IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 January 2024, accepted 16 January 2024, date of publication 25 January 2024, date of current version 8 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3358422

RESEARCH ARTICLE

Enhancing User Experience on Q&A Platforms: Measuring Text Similarity Based on Hybrid CNN-LSTM Model for Efficient Duplicate Question Detection

MUHAMMAD FASEEH[1], MURAD ALI KHAN[2], NAEEM IQBAL[3], (Member, IEEE), FAIZA QAYYUM[2], ASIF MEHMOOD[4], AND JUNGSUK KIM[4,5], (Member, IEEE)

[1]Department of Electronics Engineering, Jeju National University, Jeju-si 63243, Republic of Korea
[2]Department of Computer Engineering, Jeju National University, Jeju-si 63243, Republic of Korea
[3]School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, BT7 1NN Belfast, U.K.
[4]Department of Biomedical Engineering, College of IT Convergence, Gachon University, Sujeong-gu, Seongnam-si 13120, Republic of Korea
[5]Cellico Research and Development Laboratory, Sungnam-si 13449, Republic of Korea

Corresponding authors: Asif Mehmood (asif@gachon.ac.kr) and Jungsuk Kim (jungsuk@gachon.ac.kr)

This work was supported in part by the Ministry of Trade, Industry and Energy and the Korea Institute of Industrial Technology Evaluation and Management (KEIT), in 2023, under Grant 20022793; and in part by Gachon University Research Fund under Grant GCU-202303650001.

ABSTRACT This research introduces an innovative approach for identifying duplicate questions within the Stack Overflow community, a challenging task in NLP. Leveraging deep learning techniques, our proposed methodology combines Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks to capture both local and long-term dependencies in textual data. We employ word embeddings, specifically Google's Word2Vec and GloVe, to enhance text representation. Extensive experiments on the Stack Overflow dataset demonstrate the effectiveness of our approach, achieving an impressive accuracy of 87.09% and a recall rate of 87.%. The integration of CNN and LSTM models significantly streamlines preprocessing, making it a valuable tool for detecting duplicate questions. Future directions include extending the model to multiple languages and exploring alternative word embedding techniques. Our approach presents promising applications beyond Stack Overflow, offering solutions for identifying similar questions on various QA platforms.

INDEX TERMS Duplicate question identification, stack overflow, deep learning (DL), word embeddings, natural language processing (NLP), question-and-answer (QA) platforms.

I. INTRODUCTION

Online Question Answering (QA) platforms have been increasingly popular in our community today, as demonstrated by the dominance of Stack Overflow. Stack Overflow is an online question forum that is a popular platform for individuals seeking answers and support related to computer programming. Stack Overflow was established in 2008, boasts over 14 million registered users, and handles over 10,000 daily questions with a median response time of 11 minutes. It houses an extensive repository comprising 18 million questions, 28 million answers, 75 million comments, and 55 thousand tags [1]. Given the diversity of attributes like Question ID, Title, Body, and Tags, there's potential for users to pose similar queries [2]. Stack Overflow addresses this diligently, ensuring a secure and authentic user environment.

Detecting duplicate questions in QA platforms like Stack Overflow is vital for user convenience and quicker access to pertinent answers. It involves identifying semantic and

The associate editor coordinating the review of this manuscript and approving it for publication was Arianna Dulizia.

lexical similarities between questions to streamline the user experience. Duplicate questions include finding concise queries like ''How does Ruby code use the Unary operator?'' and ''Explain the Ruby Unary Operator.''. Much research was conducted to address this issue through various techniques, including text features, to identify shared content characteristics across experiments, making text similarity a complex yet important concept [3]. The other aspects is Short texts often contain noisy words, fewer standard terms, and informal language, including creative spelling and slang, making traditional NLP techniques designed for longer texts less applicable. For instance, most search queries contain fewer than five words and are limited to 140 characters or less [4]. To detect the duplicate question, it is essential to distinguish between semantic, syntactic, and pragmatic aspects of language and communication. These aspects relate to different layers of meaning and interpretation in natural language [5]. Table 1 highlights these three concepts' differences.

TABLE 1. Distinguishing semantic, syntactic, and pragmatic aspects.

Syntax	Semantic	Pragmatic
Study of words and their combinations in question text to form phrases	Study of words and their meanings in a question text	Study of words and their importance in a question text with concern to their context
Studies the structure of question text to focus on word order arrangement	Focus on the significance of the meaning of words in a literal sense	Focus on the intended and inferred meanings of words according to their context

Categorizing short questions involves addressing distinctive challenges, given their unique characteristics and high demand for text classification. The first steps require context-dependent selection of short question text, with difficulties due to context variability and word co-occurrences. Conventional classification and manual labeling are impractical. Short questions are difficult to manage because automatic coding frequently fails to meet accuracy standards. Challenges include non-standard words, such as misspellings, grammar issues, and jargon, leading to misconceptions. Extracting correct sentiment from short questions proves challenging due to limited-term occurrences, impairing the ability to draw valid conclusions.

DL approaches may successfully extract complex nonlinear features and better consider the text and code semantic information. DL has recently been shown to be useful for several software engineering tasks, such as code clone identification [6], [7], bug report detection [8], [9], and predicting semantically linkable information [10] in questioning and Answering websites. Xu and Yuan [11] used a semantics modeling matching technique. Wang et al. [12] utilized CNN, RNN, and LSTM on six groups of datasets. Zhang et al. [13] use W2V- CNN, W2V-RNN, and W2V-LSTM to suggest solutions to detect duplicate question detection using StackOverflow. Furthermore, in [14], the authors comprehensively reviewed the multi-model fusion approaches for QA datasets. Convolutional Neural Networks (CNN) [9], Recurrent Neural Networks (RNN) [10], Long Short-Term Memory (LSTM) [15], and other DL models have been proposed. These DL models have unique application scenarios; therefore, selecting an appropriate model and balancing effectiveness are essential challenges in successfully applying DL for this type of problem [9], [16]. On the other hand, DL models are not widely used to solve the duplicate question identification problem on Stack Overflow.

In this paper, DL algorithms for automatically detecting duplicate queries on Stack Overflow and assessing their performance. Four different deep-learning techniques are used to find duplicate question detection. These approaches are SVM, Random Forest, LSTM, and CNN combined with LSTM. Our system integrates CNN and LSTM architectures with W2V to address this challenge. Previous studies didn't use the CNN with the combination of LSTM. CNN, RNN, and other approaches are primarily used with Word2vec individually.

The primary focus of this research is to detect duplicate questions to improve user experiences for the Stack Overflow Community. In this research, the aim and Objective include Simulation Efficiency Evaluation and designing an enhanced question Detection Model.

The research questions addressed in this Study are as follows:

1) **Detecting Duplicate Question Pairs in the Stack Overflow Community**: The primary research question focuses on detecting duplicate question pairs within the Stack Overflow Community. This question aims to investigate the current state of duplicate question detection and its significance within this thriving online knowledge-sharing platform.
2) **Enhancing the Efficiency of Existing Duplicate Question Detection Algorithms**: The second research question delves into algorithmic efficiency, seeking methods to improve the performance of existing duplicate question detection algorithms. This inquiry aims to contribute to the optimization of algorithms to expedite the process of identifying the same question pairs and enhance the overall user experience within the Stack Overflow Community.

This research is categorized as Follows: Section II covers the review of related work, and Section III elucidates the proposed research methodology. Section IV is dedicated to presenting the empirical results of the proposed model. Section V presents a discussion of the proposed and overall study. Finally, Section VI summarizes the findings and future directions.

II. RELATED WORK

This Section describes the literature relevant to duplicate questions, DL, and word embedding, respectively.

A. DUPLICATE QUESTIONS

Stack Overflow is one of the most prominent QA sites for software developers. As of October 2019, Stack Overflow had over 18 million questions, 28 million answers, 76 million comments, and 56,000 tags. Each question on Stack Overflow has various attributes, such as ID, title, body, tags, creationDate, closedDate, and so on. Although users are urged to search for the forum before posting a new topic, duplicate questions regularly arise on Stack Overflow. Duplicate questions have already been created and answered on Stack Overflow. Users with high reputations are urged to manually mark duplicate questions in Stack Overflow to limit the number of duplicate queries. If two questions are identical, one will be tagged as duplicate and closed. The second question will be designated as a master(duplicate) question. A pair of duplicate questions also includes master and non-master questions. In the case of two same questions, the older question is referred to as the master question, while the newer question is referred to as the nonmaster question [17].

B. DEEP LEARNING TECHNIQUES

In recent years, DL has become pivotal in NLP tasks. Researchers have used it to identify paraphrases by examining language aspects like unigrams and bigrams [18], [19]. Some have integrated syntactic and linguistic features to measure similarity, while others have applied DL to sentiment analysis [20]. DL is also crucial for spotting duplicated questions, often utilizing CNN [5], [21], RNN, and LSTM models [22]. These models are instrumental in problem-solving and performance enhancement. CNN is particularly effective for sentence-based classification tasks [23], [24]. At the same time, DL proves valuable in software engineering applications like bug report detection, product review detection, fake news detection, and semantic knowledge prediction [25], [26], [27]. On the other hand, RNN excels in representing text features and handling word-level inputs for prediction tasks [28]. Furthermore, in [29], the authors introduced a novel self-training classifier based on CNN architecture for electroencephalography (EEG) recognition. Their approach utilizes CNN and a streamlined network, achieving commendable benchmark results. The CNN model is preferred for sentiment analysis in short text pairs. Additionally, Wang et al. [30] introduced a sentiment analysis model that combines both CNN and RNN, leading to enhanced performance in sentiment analysis tasks.

Agarwal et al. [31] developed a model using CNN and RNN, coupled with fine-grained word similarity matching, to identify paraphrases in short text messages. Similarly, in [32], the authors developed a semi-automatic approach to annotating short text tweets to enhance DL algorithms' training. In [33], the authors developed a multi-labelled k-NN classifier to perform multi-label emotion classification for improving accuracy and computational efficiency. Similarly, in [34], the authors attempted to extract semantic and emotional features to recognise emotions. On Twitter, Xu et al. [35] applied the MULTIP technique to detect paraphrases and model relationships between paraphrased words and sentences. In another study, Wieting and Gimpel [36] utilized a Tree-based LSTM model for short-text modelling. In contrast, Gupta et al. [37] introduced the Gated Recurrent Averaging Network (GRAN) based on AVG and LSTM models to detect short text similarity. Chen. et al. [38] introduced iSTS, an interpretability layer relying on sentence pair alignment for similarity assessment. Their work introduced an attention-based CNN architecture for various NLP tasks, including answer selection, paraphrase identification (PI), and text entailment (TE). Wang et al. [2], on the other hand, presented a novel DL model called BiMPM, which is employed to encode pairs of text. Following the encoding phase, the model evaluates the textual data, calculating a similarity score in the subsequent BiLSTM layer. This score serves as a crucial factor in making the final decision. In a separate investigation, Bonadiman et al. [39] devised a model to rank short text pairs. Once the ranking is complete, a CNN is utilized to compute the similarity matrix. Similarly, in [40], the authors introduced a multi-factor sequential re-ranking framework to enhance the diversity and performance of the given item recommendation by capturing information for the user's short text conversation. Yushi et al. [41] aimed to find similar questions on Quora and Stack Exchange. They used a Siamese gated neural network for Quora and a CNN model for Stack Exchange, outperforming traditional machine learning methods. In contrast, another study [42] introduced Deep Paraphrase, blending CNN and RNN techniques.

In the domain of text classification using CNN, RNN, and LSTM, Wang et al. [12] implemented DQCNN, DQRNN, and DQLSTM models to identify duplicate questions within the Stack Overflow community. These models were applied across six distinct datasets associated with questions tagged under C++, Java, Python, HTML, Object C, and Ruby. Notably, these approaches demonstrated superior performance in terms of recall rate when compared to earlier machine learning methodologies. Furthermore, in [43], the authors employed DL models for classifying peer feedback messages to explore the relationship between student burnout and peer feedback. Hoogeveen et al. [44] addressed the issue of wrongly flagged duplicate questions by utilizing user behaviour-based metadata and reputation. They also employed various text-matching models, such as DSSM, CDSSM, ARCII, Conv-KNRM [45], adaptive word matching [46], alternative re-ranking strategy (ARS) [47], dual-level representation enhancement network (DREN) [48] and RI-Match [49].

In a related study, Liting Wang and team [2] further improved this work by capturing word-level semantic information. They integrated Word2Vec (W2V) for word embedding and introduced three techniques: W2V-CNN, W2V-RNN, and W2V-LSTM, based on CNN, RNN, and LSTM architectures. These models were evaluated on a diverse programming-related question dataset, demonstrating

TABLE 2. Overview of deep learning approaches.

References	Methodology	Dataset
Huang et al. [51]	Pre-Trained word embedding by using Perceptron of Multi-Layer	Twitter
Agarwal et al. [31]	Pre-Trained word embedding by combining DL, statistical features	MSPR.
K. Dey et al. [52]	Set of Lexical, Syntactic, Semantic, and Pragmatic Features using resources Word Net, POS Tagger	Twitter, MSPR.
R. Ferreira et al. [53]	Represent pair of sentences as a combination of similarity measures using a dependency parser	MSPR.
D. Liang et al. [17]	Pre-trained Glove vectors on Wikipedia using Gated Recurrent Network	Quora.
Yushi Homma et al. [41]	Pre-trained 300 D. Glove vectors on Wikipedia using Siamese Gated Recurrent Model	Quora.
Wei Bao et al. [54]	Measuring Semantic Textual Similarity by using Attentive Siamese LSTM	SamEval 2014
ZHUOJIA XU et al. [11]	Semantic Matching Model merged framework of multi-tasking transfer learning for multi-domain forum duplicate detection of questions.	Stack Overflow
Liting Wang et al. [12]	Duplicate Questions Detection Task exploring Deep Learning approaches CNN, RNN, and LSTM on six different groups of programming-related questions dataset.	Stack Overflow
ZHUOJIA XU. Et al. [13]	Capturing semantic information at word level and for word, embedding learning W2V is used and proposed three techniques DL W2V- CNN, W2V-RNN, and W2V-LSTM on six different groups of programming related questions dataset.	Stack Overflow
Proposed Model	Google Pretrained W2V is used and proposed DL models CNN combining with LSTM for finding Duplicate Questions	Stack Overflow

superior accuracy and recall rates compared to existing methods. Furthermore, in [50], the authors proposed a novel framework called Cross-Modal Causal RelatIonal Reasoning (CMCIR) to enhance the visual question answering by capturing information between visual and linguistic semantics. Table 2 mentions a list of techniques with the utilized dataset.

Besides DL techniques, Semantic Text Similarity also played an important role in text similarity detection. Semantic Text Similarity aims to identify semantic equivalences in text fragments, like documents or paragraphs, as discussed by F. Benedetti and colleagues [55]. Initially, research focused on document-level similarity, comparing lengthy texts.

Below Table 3 shows the overview of similarity-based Approaches.

TABLE 3. Overview of similarity-based approaches.

Approach	Reference	Methodology
Vector Space Approaches	S. Boffa et al. [56], 2018	Measuring Semantic similarity among tweets pairs.
	R. Mudgal et al. [57], 2018	Measuring semantic similarity between tweets by using trigram approach.
Alignment approaches	M. Al-Samadi et al. [58] 2017	Supervised machine learning approach to find semantic similarity.
	G. Majumder et al. [59], 2021	Textual Similarity approach based on pair of sentences and for training process support vector regression is used.
Other Approach	B. Agarwal et al. [31], 2018	SVM for detecting paraphrases and semantic similarity between short text.

C. WORD EMBEDDING

Compared to other embedding techniques, W2V embedding stands out for its ability to capture semantic links between words by representing them in a continuous vector space. W2V, instead of classic methods such as TF-IDF or Bag-of-Words, provides a more nuanced understanding of context and word meaning [13]. DL W2V with CNN, W2V with RNN, and W2V with LSTM, in particular, demonstrate W2V's prowess in capturing deep semantic distinctions at the word level, particularly in the domain of programming-related topics. One significant advantage of adopting Google's W2V is that it has pre-trained models on large and diverse datasets, giving it a considerable head start on many NLP applications. This pre-trained information enables successful transfer learning while conserving computational resources [2].

In this work, we used CNN and LSTM with W2V embedding. Most studies related to sentiment analysis lack the focus on the StackOverflow dataset. Most studies didn't use more than six groups from the StackOverflow dataset. In contrast, no one used a hybrid model (CNN and LSTM combined) with the Word2vec to analyze the work. Previously, only one model with any embedding was used to analyze the performance. In our study, we use the whole dataset rather than considering only a few groups to explore the impact of our model. Overall results show the excellent performance of our proposed model.

III. PROPOSED RESEARCH METHODOLOGY

Within this chapter, we introduce our suggested approach for identifying replicated questions. The proposed framework incorporates neural network models for our analysis. We will delve into the structure of the model and provide a comprehensive overview of each component. This Section outlines the suggested approach for identifying a semantic resemblance between questions. We will elucidate the data preprocessing procedures in the initial phase. Subsequently, we will illustrate the application of word embeddings in the second phase. Moving on to the third phase, we will

incorporate various neural network architectures into our model. Finally, in the concluding phase, we will introduce the ultimate model employed for discerning semantic similarity between two questions.

The suggested approach for detecting duplicated questions is introduced in this chapter. For analysis, the proposed framework combines neural network models. This Section outlines the recommended method for identifying semantic resemblance. The process begins with the preprocessing step, and then word embedding will be used. The model is then introduced to several neural network topologies in the third phase. Finally, the ultimate model used to determine semantic similarity between two queries is presented in the final step.

A. CONVENTIONAL METHOD

The Objective for the provided input queries is to compute a correlation score that signifies the resemblance between them in terms of semantics. While many conventional techniques depend on lexical word matching, the limitations of these methods have been acknowledged [60]. Existing approaches, although capable of extracting textual attributes for the automatic identification of duplicate questions, have their constraints. As mentioned earlier, these approaches can potentially lose vital semantic information. Consequently, various alternative methods have been used to assess the semantic similarity between the two questions. However, there remains a need for further enhancements in accuracy.

$$x(q_1, q_2) \rightarrow 0 \tag{1}$$

Within the equation 1, the value '1' signifies the equivalence of q1 and q2, whereas '0' indicates their dissimilarity [12], [52]. Over the past few years, DL has emerged as a pivotal factor, yielding promising outcomes across various domains. As evidenced by previous works, many DL-based techniques have proven immensely beneficial for researchers, particularly in duplicate question detection.

B. PROPOSED MODEL

DL methods like CNN and RNN have been employed in various text-mining applications for an extended period. These deep neural networks play a pivotal role in simplifying text mining and the extraction of features. They allow for the creation of low-level text representations, obviating the need for conventional feature engineering techniques. This approach also reduces human intervention in the feature engineering process, ensuring the preservation of original data, which can be harnessed for training purposes. In the training phase of deep neural networks, features are acquired automatically without the requirement of domain-specific knowledge, rendering all processes independent of such expertise.

1) DATA PREPROCESSING

In Fig. 1, a comprehensive text preprocessing pipeline is designed to enhance the quality and relevance of textual data for subsequent analysis. The process begins with removing punctuation marks to eliminate noise and facilitate a cleaner dataset. Subsequently, all text is transformed into lowercase to ensure consistency in text analysis and prevent case-based discrepancies. Stop words, often detracting from the informative content, are eliminated from the text to reduce dimensionality. Employing stemming, we further reduce the data complexity by converting words to their base forms, preserving the essential meaning while enhancing computational efficiency.

Tokenization is the next critical step in breaking down the text into individual units, such as words or phrases, facilitating downstream processing. These tokenized sequences are then subjected to padding to ensure uniformity in sequence length, which is particularly important for specific machine-learning models. To enrich the text data, we incorporate Word2Vec embeddings, leveraging the semantic relationships between words to enhance the contextual understanding of the content.

This comprehensive preprocessing pipeline not only improves the quality of the text data but also prepares it for advanced analysis techniques, ultimately contributing to more accurate and insightful results in various NLP applications.

2) RECURRENT NEURAL NETWORK

Recurrent Neural Networks (RNNs) find application in scenarios where repetitive events unfold [61]. Expressly, Long Short-Term Memory networks (LSTM) represent a distinct category within RNNs. LSTMs are particularly adept at handling input sequences that encompass various components. In our specific context, these components pertain to words carrying substantial information. Using DNN has proven to be a highly effective approach in various machine learning tasks, especially those dealing with extensive text datasets [62]. DL techniques excel in feature extraction by uncovering intricate relationships among words and phrases, creating a condensed feature space that encapsulates meaningful information representations. In this undertaking, we consider Google's W2V an exemplary solution. W2V empowers semantic mining on extensive datasets by using mean computations [63]. It is worth noting that W2V handles out-of-vocabulary words by constructing vector representations based on words found within the established vocabulary. Among the network types closely associated with LSTM networks are CNN and traditional RNN.

C. LONG SHORT-TERM MEMORY

The architecture of a DL network comprises a distinct network category, particularly well-suited for showcasing structural phenomena. For an input sequence denoted as $x = x_1, x_2, \ldots .xT$ where the network receives input vectors $xt \in \mathbb{R}n$ and the hidden state vector $h_{t-1} \in \mathbb{R}m$ it subsequently generates the subsequent hidden state ht

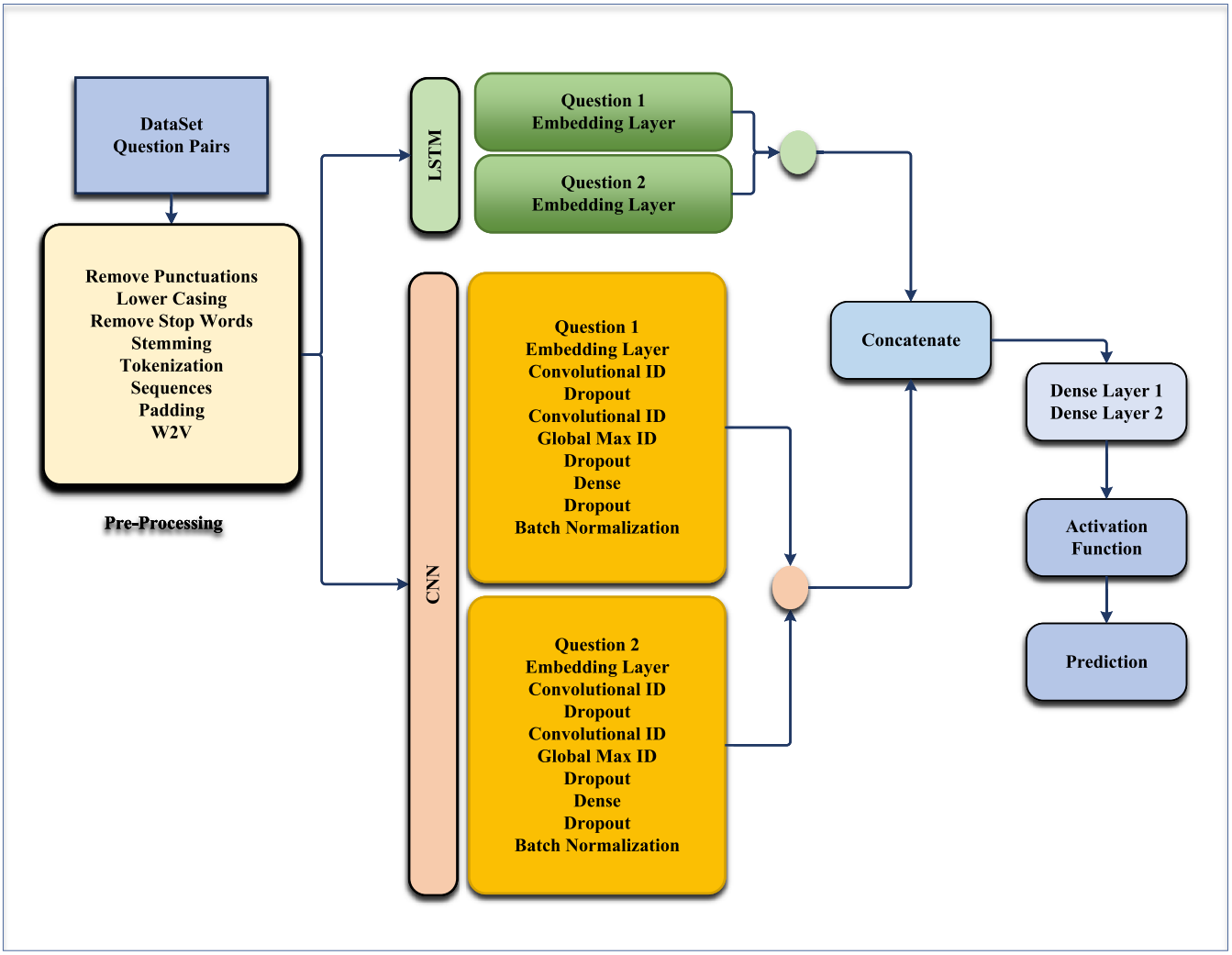


FIGURE 1. Proposed framework for finding similarity.

through the application of the subsequent recursive operation:

$$h_t = f(Wx_t + Uh_{t-1} + b) \tag{2}$$

Here, let $W \in \mathbb{R}^{m\times n}$, $U \in \mathbb{R}^m$, and $b \in \mathbb{R}^m$ represent the components of an affine transformation, while the function involving element-wise non-linearity is symbolized as f. The LSTM addresses the challenge of capturing long-range dependencies by expanding the memory cell vector at each time step within the RNN. An LSTM unit is constructed using a set of three multiplicative gates that regulate the information to retain and pass forward to the subsequent time step, denoted as t. Concretely, at each stage of the LSTM, it takes as input $x_t$it takes as input x_t, h_{t-1}, c_{t-1}, and producesh_t and c_t through the following intermediate calculations.

$$i_t = \sigma(W_i X_t + U_i h_{t-1} + b_i) \tag{3}$$
$$f_t = \sigma(W_f h_{t-1} + U_f X_t + b_f) \tag{4}$$
$$o_t = \sigma(W_o h_{t-1} + U_o X_t + b_o) \tag{5}$$
$$g_t = \tanh(W_g h_{t-1} + U_g X_t + b_g) \tag{6}$$
$$c_t = ft\theta(C_{t-1} + i_t\theta_{gt} + b_o) \tag{7}$$
$$h_t = O_t\theta tanh(c_t) \tag{8}$$

1) CONVOLUTIONAL NEURAL NETWORK

The Convolutional Neural Network, often called convents, has emerged as a significant and captivating advancement in ML over recent years. In essence, a convolutional network bears resemblance to multilayer perceptron systems but distinguishes itself through the utilization of convolutional layers. These layers establish sparse connections, linking neurons in subsequent layers to specific regions within the current input volume rather than forming relationships with every preceding neuron. Consequently, convolutional layers exhibit a notable reduction in connections compared to fully connected layers, resulting in heightened computational efficiency while maintaining a comparable number of neurons. The pivotal component of generating feature maps from these layers involves the deployment of the convolutional operator,

followed by applying nonlinear activation functions, all of which are derived from the initial layer's input space as shown in Fig. 2.

2) MAX POOLING LAYERS

Convolutional networks often employ max-pooling layers, which serve as a form of nonlinear downsampling. Within max pooling, the length of conventional layers is reduced, forming rectangular segments from subdivided outputs. For each filter, the most optimal value is selected. This approach results in greater spatial freedom for features. Max-pooling layers can be strategically added following each convolutional layer or at intervals within the network architecture.

3) DROPOUT LAYERS

In the training phase, the Dropout Layer method randomly excludes neurons, causing them to be omitted. In essence, during the activation of downstream neurons, their involvement is temporarily suspended during the forward pass, and during the backward pass, no weight updates occur. It's worth noting that this approach finds application in both Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN).

4) BATCH NORMALIZATION

In DL, another method frequently employed is known as batch normalization. It addresses the issue of shifting by standardizing each layer individually. Regarding the input layer, we aim to accelerate learning by modifying activation functions through normalization. Throughout our training procedure, batch normalization accomplishes the following tasks:

1) Computing the average and variance of input layers.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{9}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{10}$$

2) Standardizing input layers based on precomputed statistical values.

$$x_1 = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{11}$$

3) Adjusting and rescaling to acquire the layer's output.

$$y_i = \sqrt{x_i} + \beta \tag{12}$$

5) BUILDING A MODEL

The subsequent step involves constructing a prediction model. CNN extracts features from natural language data, which can be localized or deep. Extensive research has demonstrated that CNNs yield promising results in text classification tasks. RNN, like LSTM, can handle long-term dependencies. In this study, we opted for a combination of Convolutional Neural Network 1D, incorporation of dropout, dense layers, and max pooling, all in conjunction with the LSTM network. Within our model architecture, each question comprises a single translational layer. LSTM and CNN layers are initialized with word embeddings for the first two layers. Input tensors from all layers are concatenated and processed through dense layers. The final predictions are derived after passing through an activation function. Subsequently, precision, recall, accuracy, and F1 scores are computed based on these predictions.

IV. RESULTS

In this Section, we have assessed the performance of our proposed methodology, which relies on a deep neural network. Our primary objective was to identify duplicate questions within the Stack Overflow community. Through extensive analysis and experimentation, we diligently evaluated various approaches to achieve optimal results in detecting duplicate questions on the Stack Overflow platform. Our investigation was conducted on the Stack Overflow dataset, where duplicate questions were assigned a label of 1, while master questions were labelled as 0. Initially, we trained our model using DL techniques to discern similarities among questions within the Stack Overflow dataset, carefully fine-tuning the parameters to ensure optimal performance. We then tested our model on a separate testing dataset to compute the evaluation metrics.

A. DATASET

Our research project is designed to identify duplicate questions, and for this purpose, we have utilized the Stack Overflow dataset. In 2019, Stack Overflow made this dataset publicly available. Within this dataset are titles of Java Programming Language questions, some of which may be duplicates, while others are not. In this context, "duplicate" refers to questions that convey the same meaning as a master question. Each question in the dataset is assigned a unique ID, and duplicate question pairs are denoted by 1, whereas master question pairs are represented by 0. The initial rows of the dataset are presented in Fig. 3. The dataset encompasses 416,860 questions, resulting in 208,430 question pairs.

B. LABEL DISTRIBUTION

The Stack Overflow Dataset comprises pairs of question titles, and you can find the distribution of labels for this dataset in Fig. 4.

In this Section, we detail the experimental configuration employed in our model. We have used machine learning methodologies on the question dataset to pursue optimal outcomes. Initially, we gathered features including common word occurrences, question length, cosine similarity, and others. Following the feature collection, we apply machine learning techniques for analysis, specifically Support Vector Machines (SVM) and Random Forest. It's worth noting that the Random Forest model demonstrates a superior accuracy

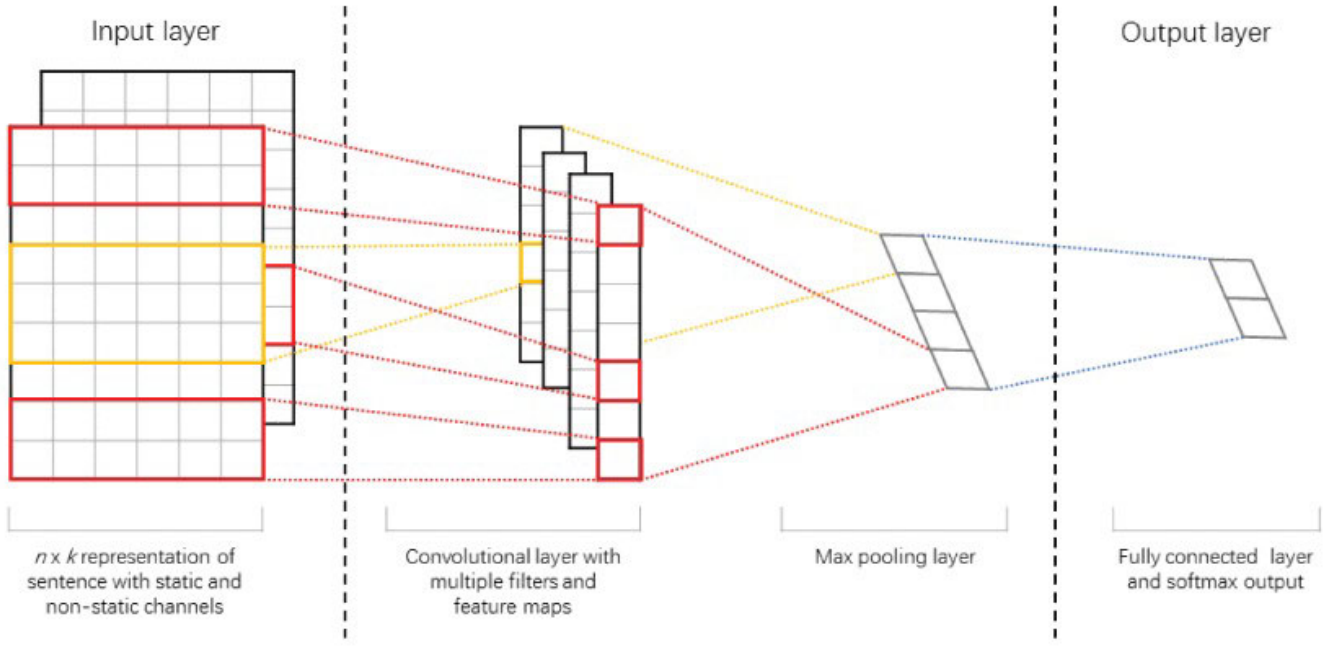


FIGURE 2. CNN architecture [2].

id	qid1	qid2	question1	question2	is_duplicate
0	1	2	use config property file variable	use property property file property file java	0
1	3	4	way use list string parameter regular expression g	java regex capture group x	0
2	5	6	overlap two date range	unable get total decrease	0
3	7	8	stringlength 2 split string	convert dot separate string array list	1
4	9	10	java 8 stream join return multiple value	new line write file	0
5	11	12	pointcut match method annotate directly inherit	spring aop annotation annotation	0
6	13	14	extract value use json path array	limit character command line argument spar	0
7	15	16	call method class inside another class	call object within object	0
8	17	18	insert blob oracle database 11g use vertx	error fetch blob data use jdbc	0
9	19	20	maven nexus find dependent project	sonatype maven repo search project depend	1
10	21	22	write file java	write console output txt file	1
11	23	24	run java web application ii	host java web project ii	1
12	25	26	android javalangnullpointerexception settext sin	label array image icon null pointer error	1
13	27	28	rmi null pointer	null pointer exception use split	1

FIGURE 3. First few rows of stack overflow dataset.

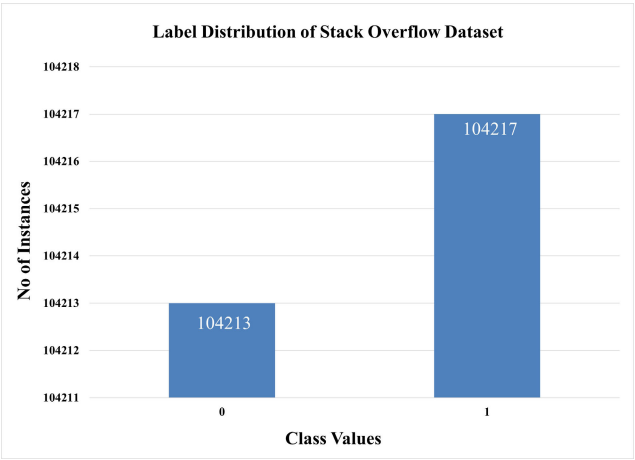


FIGURE 4. Label distribution of stack overflow dataset.

score compared to the SVM model, as evidenced in Table 4. Additionally, with the BERT and XLNET, we explore DL models and ultimately conclude that the proposed model

TABLE 4. Models applied on dataset.

Model Name	Technique	Accuracy
SVM	Measures Features and Distance	63.29
Random Forest	Measures Features and Distance	71.46
LSTM	Pretrained Google W2V	79.72
LSTM	LSTM with Glove	50.19
BERT	BERT With Pretrained corpora	73.16
XLNET	Combination of GPT and BERT	77.91
CNN, LSTM	Pretrained Google W2V	87.09

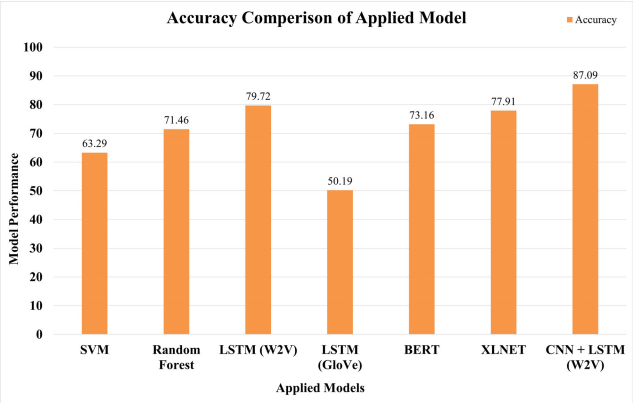


FIGURE 5. Accuracy comparison of all techniques applied on stackoverflow dataset.

yields the highest accuracy and loss Loss, as shown in Fig. 5 and Fig. 6, respectively.

1) EXPERIMENTAL SETUP

In this Section, we will verify the outcomes of our experiments using the Stack Overflow dataset. We have

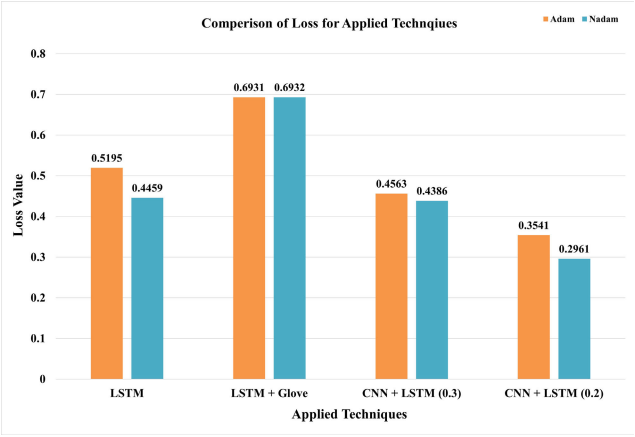


FIGURE 6. Loss comparison of all techniques applied on stackoverflow dataset.

TABLE 5. Hyperparameters and their values for Proposed model.

Parameter	Value
Max Sequence Length	25
Batch size	2,048
Maximum Number Words	1,50,000
Epochs	25
Monitor	Validation Accuracy
Loss	Binary cross entropy
Optimizer	nAdam
Activation unit	ReLU -Rectified linear units
Dropout	0.2
LR patience	5

employed various deep-learning models to derive the current model. The proposed model amalgamates features from both LSTM and CNN 1D architectures. Within our model, each question is equipped with a single translational layer. Word embeddings initialize both LSTM and CNN layers. The input tensors from all these layers are concatenated and undergo a sequence of dense layers before passing through an activation function to produce final predictions.

2) PARAMETER SETTINGS

Within this Section dedicated to our experiment, Table 5 displays the optimization of hyperparameters. This optimization begins with thoroughly analyzing the training data to determine the most suitable hyperparameters. We meticulously chose the optimal combination of the optimizer, dropout rate, and activation unit to enhance the performance of our approach. It is important to note that parameter configurations are intricately linked to the specific dataset under consideration. In the case of our dataset, we conducted a comprehensive evaluation of various parameter settings.

In the architectural design of our model, we have employed Rectified Linear Units (ReLU) as the activation function. This choice aligns with the prevailing practice in DL models. We incorporated diverse optimizers during training to ensure the best model performance. In particular, we leveraged the nAdam optimizer while simultaneously implementing a dropout layer with a rate of 0.2. This measure mitigates the risk of overfitting and results in heightened accuracy when applied to our Stack Overflow community dataset.

C. IMPLEMENTATION

We introduced an innovative method for gauging the semantic affinity between sentences. This unique approach was executed using KerStudy and the TensorFlow backend. We employed a 25-epoch training regimen to train the model, incorporating a patience value set at 5. We partitioned the dataset into an 80/20 split to facilitate training and testing. Following each epoch, we assessed the model's performance and preserved favorable results. The optimal model score, as determined on the validation set, was subsequently employed for making predictions on the testing data.

D. EVALUATION MATRICES

Here, we present the evaluation metrics for our study:
True Positives (TP): These are the instances where duplicate pairs are correctly identified as duplicates.
True Negatives (TN): These represent master pairs correctly identified as such.
False Positives (FP): occurs when master pairs are incorrectly classified as duplicate pairs.
False Negatives (FN): These are cases where duplicate pairs are mistakenly classified as master pairs.

To gauge accuracy, we utilize Equation 16, which involves adding the count of true positive and true negative samples and dividing it by the total number of samples. Precision, as defined in Equation 13, measures the proportion of predicted positive cases relative to actual positives. Meanwhile, recall, as per Equation 14, calculates the rate of correctly predicted actual positive cases. Lastly, the F-measure, computed using Equation 15, is determined as the harmonic mean of precision and recall.

Our proposed model's performance evaluation is based on these evaluation metrics.

Precision: Precision is defined as the ratio of correctly predicted positive cases to the total predicted

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

Recall: Recall represents the proportion of true positive cases that are accurately predicted.

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

F-Measure: The F-Measure is a harmonic mean of precision and recall, providing a balanced measure of both.

$$F1\ Score = 2.\frac{Precision * Recall}{Precision + Recall} \tag{15}$$

Accuracy: Accuracy is the ratio of correctly predicted true positive and true negative samples to the total number

TABLE 6. Hyperparameters and results of LSTM using sigmoid activation function.

Optimizer	Dropout layer	Accuracy	Loss
Adam	0.2	0.7471	0.5195
Nadam	0.2	0.7972	0.4459

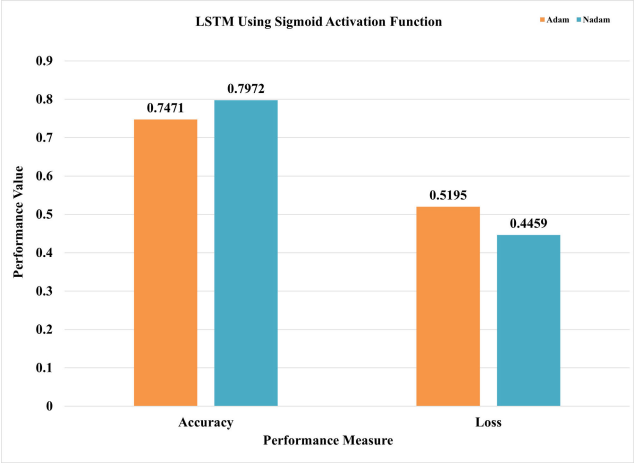


FIGURE 7. LSTM using sigmoid performance measures.

of samples.

$$Accuracy = \frac{TP + TN}{TP + FP + TF + TN} \tag{16}$$

E. RESULTS

This Section showcases and contrasts our outcomes with various established methods. Our innovative methodology is valuable for gauging the similarity between concise sentences or question pairings. Using the Stack Overflow dataset, we comprehensively display the experimental findings through tables and diagrams. To optimize model performance, a series of experiments were carried out employing various optimizers and incorporating dropout layers set at 0.2 and 0.3, which yielded improved results.

1) LSTM

The hyperparameters for our proposed approach are determined through a thorough examination of the training data, aiming to identify the most suitable hyperparameters. We carefully selected the optimizer, activation unit, and dropout settings that yielded the best performance for our LSTM model. Table 6 provides an overview of the chosen hyperparameters and their respective values configured for the LSTM model. During the application of LSTM, we have employed the Sigmoid activation function. We have also utilized the Adam and Nadam optimizers and a dropout layer set at 0.2. The outcomes of these optimizations are depicted in Fig. 7.

2) GLOVE + LSTM

Establishing the hyperparameters and their respective configurations for the Glove + LSTM model is crucial. Table 7 provides an overview of these hyperparameters and their values designated for the LSTM model. It's worth noting that the algorithm's performance falls short compared to a standalone LSTM model.

TABLE 7. Hyperparameters and results of Glove+LSTM using tanh activation function.

Optimizer	Dropout layer	Accuracy	Loss
Adam	0.2	0.5019	0.6931
Nadam	0.2	0.4996	0.6932

For the implementation of Glove + LSTM, we opted for the tanh activation function, coupled with the utilization of both the Adam and Nadam optimizers, while incorporating a dropout layer set to 0.2. The outcomes of employing the Adam and Nadam optimizers alongside the 0.2 dropout layer can be observed in Fig. 8.

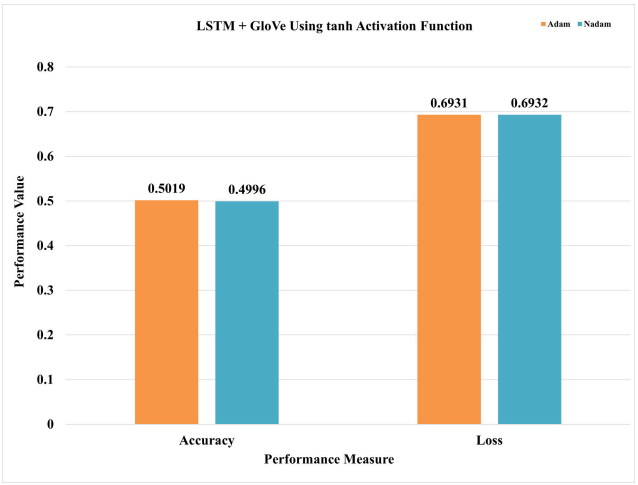


FIGURE 8. LSTM + glove performance measures.

3) BERT AND XLNET

The first step in establishing BERT and XLNet for duplicate question detection is using the existing Stack Overflow dataset. The dataset is then preprocessed to tokenize and encode the questions correctly. BERT and XLNet, being pre-trained transformer-based models, require fine-tuning on the given task. A binary classification system is used for duplicate question detection, with models trained to distinguish between duplicate and non-duplicate question pairs. During training, the models' weights are modified based on the labels. The refined BERT and XLNet models are then tested on a separate validation set to determine their performance. To acquire the best outcomes for each model, fine-tuning factors such as learning rate and batch size were adjusted. This setup process, which combines the capabilities of BERT and XLNet with the Stack Overflow dataset, makes it easier to create effective models for spotting duplicate questions. The presented accuracy scores indicate the performance of two transformer-based models, BERT and

TABLE 8. BERT and XLNET result analysis.

Model	Accuracy	Precision	Recall	F1 Score
BERT	0.7316	0.7404	0.7692	0.7801
XLNET	0.7791	0.7795	0.7912	0.7903

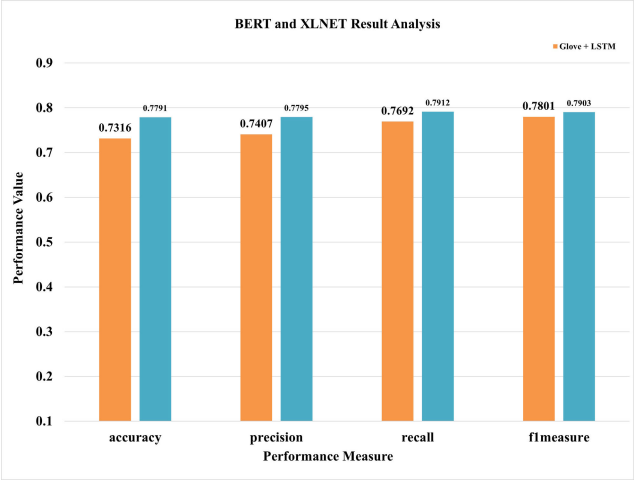


FIGURE 9. BERT and XLNET comparative analysis.

TABLE 9. Hyperparameters and results of CNN+LSTM using ReLU activation function.

Optimizer	Dropout layer	Accuracy	Precision	Recall	F1-Measure
Adam	0.3	0.7898	0.7441	0.8156	0.7669
Nadam	0.3	0.8474	0.8541	0.8384	0.846

XLNet, on a specific task as shown in Table 8. The accuracy values suggest that XLNet outperforms BERT in terms of accuracy, achieving an accuracy of 77.91% with an F1 score of 0.7903, whereas BERT achieves a slightly lower accuracy of 73.16% and an F1 score of 0.7861. These accuracy scores indicate the models' abilities to correctly classify instances, likely in a binary classification setting such as duplicate question detection.

4) CNN + LSTM

Configuring hyperparameters and their respective values to optimize performance and harnessing the capabilities of two DL models, namely CNN and LSTM, are the primary focus. The CNN DL model is employed to extract local features, enabling sentence-level analysis, while the LSTM model aids in capturing long-term dependencies within the data. The outcomes of these experiments are presented in Table 9 below:

In implementing the proposed CNN-LSTM hybrid model, the activation function ReLU is utilized. We have also employed the Adam and Nadam optimizers alongside a dropout layer with a rate of 0.3. For a visual representation of the results achieved with the Adam and Nadam optimizers and the dropout layer set at 0.3, please refer to Fig. 10.

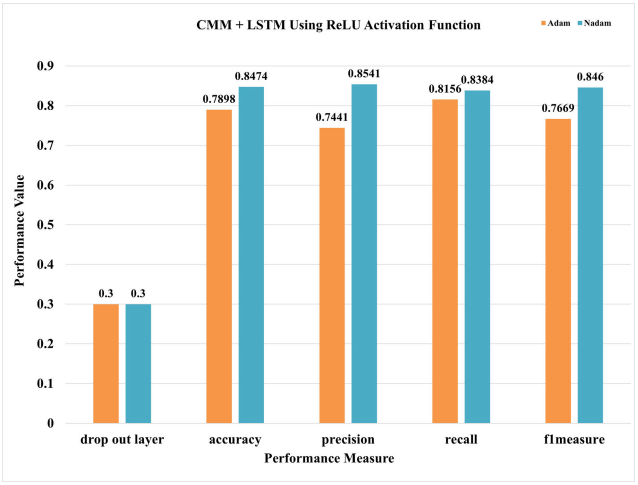


FIGURE 10. CNN + LSTM using ReLU Activation Function.

TABLE 10. Hyperparameters and results of CNN+LSTM with 0.2 dropout layer using ReLU activation function.

Optimizer	Dropout layer	Accuracy	Precision	Recall	F1-Measure
Adam	0.2	0.8354	0.9173	0.8158	0.7861
Nadam	0.2	0.8709	0.872	0.872	0.871

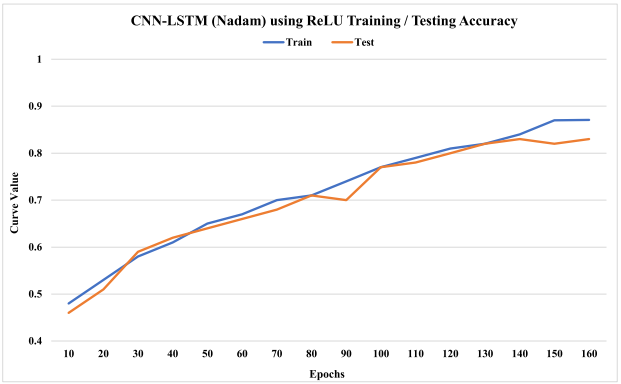


FIGURE 11. CNN + LSTM Accuracy with dropout layer 0.2.

We incorporated a dropout layer of 0.2 into our CNN-based model, which exhibited superior performance when combined with LSTM compared to earlier models. The hyperparameters specified for our model, as presented in Table 10, indicate a notable enhancement in detecting duplicate questions.

The CNN-LSTM fusion, with a dropout layer set at 0.2, achieved an accuracy rate of 87.09% as shown in Fig. 11, along with a recall rate of 87.20%, surpassing the performance of prior methodologies as shown in Fig. 12. Further, to strengthen the proposed model performance, Fig. 13 shows the loss, which is the best performance during the training and validation process among all other applied techniques.

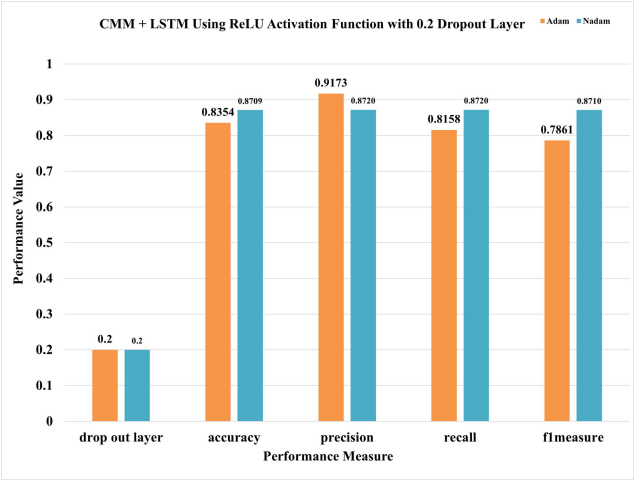


FIGURE 12. CNN + LSTM Performance measures with dropout layer 0.2.

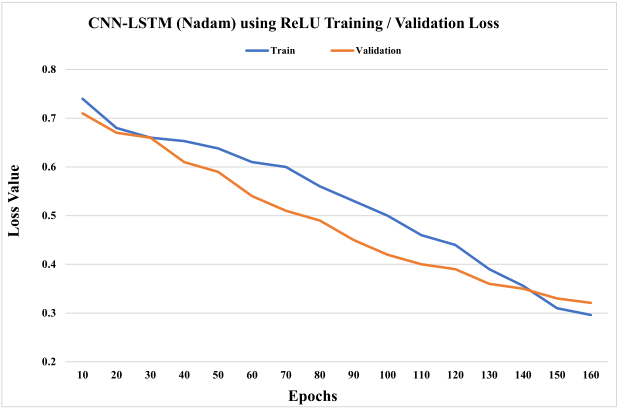


FIGURE 13. CNN + LSTM Loss with dropout layer 0.2.

TABLE 11. Comparison analysis of results.

Technique	Accuracy	Precision	Recall	F1-Measure
J Huang et al. [51]	79.20	74.80	69.40	72.00
Ferreira et al. [53]	75.13	82.00	80.20	83.10
Li Zhang et al. [13]	-	-	79.52	-
Litang Wang et al. [2]	-	-	82.20	-
Our Approach	**87.09**	**87.20**	**87.20**	**87.10**

This improvement in accuracy is evident when considering the results in Table 11, which depict the outcomes of our experiments compared to alternative approaches. Our proposed model consistently outperformed previous models, demonstrating its effectiveness in this context.

The CNN and LSTM with Word2Vec show high performance, compared to BERT and XLNet, in Duplicate Question Prediction using the Stack Overflow dataset can be attributed to several technical factors. Firstly, the CNN and LSTM architecture combines CNN for local feature extraction and LSTM networks for capturing sequential dependencies. This hybrid architecture is particularly effective in modelling textual data's complex patterns and context, allowing it to excel in tasks such as sentiment analysis and duplicate question detection. Additionally, W2V embeddings contribute to the model's success by capturing semantic relationships between words, providing a rich and contextually meaningful representation of the input text. In contrast, while BERT and XLNet are powerful transformer-based models, their sheer complexity and pre-training on diverse tasks may not align optimally with the specific characteristics of the Stack Overflow dataset for sentiment analysis and duplicate question prediction. The proposed CNN and LSTM with the W2V method might better exploit the intrinsic structure of the dataset, enabling it to outperform BERT and XLNet in this particular task.

V. DISCUSSION

In our proposed study, we utilized an advanced technique to do sentiment analysis by combining CNN and LSTM networks with W2V embedding. One significant void in the current body of literature concerns the relatively little consideration devoted to the StackOverflow dataset regarding sentiment analysis research. As many studies as there have been, one persistent drawback is the inclination to focus on a small portion of the StackOverflow dataset—typically no more than six pre-established theme groups. On the other hand, our research took a different approach by presenting a hybrid model that uses W2V embedding to integrate the advantages of both CNN and LSTM synergistically. No prior examination of the StackOverflow dataset has looked into this innovative method.

More importantly, our research approach differs from the standard practice, in which studies use one model with different embeddings to assess performance. Unlike the trend of focusing only on particular subject categories, our analysis goes beyond this traditional paradigm by analyzing the full StackOverflow dataset. We can identify subtle patterns and trends in all types of user-generated material, which provides a complete picture of the sentiment analysis within the StackOverflow community. Our efforts have culminated in the firm and excellent performance that our hybrid model has to provide, as demonstrated by the definitive findings from the thorough examination of the complete dataset.

VI. CONCLUSION AND FUTURE DIRECTIONS

Addressing the issue of identifying duplicate questions within the Stack Overflow community presents a significant challenge. Our proposed approach introduces a deep-learning method tailored to detect duplicate question pairs within this platform. We have effectively leveraged CNN and LSTM networks to capture and store long-term dependencies. This integration of CNN and LSTM DL models has proven to yield superior accuracy and recall rates compared to existing methodologies. Our technique adopts GloVe W2V for text representation, further enhancing its performance compared to prior models. In our experiments conducted on the Stack Overflow dataset, our proposed model achieved an impressive accuracy of 87.09% alongside a recall rate of 87.20%. The primary advantage of employing DL in

this context is its ability to streamline the preprocessing requirements, as it solely relies on the input of question text, saving time and human effort. Creating user-friendly interfaces and smoothly integrating the trained model into web-based applications will be the main goals of future work on duplicate question identification using the Stackoverflow dataset. Giving users immediate feedback on how closely their questions match already-posted questions will help expedite the resolution process. Scalability and speed optimization—including investigating sophisticated hashing methods for quick retrieval—will be prioritized to manage growing user traffic efficiently. The system will stay flexible and proficient in recognizing changing language patterns provided it receives regular updates, adapts to dynamic content through ongoing model training, and incorporates multimodal techniques. With the support of a robust user feedback system, this all-encompassing approach presents the duplicate question detection system as an invaluable instrument for improving user experiences, reducing redundancy, and enabling effective information retrieval in the context of online platforms.

CONFLICT

There is no conflict regarding the publication of this article.

REFERENCES

[1] M. Yazdaninia, D. Lo, and A. Sami, "Characterization and prediction of questions without accepted answers on stack overflow," in *Proc. IEEE/ACM 29th Int. Conf. Program Comprehension (ICPC)*, May 2021, pp. 59–70.

[2] L. Wang, L. Zhang, and J. Jiang, "Duplicate question detection with deep learning in stack overflow," *IEEE Access*, vol. 8, pp. 25964–25975, 2020.

[3] X. Ye and S. Manoharan, "Marking essays automatically," in *Proc. 4th Int. Conf. E-Educ., E-Bus. E-Technol.*, Jun. 2020, pp. 56–60.

[4] I. Alsmadi and G. K. Hoon, "Term weighting scheme for short-text classification: Twitter corpuses," *Neural Comput. Appl.*, vol. 31, no. 8, pp. 3819–3831, Aug. 2019.

[5] D. A. Prabowo and G. Budi Herwanto, "Duplicate question detection in question answer website using convolutional neural network," in *Proc. 5th Int. Conf. Sci. Technol. (ICST)*, vol. 1, Jul. 2019, pp. 1–6.

[6] M. White, M. Tufano, C. Vendome, and D. Poshyvanyk, "Deep learning code fragments for code clone detection," in *Proc. 31st IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Sep. 2016, pp. 87–98.

[7] Q. Xie, Z. Wen, J. Zhu, C. Gao, and Z. Zheng, "Detecting duplicate bug reports with convolutional neural networks," in *Proc. 25th Asia–Pacific Softw. Eng. Conf. (APSEC)*, Dec. 2018, pp. 416–425.

[8] B. Xu, D. Ye, Z. Xing, X. Xia, G. Chen, and S. Li, "Predicting semantically linkable knowledge in developer online forums via convolutional neural network," in *Proc. 31st IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Sep. 2016, pp. 51–62.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012.

[10] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[11] Z. Xu and H. Yuan, "Forum duplicate question detection by domain adaptive semantic matching," *IEEE Access*, vol. 8, pp. 56029–56038, 2020.

[12] L. Wang, L. Zhang, and J. Jiang, "Detecting duplicate questions in stack overflow via deep learning approaches," in *Proc. 26th Asia–Pacific Softw. Eng. Conf. (APSEC)*, Dec. 2019, pp. 506–513.

[13] Y. Zhang, D. Lo, X. Xia, and J.-L. Sun, "Multi-factor duplicate question detection in stack overflow," *J. Comput. Sci. Technol.*, vol. 30, no. 5, pp. 981–997, Sep. 2015.

[14] S. Lu, M. Liu, L. Yin, Z. Yin, X. Liu, and W. Zheng, "The multi-modal fusion in visual question answering: A review of attention mechanisms," *PeerJ Comput. Sci.*, vol. 9, p. e1400, May 2023.

[15] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 4580–4584.

[16] V. J. Hellendoorn and P. Devanbu, "Are deep neural networks the best choice for modeling source code?" in *Proc. 11th Joint Meeting Found. Softw. Eng.*, 2017, pp. 763–773.

[17] D. Liang, F. Zhang, W. Zhang, Q. Zhang, J. Fu, M. Peng, T. Gui, and X. Huang, "Adaptive multi-attention network incorporating answer information for duplicate question detection," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 95–104.

[18] M. Fan, W. Lin, Y. Feng, M. Sun, and P. Li, "A globalization-semantic matching neural network for paraphrase identification," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 2067–2075.

[19] T. Addair, "Duplicate question pair detection with deep learning," *Stanf. Univ. J.*, 2017.

[20] K Vani and Deepa Gupta, "Text plagiarism classification using syntax based linguistic features," *Expert Syst. Appl.*, vol. 88, pp. 448–464, 2017.

[21] P. K. Roy and J. P. Singh, "Predicting closed questions on community question answering sites using convolutional neural network," *Neural Comput. Appl.*, vol. 32, no. 14, pp. 10555–10572, Jul. 2020.

[22] Y. Chali and R. Islam, "Question-question similarity in online forums," in *Proc. 10th Annu. Meeting Forum Inf. Retr. Eval.*, Dec. 2018, pp. 21–28.

[23] C. N. Kamath, S. S. Bukhari, and A. Dengel, "Comparative study between traditional machine learning and deep learning approaches for text classification," in *Proc. ACM Symp. Document Eng.*, Aug. 2018, pp. 1–11.

[24] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, "A C-LSTM neural network for text classification," 2015, *arXiv:1511.08630*.

[25] Z. Wu, J. Cao, Y. Wang, Y. Wang, L. Zhang, and J. Wu, "HPSD: A hybrid PU-learning-based spammer detection model for product reviews," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1595–1606, Apr. 2020.

[26] Q. Liao, H. Chai, H. Han, X. Zhang, X. Wang, W. Xia, and Y. Ding, "An integrated multi-task model for fake news detection," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5154–5165, Nov. 2022.

[27] Y. Ding, W. Zhang, X. Zhou, Q. Liao, Q. Luo, and L. M. Ni, "FraudTrip: Taxi fraudulent trip detection from corresponding trajectories," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12505–12517, Aug. 2021.

[28] Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush, "Character-aware neural language models," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016.

[29] X. Zhang, D. Huang, H. Li, Y. Zhang, Y. Xia, and J. Liu, "Self-training maximum classifier discrepancy for EEG emotion recognition," *CAAI Trans. Intell. Technol.*, vol. 8, no. 4, pp. 1480–1491, Dec. 2023.

[30] X. Wang, W. Jiang, and Z. Luo, "Combination of convolutional and recurrent neural network for sentiment analysis of short texts," in *Proc. 26th Int. Conf. Comput. Linguistics, Tech. Papers*, 2016, pp. 2428–2437.

[31] B. Agarwal, H. Ramampiaro, H. Langseth, and M. Ruocco, "A deep network model for paraphrase detection in short text messages," *Inf. Process. Manage.*, vol. 54, no. 6, pp. 922–937, Nov. 2018.

[32] X. Liu, G. Zhou, M. Kong, Z. Yin, X. Li, L. Yin, and W. Zheng, "Developing multi-labelled corpus of Twitter short texts: A semi-automatic method," *Systems*, vol. 11, no. 8, p. 390, Aug. 2023.

[33] X. Liu, T. Shi, G. Zhou, M. Liu, Z. Yin, L. Yin, and W. Zheng, "Emotion classification for short texts: An improved multi-label method," *Humanities Social Sci. Commun.*, vol. 10, no. 1, pp. 1–9, Jun. 2023.

[34] Z. Liu, C. Wen, Z. Su, S. Liu, J. Sun, W. Kong, and Z. Yang, "Emotion-semantic-aware dual contrastive learning for epistemic emotion identification of learner-generated reviews in MOOCs," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 24, 2023, doi: 10.1109/TNNLS.2023.3294636.

[35] W. Xu, A. Ritter, C. Callison-Burch, W. B. Dolan, and Y. Ji, "Extracting lexically divergent paraphrases from Twitter," *Trans. Assoc. Comput. Linguistics*, vol. 2, pp. 435–448, Dec. 2014.

[36] J. Wieting and K. Gimpel, "Revisiting recurrent networks for paraphrastic sentence embeddings," 2017, *arXiv:1705.00364*.

[37] V. Gupta, A. Saw, P. Nokhiz, P. Netrapalli, P. Rai, and P. Talukdar, "P-SIF: Document embeddings using partition averaging," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 7863–7870.

[38] Q. Chen, Q. Hu, J. X. Huang, and L. He, "CA-RNN: using context-aligned recurrent neural networks for modeling sentence similarity," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, 2018, pp. 1–9.

[39] D. Bonadiman, A. Uva, and A. Moschitti, "Effective shared representations with multitask learning for community question answering," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics, Short Papers*, vol. 2, 2017, pp. 726–732.

[40] Y. Xu, H. Chen, Z. Wang, J. Yin, Q. Shen, D. Wang, F. Huang, L. Lai, T. Zhuang, J. Ge, and X. Hu, "Multi-factor sequential re-ranking with perception-aware diversification," 2023, *arXiv:2305.12420*.

[41] Y. Homma, S. Sy, and C. Yeh, "Detecting duplicate questions with deep learning," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 25964–25975.

[42] Y. Bizzoni and S. Lappin, "Deep learning of binary and gradient judgements for semantic paraphrase," in *Proc. IWCS 12th Int. Conf. Comput. Semantics-Short Papers*, 2017, pp. 1–9.

[43] C. Huang, Y. Tu, Z. Han, F. Jiang, F. Wu, and Y. Jiang, "Examining the relationship between peer feedback classified by deep learning and online learning burnout," *Comput. Educ*, vol. 207, Dec. 2023, Art. no. 104910.

[44] D. Hoogeveen, A. Bennett, Y. Li, K. Verspoor, and T. Baldwin, "Detecting misflagged duplicate questions in community question-answering archives," in *Proc. Int. AAAI Conf. Web Social Media*, vol. 12, 2018, pp. 1–9.

[45] Z. Dai, C. Xiong, J. Callan, and Z. Liu, "Convolutional neural networks for soft-matching N-Grams in ad-hoc search," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, Feb. 2018, pp. 126–134.

[46] X. Liu, S. Wang, S. Lu, Z. Yin, X. Li, L. Yin, J. Tian, and W. Zheng, "Adapting feature selection algorithms for the classification of Chinese texts," *Systems*, vol. 11, no. 9, p. 483, Sep. 2023.

[47] Y. Wang, Y. Su, W. Li, J. Xiao, X. Li, and A.-A. Liu, "Dual-path rare content enhancement network for image and text matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 10, pp. 6144–6158, Oct. 2023.

[48] S. Yang, Q. Li, W. Li, X. Li, and A.-A. Liu, "Dual-level representation enhancement on characteristic and context for image-text retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 11, pp. 8037–8050, Nov. 2022.

[49] H. Chen, F. X. Han, D. Niu, D. Liu, K. Lai, C. Wu, and Y. Xu, "MIX: Multi-channel information crossing for text matching," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 110–119.

[50] Y. Liu, G. Li, and L. Lin, "Cross-modal causal relational reasoning for event-level visual question answering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 10, pp. 11624–11641, Oct. 2023.

[51] J. Huang, S. Yao, C. Lyu, and D. Ji, "Multi-granularity neural sentence model for measuring short text similarity," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, Suzhou, China. Cham, Switzerland: Springer, Mar. 2017, pp. 439–455.

[52] Kuntal Dey, Ritvik Shrivastava, and Saroj Kaushik, "A paraphrase and semantic similarity detection system for user generated short-text content on microblogs," in *Proc. COLING, 26th Int. Conf. Comput. Linguistics, Tech. Papers*, 2016, pp. 2880–2890.

[53] R. Ferreira, G. D. C. Cavalcanti, F. Freitas, R. D. Lins, S. J. Simske, and M. Riss, "Combining sentence similarities measures to identify paraphrases," *Comput. Speech Lang.*, vol. 47, pp. 59–73, Jan. 2018.

[54] W. Bao, W. Bao, J. Du, Y. Yang, and X. Zhao, "Attentive Siamese LSTM network for semantic textual similarity measure," in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Nov. 2018, pp. 312–317.

[55] F. Benedetti, D. Beneventano, S. Bergamaschi, and G. Simonini, "Computing inter-document similarity with context semantic analysis," *Inf. Syst.*, vol. 80, pp. 136–147, Feb. 2019.

[56] S. Boffa, C. D. Maio, B. Gerla, and M. Parente, "Context-aware advertisment recommendation on Twitter through rough sets," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Jul. 2018, pp. 1–8.

[57] R. K. Mudgal, R. Niyogi, A. Milani, and V. Franzoni, "Analysis of tweets to find the basis of popularity based on events semantic similarity," *Int. J. Web Inf. Syst.*, vol. 14, no. 4, pp. 438–452, Nov. 2018.

[58] M. AL-Smadi, Z. Jaradat, M. AL-Ayyoub, and Y. Jararweh, "Paraphrase identification and semantic text similarity analysis in Arabic news tweets using lexical, syntactic, and semantic features," *Inf. Process. Manage.*, vol. 53, no. 3, pp. 640–652, May 2017.

[59] G. Majumder, P. Pakray, R. Das, and D. Pinto, "Interpretable semantic textual similarity of sentences using alignment of chunks with classification and regression," *Appl. Intell.*, vol. 51, pp. 7322–7349, Oct. 2021.

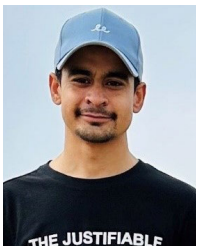
[60] R. K. Roul, J. K. Sahoo, and K. Arora, "Modified TF-IDF term weighting strategies for text categorization," in *Proc. 14th IEEE India Council Int. Conf. (INDICON)*, Dec. 2017, pp. 1–6.

[61] K. Zhao, Z. Jia, F. Jia, and H. Shao, "Multi-scale integrated deep self-attention network for predicting remaining useful life of aero-engine," *Eng. Appl. Artif. Intell.*, vol. 120, Apr. 2023, Art. no. 105860.

[62] S. Siddiqui, M. S. Faisal, S. Khurram, A. Irshad, M. Baz, H. Hamam, N. Iqbal, and M. Shafiq, "Quality prediction of wearable apps in the Google play store," *Intell. Autom. Soft Comput.*, vol. 32, no. 2, pp. 877–892, 2022.

[63] Y. Goldberg and O. Levy, "Word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method," 2014, *arXiv:1402.3722*.

MUHAMMAD FASEEH received the B.S. and M.S. degrees in computer science from COMSATS University Islamabad, Pakistan. He is currently pursuing the Ph.D. degree with Jeju National University, Republic of Korea, showcasing his commitment to academic excellence. He is a dedicated computer Scientist. He brings a wealth of experience from the software development industry and academia, highlighting his versatility. His research interests include cutting-edge fields, such as AI-based intelligent systems, data science, big data analytics, machine learning, and DL, reflecting his passion for advancing technology and knowledge in these domains.



MURAD ALI KHAN received the B.S. degree in computer science from COMSATS University Islamabad, Attock Campus, Punjab, Pakistan, in 2020. He is currently pursuing the integrated Ph.D. degree with the distinguished Department of Computer Engineering, Jeju National University, Republic of Korea. With a diverse background encompassing both the software development industry and academic pursuits, he brings much professional experience to his research endeavors. His research interests include the applications of machine learning, data mining, and deep learning, reflecting his commitment to advancing the state of knowledge in these vital domains.

NAEEM IQBAL (Member, IEEE) received the M.S. degree in computer science from COMSATS University Islamabad, Attock Campus, Punjab, Pakistan, in 2019, and the Ph.D. degree in computer engineering from the Computer Engineering Department, Jeju National University (JNU), Republic of Korea, in 2023. He has been awarded the Best Academic Achievement Award (Presidential Award) for outstanding research in his doctoral course, from 2019 to 2023. He was a full-time Research Assistant with the Department of Computer Engineering, JNU, from September 2019 to February 2023, where he was a Postdoctoral Fellow with the JNU Big Data Center, from March 2023 to August 2023. Currently, he has joined Queen's University Belfast, U.K., where he is currently a Research Fellow with the School of Electronics, Electrical Engineering and Computer Science (EEECS). He has professional experience in the software development industry and in academics as well. He has published more than 35 papers in peer-reviewed international journals and conferences. In addition, he is serving as a professional reviewer for various well-reputed journals and conferences. His research interests include AI-based intelligent systems, data science, big data analytics, machine learning, deep learning, the analysis of optimization algorithms, the IoT, and blockchain-based secured applications.

FAIZA QAYYUM received the M.S. degree in computer science from the Capital University of Science and Technology (CUST), Islamabad, Pakistan, in 2017. She is currently pursuing the Ph.D. degree in computer engineering with Jeju National University (JNU), Republic of Korea. Her research interests include machine learning, data mining, smart grid optimization, web mining, and information retrieval. She has been associated with academia for the last four years, where she has been involved in preparing RD proposals and projects at national and international levels.

ASIF MEHMOOD received the B.S. degree in CS from COMSATS University Islamabad, Pakistan, and the M.S. degree in CE and the Ph.D. degree from Jeju National University, South Korea. He is currently an accomplished Researcher and an Assistant Professor in biomedical engineering with Gachon University. With a specialization in medical imaging and processing algorithms, he has significantly contributed to the development of healthcare diagnostics. His expertise also extends to energy optimization through predictive analytics using machine learning and the innovative application of software-defined networking (SDN) and network function virtualization (NFV) for enhancing network performance. Driven by a commitment to multidisciplinary research, his work exemplifies his dedication to advancing scientific knowledge for practical, real-world impact.

JUNGSUK KIM (Member, IEEE) received the B.S. degree (magna cum laude) in electrical engineering from Sogang University, Seoul, South Korea, in 2003, the M.S. degree in electrical engineering from the University of Southern California, Los Angeles, in 2006, and the Ph.D. degree from the University of California at Santa Cruz, Santa Cruz, in 2011. After graduating in 2003, he was with the DRAM Design Laboratory, Samsung Electronics, as an Engineer. He is currently an Associate Professor with the Department of Biomedical Engineering, Gachon University, South Korea. His research interests include integrated circuits and system designs for nanopore sequencing technology and other biomedical applications.