Received 16 November 2022; accepted 1 December 2022. Date of publication 30 December 2022; date of current version 24 January 2023.
The review of this article was arranged by Associate Editor Peng Wang.

Digital Object Identifier 10.1109/OJIM.2022.3232650

An Online Parsing Framework for Semistructured Streaming System Logs of Internet of Things Systems

SUSNATA BHATTACHARYA, BIPLOB RAY (Member, IEEE), RITESH CHUGH (Senior Member, IEEE), AND STEVEN GORDON (Member, IEEE)

School of Engineering and Technology, Central Queensland University, Rockhampton, QLD 4700, Australia

CORRESPONDING AUTHOR: S. BHATTACHARYA (e-mail: s.bhattacharya@cqumail.com)

This work was supported by the School of Engineering and Technology, Central Queensland University, Rockhampton, QLD, Australia.

ABSTRACT This article presents a novel log abstraction framework based on neural open information extraction (OpenIE) and dynamic word embedding principles. Though various log parsing frameworks are proposed in the literature, the existing frameworks are modeled on predefined heuristics or auto-regressive methodologies that work well in offline scenarios. However, these frameworks are less suitable for dynamic self-adaptive systems, such as the Internet of Things (IoT), where the log outputs have diverse contextual variations and disparate time irregularities. Therefore, it is essential to move away from these traditional approaches and develop a systematic model that can effectively analyze log outputs in real-time and increase the system up-time of IoT networks so that they are almost always available. To address these needs, the proposed framework used OpenIE along with term frequency/inverse document frequency (TF/IDF) vectorization for constructing a set of relational triples (aka triple-sets). Additionally, a dynamic pretrained encoder–decoder architecture is utilized to imbibe the positional and contextualized information in its resultant outputs. The adopted methodology has enabled the proposed framework to extract richer word representations with dynamic contextualization of time-sensitive event logs to enhance further downstream activities, such as failure prediction and prognostic analysis of IoT networks. The proposed framework is evaluated on the system event log traces accumulated from a long range wide-area network (LoRaWAN) IoT gateway to proactively determine the probable causes of its various failure scenarios. Additionally, the study also provided a comparative analysis of its mathematical representations with that of the current state-of-the-art (SOTA) approaches to project the advantages and benefits of the proposed model, particularly from its data analytics standpoint.

INDEX TERMS BERT, context-aware, gateway, information extraction, Internet of Things (IoT), log parsing, natural language, streaming, transformer, triples, word-embedding.

I. INTRODUCTION

THE INTERNET of Things (IoT) devices generate a large amount of logs to record system operational activities and the execution status of the running processes that act as a set of critical information for maintaining any IoT network. These log outputs of IoT devices are a set of free-flowing semistructured textual data that are primarily designed to be human interpretable and are predominantly written in natural language texts. Typically, the output of these log messages provides situational knowledge about the operating conditions of an IoT system or a device at any given time. Hence, it is crucial to analyze these IoT logs in a live-streaming manner by performing proactive systems management and increasing the stability of the network devices, thereby improving the overall health of such complex IoT networks [27].

Automated log abstraction models can proactively analyze large amounts of system-generated IoT logs quickly and efficiently. These models can be used in conjunction with other machine learning models to perform various downstream tasks and knowledge generation activities, often required for effective decision-making purposes [27].

There is a considerable amount of research that has been conducted to analyze log data, especially in the areas of intrusion detection, forensics, anomaly detection, and fault diagnosis of IoT and networked information systems [6], [10], [41]. However, most of these studies were primarily based on predefined rule-based engines that utilized complex handcrafted heuristics' and pattern matching signatures for developing workflows, templates, and baselines to detect nonconformity from abnormal pattern deviations [31]. Moreover, these approaches resulted in system rigidity and the consequent development of these models failed to scale as they often required additional human expertise due to their domain-specific structures [7]. On the contrary, there is less research in the areas of event log parsing using information extraction (IE) and natural language processing (NLP) models that has a significant potential to detect system faults and intrusions proactively [31], [41].

Traditionally, a log parsing workflow starts with converting a semistructured text data into structured, machine-readable formats that facilitate querying, summarizing, and aggregating logs for online monitoring and improved intelligibility. This is followed by partitioning and grouping of log messages through tokenisation, parts-of-speech (PoS) tagging, and vectorization approaches for forming an event matrix [41]. The final segment of a parsing model involves log clustering to generate event patterns for performing text-mining tasks using deep learning approaches.

Literature suggests that most log mining studies have adopted offline batch processing approaches that are generally unsuitable in real-time anomaly detection and failure prediction scenarios [4], [47]. Moreover, the traditional machine learning models used in such studies are inefficient in handling frequently generated streaming log datasets ranging in thousands of megabytes resulting in a large memory footprint and consequent consumption of high computational resources. Therefore, these existing log mining models are unsuitable in predictive analytics scenarios for IoT systems [27].

Thus, there is considerable scope for devising new situation and context-aware log mining models that can capture streaming event log traces from these complex IoT networks over a prolonged time, and perform deep mining analysis to predict imminent failures proactively, rather than adopting a reactive approach [3]. In such scenarios, the integration of IE and NLP can enable an automated log parsing model to extract valuable insights with high reliability, thereby reducing human interventions to almost close to zero.

These IE systems work similarly to that of an anomaly detection model wherein, in the former, query-based searches are retrieved and presented based on their relative ranking within the corpus. Furthermore, by utilizing named entity recognition (NER) and NLP techniques, context-based information extraction and common subject-verb-object (SVO) relationships can be established that closely conform with entity extraction, classification, and tagging techniques of NLP-based log analytics [47]. Hence, adopting IE and NLP techniques in log parsing approaches provides benefits toward improving the overall accuracy of such models and significantly minimizing the occurrences of false positives.

However, the current performance benchmark of such parsing and predictive analytical models is quite limited [4]. This proposed approach intends to fill this gap by developing an agnostic model that can be beneficial in both Instrumentation and Measurement (I&M) industries from the performance perspectives of IoT networks and as well as in the academic fields of AI-based log mining and analytics. With its data-driven approach the proposed model can further automate the I&M instruments to accurately identify, track, detect, and perform critical analysis of system health with almost no human intervention [63]. Moreover, log abstraction being an integral part of such analytical models can bring a significant impact on its performance in terms of its fault detection and prediction capabilities.

The main contributions of this study are summarized herein.

1) The study proposed a streaming IoT log parsing framework that can preprocess and parse semistructured IoT gateway system logs into structured and context-aware information.
2) A hybrid mathematical model to derive and present the theoretical underpinning of open information extraction (OpenIE) in conjunction with encoder–decoder architecture.
3) The proposed model is validated using live IoT gateway log data.
4) A comparative analysis is performed to evaluate the effectiveness of the proposed model in comparison with the present state-of-the-art (SOTA) approaches from an IoT analytics perspective.

The remainder of this article is organized as follows. Section II discusses related work, followed by the proposed OpenIE framework in Section III. Section IV details the system model, the dataset collection and the experimental designs developed for performing this research. This is followed by Section V that details the experimental analysis and the numerical validation of the experimental outcome. Section VI provides a comparative study and discusses the overall perspective of performing this research. Finally, Section VII provides the concluding remarks and outlines the possible future research directions.

II. RELATED WORK

There are various approaches of log preprocessing, parsing, and summarizations that have been proposed in the literature for system anomaly detection and failure prediction scenarios of network and IoT gateways. Additionally, novel methods have also been devised using various machine learning and deep learning approaches to enhance log interpretability for further downstream troubleshooting tasks. However, there is considerably less research on leveraging the structure of log parsing approaches using OpenIE models.

This section provides a review of the existing literature and highlights their respective challenges by grouping them under multiple subtopics. In Section II-A, a thorough review of the basic principles of heuristics-based approaches pertaining to the OpenIE framework and their associated shortcomings are projected. Section II-B further expanded the use of heuristics in log analytics that were performed in the existing literature and highlighted the challenges that these models witnessed. Section II-C, discussed the benefits of moving away from heuristics-based models toward more of an autoregressive architecture using attention-based NLP models. Section II-D, highlighted the challenges of traditional long short-term memory (LSTM)-based models, and projected the benefits of adopting encoder–decoder architectures using advanced attention mechanisms that can further streamline such hybrid neural OpenIE models manifold.

A. HEURISTICS-BASED LEARNING USING OPENIE

Conventionally, OpenIE models are used to extract knowledge from large text corpora to obtain semantically structured information [12] in the form of prepositional triples or *n*-ary representations. The generic OpenIE framework led researchers to devise models that do not have a dependency on predefined ontological schema, making them perfectly suitable for downstream NLP tasks, such as knowledge base development, comprehensive question answering models, text summarizations, and so on [12].

Upadhyay and Fujii [45] proposed an approach to leverage the knowledge extraction system to effectively extract significant keywords and key sentences from large sets of document databases. The model utilized resource description framework (RDF) graph for deriving words and line triple-stores that resembled the OpenIE framework. Similarly, Glauber and Claro [20] adopted a systematic mapping methodology to extract relational tuples from past literature texts using the OpenIE framework for determining the present SOTA and the recent developments of OpenIE. The researchers used relative ranking of the derived triple-sets based on their meaningfulness to identify and extract the relevant implicit knowledge hidden within the textual documents. Likewise, the study performed by Choudhury et al. [11] proposed an innovative model of a dynamic knowledge graph based on the OpenIE framework for streaming data mining, knowledge extraction, discovery, and visualization of hidden data patterns.

Each of these models needed significant human intervention, and intelligence to extract relevant information, which often creates a risk of arbitrary elimination of pertinent knowledge that might impact the model's efficacy. Moreover, such models are domain-specific and have a significant limitation in adopting them in heterogeneous datasets. Ali et al. [2] performed a systematic literature review of the advent of the OpenIE paradigm to alleviate these OpenIE issues and overcome the challenges that most of these NLP models witness. The study projected that shallow data-based OpenIE models could be improvised using handcrafted rule-based heuristics and dependency parsing techniques. Through these techniques, rule-based relational tuples can be parsed from textual documents for effective knowledge mining and analysis [2].

B. HEURISTICS-BASED LOG PARSING

In recent years, parsing of unstructured texts and logs remained a key focus both in industries and as well as within the academic domains, which resulted in the development of a considerable amount of log parsing models [23].

In line with these techniques, various log parsing approaches are proposed earlier in the literature, and unsupervised dependency log parsers have proved to be successful in such instances since syntactic knowledge can be easily embedded within the models for achieving better probabilities [36].

He et al. [23] performed a critical analysis of the various log parsing automation models and highlighted the challenges that are present in these traditional approaches that utilize handcrafted grok patterns to extract key parameters and standard templates. The study further emphasized that voluminous log datasets can be better handled by moving away from traditional heuristics and adopting a data-driven parsing approach that can generate event log patterns automatically. The study performed by Wang et al. [51] proposed a similar log anomaly detection model based on a variant of the Word2Vec algorithm known as "LogEvent2Vec" through a sequence of steps—Log Parsing, Feature Extraction, Log Clustering, and Anomaly Detection. In line with these approaches, various models have been developed, such as LogMine [22], Drain [24], POP [23], and Spell [14]. Such models drastically reduced the processing time and error generation, making them superior to their predecessor models [23], [52].

Another study by Meng et al. [32] proposed an automated online streaming log model known as LogSummary, which is based on unsupervised algorithmic approaches and can be considered an end-to-end log summarization model that is far superior to that of the current heuristics' approaches. The study claimed that an efficient log parsing model should primarily consider three vital information—1) "entities"; 2) "events"; and 3) their "relationships." The concept is stemmed from the idea of OpenIE to generate summarized triple-sets of all the vital log messages in a given log sequence [32]. The study avoided the ranking of log summaries that are traditional in OpenIE models and instead used "Log2Vec" to learn log semantics and train the word embedding representations. This helped the model easily learn domain-specific words, and the model could handle out-of-vocabulary (OOV) words much faster, even during runtime.

C. LOG PARSING USING AUTOREGRESSIVE MODELS

Traditionally, the approaches that are based on NER methods use advanced algorithmic techniques, such as Naïve Bayes, Stochastic Optimization, Passive-Aggressive Classifiers [41] Skip-Grams model, ShallowCNN, and so forth [48].

The study by Studiawan et al. [41] proposed an event log parsing solution "NERLogparser" built upon deep learning approaches using NLP and NER techniques. The model performed both character and word level embedding that is concatenated using the forward and backward LSTM states to improve its accuracy and predictability. Furthermore, the adopted methodology is context-aware and can learn feature representations bi-directionally using these forward and backward LSTM approaches. The model also moved away from the traditional regular expressions or parsing rules for extracting log patterns and adopted a deep learning approach that can dynamically parse event flow using NLP modeling [41].

In recent years, similar recurrent neural networks (RNNs) and deep learning approaches have also shown immense promise in sensor data analysis, by moving away from the traditional machine learning approaches. This significantly increased their capabilities of extrapolating high dimensional nonlinear data structures more efficiently using complex dimensionality reduction and feature selection techniques [5]. Moreover, with the usage of deep learning approaches, the accuracy and prognosis of such models have also dramatically increased [33]. However, this demanded considerable changes in the way by which such time-sensitive, latent, and noisy signal datasets are handled from the perspectives of their data preprocessing, fusion, dependency parsing, and summarization approaches [27]. Likewise, the study performed by Zhang et al. [52] proposed a novel variant of LSTM titled the "attention-based-time-aware LSTM" (ATTAIN) network, which proved to be successful in effective parsing of time-series IoT sensor data and was primarily used for forecasting and prediction in a typical healthcare scenario. The proposed model was also capable of handling events generated from patient sensor networks at irregular time intervals, especially during critical prefailure situations when the events are likely recorded at a rapid frequency [52].

However, LSTM-based networks have their own shortcomings. Most of the studies on LSTM-based prediction networks are single-layered, which restricts their capacity to handle multiple input features. The study conducted by Gao et al. [18] proposed a deep multilayered bidirectional LSTM network that used both forward and backward states. This helped in determining the accurate weights of both closer and farther input features, consequently increasing the precision and accuracy of such models.

Aussel et al. [4] proposed a novel log parsing and mining technique using natural language understandings. The study utilized the deep learning techniques of data preprocessing, stopwords removal, tokenising, and stemming of a large corpus to reduce the vocabulary size and extract latent semantic information for data mining. It also adopted vectorization approaches to convert words and sentences into vector representations using continuous-bag-of-words (CBOWs) and n-gram models that further reduced the word ordering ambiguities and high memory usage. However, there are advanced neural networking techniques that help reduce a model's memory footprints, which are superior to the hash-tagging approach used in this study that often introduces hash collisions and consequent reduction of model interpretability.

Meng et al. [31] proposed a novel log parsing framework called "LogParse" that can provide intraservice and cross-service learning and adaptiveness through dynamic updating of the template rule-sets without the need for manual intervention or through processes from large historical log datasets. The model consists of offline and online components. During the offline stage, the model extracts templates from historical log datasets and categorizes the words into a template and variable words using a binary word classifier [31]. During the online stage, the model matches the generated words with the word templates, and if new or unknown words are found, it converts them into vector representations for distinguishing them further into a template and variable word and append the existing templates to generate new template sets. However, Wang et al. [49] have not provided adequate inference on how they dealt with the issues related to context and time sensitivities in situations where time irregularities are a concern, especially where log data is generated at high frequencies and at irregular time intervals.

D. OPENIE FRAMEWORKS USING ENCODER AND DECODER STRUCTURES

Literature suggests that attention-based models have better capabilities in handling irregular time sensitivities and can often outperform the traditional neural network-based transduction models [46]. They primarily function using the processes of sequential recurrences and convolutional approaches.

In line with this context, the study on encoder–decoder-based Transformer [46] architectures is comprehensive, since it utilized an assembly of self-attention, encoding, and decoding layers along with their positional embeddings that helped the model to mimic sequence-to-sequence operations, yet in a highly parallelable manner. The study performed by Vaswani et al. [46] has been considered a revolutionary idea that helped in alleviating most of the challenges that the traditional LSTM-based network witnessed. The model resolved the issues of handling longer sequence lengths by reducing the number of back-propagations using the positional information of the input sequence to enhance the process of learning long-term dependencies [46].

On the other hand, the fundamental adoption of heuristics' in an OpenIE framework makes them face challenges in syntactic parsing primarily due to the problems of error propagation. To overcome this, the model developed by Cui et al. [12] proposed a novel neural OpenIE approach by integrating an encoder–decoder architecture. The adopted approach was capable of dynamically learning highly confident binary extractions and relational triple-sets that are bootstrapped from a prefixed OpenIE system without the use of predefined rule-sets and handcrafted patterns. Furthermore, the study has experimentally shown that a neural OpenIE system can outperform several traditional benchmarks with higher efficiency, accuracy, and effectiveness [12].

The developments in the areas of encoder–decoder architectures gave rise to the pretrained deep bidirectional transformer architectures known as bidirectional encoder representations from transformers (also known as BERT) [13]. This architecture enabled higher levels of feature representations from unstructured nonlabeled texts by efficiently embedding contextual information in all layers of its architecture. A novel framework has been proposed by Wang et al. [49] that can significantly improve NLP performance and downstream tasks using a traditional static word embedding-based skip-gram model combined with advanced neural language architecture such as BERT. The proposed model leveraged the benefits of static word embeddings of being lightweight both in terms of its computational resource utilization and as well as its size in comparison to other contextualized word embedding models. By combining static word embedding and enhancing it with advanced contextualized word representations, the study successfully enhanced the feature representation capabilities of static word vectors that eventually lowered the cost and high usage of computational resources [49]. One of the limitations of Wang et al.'s [49] study is that it has primarily concentrated on contextualized word embedding but did not focus on the positional embedding of words which provides much richer information from a perspective of neural language modeling. This information is primarily beneficial for downstream tasks, such as log parsing and failure prediction scenarios where sequential time series data are fed to the model parallelly for achieving positional information besides their contextual representations.

III. PROPOSED OPEN INFORMATION EXTRACTION FRAMEWORK

This section provides a concise description of the proposed OpenIE framework along with a detailed explanation of the various components/subcomponents that are involved as a part of its underlying architecture. To perform log abstractions effectively, the following information extraction framework is proposed where the central architecture is based on neural OpenIE, and encoder–decoder approaches.

This proposed parsing framework is illustrated in Fig. 1. The framework has two analytical stages—1) the offline and 2) the online stages for log parsing. It is further broken down into two virtual layers/sections—1) the gateway layer and 2) the event stream processing layer. The gateway layer is the physical hardware layer and works in conjunction with the stream processing layer for log trace generation and accumulation. Additionally, this layer is physically connected to a computer system for log accumulation and storage using an MQTT bridge. The experimental setup for raw log accumulation is further explained in detail in Section IV. This accumulation process helps in the ingestion of the system logs during the offline stage for training, validation, and testing. Additionally, these offline historical logs are stored in a Syslog repository which is fed sequentially to the subsequent sections of the model for further log preprocessing. A series of steps are followed during this prestaging process that includes tokenisation, lemmatization, labeling, stopwords and lowercase removal, padding, truncation, and masking. Furthermore, the event stream processing layer also performs feature extraction, pattern matching, and feature representations from the input data stream source. The formed vocabulary dataset is fed to various deep learning models for detecting faults/anomalies in the log snippets. Moreover, this layer represents a unified information extraction framework for parsing and analysing gateway streaming logs using OpenIE, NLP, and context-aware approaches which is further explained through mathematical experiments in Sections IV-C and IV-D.

The fundamental process of detecting anomalies is finding the outliers in a traditional manner using statistic-based Gaussian cloud point; therefore, these are not generated using the same probabilistic distribution sets in a 2-D plane [37]. The NLP models come in handy to deal with multidimensional datasets, and furthermore, such models are suitable in solving common log parsing problems, such as class imbalance, NER issues, detection performances, and so forth [41].

The proposed framework intends to represent the words in the vocabulary set in a distributed manner through a domain agnostic way by embedding its semantic as well as its contextual information. The vocabulary dataset contains tri-gram word models which are inherited from the SVO relationship of a typical OpenIE methodology [32].

Most of [45] and [58] have benchmarked three adjacent words as a threshold since exceeding more than three dimensions often brings additional challenges due to large log volumes, and computational models become more complex to handle. However, such tri-gram word embedding frameworks are suitable for static word embedding, but we extend the proposed framework by incorporating components of dynamically contextualized word embedding models. This is designed to improve future downstream tasks (such as predictive analytics) with better representation capabilities by incorporating syntactic and semantic information using sequentially contextualized embedding methodologies.

Past literature suggests highly parallelable frameworks of neural language modeling, which have proved to be more successful in comparison to their autoregressive sequential predecessors [46], [49]. To that extent, the encoder–decoder architectures are far more efficient due to their lightweight structures. Such models can run relatively faster with reasonably lower computational costs primarily due to their pretrained architecture, that is highly customizable through

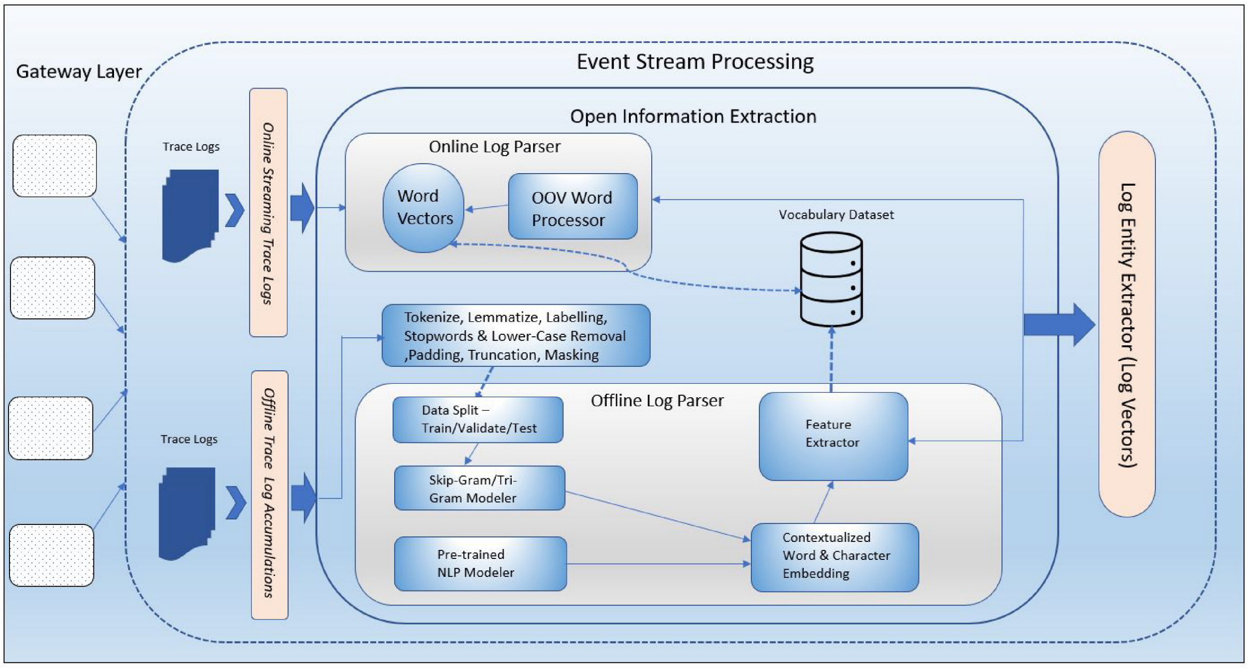
Gateway Layer
Event Stream Processing
Open Information Extraction
Online Log Parser
Word Vectors
OOV Word Processor
Vocabulary Dataset
Trace Logs
Online Streaming Trace Logs
Tokenize, Lemmatize, Labelling, Stopwords & Lower-Case Removal ,Padding, Truncation, Masking
Offline Log Parser
Data Split – Train/Validate/Test
Skip-Gram/Tri-Gram Modeler
Pre-trained NLP Modeler
Feature Extractor
Contextualized Word & Character Embedding
Trace Logs
Offline Trace Log Accumulations
Log Entity Extractor (Log Vectors)

FIGURE 1. Neural OpenIE-based log parsing framework.

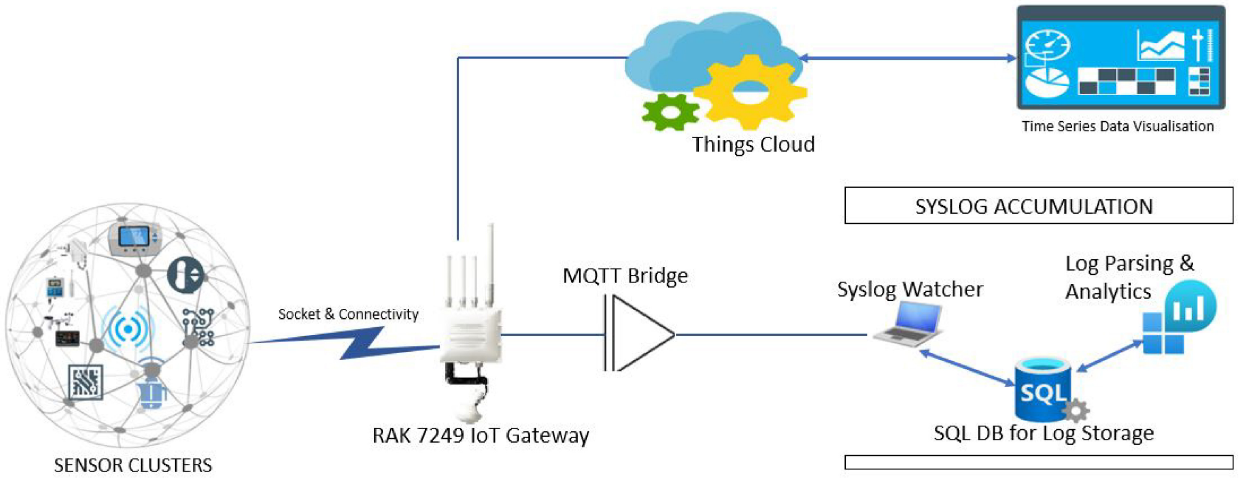

FIGURE 2. Experimental setup for log data collection and analysis.

proper fine-tuning and the adjustments of their weight parameters [13], [46].

Furthermore, this proposed model used a word-piece tokeniser [13] that gives granular contextualized representations of each word token resulting in better handling of domain-specific words, that are usually unseen. This further enhanced the predictive possibilities of handling OOV word structures by the proposed model. In addition to word-piece tokenisation, the model also performs positional contextualization and segment embeddings to better represent models even with a base model size [46].

IV. SYSTEM MODEL, DATASET DETAILS, AND EXPERIMENTAL DESIGNS

To conduct this research, accumulate raw logs, and understand its applicability in sensor data analysis, a prototypical experimental model is developed at the Centre for Intelligent Systems, CQUniversity as detailed in Section IV-A. Section IV-B provides the log dataset details which are accumulated using the experimental model explained earlier in Section IV-A. This is followed by two experimental designs in sections (Sections IV-C and IV-D) for offline log abstraction and OOV word handling during online stage, respectively.

TABLE 1. Raw system log extracts.

```
RAW LOG CHUNKS:

lora_pkt_fwd[13073]: PULL_RESP(onse) datagrams received
lora_pkt_fwd[13073]: RF packets forwarded
lora_pkt_fwd[13073]: PUSH_DATA datagrams sent
lora_pkt_fwd[13073]: PUSH_DATA acknowledged
lora_pkt_fwd[13073]: RF packets sent to concentrator
lora_pkt_fwd[13073]: TX errors
lora_pkt_fwd[13073]: BEACON queued
lora_pkt_fwd[13073]: BEACON sent so farc
lora_pkt_fwd[13073]: BEACON rejected
lora_pkt_fwd[13073]: Valid time reference
kern.info quectel-CM[1360]:requestRegistrationState2
kern.info quectel-CM[1360]:requestRegistrationState2
kern.info quectel-CM[1360]:requestRegistrationState2
kern.info quectel-CM[1360]:requestRegistrationState2
```

A. SYSTEM MODEL

The model prototype illustrated in Fig. 2 consists of a RAK7249 long-range wide-area network (LoRaWAN) gateway [57] with several soil moisture, and weather sensors connected to it. The gateway runs a customized version of Linux Operating System, known as OPEN wireless router (OpenWRT), which is specifically designed to run on embedded devices. The gateway is also connected to the cloud server via Things network middleware [56], where data is stored in Influx Database (InfluxDB) [9]. In this IoT model, sensors' data are accumulated periodically to forecast moisture levels for further visualization and analysis [9]. One of the shortcomings of this design was the absence of a proper log accumulation mechanism through which IoT gateway system logs can be accumulated over a prolonged period to determine imminent failures.

As the gateway stores the system health logs in its volatile memory, the log data gets lost once the gateway is reset/rebooted. This poses a hindrance to logging accumulation, and often failure logs remain undetected. A networked computer is connected to the gateway to collect the system logs using MQTT protocol. The monitoring system runs on a workstation-class system with Intel Core i5-8265U CPU, 1.6 GHz, 4 Core, eight logical processors, and physical memory of 16 GiB. Syslog Watcher 5.1.0 [55] is installed on the workstation for log accumulation and visualization. The Syslog Watcher is further integrated through ODBC connectivity with MySQL services for historical log storage, retrieval, and reporting.

B. DATASET DETAILS

The system logs from the IoT gateway which is collected over a period of a month is considered here for a deep dive analysis.

A sample of this log excerpt is shown in Table 1. These log snippets are basically a set of unstructured log chunks and contain a series of nonwrangled text information that is unsuitable for data processing. Furthermore, each of these log events can be broken down into a series of virtually tagged fields. These fields are—"Date," "Timestamp," "Message Type," "MessageID," "Event details," and "Value." However, for the ease of understanding, the message identifiers and the message events are only projected from which the parsed triples are derived using the proposed framework as alluded to earlier in Section III. Additionally, two sample message details generated from two separate message types (user.notice, kern.info) are also shown in Table 2.

Moreover, the machine learning models cannot interpret normal text documents, which need to be preprocessed through careful segmentation of each of the log entry into its appropriate fields for better representation and system readability. Hence, the data first needed to be tokenised into distinct chunks for segregating each log line into specific readable tokens. Furthermore, it is also converted into desired semistructured formats using suitable delimiters for easier log understandability. Separators are used for white spaces as a custom field delimiter. Additional custom parameters are used for columns at this stage for separating the log chunks into specifically named columns (as represented in Table 2) due to the lack of header information in the log snippets. This helps the model to segregate the log lines based on their timestamp, process that initiated the log entry, process id, severity of the log message, and the log events itself, which contained the actual message. These segregated log entries are generally those messages that contain valuable insights about the current system state and play an essential role in determining failure events in advance.

The next two sections (Sections IV-C and IV-D) provide the theoretical underpinning of the proposed framework using mathematical derivations to subsequently represent the offline and the online functionalities of the proposed framework. These experimental designs are developed to conceptually validate the performance, efficiency, and effectiveness of the framework in IoT datasets, and consequently evaluate its outcome.

C. OFFLINE WORD-EMBEDDING REPRESENTATIONS OF LOG EVENTS

The flow-diagram pertaining to the offline segment of the proposed model is illustrated in Fig. 3, which has four steps.

Each step in the flow-diagram are required to execute this offline segment of the proposed framework that was detailed earlier in Fig. 1. These steps are summarized as follows.

Step 1: Accumulate and input raw log text for training the model.

- Step 1.1. Preprocess the log text corpus for word tokenisation and sequencing.

Step 2: Log text data tokenisation and sequencing subroutine.

- Step 2.1. Tokenise the input raw log text.
- Step 2.2. Retrieve the log vocabulary from the input log data.
- Step 2.3. Process and encode the tokenised word sequences to a sequence of integer values.

TABLE 2. Log fields.

Date	Timestamp	MessageType	MessageID	Event
Sat Mar 13	3:07:02 2022	user.notice	lora_pkt_fwd[13073]	RF packets forwarded
Sat Mar 13	3:07:02 2022	kern.info	quectel-CM[1360]	requestRegistrationState2

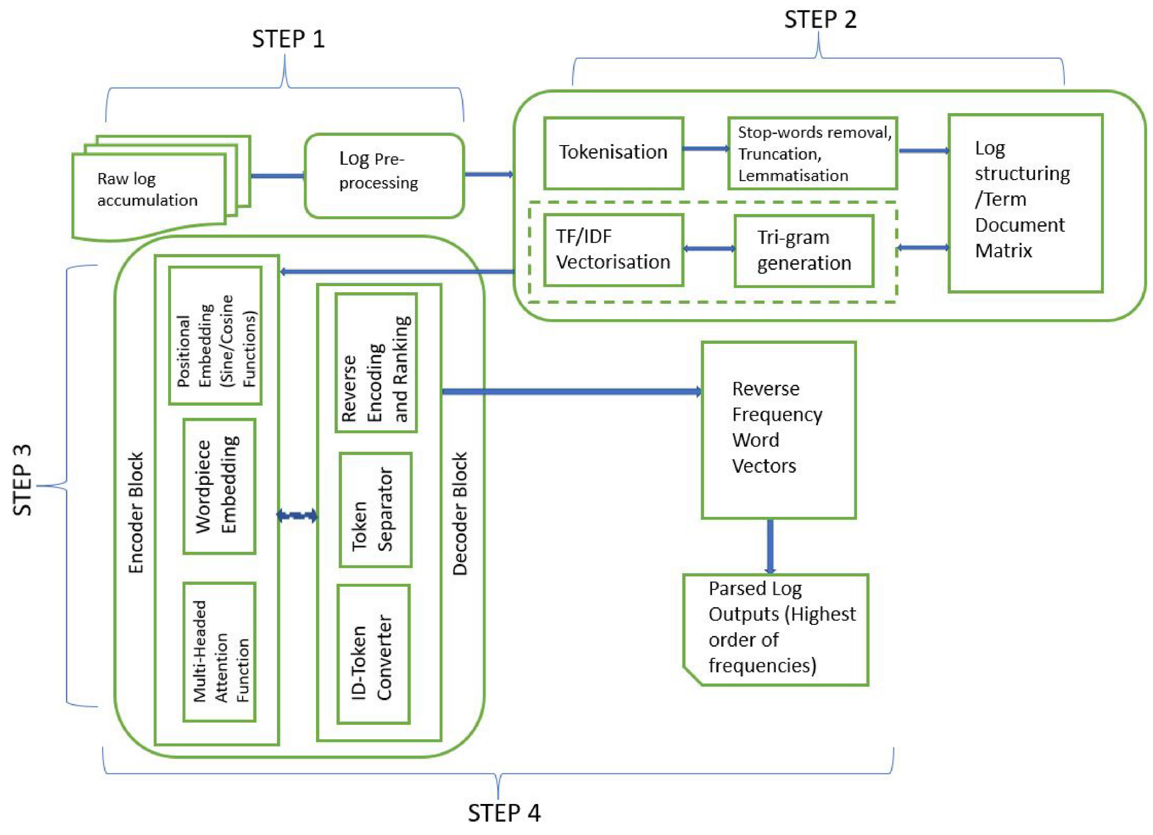
STEP 1
STEP 2
STEP 3
STEP 4
Raw log accumulation
Log Pre-processing
Tokenisation
Stop-words removal, Truncation, Lemmatisation
Log structuring /Term Document Matrix
TF/IDF Vectorisation
Tri-gram generation
Encoder Block
Positional Embedding (Sine/Cosine Functions)
Wordpiece Embedding
Multi-Headed Attention Function
Decoder Block
Reverse Encoding and Ranking
Token Separator
ID-Token Converter
Reverse Frequency Word Vectors
Parsed Log Outputs (Highest order of frequencies)

FIGURE 3. Steps of the proposed offline parsing function.

Step 2.4. Use TF/IDF Vectoriser to form a tri-gram model.

Step 3: Use tri-gram modeled output for embedding contextual information.
Step 4: Prepare data for forward and reverse encoding using pretrained neural language model.

Step 4.1. Use a pretrained encoder–decoder model (e.g., BERT/Transformer) for wordpiece tokenisation and better word representation.
Step 4.2. Use positional encoding functions for positional embedding and for sequencing the word vectors.
Step 4.3. Use decoder function to process reverse sequencing of word vectors based on their frequencies of generation.

The tri-gram output from the TF-IDF vectorizer constructs a knowledge graph in relational triple-sets using the S-V-O relationship. Considering ω_i as the target word and the S-V-O relation can be represented as (s, r, ω_i) [31], [45] where r is a variety of association relationships of ω_i. A vector representation of these association relationships is established through (1). Considering these relationships as real and meaningful from the perspective of log outputs, the inference is drawn in (1), where "$s + r$" should be closer to the target word in question [31]. A few examples of such relationships are shown later under the online stage in Fig. 6. The objective function of this relationship is further derived through

$$\mathcal{L}_{\text{Triples}} = \sum_{k=1}^{|V|} \log p\left(\omega_i | \omega_{i-v}^{i+v}\right) + \ddot{Y} \sum_{r \in R_{\omega_i}} \log p(\omega_i | s + r). \tag{1}$$

In addition, the pretrained neural language model combines the semantic and syntactic relationship along with the positional embedding to extract contextual information for a better representation of the words. During the subsequent steps, the target words ω_i is transformed into hidden input vectors, by aggregated functions through the addition of each of the word-piece token embeddings $T\omega_i$ and their

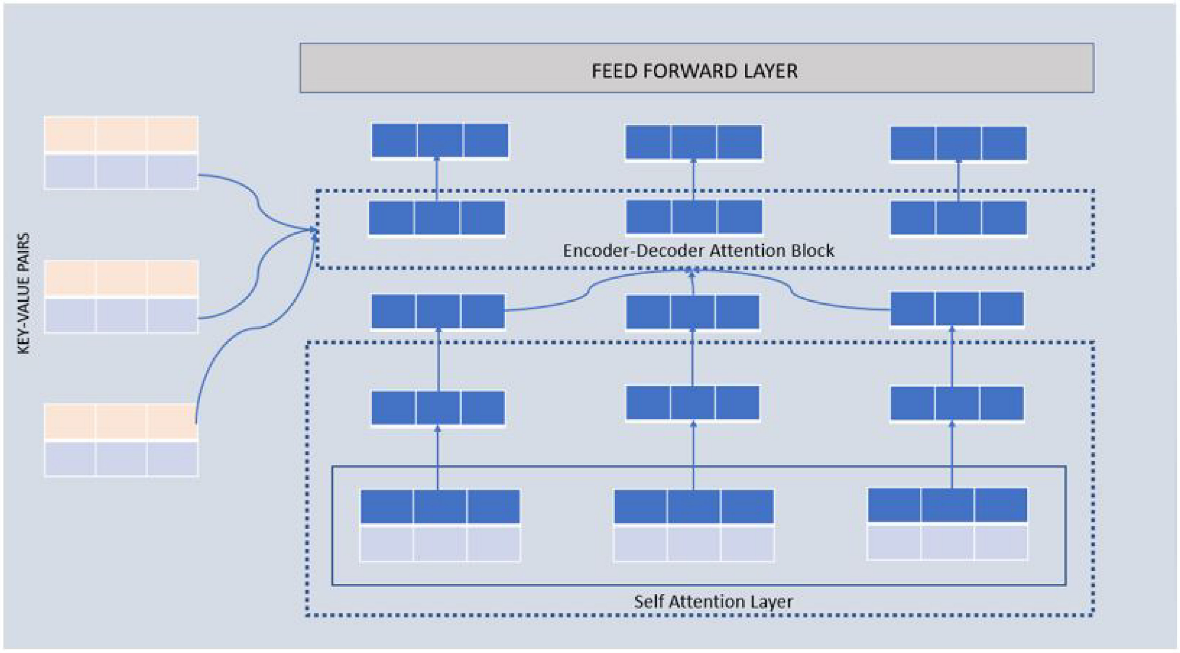

FIGURE 4. Encoder–Decoder architecture (source: adapted from AI Zone Kevin Hooke, 2020).

consecutive positional embeddings $P\omega_i$ as derived in

$$h_i = T_{\omega_i} + P_{\omega_i}. \tag{2}$$

Furthermore, considering the relationship in (2), this proposed approach explored the encoder–decoder architecture to address such contextualization issues in log analytics from a mathematical perspective. Each encoder block consists of two main layers and two sublayers called the attention heads. These main layers are feed-forward and multihead attention layers which also have two sublayers of addition + normalization. Through the attention heads, such encoder functions can efficiently handle time irregularities and long-term dependencies using their parallelable architectures. Additionally, such models handle residual data using sublayers. Fig. 4 illustrates various layers that are associated within an encoder–decoder architecture.

The scaled dot-product self-attention function with normalization is derived using the query, key, value pair in

$$\text{Attn }(Q, K, V) = \text{SoftMax}\left(\frac{Q \cdot k^T}{\sqrt{d_k}}\right).V. \tag{3}$$

Equation (3) presents an attention matrix with the attention values of each tokenised word in relation to the context words, which is further normalized using the SoftMax function. The weight matrix provides opportunities to further fine-tune and adjust each of the parameters. To derive the learned parameters, learned projection weight matrices W^K, W^Q, and W^V are derived in

$$K^T = W^K \cdot K \text{ where } \left(W^K \in R^{d_m \times d_k}\right) \tag{4}$$

$$Q^\top = W^Q \cdot Q \text{ where } \left(W^Q \in R^{d_m \times d_q}\right) \tag{5}$$

and

$$V^\top = W^V \cdot V \text{ where } \left(W^V \in R^{d_m \times d_v}\right). \tag{6}$$

This transposed weight matrix ensures dimensionality reduction to the model through each of its "m" multidimensional attention heads that has adjustable weight parameters (W) for further fine-tuning of the model. Hence, each token now has reduced dimensions of vectors expressed through K^T, Q^T, and V^T. The resulting output is a concatenation of all the individual outputs of the attention heads that finally provide an aggregated output with dimensions the same as that of its input vector representations [59]. The complete lifecycle of these vectorized token embeddings can be represented as a function of concatenation or summation of all the embedded features: Concat [Head 1$\oplus$ Head 2$\oplus$ Head 3$\oplus$ Head 4$\oplus$ Head 5$\oplus$ Head 6 $\oplus \cdots$ Head 12].

To summarize, on the original input tokens, word-piece embedding is applied to each token for multidimensional vector representations, which consequently passes through the multiattention heads where the learned projection layer helps in adjustments of the weight matrix to further fine-tune the model and reduce the dimensions of each token. Finally, the next layer performs concatenation of each of the multiattention head outputs to retain the original input dimensions. This operation is repeated and performed on the key matrix, the value matrix, and as well as on query operations to derive the key-query compatibility. The process, in turn, mimics the sequence-to-sequence output of a typical autoregressive RNN model, yet through an advanced parallelable manner.

The next layer within the attention heads is the feed-forward layer consisting of two projection sublayers which handle the positional contextualization as derived in (7), where W_1, and W_2 are the two sublayers of adjustable weight matrices in which nonlinearity is introduced through the Max function and along with bias b_1 and b_2

$$\text{FFN}(X) = \text{Max}(0, xW_1 + b_1)W_2 + b_2. \tag{7}$$

This feedforward layer is applied for each positional encoding identically yet separately to each of the input tokens. However, as the subsequent challenge for the model is in determining the order and positioning of the tokens so that they are fed sequentially, positional information through encoding needs to be established. Hence, the even and odd dimensions are represented using sine and cosine wave transformations and are expressed in the form of (8) and (9), where, on every ith token position, the phase would be different, which is represented by the function $([P \circ s/10000(2i/d)])$.

For, all even dimensions

$$\text{PE}(\text{pos}, 2i) = \sin\left(\frac{\text{Pos}}{10000^{\frac{2i}{d}}}\right). \tag{8}$$

For, all odd dimensions

$$\text{PE}(\text{pos}, 2i+1) = \cos\left(\frac{\text{Pos}}{10000^{\frac{2i}{d}}}\right). \tag{9}$$

The final implementation of the model is expressed in (10), where word embeddings are summed up with the positional embeddings, which provide the aggregated values with each of their positional signals embedded onto it, and consequently, this is fed to the self-attention and the feedforward block of the attention heads

$$h_i = E\omega_i + P \cdot E\omega_i. \tag{10}$$

The objective or the loss function is derived in (11), where X being a subset of the input tokens along with a [MASKed] token in it, to accurately predict the masked token. Additionally, $\Pi = (\omega_1, \omega_2, \omega_3, \ldots, \omega_k)$ denotes the masked tokens in the sentence X (consisting of triple-sets forming the new vocabulary dataset) while X_Π are the set of tokens that are masked and $X_{-\Pi}$ are the respective unmasked tokens

$$\mathcal{L}_{\text{MLM}} = \frac{1}{K}\sum_{k=1}^{|V|} \log p\big(\omega_{\Pi_k} | X_{-\Pi}; \theta\big). \tag{11}$$

The overall objective function is thus expressed as the summed representation in

$$\mathcal{L}_o = \mathcal{L}_{\text{Triples}} + \mathcal{L}_{\text{MLM}}. \tag{12}$$

This methodology and model help in achieving a lower dimensional word embedding to project a distributed representation of the language model, from which effective knowledge can be extracted with contextual information. Additionally, with the combination of the positional embeddings, the quality of the output word vectors have significantly improved, and a better word representation is achieved.

Hence, this is one of the significant achievements of the model as it can represent contextual words more efficiently using lower dimensional embedding by capturing their distributional representations. This process allows valuable information to be effectively extracted and shared amongst the words to project their contextual similarities. This is particularly suitable in predictive log analytics scenarios as the model can handle both contextual and unknown domain-specific words that randomly appear at different stages of the log outputs.

D. OUT-OF-VOCABULARY WORD HANDLING DURING ONLINE STAGE

The model produces the OOV word embeddings during system runtime depending on the OOV's word embedding. Though the preprocessing and word-piece tokenisation handles most of the UNKNOWN and OOV words through lemmatization and decomposition, there are possibilities of unknown domain-specific words to be present in the log vocabulary.

The process, as illustrated in Fig. 5, starts with replacing the OOV words using [MASKed] tokens, and the log entry chunk is further tokenised and converted into the model's input identifiers. These masked tokens are maximized for predicting the probabilistic distribution of all possible words at each time step of the query input. Finally, the predicted values are returned by replacing the masked words in the log texts with that of the predicted words.

Fig. 5 represents the flow-diagram of the online OOV word embedding and parsing stage. The steps in Fig. 5 are detailed in the following paragraphs, which are a continuation from the steps in Fig. 4.

In this way, the OOV word processor is trained dynamically during the model runtime, and a new embedding of the UNKNOWN words is provided when domain-specific words are generated as a part of the log output.

Step 5: Loading the pretrained model.

- Step 5.1. The pretrained language model is loaded in the neural language framework (e.g., BERT) and OOV words are detected.
- Step 5.2. The neural language tokeniser is loaded for text preprocessing and lemmatization.

Step 6: Log sequence generation.

- Step 6.1. The input log entry text is fed for further modeling.
- Step 6.2. Probability distribution of the tokenised words are predicted.
- Step 6.3. The predicted vocabulary is returned as an output.

Step 7: Streaming logs are fed to the model.

Step 8: Embedding function for the OOV words.

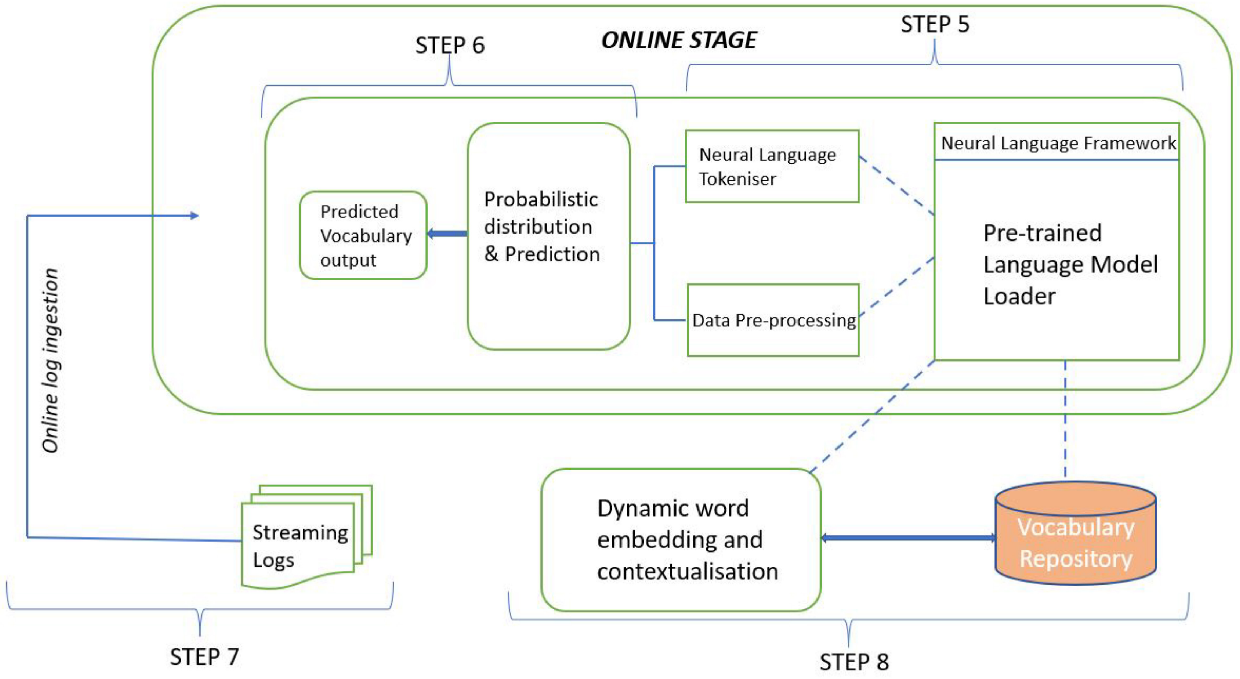
STEP 6
ONLINE STAGE
STEP 5
Predicted Vocabulary output
Probabilistic distribution & Prediction
Neural Language Tokeniser
Data Pre-processing
Neural Language Framework
Pre-trained Language Model Loader
Online log ingestion
Streaming Logs
Dynamic word embedding and contextualisation
Vocabulary Repository
STEP 7
STEP 8

FIGURE 5. Steps of the proposed online OOV parsing stage.

Step 8.1. The streaming log entry chunks are checked for OOV words.

Step 8.2. When any new OOV words/domain-specific words are found, word embedding is performed, and values are assigned to the words based on their contexts.

Step 8.3. The embedded words are updated in the vocabulary dataset by adding them to the pretrained neural language model.

This is consequently updated in the vocabulary dataset so that the word representations are retained based on their semantic and contextual information. This very approach of word embedding is adopted so that the OOV words have a contextual and relative position in the vector space, even though it is not embedded during the pretraining phases of the model.

V. EXPERIMENTAL ANALYSIS, RESULTS AND INTERPRETATION

This section provides the experimental outcome of the proposed framework through detailed theoretical analysis in Section V-A. This is followed by Section V-B, which provides a thorough interpretation of the results and outcome obtained from the mathematical experiments.

A. EXPERIMENTAL ANALYSIS

The mathematical representation of this proposed framework has enabled advanced log analytics through the extraction of varied dimensions, and assisted in augmenting the traditional log mining approaches. Furthermore, the adopted approach helped in developing suitable algorithmic models.

TABLE 3. Extracted log events.

```
LOG EVENTS:


 PULLDATA sent: 6 (100.00 acknowledged)
 PULLRESP(onse) datagrams received: 0 (0 bytes)
 RF packets sent to concentrator: 0 (0 bytes)
 TX errors: 0
 TX rejected (collision beacon): 0(req:1, rej:0)
 TX rejected (too late): 0
 TX rejected (too early): 100
 INFO: [up] PUSHACK received in 460 ms
 INFO: [down] PULLACK received in 478 ms
 BEACON queued: 0
 BEACON sent so far: 0
 BEACON rejected: 0
```

In this experiment, during the offline training stage, the raw log text corpus is preprocessed for tokenisation and further represented sequentially. After preprocessing, the log chunks are extracted, as shown in Table 3 where unnecessary field entries are removed so that the processed log chunks now contain the high-level rows. The output is a raw log vocabulary subset which is further used for determining the common word sequences and in finding outliers.

The subsequent steps involved vectorization of the raw log entry chunks for forming an event matrix from the log vocabulary. The tri-gram structures (as explained in

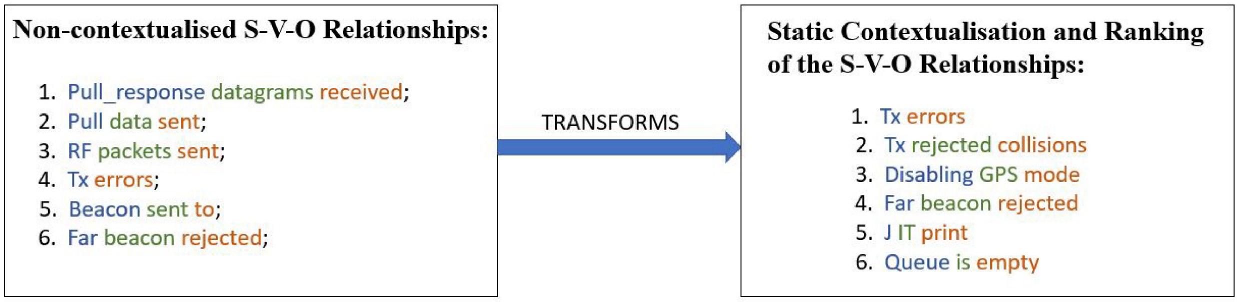

FIGURE 6. Tri-gram outputs transformed to Contextualized Tri-grams.

Section IV-C) are established using term frequency/inverse document frequency (TF/IDF) vectorization approaches to form the relational triple-sets for static word embedding representations, as presented in Fig. 6.

These triple-sets form the SVO relationships, embedding the local contextualized information within the model for static word representations in the form of a knowledge graph, as illustrated in Fig. 6. During this training phase, the model learns the context words based on their co-occurrence relations. The objective function is derived from its negative log-likelihood of the target word given a set of context words. The weights between the input layers and that of the hidden layers are finally considered the final vector representation of the modeled output. The output is a matrix representation that would help count the word sequences and determine word frequencies. The relational triples are established using this contextualized parsing method to improve the output ranking and quality of the target word vectors.

These output triples that are derived from the static word embedding are fed as an input to the encoder–decoder architecture, where the scaled dot-product attention is calculated using a SoftMax function as per (3). Here, the key and query are considered the same as the model calculates the self-attention to generate the key-query compatibility. In other words, as a part of this attention process, the context vector is calculated as a weighted summation of the values, wherein the associated weights of these values are calculated using the compatibility function of their corresponding key-query. This generates a symmetrical attention matrix which embeds the information of the attention weights for each word in the triple-sets/sentences. The weights are calculated such that the context words are given adequate weightage in relation to each of the target words ω_i which provides better word vector representations. In turn, this provides the model with the information as to which tokens are important in context with the given target word token ω_i. The SoftMax layer further normalizes the outputs so that the resultant aggregated output of each row of the matrix is 1. On the other hand, the OOV words pose additional challenges and hence the model is trained with word-piece tokenisation using the log corpus as the data distribution is different than that of the originally trained corpus. Examples of such OOV words are, "*src*/*jitqueue*.*c*:448:*jit_print_queue*(): *INFO*: [*jit*] queue is empty." Such raw log lines are decomposed into multiple subwords within the triple-sets in the format "*j_it print Queue*," "*Queue is empty*" and so on.

TABLE 4. Term frequencies of generated triple-sets.

Triple-set log entries	Count
Response datagrams received	97.5
Push ack received	97.5
Pull ack received	97.5
RF packets received	85.2
TX Errors [dummy]	3.5
Disabling GPS mode	2.42

Furthermore, for better word representation, the model considers both the left and the right context of the target word ω_i simultaneously unlike its predecessors, such as bidirectional LSTM (Bi-LSTM), embeddings for language models (ELMo), and so on, using masked language modeling. Considering the log corpus, the only parameters learned from scratch are the weights to minimize the loss of wrongly predicting a random masked word in relation to its output representations. This is handled by readjusting the weights using the objective function represented in (11).

The final output of the contextualized dynamic word embedding triple-sets is sorted and stored in reverse order within the log vocabulary dataset based on their relative ranking and frequency of generation. These outputs are highly compressed, with adequate semantic and contextual embeddings that would be immensely beneficial for further downstream activities of NLP tasks, such as NER and predictive analytics. Some of the summarized outputs extracted from the log excerpts and their generation frequency are shown in Table 4.

Similarly, during runtime, the streaming logs are fed to the model, which checks for new OOV words that are not a part of the pretrained embedding vocabulary dataset. When a completely new word is found, word piece tokenisation is

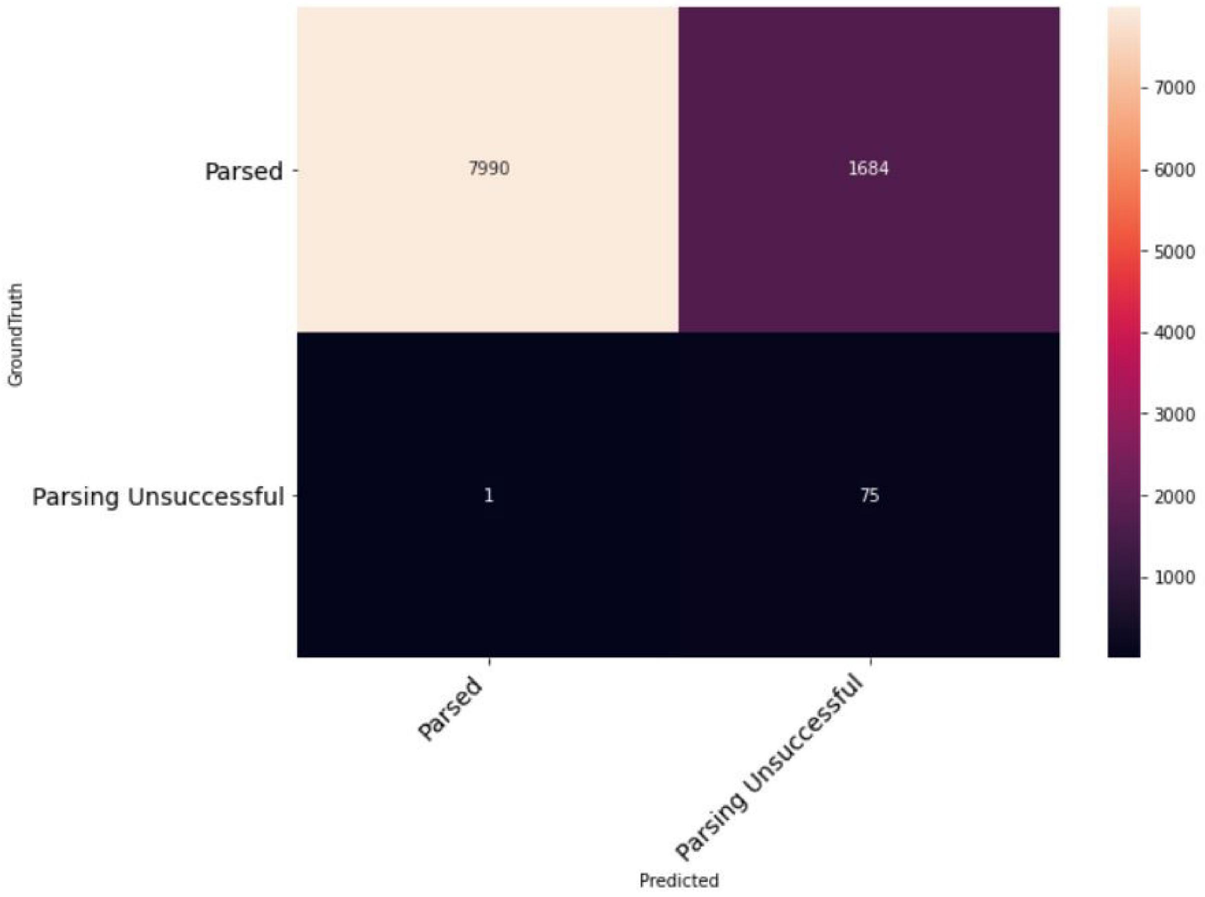

FIGURE 7. Confusion matrix output.

performed to assign a new word embedding dynamically to those unseen words in the online stage. These words are later fed back and updated with the new distributed representations within the log vocabulary dataset for future processing.

B. NUMERICAL ANALYSIS AND VALIDATION

To quantify the overall performance of the proposed model, the following numerical methods are used as part of the evaluation parameters—1) parsing accuracy; 2) runtime efficiency; and 3) overall effectiveness.

To measure the parsing accuracy—precision, recall, and $f1$-score are calculated [60] using the expressions in

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{13}$$

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{14}$$

$$f1\text{-Score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{15}$$

Here, the $f1$-score is basically the harmonic mean of the precision and recall that helps in accurately estimating the performance of the model in terms of its parsing accuracy. More precisely, the parsing accuracy is the metric that determines the ratio of the correctly parsed log events over the total population.

TABLE 5. Parsing classification report.

	precision	*recall*	*f1-score*	*support*
parsed	1.00	0.83	0.90	9674
parsing unsuccessful	0.04	0.99	0.08	76
accuracy			**0.83**	9750
macro avg	0.52	0.91	0.49	9750
weighted avg	0.99	0.83	0.90	9750

Moreover, it has been observed that the overall parsing accuracy of the proposed model has been greater than or equal to 0.83. This is represented using a confusion matrix plot in Fig. 7. The model yields a score between 0.8 and 1 as shown in Table 5. The SVO relations were initially extracted from the raw logs manually to establish the ground truth of the dataset for the purposes of evaluation. The parsing accuracy is calculated against this manual classification data with that of the parsed outputs generated by the proposed framework. To perform the model evaluation, logs are randomly selected to calculate the true-positive, true-negative, false-positive, and false-negative iterations. For the "parsed accurately" or the positive class, the true positive is the successfully parsed ones that were determined to be correctly predicted after comparison with the manual ground truth data. The false positive is the successfully parsed ones that were determined to be incorrect after comparing it with the manual ground-truth data. Similarly, for the "parsed inaccurately" or the negative class, the true negative is the unsuccessfully parsed ones that matched correctly with the manually incorrect log tuples. The false-negative ones are the ones that were correctly learned by the proposed model but did not generate the same log triple outputs as that of the manually parsed sampled data. The log excerpts used for these experimental purposes have a chunk of 9750 log lines. The classification report in Table 5 provides a snapshot summary of the various numerical outputs against this support of 9750 events. The positive and the negative classes are the respective logs that are parsed successfully and the logs that could not be parsed accurately. The test sample that belongs to the positive class is 9674, while the test sample that belongs to the negative class is 76. Additionally, to handle the class imbalance in the dataset, the macro average is calculated by providing equal weightage to each sampled class. This can be represented by the expression in

$$\begin{aligned}&\text{score}_{\text{macro-avg}}\\&\quad= 0.5\cdot\ \text{score}_{\text{class [parsed]}}\\&\quad\ + 0.5\cdot\ \text{score}_{\text{class [parsing unsuccessful]}}\cdot\end{aligned}\tag{16}$$

Hence, the macro average for the precision of the positive and the negative class is computed to be 0.52. The one for recall has been calculated to be 0.91, and for $f1$-score to be 0.49. Similarly, to determine the impact of precision, recall, and $f1$-score due to class imbalance against its average output, the weighted average is calculated. The weighted average is expressed by

$$\begin{aligned}&\text{score}_{\text{weighted-avg}}\\&\quad= 0.998\cdot\text{score}_{\text{class [parsed]}}\\&\quad\ + 0.002\cdot\text{score}_{\text{class [parsing unsuccessful]}}\cdot\end{aligned}\tag{17}$$

The weighted average for precision has been computed to be 0.99. The one for recall is 0.83 and for $f1$-score to be 0.90. It can be inferred from the overall macro and the weighted average that the model performed well in all classes despite data imbalance. Moreover, the sampled data has high true positives and fewer false negatives, which signifies that the overall performance of the model is significantly higher considering this sampled dataset. Though due to this class imbalance, there is a possibility that the parsing accuracy may not reflect the actual performance of the model as it is only projecting the underlying class distribution. This can be mitigated by using larger datasets and through proper parameter tuning and effective optimization of the objective or loss functions of the model. It was observed that with the offline model-run, the parsing accuracy was 0.68. The model by virtue of its OOV word handling framework using dynamic word embeddings could significantly increase the abstraction capabilities of the model.

Moreover, the proposed model carries a few advantages over the traditional log mining techniques, in terms of its efficiency as it is only concerned with the free-text string-based log events that make preprocessing of the data faster, eventually increasing runtime. Additionally, the model adopted parallelization techniques over standard clustering methods, as log clustering does not scale well with larger datasets. The transformer-based architecture further provided additional benefits as it did not require complex hyper-parameter tuning and the tuning was performed on sample datasets with the intent of adopting them on larger datasets for future predictive modeling [61].

However, it is observed that the runtime efficiency of standard machine learning models is comparatively faster than the proposed model, but when compared with RNN-based architectures, the model runs significantly faster. A numerical evaluation of the execution time against chosen standard models is presented under Section VI.

Additionally, the effectiveness of the model is evaluated based on its parsing accuracy. It has been observed earlier that the overall parsing accuracy is high (>0.83) compared to the SOTA approaches. Moreover, the effectiveness is not uniform and depends on the log size as well. Nevertheless, the model exhibited high effectiveness on smaller log subsets. This would be evaluated further on larger log datasets while performing downstream failure prediction tasks as a part of the future expansion of this research.

VI. COMPARATIVE STUDY AND DISCUSSION

There are several log parsing and analytics techniques that are proposed in the literature. However, the majority of these models have focused on predefined rule-based heuristics and template pattern generations that cannot contextualize to discover the latent meaning hidden in the log outputs. Additionally, none of these models has approached log parsing from a time sensitivities perspective where attention-based sequence to sequence log abstraction is performed for deriving triple-sets from raw log sequences. This is particularly beneficial in IoT systems' log data analytics where data are generated inherently from low-powered signals that may contain time irregularities. Table 6 presents a model comparison between the proposed and traditional parsing approaches for time sensitivity, used method/techniques and, parallelability. The majority of the methods shown in Table 6 did not consider the time irregularities except "DeepLog" which used both log-key and time-sensitive information. These studies were somewhat biased in knowing the information beforehand to fulfil the future downstream activities, such as log anomaly detection and failure prediction scenarios. By using this word embedding and attention-based mechanism, the proposed model achieved high levels of efficiency even without the availability of prior heuristics-based information. Additionally, the approach can be considered as the first work that imbibed contextualization within the framework

TABLE 6. Model comparisons.

Model	Algorithmic approach	Method	Time Sensitive	Mode	Parallelisability
Spell	Lowest Common Subsequence	Log key based	No	Online	Yes
Drain	Fixed Depth Parse Tree	Heuristics based	No	Online	No
POP	Hierarchical Clustering & Partitioning	Heuristics based	No	Offline	Yes
DeepLog	N-Gram & LSTM	NLP & Log Key-based	Yes	Online	No
NERLogParser	Bi-LSTM	NLP & NER based	No	Offline	No
LogEvent2Vec	TF-IDF & Deep Neural Network	Word Embedding	No	Offline	No
LogTransfer	Transfer Learning & LSTM	Template based	No	Offline	No
Graphical Log Mining	Stream Process & Graph Partition	Probabilistic Penalty Graph	No	Online	No
Log2Vec	Lexical & Semantic Embedding	Word Embedding	No	Online	No
LogParse	OpenIE & Log Summarization	Word Embedding & Template based	No	Online	No
Proposed Model	Neural OpenIE & Encoder-Decoder	Dynamic Word Embedding & Attention based	Yes	Online	Yes

TABLE 7. Numerical comparisons.

Model	Parsing Accuracy	Runtime Efficiency (in Seconds)	Overall Effectiveness
Random Forest	0.91	0.35	P : 0.42 R : 0.65 F1: 0.51
Spell	0.60	0.69	P : 0.71 R : 0.86 F1: 0.77
Drain	0.69	0.26	P : 0.79 R : 0.92 F1: 0.86
DeepLog	0.69	5.99	P : 0.38 R : 0.97 F1: 0.54
LogMine	0.61	1.48	P : 0.59 R : 0.66 F1: 0.62
Proposed Model	0.83	0.20	P : 1.00 R : 0.83 F1: 0.90

*P *denotes* Precision; R *denotes* Recall; F1 *denotes* F-Score

through the combination of neural OpenIE in conjunction with encoder–decoder architectures, which not only process data faster but are also significantly lightweight, resulting in less consumption of computational resources during its preprocessing and runtime. As a part of the research, a comparative numerical analysis of the overall parsing accuracy is evaluated and compared with the related SOTA approaches. The proposed model has an overall parsing accuracy of **0.83**. This is noted to be significantly higher than the other offline and online parsing approaches as shown in Table 7. The parsing accuracy for the other models, such as LogMine, Drain, Spell, and so forth were comparatively lower than the proposed model.

Though Random Forest is an offline model and is based on standard machine learning approaches, but it exhibited better accuracy on this log subset. However, such models are not based upon neural language approaches and does not consider contextualization and semantics of logs that are essential for predictive modeling. As a part of the future extension of this research, these models would be exhaustively evaluated in larger IoT log datasets to more accurately estimate their performances. Additionally, the execution time and the overall effectiveness is also compared with the present SOTA to demonstrate the capability of the proposed log parsing framework. The evaluation results are presented in Table 7.

The average runtime of the models are less than or equal to 1.5 s on raw log datasets without prefiltering. Traditionally, the runtime increases with the increment of log sizes and the processing becomes considerably slower, resulting in system timeouts.

It is observed that the standard machine learning models took on an average of < 0.35 s and was comparatively faster than the other deep learning approaches. Such models are inefficient and have limited use in future downstream tasks as they are not contextually aware and sensitive to infrequent log event generations. On the other hand, Spell took 0.69 s and Drain executed in 0.26 s, while DeepLog and LogMine took comparatively longer and ran for 5.99 and 1.48 s, respectively. This proposed model with its efficient prefiltering and preprocessing architecture could effectively parse the log subset within 0.20 s during offline run. Though there are possibilities to further fine-tune by prefiltering a considerable chunk of this log dataset prior to training and cross validating each of the models to reduce any data outliers. This may further improve the runtime efficiency of these existing models as well, but may not be a feasible approach in case of future analysis of larger log datasets. Furthermore, the numerical analysis in Table 7 shown that the proposed model achieved the best performance as it utilized dynamic word embedding and word-piece tokenisation approach for appropriately tokenising unseen OOV words yet representing accurately the semantics of each of the log messages. It is observed that there may be a slight performance hit of the proposed model, if the number of attention heads and the feedforward network sizes are reduced. Though, from the perspective of overall effectiveness, the proposed model achieved higher precision (1.00) and recall (0.83) as that of other advanced machine learning-based graph partitioning approaches [23], [62]. Additionally, the $F1$-score (0.90) achieved by the proposed model using larger feedforward neural layers, proved that the model has performed significantly better in terms of its overall effectiveness and has the potential to enhance further downstream tasks, such as dynamic learning of failure patterns and anomaly prediction scenarios.

As elucidated in the previous sections, the primary intent of this study is to move away from the traditional log key and heuristics-based approaches toward that of a neural language modeling-based approach. The reason is that, most large-scale systems generate a considerable number of logs with high levels of agility. In turn, these traditional rule-based approaches become ineffective as various micro/modular services of the parent systems generate logs with varied dimensions. Additionally, the modern agile-based practices lead systems to dynamically update them, resulting in the generation of a proliferation of OOV words that are difficult to handle using the traditional statistical and key-rule-based approaches. These are some reasons for adopting a word embedding method in this study so that the model can handle such robustness with higher precision and accuracy. Moreover, with the integration of an attention-based mechanism, the model can now handle dynamic word representations, which further alleviated its capability of working in typical IoT environments where time irregularities of log generations are a concern.

Furthermore, as the premise of the log parsing framework is oriented toward event stream processing of online logs, this model may be adapted in other log datasets with minor fine-tuning and modifications. Moreover, the current research is evaluated only on IoT log datasets, captured from the IoT gateways, and is based on the premise of IoT networks.

VII. CONCLUSION

The study investigated the method of integrating neural OpenIE with pretrained encoder–decoder architecture for better word representations in scenarios of IoT log analytics. The mathematical model further proved that time sensitivities could be effectively handled using an attention-based mechanism compared to rule-based frameworks that are prevalent in current SOTA approaches. Furthermore, the static word embedding and contextual information representations using the relationship of the derived triple-sets from the TF-IDF architecture enhanced the model's performance by significantly reducing the computational time and its allocated resources.

Future extension of this work will focus on optimizing the proposed model so that it can be validated using multiple NLP-based failure prediction scenarios and further enhance its adaptive capabilities in IoT-based log abstraction and service management activities.

In terms of its limitations, the disparity in accuracy that is noticed for various approaches is the way in which the manual labeling of the ground-truth data is organized. Since this is specific to the IoT log instances and the way such data was handled, it may have imposed further limitations in accurately estimating the overall performance of these SOTA approaches. This would be investigated further on larger datasets and with additional attributes pertaining to various failure prediction case studies. The expansion of this research would consider the performance of these SOTA approaches, not just on its parsing capabilities but also from the perspectives of its accurate determination and reporting of failures/anomalies in the log snippets. Furthermore, there is a also a possibility that the proposed model can suffer from false positives and false negatives during future downstream task implementations while extracting information in terms of segregating anomalies from informational logs. This can be potentially mitigated by proper parameter tuning during the training phases of the model while developing such future predictive and prescriptive log analytics tasks.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their valuable comments that helped them in improving this research.

REFERENCES

[1] K. Adnan and R. Akbar, "An analytical study of information extraction from unstructured and multidimensional big data," *J. Big Data*, vol. 6, no. 1, p. 91, 2019, doi: 10.1186/s40537-019-0254-8.

[2] S. Ali, H. Mousa, and M. Hussien, "A review of open information extraction techniques," *Int. J. Comput. Inf.*, vol. 6, no. 1, pp. 20–28, 2019, doi: 10.21608/ijci.2019.35099.

[3] R. B. Almeida et al., "A distributed event-driven architectural model based on situational awareness applied on Internet of Things," *Inf. Softw. Technol.*, vol. 111, pp. 144–158, Jul. 2019, doi: 10.1016/j.infsof.2019.04.001.

[4] N. Aussel, Y. Petetin, and S. Chabridon, "Improving performances of log mining for anomaly prediction through NLP-based log parsing," in *Proc. IEEE 26th Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst. (MASCOTS)*, 2018, pp. 237–243.

[5] Z. Ballard, C. Brown, A. M. Madni, and A. Ozcan, "Machine learning and computation-enabled intelligent sensor design," *Nat. Mach. Intell.*, vol. 3, no. 7, pp. 556–565, 2021, doi: 10.1038/s42256-021-00360-9.

[6] N. Basha, M. Z. Sheriff, C. Kravaris, H. Nounou, and M. Nounou, "Multiclass data classification using fault detection-based techniques," *Comput. Chem. Eng.*, vol. 136, May 2020, Art. no. 106786, doi: 10.1016/j.compchemeng.2020.106786.

[7] H. Bast and E. Haussmann, "More informative open information extraction via simple inference," in *Proc. ECIR*, 2014, pp. 585–590.

[8] P. Braun, A. Cuzzocrea, C. K. Leung, A. G. M. Pazdor, and K. Tran, "Knowledge discovery from social graph data," *Procedia Comput. Sci.*, vol. 96, pp. 682–691, Jan. 2016. [Online]. Available: https://doi.org/10.1016/j.procs.2016.08.250

[9] V. Y. Chandrappa, B. Ray, N. Ashwath, and P. Shrestha, "Application of Internet of Things (IoT) to develop a smart watering system for cairns parklands—A case study," in *Proc. IEEE Region 10 Symp. (TENSYMP)*, Dhaka, Bangladesh, Jun. 2020, pp. 1118–1122, doi: 10.1109/TENSYMP50017.2020.9230827.

[10] Y. Chen, Z. Zhen, H. Yu, and J. Xu, "Application of fault tree analysis and fuzzy neural networks to fault diagnosis in the Internet of Things (IoT) for aquaculture," *Sensors*, vol. 17, no. 1, p. 153, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/1/153

[11] S. Choudhury et al., "NOUS: Construction and querying of dynamic knowledge graphs," in *Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE)*, Apr. 2017, pp. 1563–1565, doi: 10.1109/ICDE.2017.228.

[12] L. Cui, F. Wei, and M. Zhou, "Neural open information extraction," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguist. (Vol. 2 Short Papers)*, Melbourne, VIC, Australia, Jul. 2018, pp. 407–413. [Online]. Available: https://aclanthology.org/P18-2065. doi: 10.18653/v1/P18-2065.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[14] M. Du and F. Li, "Spell: Online streaming parsing of large unstructured system logs," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 11, pp. 2213–2227, Nov. 2019, doi: 10.1109/TKDE.2018.2875442.

[15] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Dallas, TX, USA, 2017, pp. 1285–1298. [Online]. Available: https://doi.org/10.1145/3133956.3134015

[16] A. Elragal and M. Haddara, "Design science research: Evaluation in the lens of big data analytics," *Systems*, vol. 7, no. 2, p. 27, 2019. [Online]. Available: https://www.mdpi.com/2079-8954/7/2/27

[17] T. Elsaleh, S. Enshaeifar, R. Rezvani, S. T. Acton, V. Janeiko, and M. Bermudez-Edo, "IoT-Stream: A lightweight ontology for Internet of Things data streams and its use with data analytics and event detection services," *Sensors*, vol. 20, no. 4, p. 953, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/4/953

[18] J. Gao, H. Wang, and H. Shen, "Task failure prediction in cloud data centers using deep learning," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 1111–1116, doi: 10.1109/BigData47090.2019.9006011.

[19] A. Ghafouri, W. Abbas, A. Laszka, Y. Vorobeychik, and X. Koutsoukos, "Optimal thresholds for anomaly-based intrusion detection in dynamical environments," 2016, *arXiv:1606.06707*.

[20] R. Glauber and D. B. Claro, "A systematic mapping study on open information extraction," *Expert Syst. Appl.*, vol. 112, pp. 372–387, Dec. 2018. [Online]. Available: https://doi.org/10.1016/j.eswa.2018.06.046

[21] C. Gutschi, N. Furian, J. Suschnigg, D. Neubacher, and S. Voessner, "Log-based predictive maintenance in discrete parts manufacturing," *Procedia CIRP*, vol. 79, pp. 528–533, Jan. 2019. [Online]. Available: https://doi.org/10.1016/j.procir.2019.02.098

[22] H. Hamooni, B. Debnath, J. Xu, H. Zhang, G. Jiang, and A. Mueen, "LogMine: Fast pattern recognition for log analytics," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manag.*, Indianapolis, IN, USA, 2016, pp. 1573–1582. [Online]. Available: https://doi.org/10.1145/2983323.2983358

[23] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "Towards automated log parsing for large-scale log data analysis," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 6, pp. 931–944, Nov./Dec. 2018, doi: 10.1109/tdsc.2017.2762673.

[24] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 33–40, doi: 10.1109/ICWS.2017.13.

[25] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quart.*, vol. 28, no. 1, pp. 75–105, 2004, doi: 10.2307/25148625.

[26] A. Ismaili-Alaoui, O. Kasmi, A. Baïna, K. Baïna, K. Benali, and M. Bellafkih, "Priority-based event management using fuzzy logic for an IoT-BPM architecture," in *Proc. IEEE 12th Conf. Services Orient. Comput. Appl. (SOCA)*, Nov. 2019, pp. 111–118, doi: 10.1109/SOCA.2019.00024.

[27] R. Krishnamurthi, A. Kumar, D. Gopinathan, A. Nayyar, and B. Qureshi, "An overview of IoT sensor data processing, fusion, and analysis techniques," *Sensors*, vol. 20, no. 21, p. 6076, Oct. 2020, doi: 10.3390/s20216076.

[28] Y. Liao, X. Jiang, and Q. Liu, "Probabilistically masked language model capable of autoregressive generation in arbitrary word order," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguist.*, Jul. 2020, pp. 263–274. [Online]. Available: https://aclanthology.org/2020.acl-main.24. doi: 10.18653/v1/2020.acl-main.24.

[29] X. Ma et al., "A survey on deep learning empowered IoT applications," *IEEE Access*, vol. 7, pp. 181721–181732, 2019, doi: 10.1109/access.2019.2958962.

[30] M. Madkour, D. Benhaddou, and C. Tao, "Temporal data representation, normalization, extraction, and reasoning: A review from clinical domain," *Comput. Methods Programs Biomed.*, vol. 128, pp. 52–68, May 2016. [Online]. Available: https://doi.org/10.1016/j.cmpb.2016.02.007

[31] W. Meng et al., "LogParse: Making log parsing adaptive through word classification," in *Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2020, pp. 1–9, doi: 10.1109/ICCCN49398.2020.9209681.

[32] W. Meng et al., "Summarizing unstructured logs in online services," 2020, *arXiv:2012.08938*.

[33] S. Namuduri, B. N. Narayanan, V. S. P. Davuluru, L. Burton, and S. Bhansali, "Review—Deep learning methods for sensor based predictive maintenance and future perspectives for electrochemical sensors," *J. Electrochem. Soc.*, vol. 167, no. 3, 2020, Art. no. 37552, doi: 10.1149/1945-7111/ab67a8.

[34] D. Nguyen, C. Nguyen, T. Duong-Ba, H. Nguyen, A. Nguyen, and T. Tran, "Joint network coding and machine learning for error-prone wireless broadcast," in *Proc. IEEE 7th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2017, pp. 1–7, doi: 10.1109/CCWC.2017.7868415.

[35] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, Dec. 2007, doi: 10.2753/MIS0742-1222240302.

[36] X. Qiao, H. Cao, and T. Zhao, "Improving unsupervised dependency parsing with knowledge from query logs," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 16, no. 1, p. 3, 2016, doi: 10.1145/2903720.

[37] L. Ruff et al., "A unifying review of deep and shallow anomaly detection," *Proc. IEEE*, vol. 109, no. 5, pp. 756–795, May 2021, doi: 10.1109/jproc.2021.3052449.

[38] M. Singh et al., "OCR++: A robust framework for information extraction from scholarly articles," in *Proc. COLING 26th Int. Conf. Comput. Linguist. Tech. Papers*, Osaka, Japan, Dec. 2016, pp. 3390–3400. [Online]. Available: https://aclanthology.org/C16-1320

[39] V. C. Storey and I.-Y. Song, "Big data technologies and management: What conceptual modeling can do," *Data Knowl. Eng.*, vol. 108, pp. 50–67, Mar. 2017. [Online]. Available: https://doi.org/10.1016/j.datak.2017.01.001

[40] H. Studiawan, F. Sohel, and C. Payne, "Automatic event log abstraction to support forensic investigation," in *Proc. Aust. Comput. Sci. Week Multiconf.*, Melbourne, VIC, Australia, 2020, pp. 1–9. [Online]. Available: https://doi.org/10.1145/3373017.3373018

[41] H. Studiawan, F. Sohel, and C. N. Payne, "Automatic log parser to support forensic analysis," in *Proc. 16th Aust. Digit. Forensics Conf.*, 2018, pp. 1–10, doi: 10.25958/5C5268C766686.

[42] M. Suzuki, K. Komiya, M. Sasaki, and H. Shinnou, "Fine-tuning for named entity recognition using part-of-speech tagging," in *Proc. 32nd Pacific–Asia Conf. Lang. Inf. Comput.*, Hong Kong, Dec. 2018, pp. 632–640. [Online]. Available: https://aclanthology.org/Y18-1072

[43] L. Uden and W. He, "How the Internet of Things can help knowledge management: A case study from the automotive domain," *J. Knowl. Manag.*, vol. 21, no. 1, pp. 57–70, 2017.

[44] P. Ulrich, W. Becker, A. Fibitz, E. Reitelshöfer, and F. Schuhknecht, "Data analytics systems and SME type—A design science approach," *Procedia Comput. Sci.*, vol. 126, pp. 1162–1170, Jan. 2018. [Online]. Available: https://doi.org/10.1016/j.procs.2018.08.054

[45] R. Upadhyay and A. Fujii, "Semantic knowledge extraction from research documents," in *Proc. Federated Conf. Comput. Sci. Inf. Syst. (FedCSIS)*, Sep. 2016, pp. 439–445.

[46] A. Vaswani et al., "Attention is all you need," 2017, *arXiv:1706.03762*.

[47] C. Wang, H. T. Vo, and P. Ni, "An IoT application for fault diagnosis and prediction," in *Proc. IEEE Int. Conf. Data Sci. Data Intensive Syst.*, Dec. 2015, pp. 726–731, doi: 10.1109/DSDIS.2015.97.

[48] J. Wang, Z. Wang, D. Zhang, and J. Yan, "Combining knowledge with deep convolutional neural networks for short text classification," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, 2017, pp. 2915–2921.

[49] Y. Wang, L. Cui, and Y. Zhang, "Improving skip-gram embeddings using BERT," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 29, pp. 1318–1328, Mar. 2021, doi: 10.1109/TASLP.2021.3065201.

[50] Z. Wang, W. Bi, Y. Wang, and X. Liu, "Better fine-tuning via instance weighting for text classification," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 7241–7248, doi: 10.1609/aaai.v33i01.33017241.

[51] J. Wang et al., "LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in Internet of Things," *Sensors*, vol. 20, no. 9, p. 2451, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/9/2451

[52] K. Zhang, J. Xu, M. R. Min, G. Jiang, K. Pelechrinis, and H. Zhang, "Automated IT system failure prediction: A deep learning approach," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 1291–1300, doi: 10.1109/BigData.2016.7840733.

[53] Y. Zhang, X. Yang, J. Ivy, and M. Chi, "ATTAIN: Attention-based time-aware LSTM networks for disease progression modeling," in *Proc. IJCAI*, 2019, pp. 4369–4375.

[54] J. Zhu et al., "Tools and benchmarks for automated log parsing," in *Proc. 41st Int. Conf. Softw. Eng. Softw. Eng. Pract.*, Montreal, QC, Canada, 2019, pp. 121–130. [Online]. Available: https://doi.org/10.1109/ICSE-SEIP.2019.00021

[55] "Syslog watcher." Accessed: Sep. 5, 2021. [Online]. Available: https://ezfive.com/syslog-watcher/

[56] "The things network." 2015. [Online]. Available: https://www.thethingsnetwork.org/country/australia/

[57] "WisGate edge max, RAK7249." Accessed: Apr. 3, 2022. [Online]. Available: https://store.rakwireless.com/products/rak7249-diy-outdoor-gateway?variant=39881272033478

[58] W. Meng et al., "A semantic-aware representation framework for online log analysis," in *Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2020, pp. 1–7, doi: 10.1109/ICCCN49398.2020.9209707.

[59] D. Sundararaman et al., "Syntax-infused transformer and bert models for machine translation and natural language understanding," 2019, *arXiv:1911.06156*.

[60] I. Sedki, A. Hamou-Lhadj, and O. Ait-Mohamed, "AWSOM-LP: An effective log parsing technique using pattern recognition and frequency analysis," 2021, *arXiv:2110.15473*.

[61] 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 *Proc. 46th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw. (DSN)*, 2016, pp. 654–661.

[62] M. Cinque, R. D. Corte, V. Moscato, and G. Sperlí, "A graph-based approach to detect unexplained sequences in a log," *Expert Syst. Appl.*, vol. 171, Jun. 2021, Art. no. 114556. [Online]. Available: https://doi.org/10.1016/j.eswa.2020.114556

[63] B. Y. Ooi and S. Shirmohammadi, "The potential of IoT for instrumentation and measurement," *IEEE Instrum. Meas. Mag.*, vol. 23, no. 3, pp. 21–26, May 2020, doi: 10.1109/MIM.2020.9082794.

SUSNATA BHATTACHARYA was born in India. He received the B.Tech. degree in electronics and communication engineering from the University of Kalyani, Kalyani, India, in 2000, the master's degree from BITS Pilani, Pilani, India, in 2014, and the M.Phil. degree in knowledge and information sciences from the University of Cape Town, Cape Town, South Africa, in 2018. He is currently pursuing the Ph.D. degree in business and informatics from the Centre for Intelligent Systems, Central Queensland University, Rockhampton, QLD, Australia.

He has extensive experience both in Industry and as well as in Academia and his strong focus is on IT Operations and IT Service Management. He has got extensive project management experience in executing IT infrastructure projects and a natural acumen in technical service delivery. He is passionate about research and his research interests are in the areas of neural language understanding, intelligent information processing, research data management, Internet of Things, artificial intelligence, knowledge management practices, information sciences, and data analytics.

Mr. Bhattacharya is Microsoft and ITIL certified, and has bagged numerous rewards and accolades from various Organizations of International repute.

BIPLOB RAY (Member, IEEE) is currently working as a Senior Lecturer with Central Queensland University, Rockhampton, QLD, Australia, with a background mix of research, academic, and industry experience. He has been working in several Australian federal government and industry-funded research projects since 2016. He has more than 46 book chapter, journal, and conference publications, which has been recognized by peers and cited extensively. He is highly interested in multi-disciplinary research with core interests in secure communication protocols of cyber–physical systems, artificial intelligence, and Internet of Things.

Dr. Ray has received the CQU Vice-Chancellor's Award for Exemplary Practice in Learning and Teaching and Student Voice Commendation in 2019 and 2021. He was also awarded the Deans' Award and the Vice-Chancellor's Award for Outstanding Researchers Commendation in 2019. He has served as a guest editor, editorial board member, and reviewer for reputable journals. He has also been serving as a keynote/plenary speaker, organizing chair, PC member, and reviewer for several conferences since 2012.

RITESH CHUGH (Senior Member, IEEE) received the Ph.D. degree in knowledge management from Victoria University, Footscray, VIC, Australia, in 2014.

He is an Associate Professor of Information and Communication Technologies with Central Queensland University, Rockhampton, QLD, Australia. His research as an Information Systems Socio-Technological Expert focuses on the social role of emerging information systems and their influence on humans and organizations in real-world settings. He takes an interdisciplinary approach to research, which includes information systems management, knowledge management, educational systems, and technology-enhanced learning. He has published widely in these areas, with his research published in several leading academic journals.

Dr. Chugh has received several teaching awards recognizing his teaching excellence, commitment to improved student outcomes, and engagement in reflective learning and teaching activities. He is a Senior Member of the Australian Computer Society.

STEVEN GORDON (Member, IEEE) received the Ph.D. degree in telecommunications from the University of South Australia, Adelaide, SA, Australia, in 2001.

He was a Senior Lecturer with the University of South Australia and an Associate Professor with Thammasat University, Bangkok, Thailand. He is currently a Senior Lecturer with Central Queensland University, Rockhampton, QLD, Australia. His current research interests include formal analysis of protocols, design of IP-based wireless ad hoc, sensor, and vehicular networks, and Internet security and privacy protocols.

Dr. Gordon has served for the editorial board and TPC of various international journals and conferences in networking. He is a member of ACM and IEICE.