Technology review: NLP and ML techniques for log analysis to detect anomalies

Introduction

Digital transformation around the world along with the fast advances in the computing technologies (e.g. cloud computing, mobile apps, etc.) create a new reality of exponential explosion of the software systems, applications and services supporting all the aspects of our life. These systems running 24/7, they're characterized with high levels of complexity in architecture, constantly scaled up workloads and highly distributed (e.g. micro-service architecture, containerization, multi-region clustering, etc.). While the availability and reliability of these systems are of the most critical aspects, monitoring and operating them with the old ways don't scale. One of the type of tasks which is becoming extremely challenging in this new reality is log analysis to detect of errors and faults for timely corrective action. Beyond the challenge of a huge scale of the generated logs to be manually processed by operator, the old search and rule-based filtering methods have hard time to generate valuable insights due to architectural complexity.

In this review, we learn and present several approaches suggested and published by a few researchers over recent years. They apply machine learning and NLP methods and algorithms to log analysis in attempt to build mechanisms for automatic detection of the anomalies in the systems behaviors.
"In data analysis, ***anomaly detection*** *(also* ***outlier detection****)[1] is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data."*
(wikipedia definition based on *Zimek, Arthur; Schubert, Erich (2017), "Outlier Detection", Encyclopedia of Database Systems, Springer New York, pp. 1–5*)

Review approach

We start with presenting the ML methods surveyed by He et al which were presented at 2016 IEEE 27th International Symposium on Software Reliability Engineering. They explored, implemented and evaluated both supervised and unsupervised methods of anomaly detection – 3 per each type.
Then we present an experience report from Bertero et al. which they presented at 2017 IEEE 28th International Symposium on Software Reliability Engineering. This team built experimentation framework for anomaly detection in systems using syslogs mining. They implemented and evaluated 3 supervised learning methods though the novelty in their approach was to focus on the unstructured part of a log message and leverage NLP, specifically *word2vec* embedding for text data representation.
Finally, we discuss the above approaches and try raising some further ideas for leveraging NLP techniques in the domain of log analysis for anomaly detection.

ML experience report

He et al. used 2 labeled datasets – HDFS and Blue Gene/L (BGL further). They've implemented a log parser and anomaly detection tool applying ML algorithms on the datasets. The approach used here was to transform unstructured text of log messages into structured data through identification of templates. Then log messages are mapped into event templates and grouped into log sequences with one of the ways – fixed, sliding or session window (where session ID is available). Feature extraction is done by encoding each group into the vector of aggregated counts per event type in the sequence. The set of the vectors comprise a matrix which is used to train the ML models. The ML methods which this group of researchers implemented were Logistic regression, Decision tree, SVM, Log Clustering, PCA and Invariants Mining.
The researchers used Precision, Recall and F1 measure in evaluation of the learning methods accuracy. Their observations included: 1. supervised anomaly detection methods performed better from the precision standpoint though on the recall parameter it depends on the dataset; 2. sliding windows approach performed better than the fixed window's; 3. invariants mining has stable high performance; 4. window size and step size affect differently the accuracy of the supervised and unsupervised methods; finally – 5. wrt efficiency most of the models perform linearly, except for the log clustering method.

ML experience augmented with NLP

A year after He et al. published their experience report a group of researchers from LAAS-CNRS published an experience report on leveraging the NLP to detect anomalies in syslogs. Obviously, NLP doesn't substitute the ML algorithms but rather use an NLP technique to preprocess the log messages before training the model. Specifically, they used *word2vec* to generate embeddings and this allowed to incorporate similarities between words meanings into the modeled data. In this case the researchers team built a dedicated experimentation platform to prepare the datasets. They deployed an open source Clearwater VNF system for video calls over SIP protocol and decorated the deployment with their components - experimentation campaigns runner, errors emulator and logs collector. Errors emulator provides an ability to emulate errors of 4 types – CPU, memory, disk i/o and network – by creating stress, disk writing workers, memory reservations without release and iptables manipulation. Experimentation campaign is a run with a set of configurable parameters – subset of target components of Clearwater, subset of error types for emulation {CPU, memory, disk, latency, packet_loss}, injection and clean run duration as well as offset.
For machine learning a dataset of 660 prelabeled files was synthetically generated using the experimentation campaigns – 330 normal and 330 anomalous. *cbow* method of *word2vec* was applied to the concatenation of the words from the entire dataset (660). 233K words populated the corpus of the vector space. At the next step each of

660 log files was mapped into the vector space through *centroid* method (a file vector is computed by averaging each line whereas each line's vector is computed by averaging the contained words vectors) or *tfidf*. Then the matrix of 660XI*space*l with the labels was used to train the models. Three supervised learning classifiers were used in this experiment – Naïve Bayes, Random Forest and Neural Network. For evaluation this experiment used 10-fold validation approach (i.e. 90% of the dataset was used for training the models and the rest 10% as tested instances).

Discussion

Doing relative evaluation of the performance (i.e. effectiveness and efficiency) of two approaches may not make much sense since the experiences vary on all the aspects – datasets, applied algorithms, experimentation environments, collected and published measures, etc. Instead we'll try to summarize interesting characteristics of each approach on some relevant dimensions and spot the strengths of each. We'll further suggest some ideas and directions for future exploration.

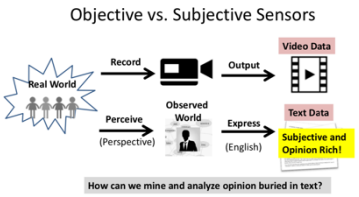
Aspect/ Report	Experience Report: System Log Analysis for Anomaly Detection (He et al.)	Experience Report: Log Mining using Natural Language Processing and Application to Anomaly Detection (Bertero et al.)
Dataset type	Manually labeled real files	synthetically generated labeled log files
Datasets size	15,923,592 log messages and 365,298 anomalous instances	330 normal log files + 330 file under stress (233K words in total)
ML/ NLP methods applied	Logistic regression, Decision tree, SVM, Log Clustering, PCA, Invariants Mining	*word2vec* (cbow and skip-gram), Naïve Bayes, Random Forests, Neural Networks
Implementation tools	(mostly) scikit-learn	scikit-learn
Learning approach	Identify log messages skeletons stripped of the parameters; group consequent log messages in a fixed, sliding or session window; use log event counts as a feature vector	Apply *word2vec* on the 660 files corpus vocabulary; compute the feature vectors of files by aggregation of the contained word vectors; train and test the models with 10-fold validation approach

The contribution of the experience and research done by He et al. was in the systematic approach of implementation and evaluation of six different ML algorithms based on the

prior work of different researchers. They were able to compare the performance of those and spot interesting findings.
Bertero et al. incorporated NLP method in their experience. Specifically, they used *word2vec* algorithm to represent the words for ML training. This allowed them to refrain from tedious log preprocessing, e.g. parsing as well as tagging of the events. When implementing this approach they intentionally trade in some portion of accuracy due to ignorance of the log messages specific structures. Though they achieve more generality as their model is agnostic to log messages format.

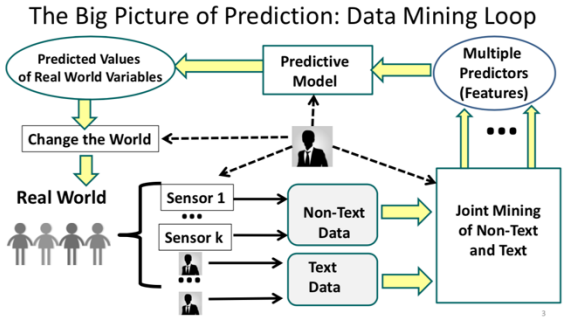
Future work

Inspired by the approach of Bertero et al. of augmenting the ML models with NLP for anomaly detection by means of log mining we'd like to present an idea of modeling this problem in terms of joint data mining. First let's try to create a model of this use case:



Real World -> Runtime system
Objective sensors -> {metrics (e.g. OS, daemons, container facts, etc.), behaviors (e.g. crashes, scale-out, etc.)
Subjective sensors -> developers who code the log messages based on the observed state of the runtime system

Now we can try to formulate a data mining loop in terms of this problem:



Apply sentiment analysis on log messages to mine state of the running system/ components
Use runtime metrics coming from sensors to improve the accuracy of the features and the predictive model
Adjust runtime system/ application configuration based on the predicted (e.g. load, failures, latency)
Refine the set of sensors to improve the accuracy of learning

Conclusion

Detecting anomalies through analysis of log messages became an important application of text mining to avoid "manual" human consumption of huge volumes of data while operating and monitoring and operating modern systems.
Over recent years various methods of both text clustering and text categorization were suggested and explored. He et al. built a tool which implemented six ML algorithms and the team published the experience report with the relative evaluation of the algorithms performance on the real pretrained HDFS and BGL data. Although this team used textual part of data of the log messages they parsed their dataset items basically "structurizing" their data ahead of feature extraction for ML models.

An interesting novelty of leveraging NLP on log messages before applying the ML algorithms was suggested by Bertero et al. By using that technique they avoided the burden to parse and structure the text data though compromised the accuracy of classification.
Towards the end of the approaches discussion we suggested some ideas of using sentiment analysis and applying joint mining of log text messages to this problem space. Log messages may be referred to as developers sentiments while the running system metrics serve as non-text data generated by sensors.
Anomaly detection through log messages mining is a hot research area where ML models augmented with NLP techniques can be successfully applied.

References

https://hal.laas.fr/hal-01576291/document
https://jiemingzhu.github.io/pub/slhe_issre2016.pdf
https://www.slideshare.net/japerk/nlp-techniques-for-log-analysis
https://blogs.oracle.com/datascience/introduction-to-anomaly-detection

https://www.youtube.com/watch?v=Dt81qwza-zA&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v
https://www.youtube.com/watch?v=PJ_kx9-OPgc&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=2
https://www.youtube.com/watch?v=At19CBGpbMI&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=3
https://www.youtube.com/watch?v=5vrY4RbeWkM&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=5
https://www.youtube.com/watch?v=mG4ZpEhRKHA&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=6
https://www.youtube.com/watch?v=5nfe835TVcY&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=7
https://www.youtube.com/watch?v=H4J74KstHTE&list=PL7-o3Fa2I_AluoTm_Y490nSAq9C4qps_v&index=8