A Review of Unstructured Data Analysis and Parsing Methods

Shubham Jain*, Amy de Buitléir†, and Enda Fallon*
*Software Research Institute, Athlone Institute of Technology, Athlone, Ireland
sjain@ait.ie efallon@ait.ie
†Network Management Lab, Ericsson, Athlone, Ireland
amy.de.buitleir@ericsson.com

***Abstract*—Computer applications generate an enormous amount of data every day through their logs, system-generated files or other reports. This generated data depicts the state of the running system and contains abundant information that can be used for system diagnostics and monitoring. Network monitoring systems produce a wide variety of unstructured information, so there is a need for an automated way to extract the relevant data, which currently requires multitude of custom parsers. Developing and testing custom parsers can be time-consuming. Instead, data can be automatically processed and parsed into a machine-readable format, building a generic model for standard or vendor-specific data, and generating insights for analytics, anomaly detection, intrusion detection, node failures and various other applications. This paper reviews some existing approaches for unstructured data mining and parsing and discusses the challenges in information extraction, creation of knowledge bases and presents a generic framework for automatic parsing.**

***Index Terms*—Data Mining, Information Extraction, Similarity, NLP, Knowledge base**

I. Introduction

Modern systems generate useful insights which depict the run-time behavior through logs or trace messages. Due to the complexity and scalability of modern applications the volume of meta-data generated is enormous [1]. These unstructured log messages can be processed and parsed to be made available for anomaly detection [2] [3] [4], cognitive management [5], autonomous problem validation [6] [7], incident management or root cause analysis. However, data from logs, trace messages or other print statements can be non-linguistic and unstructured, and thus cannot be used directly for analysis. Creation of a knowledge base from unstructured log messages has been widely studied and implemented in recent years. However, manual domain-specific construction of knowledge base dominates the current literature as there are no domain-independent automatic frameworks available for creation of knowledge bases [8].

Constructing a knowledge base even for a single domain in a large scale system can be very challenging. One such domain is the telecommunications, where despite the presence of logging standards such as BSD [1], the structure of different components can have great variance. Radio Base Stations (RBS) generate tons of log messages every hour for a large scale system, and the components are developed by various engineers in different programming languages; thus the structure varies considerably. Explicit programming for construction of knowledge bases from these logs needs expertise in recognising the relevant structures that contain information and disposal of redundant information. Even after we automate the process for information extraction through custom parsers, it still needs to be re-run periodically to adapt to changes in policies or standards. Thus, the research suggests that it is not practical to use a supervised approach for extracting all relevant data, even for a single large-scale domain.

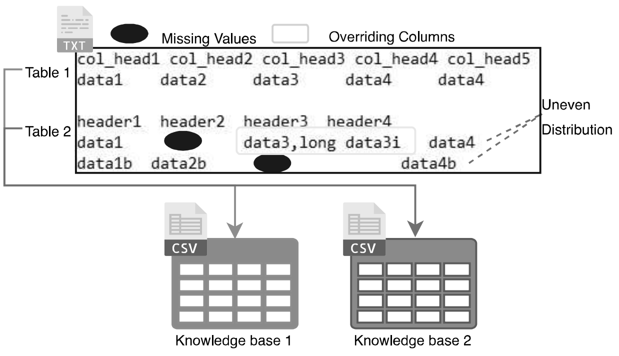
TXT
Missing Values
Overriding Columns
Table 1
col_head1 col_head2 col_head3 col_head4 col_head5
data1 data2 data3 data4 data4
Table 2
header1 header2 header3 header4
data1 data3,long data3i data4
data1b data2b data4b
Uneven Distribution
CSV
CSV
Knowledge base 1
Knowledge base 2

Fig. 1: An Illustrative Example of Unstructured Data Extraction

In this paper, we survey different algorithms and approaches utilized for unstructured information extraction using semi-supervised or unsupervised learning to recognise relationships for template extraction. A simple illustration of unstructured data extraction is shown in Fig. 2 where a plain text document is analysed and parsed to generate two CSV files. As seen in the figure, the data contains missing values, is misaligned and includes overriding columns. Furthermore, the tables in Fig. 2 have different structures. There can be thousands of such structures with large variance in text distribution and formatting in a large scale system. Current research focuses on achieving automation using natural language processing, data mining, machine learning and deep learning. Thus, the problem of pattern or relationship extraction is extremely challenging due to the uneven distribution and high degree of variation between different structures [9].

This paper reviews the most commonly used algorithms for unstructured data mining and parsing, groups them into

[1]https://www.hjp.at/(de)/doc/rfc/rfc3164.html

multiple components according to function, and organises the components into a generic framework for automated information extraction. This framework (shown in Fig. 3) allows the reader to select and combine the best components for a particular solution.

This paper is organised as follows: §II describes an overview of different available algorithms and tools and compare different sources that utilize them, challenges and their limitations. We also provide the summary of related work in a sequential manner by using a layered component diagram. §III details some experiments to compare different algorithms for unstructured text analysis. §IV describes our conclusions.

II. Unstructured Information Extraction

In this section we discuss the different approaches for information extraction from unstructured data. We divide them into relationship-based, template-based and deep learning-based approaches. We further discuss the challenges in extraction using the techniques described in the literature.

A. Relationship-based Extraction

Unstructured data is a text-heavy collection of characters with no predefined model, that often contains a stream of information that can be classified based on the relationships between the characters. The authors of [10] made use of named entity recognition and generic text classification to extract relevant information from an unstructured message (see Fig. 2 for example). They also made use of Conditional Random Fields (CRFs)-based statistical Named-Entity Recognition (NER) which extracts real world entities and classifies them into pre-defined category based on the data-model. CRFs recognize the pattern of words rather than individual words which allows the classifier to recognize entity by the patterns of sequence containing it. Text was classified into 3 types: Text-only, Features-only and both, using a Support Vector Machine (SVM) classifer. The use of Term Frequency / Inverse Document Frequency (TF-IDF) [11] to provide weights to the word distribution in the document along with the attribute selection method made it even better.

The experiments by [12] supported that TF-IDF generated better results compared to simple features such as character bi-gram and timestamp statistics. They also experimented with different learning algorithms and concluded that variants of Self Organizing Feature Map (SOFM) [13], which creates a low-dimensional representation of the input space for dimensionality reduction, yielded better results than basic clustering algorithms like K-means [14] and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [15] which groups together points that are closer to each other in a bi-dimensional space by calculating distance between them. These variants solved the problem of matching multiple clusters into required categories. The authors analyzed high volume data and suggested the use of dimensionality reduction to remove irrelevant feature terms and noise reduction.

The authors of [8] focused on meta data extraction from various sources to predict user preferences. The straightforward algorithm is based on determining boundaries of statements, identifying nested words and assigning them labels based on the context to extract features. They made use of part-of-speech-tagging followed by morphological analysis to generate implicit meta-data about the features on a linguistic model.

On the other hand, [16] proposed a feature learning framework that takes advantage of the vast amount of expert knowledge available in unstructured form on the Web. They learn a Hierarchical Pachinko Allocation Model (HPAM) [17] to discover set of latent variables such as super-topics and sub-topics. Using a directed acyclic graph for topics and calculating the Earth mover's distance between the latent variables they generate a similarity matrix which is used to partition messages based on similarity score.

Relationship based extraction are feature-driven and require linguistic data or a meta-dictionary to perform analysis. Most of the algorithms can be implemented using Natural Language Processing-based libraries.

B. Template-based Extraction

Unstructured information can be non-linguistic and may not have a standardized dictionary of words. Reference [3] proposed a blackboard architecture, an agent-based approach to extract and interpret table templates in heterogeneous sources. Each agent worked to analyse data and apply a set a protocols to label it or to classify it. Output data was exchanged between different agents and used as an input for next processes. The blackboard architecture together with multi-level evaluation measures, reported the effectiveness of certain agents and their collaboration. However, most handcrafted rules for learning and validation are not viable due to frequent changes in the system. Each agent has their own set of protocols that are not valid for different unstructured messages in a corpus.

In contrast to the agent-based approach, new pattern learning techniques to form event templates have been introduced and implemented by various researchers. The goal of such techniques is to differentiate between the constant part and the variable part in a message and prepare templates of variable length. Simple Logfile Clustering Tool (SLCT) [18] is among the first techniques that worked on data-driven automated log parsing. It is widely used for various mining tasks due to its availability as an open source tool. The algorithm is based on association rule mining with a three step procedure. The first step is word vocabulary construction. It iterates through the data and captures meta-data about word frequency and distribution. The second step includes cluster construction with the inputs from word vocabulary. In the third step, clustered log messages are then combined to to create a template.

On the other hand, the authors of Iterative Partitioning Log Mining (IPLOM) [19] did a heuristics-based parsing of log messages that were dependent on the characteristics of unstructured messages. The template construction is inspired from SLCT, but it performs a three-step hierarchical partitioning procedure followed by template construction in the last step. They partition log messages based on message lengths, and for each partition, the position word distribution

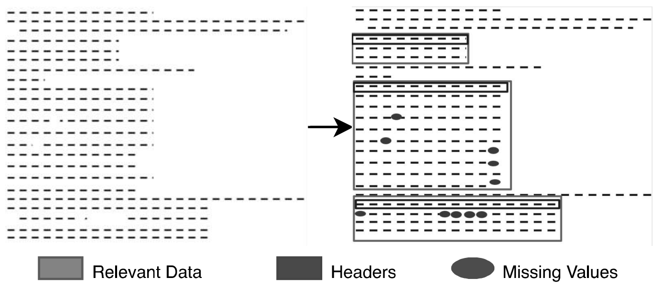

Fig. 2: An Illustrative Example Detecting Tables using Content Information

is calculated. Then, they dynamically split messages at the position where the number of constant words is the least. They further partition messages by mapping set of unique tokens with the tokens returned by the heuristic criterion. The last step includes log creation using SLCT.

LKE (Log Key Analysis) [20] is a log parsing method which combines clustering and heuristic rules. Messages are clustered with predefined weights using hierarchical algorithms. Clusters are split using the heuristic rules. Log templates are created, as in other approaches.

Whereas LKE partitions messages using hierarchical clustering and IPLOM clusters using word distribution, LogSig [21] calculates a value for each log messages by generating a word-value pair for each message and clusters them after a number of iterations. The method works well when there are relations between words in an unstructured message. Those relations can be converted to key-value pair for a template generation. The process for template generation is same as SLCT and IPLOM.

Additionally, various data mining techniques have been proposed by researchers to extract non-linguistic data patterns. Reference [22] recognised templates by clustering messages using sequential algorithms in a log file. They focused on clustering information using an algorithm they call PARIS (Principle Atom Recognition In Sets) which extracts event-based log messages by dynamic creation of event dictionaries. The study suggested that most of the approaches utilize either 1) Frequent pattern mining, 2) Clustering or 3) Heuristics based analysis for template-based information extraction.

C. Deep Learning-based Extraction

Deep learning approaches that analyse data to extract information are data-driven and efficient and don't rely on heuristics or rule-based methods. DeepDeSRT [23] is a deep learning-based solution to extract templates within a document using a pre-trained model of Faster R-CNN [24]. DeepDeSRT performs image analysis on document to detect a table-like structure and extract and store information from the recognised template. The approach differs from traditional data analysis algorithms as instead analysing raw text, an image of the document is analysed. The authors focused on two things 1) table detection and 2) structure recognition. They used the concept of object detection in natural scene images and applied it to table detection in documents. They detected tables by generating region proposals based on input images and classifying it with Fast-RCNN. After detecting a table, the authors segmented the table into regions such as rows, headers and columns using a deeper FCN-Xs architecture [25] that can extract extra details in the shallow layers and provides delineation of rows and columns and X is the number of shallow layers. The approach has been only used on PDF tables, but can be used to detect structures in plain-text unstructured documents that look like tables. The diagram in Fig. 2 shows a structure that can be processed using object detection algorithms to detect table and header structures. The green border shows the table boundaries that would be identified and the blue border shows the header boundary within the table. The missing values are shown using red ovals that can be detected using this approach.

Another deep learning technique is presented in DeepLog is presented in reference [26]. It does not itself perform data extraction but recognizes patterns in word distribution using Long Short-Term Memory (LSTM) [27]. The authors propose a learning framework for anomaly detection on unstructured system logs and cluster them using a neural network model. The patterns generated in this literature can be utilized to extract and store event messages.

D. Problems

Unstructured text files may contain abundance of valuable information about the system, but the extraction of this information can be quite challenging depending on the structure and format of information.

Missing Values. Template extraction in an unstructured message depends on the semantic characteristics such as length and white-space distribution. The red ovals in Fig. 2 represent missing values, a common problem in unstructured messages. Most of the pattern-based extraction methods [18] [19] [20] [21] create event templates for a single message and cluster them together to form a pattern. Such structures can be combined together to form a table-like structure. But, with missing values the length of the message changes and a novel event template is generated in the algorithm.

Non-linguistic. Log messages, trace logs and print statements from domain-specific servers are usually technical and non-linguistic. Natural language processing (NLP) has been widely used for information extraction in log files [28]. However, NLP techniques are limited to a language dependent data that has a predefined vocabulary for analysis. An unstructured corpus of words in system logs contains technical words which are nested form of base words. Analysis of such documents using NLP requires creation of meta-dictionaries. Thus NLP techniques fall short for extracting information from a document that contains technical non-language-specific words.

No meta-data. Data-driven extraction techniques follow a heuristic-rule based extraction. The rules are dependent on the characteristics of an unstructured message which are decided based on the meta-dictionary of the words. With no information about the contents of a message, it becomes challenging to extract relationships or templates. Feature-driven techniques

[8] [10] [16] require a dictionary of the words before analysis. However, the techniques fall short in addressing extraction in the absence of meta-data.

Multiple templates. Execution traces, system logs in a streaming application constantly generate information. Such information can be parsed in documents, where each document can be analysed to find structures. Deep learning techniques [23] that extract a table-like structure, process the document to detect the tables. Various machine learning techniques detect table structures by white-space distribution analysis and detect components such as header, row and column. There can be multiple table-like patterns in a single document which can be extracted and parsed directly. However, little research has been conducted on extracting multiple templates using context analysis and white-space distribution.

III. Automatic data Analysis and Parsing

In this section we summarise the algorithms and techniques discussed earlier, and show how they can be combined to automate parsing of unstructured text. Fig. 3 aggregates a variety of approaches used by many researchers to develop a tool for unstructured data parsing. It also categorises common tools and algorithms according to their role in a generic solution. Each component describes a step that provides input to the next layer for further computation. The literature survey suggests that the entire process of unstructured information parsing can be divided three major steps which are *extraction*, *computation*, *bench-marking*, *evaluation* and *application*. We further classified *computation* into four layers that are *transformation*, *modelling*, *execution* and *storing* on top of *parallelization* layer. Each component interacts with the next component using combination of one or more algorithms described in each layer. The components are discussed further below.

Extraction. In this step, log messages, execution traces and other print sequences are collected from heterogeneous sources. These sources have a predefined data-model that makes it possible to develop a heuristic rule-based extraction technique. These sources provide meta-data about the characteristics of log messages. Based on the literature review, sources can be categorised as one of the two types 1) Standard-specific, 2) Vendor-Specific. *Standard-specific* messages follow a standard-specified format such as IEEE, IETF and 3GPP. *Vendor-specific* sources are from vendors such as Ericsson, Huawei and Verizon that generate log statements and follow a specific standard on top of their own set of rules. Determining the characteristics and writing rules is possible in this component.

Transformation. The first step after data collection is transformation of data to make it available for processing. Various authors have used regular expressions to format data. SLCT, LKE, IPLOM and LogSig use regular expressions to create event templates. The documents are normalized using NLP techniques such as lemmatization and stemming [29] which is a method to remove inflectional endings and return the base dictionary form of the word. Data-driven extraction techniques

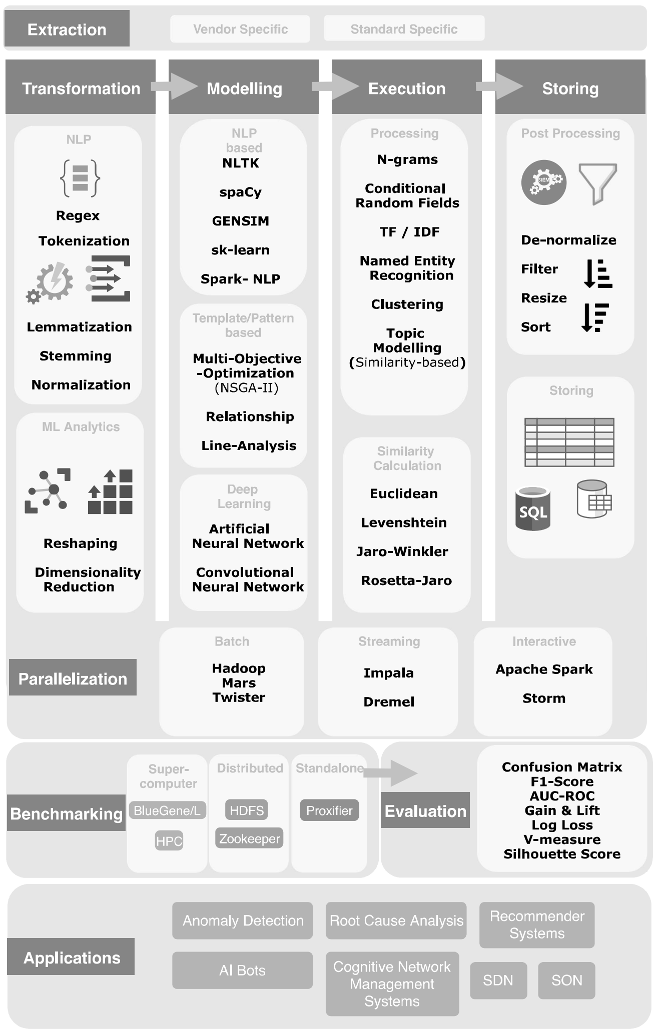

Fig. 3: Components diagram for Automatic Parsing of Unstructured Text

make use of regular expressions to extract patterns in standard-specific data. Data reduction and removing unnecessary data such as file-paths and timestamps is done in this step.

Modelling. Processed data is then analysed through machine learning or deep learning techniques. The literature focuses on using NLP-based techniques for data that has a model, template-based techniques for data that does not have a meta-dictionary available and deep learning-based extraction techniques for sequence [26] and image [9] based analysis of the unstructured data. While these techniques have been used individually. Our survey of the literature found no experiments where two or more techniques have been combined.

Execution. The component describes approaches that can be used with the combination of technique in the previous component. Some researchers use n-grams to count the probability of contiguous sequence of n-words for a collection of messages. TF/IDF (described in §II) is used by some authors with a combination of n-gram variants such as bi-gram and tri-gram which analyse sequence of two and three words in a document. Our review of the literature found that the use of a

Dataset	Logs	Nodes	Events
BlueGene/L (BGL)	4,747,963	6152	376
High Performance Computing (HPC)	433,490	49	105
Hadoop Distributed File System (HDFS)	11,175,629	203	29
Zookeeper	74,380	32	80
Proxifier	10,108	32	8

TABLE I: Benchmarks available for Log Parsing

similarity threshold between different messages to cluster them is prominent. To calculate similarities between messages various algorithms have been used by researchers depending on the data structure which are mentioned in the component. One popular algorithm is Levenshtein distance [30] that calculates the distance between two message sequences by measuring the minimum number of edits required to change one word into another.

Storing. The processed data is stored in a CSV file, SQL database or the distributed file system. This may require some post-processing such as de-normalizing data, filtering data and sorting it according to the requirements. The step transforms the matrix or vector used for analysis into a structured message for loading and storing.

Parallelization. Computer applications generate an enormous amount of data everyday through their logs, system-generated files or other streaming statements. Processing huge volume of data on a framework can be time consuming. Some industrial applications for log parsing have a parallelization component for all the computations in the processing. A popular parallelization component is Apache Hadoop [31] for batch processing of data. For non-batch data, new solutions have been implemented and categorized into streaming data processing and interactive data processing. Some known frameworks for processing streaming data are Dremel by Google, [32] and *Impala*, an open-source implementation of Dremel. Apache Spark [33] is another in-memory computing framework on a distributed storage that supports real-time and interactive data processing.

Benchmarking and Evaluation. To evaluate state-of-the-art parsing methods, the authors of [1] released a collection of 5 log data-sets spanned across distributed systems, super-computers and standalone systems. The benchmark for data parsing contains 16,441,570 log lines. A summary of data-sets available for bench-marking is illustrated in Table I. To study the accuracy of parsing methods on different benchmarks, multiple evaluation metrics have been used by researchers which are mentioned in Fig. 3. A commonly used metric is F-score which is the weighted harmonic measure of precision (number of relevant instances over total number of instances) and recall (number of relevant instances over total number of relevant instances).

IV. Conclusions

In this paper we discussed the state-of-the-art algorithms for information extraction from unstructured text. We divided the algorithms into relationship-based, which used feature-driven approaches, template-based, that used data-driven approaches and deep learning based techniques. Furthermore, we discuss the challenges in parsing and analysis faced by existing techniques. We summarised the process of data parsing and identified common algorithms for each component (see Fig. 3).

Acknowledgments

This work is funded by Irish Research Council Enterprise Partnership Scheme Postgraduate Scholarship 2020 under Project EPSPG/2020/7.

References

[1] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "An evaluation study on log parsing and its use in log mining," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2016, pp. 654–661.

[2] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM, 2009, pp. 117–132.

[3] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *2009 Ninth IEEE International Conference on Data Mining*. IEEE, dec 2009.

[4] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," in *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, oct 2016.

[5] S. Ayoubi, N. Limam, M. A. Salahuddin, N. Shahriar, R. Boutaba, F. Estrada-Solano, and O. M. Caicedo, "Machine learning for cognitive network management," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 158–165, jan 2018.

[6] I. Beschastnikh, Y. Brun, S. Schneider, M. Sloan, and M. D. Ernst, "Leveraging existing instrumentation to automatically infer invariant-constrained models," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*. ACM, 2011, pp. 267–277.

[7] W. Shang, Z. M. Jiang, H. Hemmati, B. Adams, A. E. Hassan, and P. Martin, "Assisting developers of big data analytics applications when deploying on hadoop clouds," in *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 402–411.

[8] W. Paik, S. Yilmazel, E. Brown, M. Poulin, S. Dubon, and C. Amice, "Applying natural language processing (nlp) based metadata extraction to automatically acquire user preferences," in *Proceedings of the 1st international conference on Knowledge capture*. ACM, 2001, pp. 116–122.

[9] S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed, "Deepdesrt: Deep learning for detection and structure recognition of tables in document images," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1162–1167.

[10] C. Suh-Lee, "Mining unstructured log messages for security threat detection," 2016.

[11] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, 2004.

[12] W. Li, "Automatic log analysis using machine learning: awesome automatic log analysis version 2.0," 2013.

[13] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[14] K. Krishna and N. M. Murty, "Genetic k-means algorithm," *IEEE Transactions on Systems Man And Cybernetics-Part B: Cybernetics*, vol. 29, no. 3, pp. 433–439, 1999.

[15] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[16] A. Bafna and J. Wiens, "Automated feature learning: Mining unstructured data for useful abstractions," in *2015 IEEE International Conference on Data Mining*. IEEE, 2015, pp. 703–708.

[17] D. Mimno, W. Li, and A. McCallum, "Mixtures of hierarchical topics with pachinko allocation," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 633–640.

[18] R. Vaarandi, "A data clustering algorithm for mining patterns from event logs," in *Proceedings of the 3rd IEEE Workshop on IP Operations & Management (IPOM 2003)(IEEE Cat. No. 03EX764)*. IEEE, 2003, pp. 119–126.

[19] A. Makanju, A. N. Zincir-Heywood, and E. E. Milios, "A lightweight algorithm for message type extraction in system application logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 11, pp. 1921–1936, 2011.

[20] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *2009 ninth IEEE international conference on data mining*. IEEE, 2009, pp. 149–158.

[21] L. Tang, T. Li, and C.-S. Perng, "Logsig: Generating system events from raw textual logs," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 785–794.

[22] M. Aharon, G. Barash, I. Cohen, and E. Mordechai, "One graph is worth a thousand logs: Uncovering hidden structures in massive system event logs," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2009, pp. 227–243.

[23] S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed, "Deepdesrt: Deep learning for detection and structure recognition of tables in document images," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 01, Nov 2017, pp. 1162–1167.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 91–99.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[26] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1285–1298.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] C. Bertero, M. Roy, C. Sauvanaud, and G. Trédan, "Experience report: Log mining using natural language processing and application to anomaly detection," in *2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2017, pp. 351–360.

[29] T. Korenius, J. Laurikkala, K. Järvelin, and M. Juhola, "Stemming and lemmatization in the clustering of finnish text documents," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 2004, pp. 625–633.

[30] V. I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb 1966.

[31] T. White, *Hadoop: The definitive guide*. " O'Reilly Media, Inc.", 2012.

[32] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis, "Dremel: interactive analysis of web-scale datasets," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 330–339, 2010.

[33] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets." *HotCloud*, vol. 10, no. 10-10, p. 95, 2010.