

Towards Improved Fault Localization using Transfer Learning and Language Modeling

Yusufu Shehu
Moogsoft Ltd
River Reach, 31-35 High Street
Kingston-Upon-Thames, UK
yusufu.shehu@moogsoft.com

Robert Harper
Moogsoft Ltd
River Reach, 31-35 High Street
Kingston-Upon-Thames, UK
rob@moogsoft.com

Abstract—Word embeddings provide efficient vector representations of words that capture the syntactic and semantic relationships in a corpus. Within an IT infrastructure, event data can be interpreted as sequences of tokens and can be represented in a continuous vector space. Similar tokens cluster together in the vector space, which can provide insight into patterns of failures and enable detection of actionable incidents. Fault localization techniques need to be adaptable without the requirement of building knowledge bases from scratch to account for new services or hardware deployed on existing infrastructures, or semantically equivalent incidents described by different lexicons across different infrastructures. Using the paradigm of transfer learning, word embeddings can be built and incrementally updated to introduce new vocabulary and alter the relationships of existing tokens, whilst persisting the general contextual information of the initial embedding. Features in event data procured from IT infrastructures are typically sparsely distributed, with many events being duplicated with minor character mutations. We use unsupervised clustering techniques to analyse the vector representation of the event data. Our analysis shows that clustering vector representations of event data based on semantic similarity produce interpretable categories, which can be used to improve fault localization and identification of root cause.

I. INTRODUCTION

Fault localization [1] can be described as a method or collection of methods that help isolate service-impacting failures and provide the remediation actions required. Techniques such as Root Cause Analysis (RCA) and alert correlation techniques such as Tempus [2] and Cookbook [3] are examples of fault localization. Text-based fault localization methods have two distinct approaches. Firstly, there are rule-based methods that use regular expressions and programmatic constructs such as exact matching and if-then-else statements. Secondly, there are token based similarity matching techniques such as Cookbook. Rule-based methods are brittle and become difficult to maintain with the addition of new rules for different text structures. Data-driven methods like Cookbook provide unsupervised deduction of localized faults and do not rely on extensive rule management. None of these methods implement semantic similarity, and instead rely on lexical comparisons.

To tackle these limitations we implement word embeddings, which are a robust method of persisting semantic relationships.

Word embeddings are fixed-length, dense and distributed representations of words that encode syntactic and semantic information [4] and are widely used in modern Natural Language Processing (NLP) tasks, including information retrieval, semantic analysis, question answering and machine translation. Pretrained word embeddings are effective for general purpose tasks but lack the specificity of unique lexicons from different IT infrastructure. As a result, fault localization performed with the word vectors will lose precision. Through transfer learning, word embeddings can be updated with new vocabulary and text sequences to introduce new contexts and allow more precise vectors to be extracted.

Transfer learning aims to mimic the human ability to acquire knowledge while learning one task and apply this knowledge to solve a related but separate task. This requires training a source model in an unsupervised setting typically on a massive unlabeled dataset then persisting the ‘knowledge’ to a target model which will typically be trained on a smaller and more specific dataset in a supervised learning setting. General features and broader context will be inherited by the target model without having to be trained from scratch on the source data. This facilitates the use of stockpiled general purpose models that can save training time and improve learning for downstream tasks. This approach has led to a recent breakthrough in computer vision (CV). The combined arrival of very large image datasets, e.g. ImageNet [5], deep convolutional neural networks e.g. ResNet [6] and graphical processing units provided a ‘critical mass’ in research leading to pretrained models achieving state-of-the-art results in CV tasks such as object detection, image segmentation, human pose estimation and video recognition.

NLP, which has direct applications to network and operations management problems such as anomaly detection, clustering of network events, and document classification is on the brink of a similar breakthrough. The potential for these advancements has been spurred by language models [7], [8] and deep contextual word embeddings [9]. Combining the use of word embeddings and transfer learning, we propose an incremental fault localization method that can capture new token contexts and find analogues for existing tokens from two or more lexicons derived from client networks. This paper begins in Section II by describing the objectives and

motivation of the paper and to introduce the relevant technical terms used throughout the paper.

Section III outlines the previous work and recent developments in techniques that will be used in Section VI. A patented sorting algorithm used to optimize corpus sentences is described in Section V. The analysis is described in Section VI, with the data used in the analysis described in Section VI-A. The results obtained from the analysis are presented in Section VII. The results are discussed and further enhancements and optimizations are discussed in Section VIII.

II. OBJECTIVES

There are two goals of this paper. Firstly, to demonstrate the usefulness of domain specific word embeddings. Secondly, to demonstrate the efficacy of fine-tuning word embeddings pretrained on generic corpora with a subset of domain-specific tokens. The second goal shows that updating embeddings can be more efficient in terms of total data used and can achieve the same performance from using the entire dataset. This is beneficial for network and operations monitoring tasks where unique data points can be scarce. An incremental approach to word embeddings [10] enables pretrained embeddings to be deployed on a different infrastructure with different diagnostic vocabulary and tune them without requiring millions of new tokens. We provide an assessment method for the quality of these word embeddings, where quality refers to the reliability of the most similar tokens generated from an embedding given particular test tokens. The importance of capturing semantic similarity is of particular importance in network and operations management as the vector representations of tokens can be clustered together in the embedding space, which drives information retrieval and detection of root cause.

A. Definitions and Nomenclature

The following definitions are used throughout this paper:

- Event** An indivisible notification from a network device, application or monitoring system. An event represents a state change and is not necessarily representative of a fault.
- Incident** An actionable support item, such as a service shutdown, that must be resolved.
- Token** A word or phrase identified by standard sentence segmentation techniques.

III. PREVIOUS WORK

Deeplog [11] proposes a semi-supervised anomaly detection solution, which employs a Long Short-Term Memory (LSTM)-based architecture to capture anomalous patterns in log entries. It can be deployed in an online fashion by incrementally adding new log patterns over time. Workflows can be constructed from the system log to perform root cause analysis. LogCluster [12] proposes an unsupervised, cluster-based approach for log-based anomaly detection. It clusters historical and incoming logs using a knowledge base. Only unseen logs are examined to identify incidents, significantly reducing the number of logs required for detection. It relies on document

frequency of tokens to vectorize log sequences. Whilst the previous works described do share the common theme of using machine learning techniques for fault localization, they focus more on log-based anomaly detection. Whereas we describe the enhancement of fault localization using transfer learning and use unsupervised learning techniques to explore this. The main contribution of this work is the ability to use semantic similarity to represent event data and update these representations in an online fashion. The work presented is independent of the label information of the training set and is capable of forming incidents without prior expert knowledge.

IV. UNSUPERVISED LEARNING METHODS EXPLORED

To extract word embeddings from text data, we use the skip-gram flavor of the Word2Vec model [13]. Skip-gram works well with a small amount of the training data and can effectively represent rare tokens and phrases, whereas the Continuous-Bag-of-Words flavor smoothes over rare tokens in favour of more frequent tokens.

We use clustering to interpret embeddings by grouping on proximity of vectors to give a measure of semantic similarity for groups of tokens. The vectors produced from our embeddings have 100 dimensions and are considered as high-dimensional data points. A key challenge common to most clustering algorithms, including the popular k-means algorithm [14], is that they do not scale well to high-dimensional data. This drop in efficiency and accuracy is called the ‘curse of dimensionality’ and is caused by the exponential increase in volume associated with adding extra dimensions to Euclidean space [15]. We use Affinity Propagation [16], which is tolerant to high-dimensional data to produce clusters with extracted vectors from word embeddings.

V. FREQUENCY-BASED SORTING ALGORITHM

In this section we define the problem of feature sparse datasets, and the sorting algorithm design and implementation to condense sparse datasets. The algorithm is novel and of specific benefit to network and infrastructure management, with the end goal of commercial use at a number of enterprise scale networks and is the subject of a patent filing by our employer, Moogsoft. The distributional hypothesis in NLP informs us that semantically similar tokens will be distributionally similar and thus occur in similar linguistic contexts. If the number of contexts is small, the language model will provide vectors with limited semantic scope. In the domain of network and infrastructure management, events can be generated at rates in excess of many thousands per second. Despite the large volume of data ingested, a large proportion of the tokens are duplicates only varying by a unique identifier for example, MySQLTM user IDs *mysql_dbuser_ID:103*, *mysql_dbuser_ID:108*. Masking specific token types such as dates, URLs and IP Addresses can contract millions of events to only a few hundred unique alert templates, for example, masking database access based on dates, *db_accessed:12/04/2010* and *db_accessed:05/06/2003* contract to *db_accessed:ddmmyy*. This can reduce total data

volume by several orders of magnitude, which makes it increasingly difficult to obtain a feature-rich dataset containing a sufficient volume of unique templates. Without using templates we would be training with the same sentences thousands of times, which would skew the results to the most frequent tokens and not provide realistic semantic equivalence. If a sentence is defined as a sequence of tokens and unique tokens are defined as tokens which exist only in a domain-specific corpus, then the aim of the sorting algorithm is to rearrange the ordering of sentences in a corpus such that all of the unique tokens are introduced at least once by a subset of the entire corpus. The remaining part of this section describes the method of optimizing the corpora used to train word embeddings. C_g refers to the corpus that is used for generic purpose fault localization, i.e. contains a broad vocabulary but lacks specific terms that are unique to a specific client's infrastructure. C_d refers to the corpus that contains vocabulary derived from a specific IT infrastructure. The unique vocabulary from the *Domain* corpus, v_d , is derived by subtracting the deduplicated list of tokens from the *General* corpus set from that of the *Domain* corpus set. The steps of the algorithm are divided into two for each stage of sorting and are described in Algorithm 1 and 2. The impact of the sorting algorithm on the domain specific IT corpus is shown in Figure 1

Algorithm 1 Single Sort C_d

Require: $\text{len}(C_g) \geq 0$ and $\text{len}(C_d) \geq 0$ and $\text{len}(v_d) \geq 0$
for *sentence* in C_d **do**
 if token in v_d is in *sentence* **then**
 deduplicate tokens, count and append to $\text{list}[\text{count}_{d_{\text{found}}}]$
 end if
end for
 $C_{d_{1x\text{-sorted}}} = \text{reverse sort on count}(C_d)$
return sorted corpus $C_{d_{1x\text{-sorted}}}$

Algorithm 2 Double Sort C_d

Require: $\text{len}(C_g) \geq 0$ and $\text{len}(C_d) \geq 0$ and $\text{len}(v_d) \geq 0$ and $\text{len}(C_{d_{1x\text{-sorted}}}) \geq 0$
for *sentence* in $C_{d_{1x\text{-sorted}}}$ **do**
 if token in v_d is in *sentence* **then**
 deduplicate tokens, count and append to $\text{list}[\text{count}_{d_{\text{found}}}]$
 $v_d = v_d - \text{tokens found}$
 end if
end for
 $C_{d_{2x\text{-sorted}}} = \text{reverse sort on count}(C_{d_{1x\text{-sorted}}})$
return 2x-sorted corpus $C_{d_{2x\text{-sorted}}}$

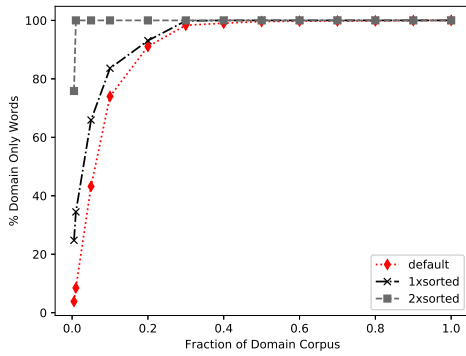


Fig. 1: Unique vocabulary spread for different levels of sorting.

VI. ANALYSIS

In this section we demonstrate the efficacy of incremental training of word embeddings by applying high dimensional clustering to segment and produce specific categories in IT infrastructure data.

A. Dataset Description

This section describes the datasets used to perform the analysis. For each of the raw datasets, a subset of the data is taken (roughly 10,000,000 tokens each) to give sufficient representation of the possible tokens and contexts encountered, whilst limiting CPU usage and allowing for more efficient re-iterations and tuning of the algorithms used in the experiment to be achieved.

- **WikiText-103** [17] The WikiText language modeling dataset (*WT-103*) is a collection of 103 million words extracted from a set of high quality articles on Wikipedia. The dataset has roughly 267,000 unique words. After stop word removal, to meet the 10 million token threshold we took 20% of the raw corpus giving 767,720 sentences and 9,954,433 tokens.
- **Stack Exchange** Stack Exchange (*Stack-Ex*) is a network of question-and-answer websites on topics in diverse fields. The general IT corpus is a culmination of seven topics: Database Administrators, Development Operations, Network Engineering, Security, Server fault, Superuser, and Unix. Question-and-answer data for each of the topics were downloaded from the Stack-Ex archive [18] to form a raw corpus of 100 million tokens. After stop word removal, to meet the 10 million token threshold we took 16% of the raw corpus giving 614,709 sentences and 10,065,277 tokens.
- **Cisco** Cisco is a multinational provider of IT, networking and cybersecurity solutions. They provide documentation for error messages and state change messages for their networking, software and security products called System Error Messages or SEM [19]. Some examples of the categories are: application layer, internet protocol, packet infrastructure, routing, high availability. The total dataset (*Cis*) was obtained by downloading the spreadsheets describing the SEM and has roughly 30 million tokens. After stop word removal, to meet the 10 million token threshold we took 50% of the raw corpus giving 894,013 sentences and 10,096,894 tokens.

B. Training Incremental Word Embeddings

Word embeddings were trained using *Gensim* [20]. Two general purpose embeddings were trained to form the basis for two incremental training configurations. The first embedding is trained on the WT-103 dataset and represents the ‘general English’ language model, in that it contains many examples of natural language and typical English phrases. The second embedding is trained on the combination of WT-103 and Stack-Ex, which represents an enrichment of the ‘general English’ model with ‘general IT infrastructure’ terms and vocabulary. For the next steps of the analysis the combined embedding is incrementally enriched with subsets of the Cisco corpus. The

amounts of the corpus used to enrich the embedding are 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100%, with both sorted and not-sorted configurations of the corpus used.

C. Initial Clustering Using Affinity Propagation

We use AP to select regions of the embedding space that have highly correlated tokens predominantly belonging to the IT corpora, achieving the goal of data space reduction and cluster population modulation. k-means is used to group the correlated tokens in the reduced space for categorization and visualization of token groups. Initializing k-means clustering with affinity propagation has been used with success in hybrid intelligent systems, given affinity propagation’s tolerance of high dimensional data and speed in generating good cluster exemplars [21]. Using Affinity propagation requires the specification of a single hyper-parameter called *preference*. Large preference values result in a large amount of clusters, with the populations of each cluster tending to be smaller. Smaller preference values result in a smaller amount of clusters with higher populations. We choose a preference value to give us of the order of 100 exemplars, given the size of the datasets and the dimensionality of the vectors. This also provides a basis for diversity in WikiText, Stack-Exchange or Cisco dominant clusters. A set of 6,000 test words are obtained which are an equal-parts combination of words unique to either the WikiText, Stack-Ex or Cisco corpora. These words pass the initial rare words filter requirement of occurring at least 20 times in the entire corpus to remove words with insufficient context. The corpus specific dominant clusters are found by finding the largest amount of tokens in the cluster belonging to either corpus. For each token in the cluster, the top ten most similar words are obtained by using the Gensim API’s most similar method. These most similar token sets can be described as ‘token clouds’, which give an estimation of those tokens that are close to the selected token using cosine similarity.

We use principal component analysis (PCA) [22] to reduce the dimensionality of the grouped vectors to three principal components for further analysis and visualization of the k-means produced clusters.

VII. RESULTS

In this section, we demonstrate the evolution of clusters and cluster membership as the word embedding is enriched with increasing amounts of the sorted Cisco corpus, where the clusters only include Cisco tokens. The improvement in cluster quality and stability with increased amount of the sorted Cisco corpus is shown. The cosine similarity threshold used is 0.9. A tight threshold is used to show cluster membership with a smaller collection of tokens, this makes it clearer to observe categories within each cluster. Having a tighter threshold produces a set of tokens that are highly correlated and will produce tighter and more comprehensible clusters. With a looser threshold, the clusters become more dispersive, meaning the compactness and separation deteriorates, in turn making it more difficult to show the improvement in clustering performance metrics. In practice, we would expect a similarity

threshold closer to 0.6-0.7 in order to be more inclusive of alerts that are not near identical but should be grouped together based on the majority of the alert content.

We demonstrate the emergence of Cisco tokens alongside Stack-Ex tokens in Stack-Ex dominant clusters. If Cisco dominant clusters were used this could bias the results and mask the effect of the enrichment. We use a similarity threshold of 0.6 to produce large population clusters, which may be more dispersed, but contain more co-occurrences of Cisco and Stack-Ex tokens of similar context in the same cluster. For clarity, [CIS] refers to a Cisco token and [STK] refers to a Stack-Ex token. The number of clusters used in both sets of results is 8. This was chosen by observing the silhouette scores [23] for a range of k values from 1-10 and 8 was observed to give the highest overall scores.

Table I displays some of the Cisco tokens that are grouped into the 8 clusters. For each cluster some options of potential categories that the clusters represent are given. The word vectors are extracted from an embedding that has been enriched with 100% of the sorted Cisco corpus. Multiple categories have been shown to appear in single cluster, more importantly tokens belonging to the same category appear in the same cluster.

TABLE I: Tokens obtained in k-means derived clusters

Cluster	Tokens	Categories
0	instance-0n[CIS], instance-1n[CIS], instance-3n[CIS], NPU-0[CIS], NPU-1[CIS]	Processors/ID
1	sysdb_item_delete[CIS], sysdb_item_set[CIS], SYSMGR_GET_PLACEMENT_LIST[CIS], sysmgr_mbi_boot_hold[CIS], sysmgr_msg_sendv[CIS], ether_spa_init[CIS], event_send_async[CIS], MFWD-FRETTA[CIS], MFWD-VIKING[CIS], sysdbedm[CIS], eth_server_readv[CIS], event_conn_get_fd[CIS], MFWD-ASR9K[CIS], MFWD-SKYWARP[CIS]	System/Connectivity/Events/Multicast Forwarding
2	CPAK[CIS], QSEFP+[CIS]	Router systems/Network Transceivers
3	bridge-0[CIS], bridge-1[CIS], bridge-2[CIS], bridge-3[CIS]	Network bridging devices
4	FIB-MPLS[CIS], IPV4-EA[CIS], IPV6-EA[CIS]	Forward Information Base, Network layer IP Addresses
5	instance-0[CIS], instance-1[CIS], instance-2[CIS]	Instance ID
6	ASIC-0[CIS], autorp_map_agent[CIS], EOP-1[CIS], FLA00[CIS], FLA01[CIS], FLA10[CIS], FLA11[CIS]	Multicast traffic/Packet Infrastructure
7	ospf_uv[CIS], ospfv3_uv[CIS], pbr_ea[CIS], qosmgr[CIS], SBCSVI[CIS], SESVI[CIS]	Security/Routing Protocol/Quality of Service

The k-means distributions in Figures 2-4 show that distinct clusters form that occupy different regions of the 3-D space for increasing amounts of the Cisco corpus. Each top figure in these figures shows the clusters for the default corpus. Each bottom figure shows the clusters for the sorted corpus. Following the central cluster in each plot (red/orange), we can see that as the amount of corpus used to train the embedding increases, the central cluster increases in size, indicating that more semantically similar terms are being grouped. For the sorted corpus figures - observing the same central cluster - the distribution remains roughly constant after 50% of the corpus is introduced, whereas for the default corpus the distributions become more dispersed. Using the sorted corpus provides greater stability in cluster evolution, with the clusters only slightly changing in positioning after 50% of the Cisco corpus is introduced, whereas with the default corpus the cluster centres continue to fluctuate until 100% of the corpus is introduced. This is further demonstrated by the stability of the silhouette scores in Figure 5 after 50% of the sorted Cisco corpus is introduced compared to the default corpus. The cluster stability is important in fault localization, because

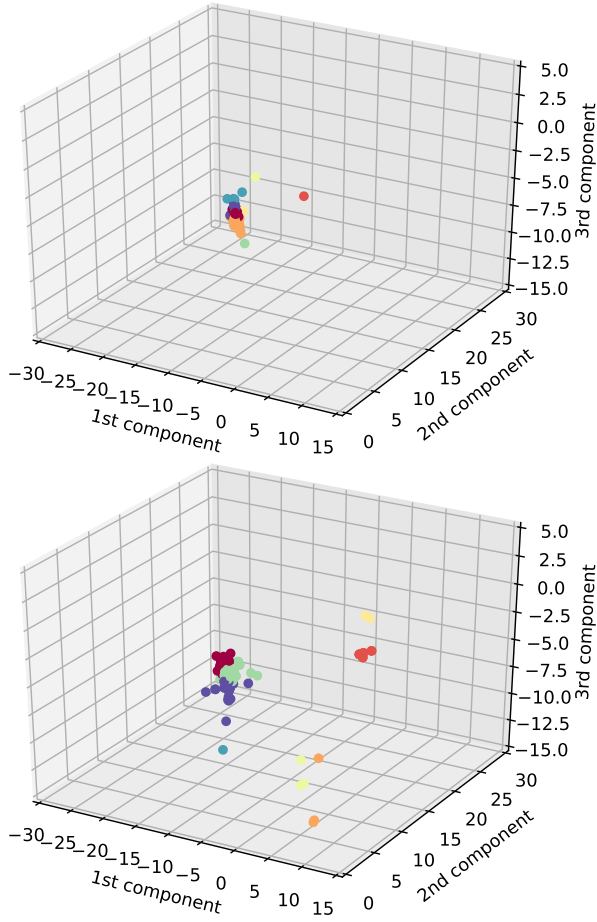


Fig. 2: Clusters from token clouds with 5% cisco corpus.

we want to be able to deduce any incidents and have them persist even while new data is incoming.

We observe the co-occurrence of Cisco tokens with Stack-Ex tokens in Table II. This result is used to affirm that clusters of generic IT tokens can be enriched with domain specific IT token sets. A selection of the Stack-Ex and Cisco tokens that are present in the same cluster are shown in Table II. We observe that, of the Cisco and Stack-Ex tokens to co-occur in the same clusters, they can roughly belong to similar categories. We do not expect complete category matching between Cisco and Stack-Ex, as some of the contexts may be exclusive to either corpus, but tokens that may be related when creating a network related incident should co-occur in the same cluster. Using the Cisco SEM documentation [19], some of the potential overlapping categories are described in the third column of Table II. For example in the row belonging to category ‘Router/Network Layer’ there are Cisco terms relating to HFR, one of Cisco’s routers. These occur in the same cluster as generic IT tokens describing router characteristics, such as buffer size. The co-occurrence of the Cisco and Stack-Ex tokens in the same cluster can be produced by introducing as little as 1% of the sorted Cisco corpus.

These results show that enriching an existing embedding can lead to an increased ability for tokens related to network incidents to co-occur. Using a domain-specific corpus, with a

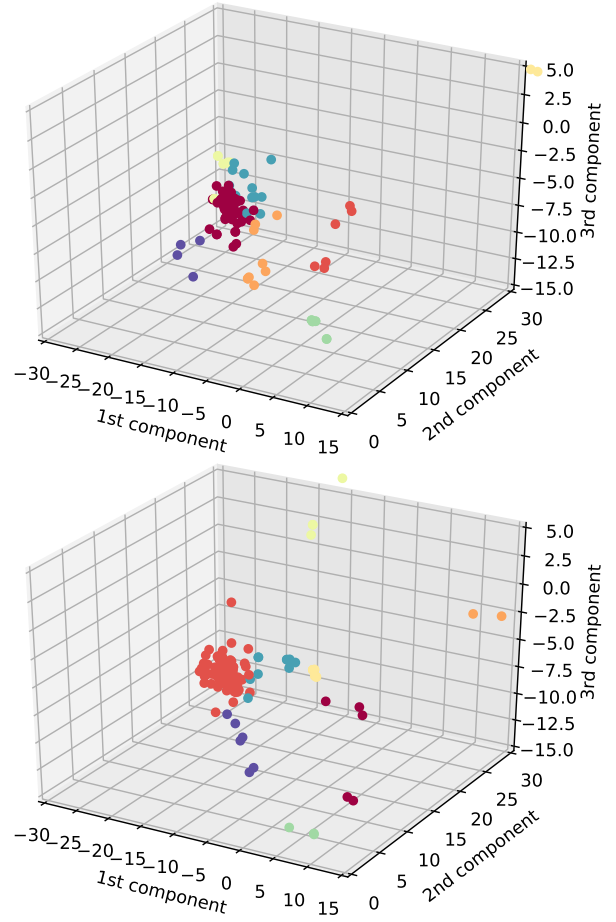


Fig. 3: Clusters from token clouds with 50% cisco corpus.

TABLE II: Overlapping Tokens in Cluster using 10% of Sorted Corpus

Cisco Tokens	Stack-Ex Tokens	Categories
autoboot[CIS], BOOTING[CIS], Membooted[CIS], membooting[CIS]	Bootloader[STK], Bootloader[STK], bootrec[STK], fixboot[STK]	Services
imdr_debug_init[CIS], imdr_trace_init[CIS]	IRQL_NOT_LESS_OR_EQUAL[STK]	Driver errors
mysmgr_msg_sendv[CIS], Sysmgr_proc_ready[CIS]	HKCU[STK], HKCUControl[STK]	System management, processes
hfr_lcpm_show[CIS], hfr_pm_qos_gran_show[CIS], hfr_pm_test[CIS]	buffer-size[STK], buflist_findname[STK], buflist_new[STK], ip-forwarding[STK]	Router, Network layer
ip4_rib_batch_buffer_TYPE[CIS], ipv6rpl[CIS]		
Err_str[CIS], error-1[CIS], error-2[CIS], error-3[CIS], cleanup_dead_node[CIS]	SevereCritical[STK], Cleanup[STK]	Error Messaging

unique vocabulary can lead to the co-occurrence of existing generic tokens with the related domain-specific tokens. This can facilitate the evolution of a fault localization application to detect and persist new incidents given new vocabulary or semantically similar but lexically different tokens.

VIII. CONCLUSION

In this paper we propose a new contribution to fault localization with IT infrastructure event data by expressing semantic relationships in data with word embeddings and clustering the extracted vectors by semantic similarity. This differentiates from the approach in LogCluster, which encodes event data based on frequency. We use transfer learning to incrementally enrich a generic word embedding with a subset of domain-specific sequences of tokens, providing a method for online fault localization. Through transfer learning, an embedding belonging to one client can be updated using the

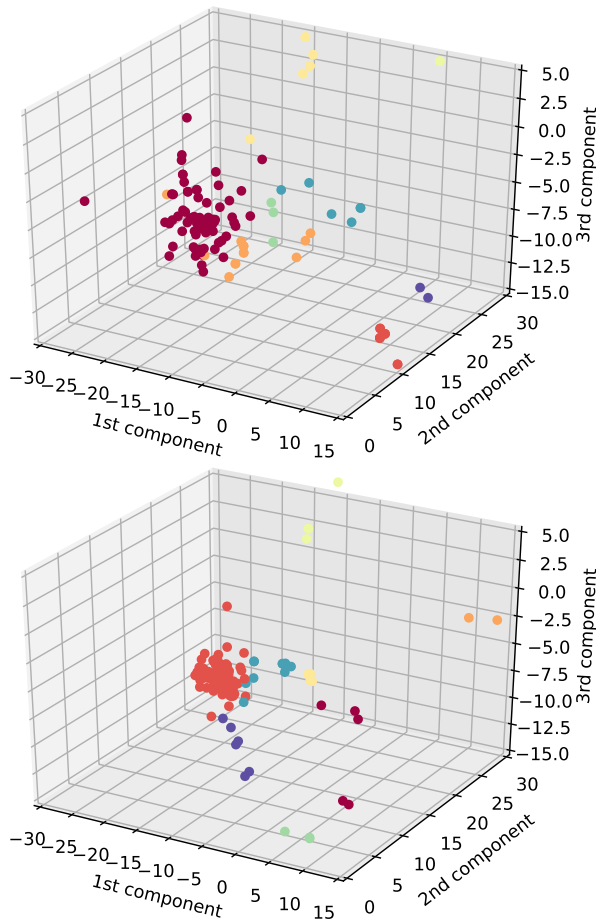


Fig. 4: Clusters from token clouds with 100% cisco corpus.

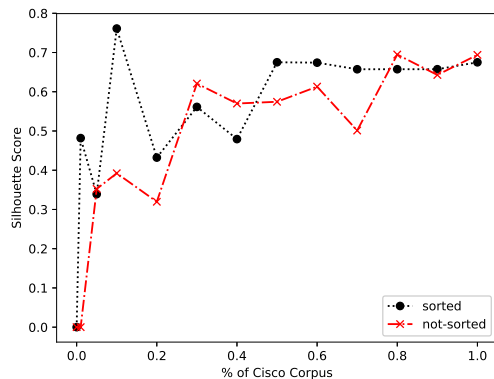


Fig. 5: Silhouette scores for clusters formed on token clouds.

bespoke network vocabulary of a different client for detecting actionable incidents on the different network. The method is entirely unsupervised as opposed to DeepLog, which requires some pre-labeling of event data. The efficacy of this approach is demonstrated using a combination of affinity propagation, PCA and k-means clustering to group the word vectors at different stages of the enrichment. The domain-specific words form distinct IT infrastructure categories, with either multiple categories in one cluster or a cluster having membership entirely from one category. The cluster membership becomes more defined and stable as more domain specific words are

introduced into the word embedding. Using the visualization of the clusters and the silhouette score distribution, we observe that the clusters formed with sorting are more compact, less of the corpus is required to form initial clusters (1% compared to 5%) and with 50% of the sorted domain-specific corpus introduced, we find that the clusters become stable and do not mutate with further enrichment of the embedding. The analysis presents an opportunity for further research to provide embeddings for previously unseen tokens, e.g. through the implementation of character n-grams. The word embeddings produced in this analysis can be used as an embedding layer of a deep neural network to be used for event and incident classification tasks and predictive language modeling.

REFERENCES

- [1] A. S. Malgorzata Steinder, "A survey of fault localization techniques in computer networks," *Sci. Comput. Program.*, 2004.
- [2] R. Harper and P. Tee, "A Method for Temporal Event Correlation," *IFIP/IEEE Symposium on Integrated Netw. Service Manag.*, 2019.
- [3] —, "Cookbook, a recipe for fault localization," *IEEE/IFIP Network Operations and Management Symposium*, 2018.
- [4] Y. Chen *et al.*, "The Expressive Power of Word Embeddings," *Deep Learning for Audio, Speech and Lang. Proc. Workshop*.
- [5] J. Deng *et al.*, "ImageNet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.
- [6] K. He *et al.*, "Deep Residual Learning for Image Recognition," *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [7] J. Howard *et al.*, "Universal Language Model Fine-tuning for Text Classification," *Proc. of the 56th Ann. Meeting of the Assoc. for Comp. Ling.*, 2018.
- [8] J. Devlin *et al.*, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proc. of the 2019 Conf. for Comp. Linguistics: Human Language Technologies*, 2019.
- [9] M. E. Peters *et al.*, "Deep contextualized word representations," *Proc. of the 2018 Conf. of the North American Chapter of the Assoc. for Comp. Ling.: Human Language Technologies*, 2018.
- [10] A. Kabbach *et al.*, "Towards Incremental Learning of Word Embeddings Using Context Informativeness," in *Proc. of the 57th Ann. Meet. of the Assoc. for Comp. Linguistics*, 2019.
- [11] D. Min *et al.*, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," *Proc. of the 2017 ACM SIGSAC Conf. on Comp. and Comms. Security*, 2017.
- [12] Q. Lin *et al.*, "Log clustering based problem identification for online service systems," *Seventh IEEE Int. Conf. on Data Mining*, 2007.
- [13] T. Mikolov *et al.*, "Efficient Estimation of Word Representations in Vector Space," *Int. Conf. on Learning Representations*, 2013.
- [14] S. P. Lloyd, "Least squares quantization in pcm," *Technical Report RR-5497, Bell Lab*, 1957.
- [15] R. E. Bellman, *Dynamic programming*. Princeton Univ. Press, 1957.
- [16] B. J. Frey *et al.*, "Clustering by passing messages between data points," *Science* 315.
- [17] S. Merity *et al.*, "Pointer Sentinel Mixture Models," *Int. Conf. on Learning Representations*, 2016.
- [18] "Stack Exchange XML Data Resource," 2019. [Online]. Available: <https://ia600107.us.archive.org/27/items/stackexchange/>
- [19] Cisco, "Cisco IOS XR System Error Messages Reference Guide." [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios_xr_sw/error/message/ios-xr-sem-guide.html
- [20] R. Rehrrek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proc. of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 2010.
- [21] C. J. Y. Zhu, J. Yu, "Initializing k-means clustering using affinity propagation," *Conf.: Hybrid Intelligent Systems*, 2009.
- [22] H. Hotelling, "Analysis of a complex statistical variables into principal components," *J. Educ. Psychol.*, 1933.
- [23] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comp. and Applied Math.*, 1987.