that was discovered. If however it is found that machine learning analysis proves to find useful information, then the next step is to move onto real world data testing.

4. The next stage would be to move onto data from a real system and perform analysis on their logs which are likely stored in a NoSQL type database; this may require becoming familiar with the type of log file storage system that the system is implementing to be able to retrieve the log file data. Once the data can be retrieved the findings from what was discovered in the previous experiment will be used to redo the testing on the new real world data.

4 Evaluation

The techniques and tools that show promise will be assessed on how well they identify events and failures if possible or just whether they actually can identify any events. It will then be decided whether they need further testing and tweaking of parameters.

However when monitoring these logs for events it would be hard to give a quantitative measurement to how well the algorithm is identifying events due to the lack of data available on these log files; since the log files are so large to manually go through and identify events it would be near impossible due to how long it would take and what the person reading the log would miss because they would not be able to remember and recognise every pattern. A method may need to be devised to give a measurement of how well the algorithm performs.

5 Conclusion

To discover if machine learning log file analysis can effectively identify events and failures in a system, real world log file data will be processed with text based classification tools and unsupervised learning algorithms to discover whether events and failures can be identified.

References

[1] Apache cassandra. `http://cassandra.apache.org/`. Accessed: 13/3/2015.

[2] Apache uima. `https://uima.apache.org/`. Accessed: 13/3/2015.

[3] Couchdb. `http://couchdb.apache.org/`. Accessed: 13/3/2015.

[4] Fluentd. `http://www.fluentd.org/`. Accessed: 17/3/2015.

[5] Graylog2. `https://www.graylog.org/`. Accessed: 17/3/2015.

[6] Logstash. `http://logstash.net/`. Accessed: 17/3/2015.

[7] Mallet. `http://mallet.cs.umass.edu/`. Accessed: 13/3/2015.

[8] Mongodb. `http://www.mongodb.org/`. Accessed: 13/3/2015.

[9] Natural language toolkit. `http://www.nltk.org/`. Accessed: 13/3/2015.

[10] Splunk. `http://www.splunk.com/en_us/homepage.html`. Accessed: 14/3/2015.

[11] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, Mikko Terho, and Jelena Vlasenko. Failure prediction based on log files using random indexing and support vector machines. *Journal of Systems and Software*, 86(1):2–11, 2013.

[12] Youngjoong Ko and Jungyun Seo. Automatic text categorization by unsupervised learning. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 453–459. Association for Computational Linguistics, 2000.

[13] Wenke Lee. Applying data mining to intrusion detection: the quest for automation, efficiency, and credibility. *ACM SIGKDD Explorations Newsletter*, 4(2):35–42, 2002.

[14] Jon Stearley, Sophia Corwell, and Ken Lord. Bridging the gaps: joining information sources with splunk. In *Proceedings of the 2010 workshop on Managing systems via log analysis and machine learning techniques*, pages 8–8. USENIX Association, 2010.

[15] Patrick Trinkle. An introduction to unsupervised document classification. `http://userpages.umbc.edu/~tri1/docs/unsuperdocumentclass.pdf`, 2009. Accessed: 17/3/2015.

[16] Risto Vaarandi et al. A data clustering algorithm for mining patterns from event logs. In *Proceedings of the 2003 IEEE Workshop on IP Operations and Management (IPOM)*, pages 119–126, 2003.

[17] Gary M Weiss and Haym Hirsh. Learning to predict rare events in event sequences. In *KDD*, pages 359–363, 1998.

[18] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 117–132. ACM, 2009.

[19] Kenji Yamanishi and Yuko Maruyama. Dynamic syslog mining for network failure monitoring. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 499–508. ACM, 2005.