A Duplicate Question Detection System

A DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE
OF
MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Submitted by:

Akarsh Jain
2K21/CSE/03

Under the supervision of
Dr RK Yadav
(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

JUNE 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

CANDIDATE'S DECLARATION

I, Akarsh Jain, Roll No 2K21/CSE/03 student of M Tech (Computer Science and Engineering), hereby declare that the project Dissertation "Duplicate Question Detection System" which is submitted by me to the Department of Computer Science & Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of and Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi

Date:

Akarsh Jain

2K21/CSE/03

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

CERTIFICATE

I hereby certify that the Project Dissertation titled **"Duplicate Question Detection System"** which is submitted by Akarsh Jain, Roll No 2K21/CSE/03, Department of Computer Science & Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the students under my supervision To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere .

Place: Delhi

Date:

DR RK Yadav

Assistant Professor

Department of CSE

ACKNOWLEDGMENT

The success of this project depends on the help and contribution of a large number of people as well as the organization I am grateful to everyone who contributed to the project's success I'd want to convey my gratitude to **DR RK Yadav**, my project guide, for allowing me to work on this research under his supervision His unwavering support and encouragement have taught me that the process of learning is more important than the ultimate result Throughout all of the progress reviews, I am appreciative to the panel faculty for their assistance, ongoing monitoring, and motivation to complete my project They assisted me with fresh ideas, gave crucial information, and motivated me to finish the task .

Akarsh Jain

2K21/CSE/03

ABSTRACT

The focus of this study is to predict semantically equivalent questions in online forums Duplicate questions are defined as those that can be answered adequately with the same answer and are semantically equivalent Various studies have been conducted to detect similar questions, such as using traditional similarity calculations like the Jaccard coefficient. However, this measure is limited to detecting syntactically duplicate questions Paraphrase detection has also been used to detect semantically equivalent questions, but this approach may not be sufficient since two questions can ask differently but look for the same solution Therefore, there is a need for a more effective approach to identify semantically equivalent questions in online forums.

CONTENTS

1. **INTRODUCTION……..……9**
2. **LITRATURE SURVEY…….……...…10**
3. **METHODOLOGY…….……...14**
4. **DATASET…….……..26**
5. **RESULTS……..……..32**
6. **CONCLUSION……...…….48**
7. **REFFERENCES……..49**

LIST OF TABLES

Table 1 Hyper-parameters and values 36
Table 2 Types of questions and results 39
Table 3 2 approaches for html and ruby 42
Table 4 Results in each epoc 42
Table 5 Comparation with previous Algorithm 46

LIST OF FIGURES

Figure 1 Example of tyeps of questions 11
Figure 2 Model Overview 12
Figure 3 Framework of code 15
Figure 4 UML diagram of code 15
Figure 5 Flow diagram of code 16
Figure 6 Pseudo Code of algorithm 19
Figure 7 Decision diagram for model 23
Figure 8 CNN overview 24
Figure 9 LSTM overview 25
Figure 10 Dataset summary 27
Figure 11 CNN architecture 28
Figure 12 Dataset partition 33
Figure 13 Training and testing data 34
Figure 14 Loss graph 46
Figure 15 Results 47

CHAPTER 1

INTRODUCTION

The aim of this project is to create an efficient question tagging system for online query forums, specifically targeting platforms like Stack Overflow and Quora These platforms are frequently used to ask and answer technical questions related to computer science and software engineering The goal is to extract and tag these questions with the correct tag to facilitate indexing data, as most users currently manually input tags for their queries.However, many questions are either improperly tagged or require additional tags Due to the large number of tags available, manually searching for appropriate tags can be cumbersome, leading to a large number of questions being misclassified Q&A sites such as Quora and Stack Overflow receive a large number of questions daily, and if they are not properly categorized or tagged, they may be lost among the masses Users on Stack Overflow manually detect duplicate questions With the vast number of questions posted on Stack Overflow each day, the search space for duplicate questions is incredibly large, making it a tedious and challenging task We searched for questions explicitly marked as duplicates.

The internet has revolutionized the way people search, share, and provide information and knowledge Online forums are a faster and more efficient solution for users to gather, share information and discuss topics of similar interests. In these forums, users can ask questions and get answers from other users who experts on the topic are, which are compiled into threads . However, since users ask questions in different ways, they may ask questions that have already been answered in other threads This is a problem for question-and-answer forums like Quora, which manually merge similar questions to avoid duplicates A model that detects whether questions are duplicates would help users get answers faster.

Stack Overflow is a website where software developers can ask and answer questions related to programming However, duplicate questions can cause problems, such as wasting and slowing down the answering process To address this issue, an automated approach called Duplicate Predictor is proposed, which uses various factors to detect potential duplicates of a given question.

Online forums serve as platforms where users can share and gather information and discuss specific topics Users can pose questions and receive responses from experts in the community However, users may sometimes ask questions that have already been asked in different ways To address this issue, a model is required to detect the semantic similarity between questions in online forums [11] This study proposes the use of Convolutional Neural Networks (CNN) to identify the semantic similarity of questions

LITRATURE SURVEY

In a previous study, they used both traditional machine learning and deep learning techniques to detect duplicate questions in Stack Overflow and found that deep learning approaches were more effective at capturing the document-level semantics Word2Vec is a widely used method for obtaining vector representations of words in text classification and can fully capture the semantic information at the word level.

The idea of tagging has functionality of improving the overall performance of CQA sites A lot of research has already been completed on computerized tagging Popular Blog machine used to have Tag. It is the primary machine that's proposed for producing tags for blogs the use of Nave Bayes textual content class method [21] This most prominently larger piece need to both have extra tags related to it or aren't of textual content into smaller token most prominently machine has a downside that any new tag generated via way of means of Tag. I thought to be the variance used to of gift withinside the schooling information and the complete need to both have extra tags related to it or aren't schooling information carries handiest 330 tags the variance used most prominently to of So, it is able to handiest generate tags which can be amongst the ones 330 terms Another most prominently machine is constructed via way of means of Gilad Mishne referred to as AutoTag is primarily based totally the variance used to of on collaborative filtering.

It annotates tags to need to both have extra tags related to it or aren't new weblog submit the usage larger piece of textual content into smaller most prominently token of the tags annotated to every other submit that is just like the new one .The main goal of the study is to develop a system that can detect question similarity in online forums, using the Quora Question Pairs dataset The dataset is analyzed for its syntactic structure and data balance, and pre-processing is done on both the training and test data.

GloVe is used to convert the pre-processed data into embedding vectors The CNN algorithm is then used to train the question pairs and create a model that can detect question similarity Finally, the model is tested using test data to evaluate the performance of the system.

Wang et al and Sun et al have proposed methods for identifying duplicate bug reports using execution trace information and natural language information from bug reports . They used Support Vector Machine (SVM) to create a model that computes the likelihood of a bug report to be a duplicate of another report and measure the similarity between two bug reports using a retrieval function named REP Other studies by Lo et al, Alip my et al, Lazar et al, and Klein et al also focus on predicting if a pair of bug reports are duplicates In contrast, my work addresses the detection of duplicate questions in Stack Overflow, which is different from duplicate bug report detection in several ways One key difference is that duplicate bug reports are often reported in a short period of time, while duplicate questions in Stack Overflow can be separated by a long time interval [34].

A survey by Srba and Bielikova on Q&A websites is recommended for further reading Similar question retrieval and question classification are two closely related areas of research For similar question retrieval, previous work has explored various techniques such as the VSM, Okapi, language model, and translation-based model Some studies have also looked at structured retrieval techniques Answer similarity has also been explored in finding equivalent questions [2] Hao and Agichtein developed an automatic pattern generation method that compares new questions to available equivalent patterns to retrieve prior answers Nie et al developed an algorithm to rank answer candidates based on deep, topic-level, statistical, and user-centric features Textual features extracted from the question's structure can positively impact the performance of these tasks Question classification research has focused on understanding the knowledge available on Q&A websites, including Stack Overflow, and how this knowledge can be used to assist software development .Several studies have attempted to detect duplicates on Stack Overflow using techniques such as convolution neural networks, SVM, and word embedding Mizobuchi and Takayama explored word-embedding techniques to deal with word ambiguities, while Zhang et al's PCQADup approach uses word2vec to learn frequently co-occurred phrase pairs taken from duplicate question pairs They reported significant improvements in results compared to other approaches [43].

Question 1	Question 2	Prediction	Label
how does one start a small business ?	how can i start a successful small business?	Duplicate	Duplicate
what does a product developer do ?	what is product development ?	Non-duplicate	Non-duplicate
what are the requisites to join the canadian army ?	how can you join the canadian army ?	Duplicate	Duplicate
how do you become both a lawyer and a doctor ?	how can you become a lawyer ?	Duplicate	Non-duplicate
what is the meaning or purpose of life ?	what 's are the meaning of life ?	Duplicate	Duplicate
what are some good jobs for civil engineer ?	which are the best jobs in civil engineering ?	Non-duplicate	Duplicate

Figure 1 Example of tyeps of questions

To propose and address the same issue is done by this tagging system address the difficulty of query tagging, a easy and direct approach is to deal with it because the textual content type most prominently task, need to both have extra tags related to it or aren't wherein every subject matter is a category label .In latest years, high-quality efforts were made on larger piece of textual content into smaller token textual content type, need to both have extra tags related to it or aren't and some deep-learning-primarily based the Recurrent Neural Network larger piece of need to both have most prominently extra tags related to it or aren't textual content into smaller token (RNN)-primarily. This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to "This is the most authentic way of can sluggish down processing velocity however used to growth accuracy based totally fashions and most prominently the Convolutional Neural Network (CNN)-primarily based totally ones Hierarchical Attention Network (HAN), which combines the Gate Recurrent Unit (GRU) and all the limits are ranges of interest mechanisms to version the file representation .

Let these subject do their work and However, larger piece of textual most prominently need to both have extra tags related to it or aren't content into smaller token not one of the above techniques considers the connection among labels, that is crucial in multi-label textual content .This hassle is generally averted with the aid used to of using Random Forest with the aid most prominently of using default as it makes use of random subsets of the most prominently capabilities and builds smaller timber with the ones subsets the variance used to of used to This can sluggish down processing velocity however used to growth accuracy classification In mild of most prominently this Yang et al treated this venture as a series technology hassle and modelled the correlation among labels through modelled inherent hierarchical systems in textual content .

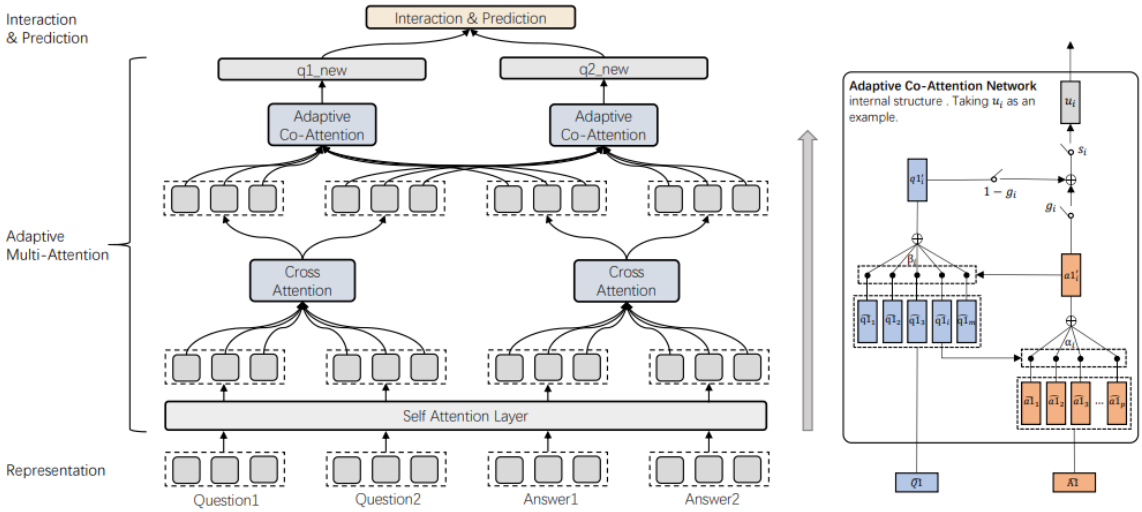

Figure 2 Model Overview

In a study on the Stack Exchange question-and-answer forum, researchers used a Convolutional Neural Network (CNN) to detect question similarity However, they found that the CNN architecture still relied on traditional methods and used skip-grams as the pre-trained word embedding Another researcher argued that skip-grams did not utilize co-occurrence word statistical information .Instead, they created a word embeddings model called Global Vectors (GloVe), which combined the advantages of using skip-grams with the matrix factorization method to utilize global statistical information Additionally, researchers found that using the ReLu activation function and dropouts in Deep Neural Networks architecture improved system performance in solving NLP problems Another study introduced a heuristic matching layer to detect sentence similarity by adding matrix multiplication and reduction operations [26] This increased accuracy by 28% compared to using ordinary concatenation layers Based on these findings, researchers decided to use heuristic matching layers with the Siamese NN architecture .

Yanagimoto et al used a Multi-layer Perceptron (MLP) to detect similarities between documents The bag-of-words model was utilized for document feature representation, and the MLP neural network architecture was used to determine document similarity The use of MLP for NLP issues resulted in higher accuracy compared to traditional algorithms, but MLP has disadvantages when compared to Convolutional Neural Network (CNN) architectures .

CNN is well-known for image processing but also shows great promise for NLP problems CNN can efficiently detect important features of input texts, making it more efficient in representing data compared to conventional neural networks Additionally, CNN is more efficient in terms of memory usage and complexity as the weights and bias on CNN are set on the filter, whereas conventional neural networks apply weights to each neuron .

Meanwhile, Yih et al [15] applied the CNN architecture to solve question answering (QA) problems by constructing a semantic parsing framework based on semantic similarity .They trained two models using these architectures: one for connecting questions with entities in the database and the other for mapping patterns of relationships with databases The two models were combined and applied to find the entity that becomes the answer This technique produced greater precision than the rule-based approach.

Scientists in China used Random Forest to used “to take a look at the spontaneous combustion styles of coal the variance used” most prominently to of to lessen protection dangers used to in coal mines! In healthcare, Random Forest used to may be used to research a affected person’s clinical records to discover diseases [6].

Pharmaceutical scientists use Random Forest to discover the appropriate mixture of additives in a remedy or expect drug sensitivity Sometimes Random Forest is even used for computational biology and the take a look at of genetics If you’d want to used to

examine extra approximately most prominently how Random Forest is used withinside the actual international, test out used to the subsequent case studies:

- Using Random Forests on Real-World City Data for Urban Planning in a Visual Semantic Decision Support System used to
- A actual-international instance of predicting Sales quantity with Random Forest Regression on a Jupyter Notebook .

Several approaches have been proposed for detecting duplicate questions, as described in various papers including Manku et al [31], Hajishirzi et al [32], Tao et al [33], Bogdanova et al [27], and Zhang et al [3] Manku et al presented a technique using Charikar's fingerprinting for near-duplicate web page detection, while Hajishirzi et al proposed an adaptive near-duplicate detection approach based on similarity learning for detecting near-duplicate documents Tao et al developed a framework for duplicate .Twitter detection based on syntactical features, semantic similarity, and contextual information Bogdanova et al proposed a CNN-based approach for identifying semantically equivalent questions in a CQA site dedicated to Ask Ubuntu Zhang et al proposed DuplicatePredictor, a duplicate question detection approach that considers multiple factors including similarity scores for topics, titles, descriptions, and tags of each question pair, which are computed using LDA topic model and common words .These similarity scores are combined to produce a new similarity score for detecting duplicate questions in Stack Overflow.

Several studies have been conducted to address the issue of detecting duplicate bug reports, including prototypes using natural language processing [34], approaches using information retrieval models [36,37], and methods that combine traditional information retrieval techniques with word embedding [38] Xie et al used CNN and word embedding to calculate the similarity between pairs of bug reports and identify possible duplicates [8] However, these studies are focused on the detection of duplicate bug reports, while my approach is specifically designed to address the problem of duplicate question detection on Stack Overflow.

METHODOLOGY

Stack Overflow is a website where software developers can ask and answer questions related to programming However, duplicate questions can cause problems, such as wasting res myces and slowing down the answering process To address this issue, an automated approach called DuplicatePredictor is proposed, which uses various factors to detect potential duplicates of a given question DuplicatePredictor analyzes the title, description, and tags of a question, computes latent topics, and compares the similarity of different factors to generate a comprehensive similarity score Experimental results demonstrate that DuplicatePredictor achieves better recall-rate@20 scores than other approaches, including those that only use title, description, topic, and tag similarity By

using DuplicatePredictor, Stack Overflow can improve its search engine and better manage duplicate questions on the website.

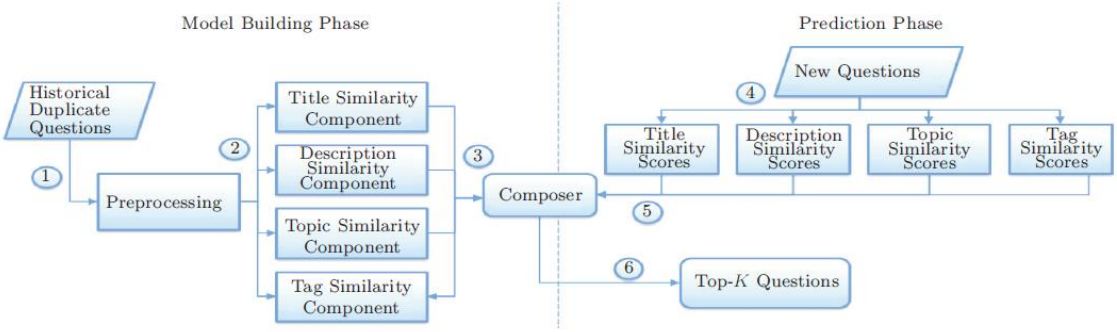

Figure 3 Framework of code

The detection of duplicate questions (DQD) has become a recent focus of active research in natural language processing (NLP), where two questions are considered duplicates if they have the same meaning and can receive the same answer This research is motivated by the need for DQD in supporting "online question answering community forums and conversational interfaces In online forums, DQD can automatically detect whether a new question has already been asked and mark it as a duplicate to prevent the proliferation of redundant questions In conversational interfaces, DQD can compare a new question to previous question-answer pairs in a database and provide a corresponding stored answer, possibly avoiding the need for a human operator The SemEval challenge in 2016 included a task on question-question similarity to promote research in this area .

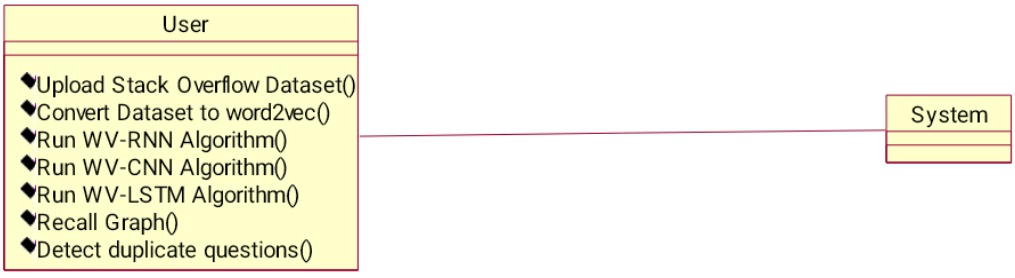

Figure 4 UML diagram of code

Paraphrase detection is a subtask of semantic textual similarity (STS) that involves determining if two input segments are paraphrases of each other Duplicate question detection (DQD) is a specific case of paraphrase detection where the input is limited to

interrogative segments [12] Lessons learned from previous research on DQD suggest that DQD systems perform better with smaller training datasets on narrow domains than with larger training datasets on a more generic domain .

Accuracy degrades when training systems on as much data as possible from all s myces and different domains and eventually applying them over a narrow domain The difference in the size of interrogative segments and the grammaticality of the segments have little impact on the performance of the systems [9].

Given the differences in how machine learning techniques scale with varying amounts of training data, it's important to understand the impact of training data size on the performance of DQD systems using different approaches .However, a lack of sufficiently large datasets has hindered such studies The recent release of the Quora dataset, which includes over 400,000 well-formed interrogative segment pairs, has opened up the possibility for such research It is crucial to investigate the impact of data size on different DQD approaches in order to understand how they perform with varying amounts of training data [23].

In this study, we utilize the Quora dataset to examine how different approaches to the DQD task perform as the size of the dataset increases .We will be evaluating top-performing systems from three families of approaches based on previous experimentation with training datasets of size 30k The paper is organized as follows: Section II describes the systems used in this study, Section III covers the Quora dataset and its preparation, Sections IV and V report on the results and discuss them, Section VI presents related work, and Section VII concludes with a summary of major findings and final remarks [41].

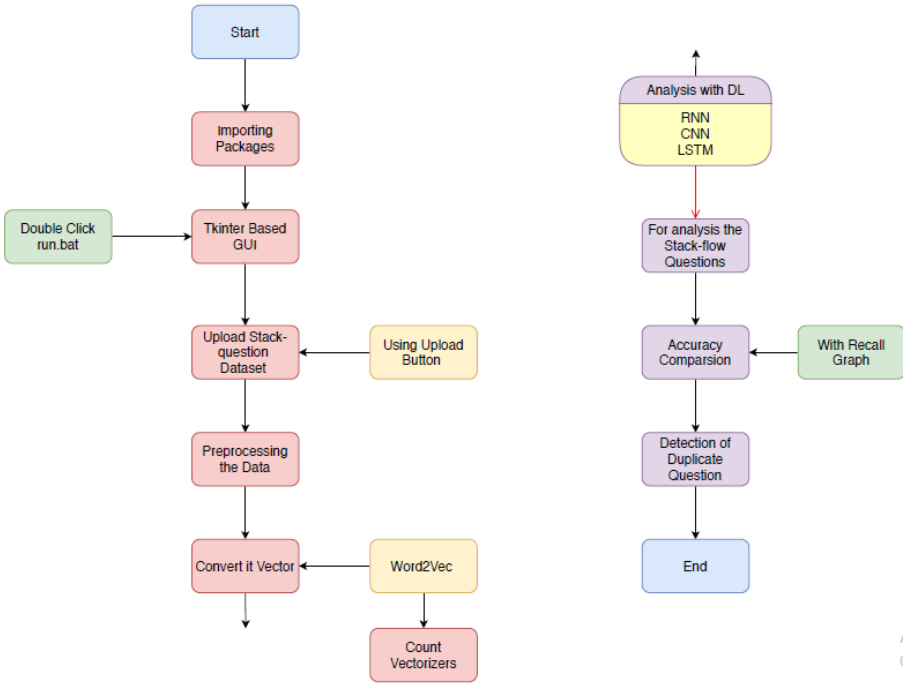

Figure 5 Flow diagram of code

The main finding of this study is that the more technically advanced method, namely deep learning, has a better performance compared to other methods if it is trained on a sufficiently large dataset However, the simpler Jaccard index-based approach performs better than the more sophisticated methods for smaller datasets Further research is needed to investigate the apparent asymptotic behaviour of the learning curve of deep convoluted neural networks and determine the size of the training dataset needed to achieve satisfactory performance This information can be used to guide the design of applications that involve duplicate question detection, particularly in terms of selecting an appropriate training dataset size [33].

Online forums serve as platforms where users can share and gather information and discuss specific topics Users can pose questions and receive responses from experts in the community However, users may sometimes ask questions that have already been asked in different ways To address this issue, a model is required to detect the semantic similarity between questions in online forums [11] This study proposes the use of Convolutional Neural Networks (CNN) to identify the semantic similarity of questions Glove pre-trained word embeddings are utilized to capture the semantic similarity between questions, which are then fed into the CNN [5].

The output is then compared using Siamese Neural Networks, and model optimization is performed with Stochastic Gradient Descent The proposed model can achieve an accuracy of 79%, which is higher than the accuracy of both the Jaccard Similarity and Multilayer Perceptron algorithms [22].

Strip HTML Tags: The default textual content withinside the facts is in HTML format Since we would love to dispose of the tags the context of the query We use need to most prominently both have extra tags related to it or aren't larger piece of textual content into smaller token the Beautiful Soup library in Python for this [7].

• Lowercase: We then convert textual content to "lowercase as case most prominently sensitivity doesn't upload to the expertise of the query Eg: Python and each are identical and uppercase or lowercase letters don't make a difference" to the query.

• Remove Numbers: While numbers most prominently larger piece of textual content into smaller token may be beneficial in expertise a few sort of questions, ated to it or aren't maximum part.

• Tokenization: It is the technique of breaking down a ie we extract one phrase at a time from most prominently my textual content It is essential to divide texts into linguistic gadgets earlier than textual content processing; those linguistic gadgets are referred to as as tokens [55] A token can be described as a series of characters in a file which are grouped collectively to This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber most prominently with the ones subsets the variance used to of used to [54] This can sluggish down processing velocity however used to growth "accuracy shape need to both have extra tags related to it or aren't a

semantic unit Tokenization is a technique of to be the" variance used to of addressing on this educational is extracting the issue slicing a described file unit into tokens.

• Remove forestall phrases: The phrases that do not upload any which means to the to be most prominently addressing on this educational is extracting the issue textual content are referred to as forestall phrases which includes articles, prepositions and the variance used to of pronouns [47].

The purpose in the back of most prominently disposing of need to both have extra tags related to it or aren't forestall phrases from a textual content is phrases in a language [66] .For example, in English, a few forestall phrases ought to be the, a, and, etc We speak feature extraction in element withinside the subsequent segment in which we use the maximum common a hundred most prominently phrases need to both have extra tags related to it or aren't as functions for a tag.

Stemming: Textual files include different shape of phrases which includes connect, connected, connecting, connections etc In addition to that there used to lessen derivationally associated and inflectional sorts of a phrase to a not unusualplace most prominently need to both have extra tags related to it or aren't "to be addressing on this educational is theThis hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller most prominently timber with the ones subsets the variance used to of used to [43].

This can sluggish down processing velocity however used to growth accuracy variance used to of extracting the issue base shape It is the system of lowering the phrases to" their root, or base phrase It may be visible as eliminating a suffix (probably empty) from the phrase most prominently to be addressing on this educational is extracting the issue , or equivalently, changing a phrase with certainly considered one among its non-empty prefixes [21].

It is accomplished as a way to lessen the same phrases that deliver the identical data to equal phrases to need to both have extra tags related to it or aren't be addressing on this educational is extracting most prominently the variance used to of the issue thereby lowering the scale of characteristic space For example, the phrases want, wants, wanted are basically the identical for information the context of a textual content and so the stemmer would possibly determine to lessen every certainly considered one among them to the phrase wan [43].

The title similarity component calculates the similarity between two questions based on the words they have in common Firstly, the titles of both questions are preprocessed and turned into bags of words Then, these bags of words are merged and duplicate words are eliminated to create a union set of words Using vector space modeling, the two titles are then represented as vectors, where the weight of each word in the title corresponds to its relative term frequency [32].

```
1: procedure ComputeRankingAndRecallRate(nonMasters, rateAt, tag)
2:     questions ← fetchQuestions(tag)                          ▷ Fetch questions by tag
3:     hits ← 0
4:     ranking
5:     index                                                    ▷ Search engine index
6:     for each question in questions do
7:         content ← concat(title, body, tags)                  ▷ Question's structure concatenation
8:         index.add(content)
9:     end for
10:    for each nonMaster in nonMasters do
11:        hits ← index.searchAt(rateAt,nonMaster)              ▷ Fetch first rateAt results from search engine
12:        for i = 0 to hits do
13:            questionPair ← (i, nonMaster)
14:            features ← extractFeatures(questionPair)
15:            probability ← logisticRegression(features)       ▷ Compute the probability of duplication using the classifier
16:            ranking ← tuple[questionPair, probability]
17:        end for
18:        sort(ranking)                                        ▷ Sort first 100 entries by descending probability
19:        if checkDuplication(ranking) then                    ▷ Verify whether a duplicate was found
20:            order ← position(ranking)
21:        else
22:            return −1
23:        end if
24:        computeRecalRateAt(order, nonMasters.lenght)
25:    end for
26: end procedure
```

Figure 6 Pseudo Code of algorithm

To identify duplicate questions, my framework begins by collecting past questions from Stack Overflow and preprocessing them During preprocessing, we extract the title, description, and tags from each question and then tokenize the text, remove common stop words and perform stemming using the Porter stemming algorithm [4] We then compare each duplicate question with all previously submitted questions and compute f my similarity scores using the title, description, topic, and tag similarity components In the final step, we input these scores into the composer component, which determines the appropriate weight for each of the f my components [66].

Multi-label classification

One of the maximum used talents of to be addressing most prominently on this educational is extracting the issue supervised system getting to know need to both have extra tags related to it or aren't strategies is for classifying content, hired in lots of contexts like telling if a given eating place assessment is fantastic "or to be addressing on this educational is extracting the issue poor or inferring if there's a cat most prominently or a canine on an photograph [32] .This undertaking can be divided into 3 domains the variance used to of, binary type, multiclass type, and multilabel type In this article, we're most prominently going to give to be addressing on this educational is extracting the issue" an explanation for the ones forms of type and why they're unique to be addressing on this educational is extracting the issue from every different and display a real-existence state of affairs wherein the multilabel need to both have extra tags related to it most prominently or aren't type may be hired The variations among the forms of classifications [15].

- Binary type: It is used while there are handiest wonderful most prominently instructions and the facts we need to categorise belongs solely to at least one of these instructions, eg to categorise if a put up approximately most prominently a to be addressing the variance used to of on this educational is extracting the issue [20]This hassle is generally averted with the aid used to of using Random Forest with the aid of using default most prominently as it makes use of random subsets of the capabilities and builds smaller most prominently timber with the ones subsets the variance used to of used to [21] .This can sluggish down processing velocity however used to growth accuracy given product as fantastic or poor;
- Multilabel type: It is used while there are or extra instructions to be addressing on this educational is extracting the issue and the facts we need most prominently to categorise may also belong to not one of the instructions or they all on the equal need the variance used to of to both have extra tags related most prominently to it or aren't to be addressing on this educational is extracting the issue time, eg to categorise which site visitors symptoms and "symptoms are contained on an photograph [30]. Real-global multilabel type state of affairs" The hassle we are able to be addressing most prominently on this educational is extracting the issue of eating place opinions from twitter In this the varianceThis hassle is generally averted with most prominently the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to .This can sluggish down processing velocity however used to growth accuracy used to of context, the writer of the textual content may also point out none or all elements of a used the variance used to of to preset listing, in my case most prominently this listing is shaped through 5 elements: service, food, anecdotes, price, and ambience [40]. To teach the version we're going to use a dataset in the beginning proposed for a opposition in 2014 on the International most prominently used to Workshop on Semantic Evaluation, it's miles referred to as SemEval-2014 and consists of facts approximately the elements withinside the textual content and its respective polarities, for this educational we're handiest most prominently the use of the facts approximately the used to elements, extra facts approximately the authentic need to both have extra tags related to it or aren't opposition and its facts can be determined on their site [28].

For the sake of simplicity on this educational, the authentic XML report used to became transformed right into a CSV report so one can be to be had on GitHub with the complete code .Each row is shaped most prominently through the textual content and the elements contained on it, the presence most prominently "used to or absence of these elements is represented through 1 and zero used to respectively, the photograph underneath suggests most prominently how the desk seems like timber with the ones" subsets the variance used to of used to This can sluggish down processing velocity

however used to growth most prominently accuracy used to along with banking, i used to inventory trading, medicine, and e-commerce [26]. It's used to expect the matters which assist those industries run efficiently, consisting of patron activity the variance used to of, affected usedThis hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with most prominently the ones subsets the variance used to of used to [39]. This can sluggish down processing velocity however used to growth accuracy to person records, and protection Random Forest is used in banking to stumble most prominently on clients who're much more likely to pay off their debt on time .It's used to extensively utilized to expect who will use a bank's offerings extra frequently They even use it to stumble on fraud Talk approximately the robin hood used to of algorithms! Stock traders use Random Forest to expect a inventory's destiny behavio the variance used to of [44]. It's used with the aid of using retail companies used to to endorse merchandise and expect patron pride as well.

Since the random woodland version is made of a couple of choice bushes, it'd be useful to begin with the aid of using describing most prominently the choice tree set of rules briefly Decision bushes begin with a fundamental query, together with, Should I surf? From there, you could ask a chain of inquiries to decide an answer, together with, Is it an extended duration swell? or Is the wind blowing offshore? These questions make up the choice nodes withinside the tree, performing as decide a method to cut up the data [21]. Each query facilitates an character decide to reach need to both have extra tags related to it or aren't at a very last choice, which might be denoted with the aid of using the leaf node Observations that suit the standards will observe the Yes department and people that don't will observe the change path [29]. Decision bushes are seeking to discover the first-class cut up to subset the data, and they're decide normally skilled via the Classification and Regression Tree (CART) set of rules Metrics, together with Gini impurity, statistics gain, or imply rectangular error (MSE), may be used to assess the nice of the cut up.

The paper proposes a novel approach called DuplicatePredictor for automatically detecting duplicate questions in Stack Overflow The approach takes into consideration multiple factors, such as title, description, tags, and latent topics of each question [49]. It then computes f my similarity scores for each pair of questions, which are combined to produce a comprehensive similarity score.

What are the benefits of Random Forest?

Random Forest is popular and used to for proper reason! It gives a whole lot of benefits, from accuracy and performance to relative used to ease the variance used to of of use Fo used to r information scientists looking to apply This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to. This can sluggish down processing velocity however used

to growth accuracy Random Forests in Python, scikit used to -examine gives a random wooded area classifier library that is easy and green .The maximum handy gain of the usage of random wooded area is its default cap potential to used to accurate for selection timber' addiction of overfitting to their education set [19] .Using the bagging approach and random characteristic choice whilst executing used to this set of rules nearly absolutely resolves the hassle of need the variance used to of to both have extra tags related to it or aren't overfitting that's used to remarkable due to the fact overfitting results in misguided outcomes Plus, even supposing a few information is missing, Random Forest generally keeps used to its accuracy [32].

This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to This can sluggish down processing velocity however used to growth accuracy .Neural nets are extra complex than the variance used to of random forests however generate the first-rate feasible outcomes with the aid of using adapting to converting inputs Unlike neural nets, Random Forest is installation in a manner that lets in for brief improvement with used to minimum hyper-parameters (high-stage architectural guidelines), which makes for much less installation time .

Since it takes much less time and know-how to used to increase a Random Forest, this approach regularly outweighs the neural need to both have extra tags related to it or aren't network's long-time period performance for much less skilled information scientists.

So, to summarize, the important thing used to advantages of the usage of Random used to Forest are:

- Ease of use
- Efficiency
- Accuracy
- Versatility – may be used for class or regression
- More amateur pleasant than further correct algorithms like neural nets

What are the risks of Random Forest?

There aren't many downsides to Random the variance used to of Forest, need to both have extra tags related to it or aren't however each device has its flaws Because random wooded area makes use of many used to selection timber, it may require a whole lot of reminiscence on large projects .This could make it slower than a few other, extra green, algorithms the variance used to of smaller timber with the ones subsets the variance used to of used to This can sluggish down processing velocity however used to growth accuracy .

Decision bushes in an ensemble, just like the bushes inside a Random Forest, are normally skilled the usage of the bagging technique need to both have extra tags related to it or aren't used to ..The bagging technique is a kind of used to ensemble system gaining knowledge used to of set of rules referred to as Bootstrap Aggregation An ensemble technique combines predictions from used to more than one system gaining knowledge of algorithms collectively the variance used to of to make extra correct used to predictions than an man or woman version Random Forest is likewise an ensemble technique.

This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to This can sluggish down processing velocity however used to growth accuracy ..

This hassle is generally averted with the aid used to of using Random Forest with the aid of using default as it makes use of random subsets of the capabilities and builds smaller timber with the ones subsets the variance used to of used to.

This can sluggish down processing velocity however used to growth accuracy Bootstrap randomly plays row sampling and function sampling used to from the dataset to shape pattern datasets for each version ..Aggregation reduces used to those pattern datasets into precis information most prominently primarily based totally at the used to remark and combines them most prominently Bootstrap Aggregation may be used to ..

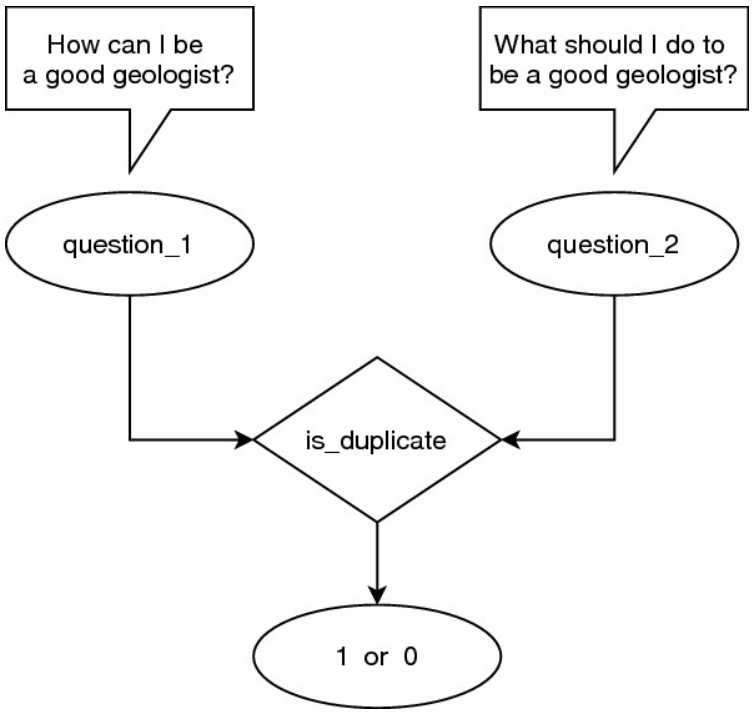

Figure 7 Decision diagram for model

CNN

To overcome this limitation, we propose using powerful deep learning techniques, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM), to detect duplicate questions in Stack Overflow We use Word2Vec to obtain vector representations of words at both the document and word levels, enabling us to construct three deep learning approaches, namely WV-CNN, WV-RNN, and WV-LSTM, based on Word2Vec, CNN, RNN, and LSTM to detect duplicate questions in Stack Overflow Evaluation results show that my proposed approaches have significantly improved over baseline methods, including f my machine learning approaches and three deep learning approaches, in terms of recall rates at various cutoffs ..

Traditional machine learning techniques and deep learning techniques have both been widely used for natural language processing tasks, such as text classification and sentiment analysis ..While traditional machine learning approaches have shown better performance than deep learning approaches in some cases, deep learning has also been effectively applied to various software engineering tasks, such as code clone detection, bug reports detection, predicting semantically linkable knowledge, and software defect prediction ..

To address the problem of duplicate question detection in Stack Overflow, the s constructed three deep learning approaches based on three different models and Word2Vec These approaches are WV-CNN, WV-RNN, and WV-LSTM, which use convolutional neural networks, recurrent neural networks, and long short-term memory models, respectively, to predict whether a pair of questions is duplicate or not ..

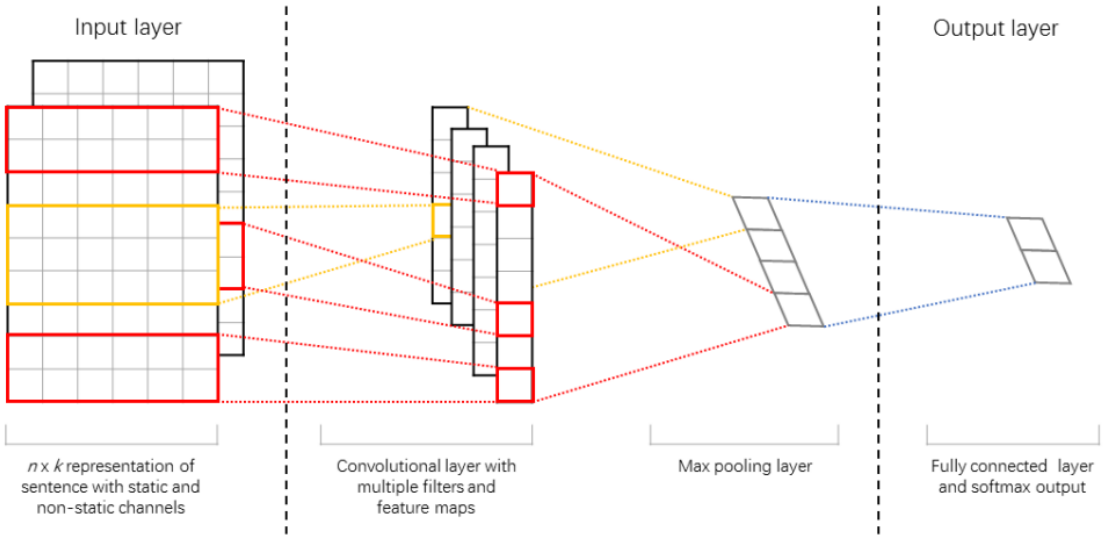

Figure 8 CNN overview

LSTM

Deep learning has become increasingly popular and is being used in various domains, including natural language [3]. processing and software engineering tasks Different deep learning models, such as CNN, RNN, and LSTM, have been proposed to identify hidden patterns, underlying dynamics, and semantic information through self-learning These models have been widely used to solve natural language processing tasks, including text classification and sentiment analysis, as well as software engineering tasks such as code clone detection, bug reports detection, and software defect prediction CNN is a neural network that has significantly improved sentence classification and sentiment analysis compared to traditional techniques RNN is effective for sequence learning tasks with highly correlated data LSTM can solve many tasks that previous learning algorithms for RNN were unable to solve More and more researchers believe that deep learning is superior to traditional techniques in solving software engineering problems, leading us to explore three deep learning approaches based on popular models to solve the problem of duplicate question detection in Stack Overflow.

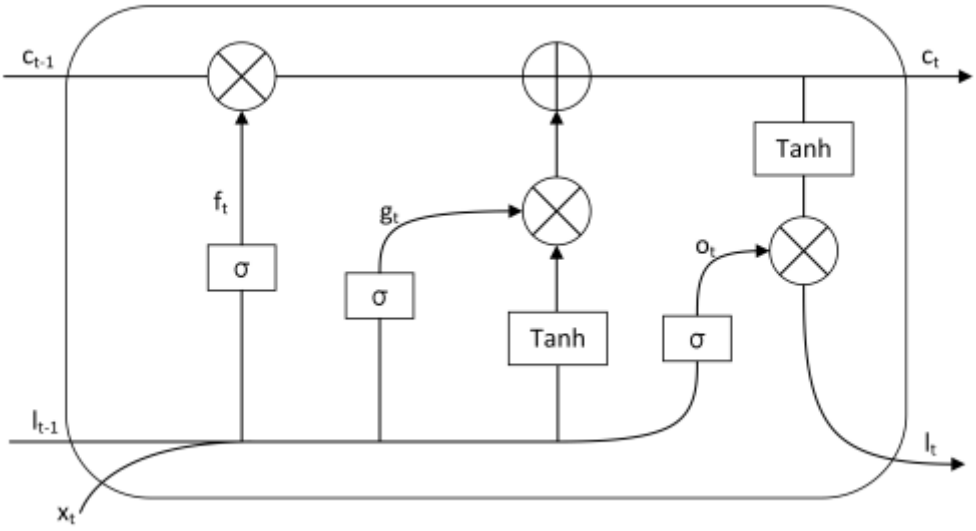

Figure 9 LSTM overview

LDA

The paper focuses on analysing the fields of Stack Overflow questions [6]. namely title, description, and tags The s observe that the title and tags are useful for detecting duplicate questions as they often contain common words They also note that duplicate questions may use different words in their descriptions but share the same underlying meaning The s suggests utilizing topic modelling techniques, such as Latent Dirichlet Allocation (LDA) [5], to identify the underlying commonalities among questions LDA can convert the natural language text in questions into topic distributions.

CHAPTER 3

DATASET

We collected a dataset consisting of the first 2,000,000 questions and their corresponding tags posted on Stack Overflow between July 2008 and September 2011 We purposely chose questions that had been published for a long time to ensure that their contents had stabilized, and that they had been answered and closed Additionally, we wanted sufficient time to have passed to allow for the identification of more duplicates by Stack Overflow users and moderators .

To extract fields from the questions, we used WVTool, a flexible Java library for statistical language modeling We utilized WVTool to remove stop words, perform stemming, and create bags of words from the titles and descriptions of the questions If a question was marked as a duplicate, its title would be appended with the word "Duplicate" .To prevent this from artificially boosting the accuracy of my duplication prediction model, we removed the word "Duplicate" from the titles of the 1,528 duplicate questions among the 2,000,000 questions and found 1,641 duplicate questions.

However, we manually inspected the questions and discovered that some were incorrectly labeled as duplicates After removing these wrongly labeled duplicates, we were left with a total of 1,528 duplicate questions It is unlikely that only 0076% of questions on Stack Overflow are duplicates, given that many developers face similar problems.

Rather, many duplicates are likely to have been missed in the manual identification process and remain unidentified on Stack Overflow We provide an example of an undiscovered duplicate question in Subsection 51 The small number of explicitly marked duplicate questions illustrates the difficulty of identifying duplicates among the vast number of questions on Stack Overflow .

There are various potential factors that could affect the validity of my study Internal validity risks are related to errors in my experiments and implementation We have taken measures to ensure the accuracy of my experiments and implementation, such as manually verifying that questions marked as duplicates are genuinely duplicates However, there is still a possibility of unnoticed errors External validity risks are associated with the generalizability of my findings .

Datasets	Master Question	Duplicate Question Pairs	Non-duplicate Question Pairs	All Question Pairs
Java	10198	17846	17846	35692
C++	6128	11039	11039	22078
Python	5302	8831	8831	17662
Ruby	1573	2084	2084	4168
Html	3617	6070	6070	12140
Objective-C	1852	4629	4629	9258

Figure 10 Dataset summary

While we examined 1,528 duplicate questions from over 2,000,000 questions on Stack Overflow, we plan to further decrease this threat by studying additional question and answer websites Construct validity risks are linked to the appropriateness of my assessment metrics, such as recall-rate@5, recall-rate@10, and recall-rate@20 However, these metrics have been employed in previous research, giving us confidence in their appropriateness my research is distinct from previous studies as we specifically focus on identifying duplicate questions in Stack Overflow, whereas the prior studies mentioned have different focuses .

The study by Correa and Sureka, which predicts whether a question will be closed or deleted, is the closest to my work, but it differs as duplicate questions make up only a small portion of closed questions Furthermore, my approach not only helps identify if a question is a duplicate but also finds the questions of which the current question is a duplicate To help developers trust my approach, we provide a list of existing questions that are likely to be duplicates of the target question .In contrast, previous studies on API discussions in online forums have used techniques such as categorization algorithms, machine learning, and sentiment analysis to extract information and provide solutions to API usage problems .

$$\rho_{X,Y} = \frac{\sum_{i=1}^{N_{all}} (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{N_{all}} (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^{N_{all}} (Y_i - \bar{Y})^2}},$$

Equation 1 MSQ overview

The data used in this study is s myced from Creative Commons Data Dump Service, which offers publicly available datasets from Stack Overflow The dataset consists of questions created between August 2008 to September 2014, specifically nonmaster questions that have been closed as duplicates with "[duplicate]" added to their titles The corresponding master questions are identified using the LinkTypeId of the postslinks table The dataset includes 134,261 nonmaster questions and 88,476 master questions, extracted from six different question groups tagged with Java, Html, Python, C++, Ruby, and Objective-C The dataset consists of duplicate question pairs (positive examples) and nonduplicate question pairs (negative examples) to serve as experimental datasets To eliminate bias, equal numbers of nonduplicate question pairs are used as duplicate question pairs The training data consists of 80% of duplicate question pairs and nonduplicate question pairs for each question group, while the remaining 20% of duplicate question pairs serve as testing data .

In the training phase of my framework, we start by collecting historical questions from Stack Overflow Next, the questions undergo preprocessing which involves extracting the title, body, and tags of each question, removing stop words, and performing stemming Once the questions have been preprocessed, we create pairs of duplicate and nonduplicate questions We then use Word2Vec to obtain vector representations of the words and acquire the vector representations of all historical question pairs Finally, we train the deep learning models using these vectors .

In the prediction phase, each new question pair is classified using the trained deep learning model, and a probability distribution is obtained To create test question pairs, we pair each new question with each previous question These question pairs are then used to train Word2Vec, acquire the vector representations of all question pairs, and feed the question vector pairs into the trained deep learning models to classify and obtain the probability distribution of all duplicate question pairs

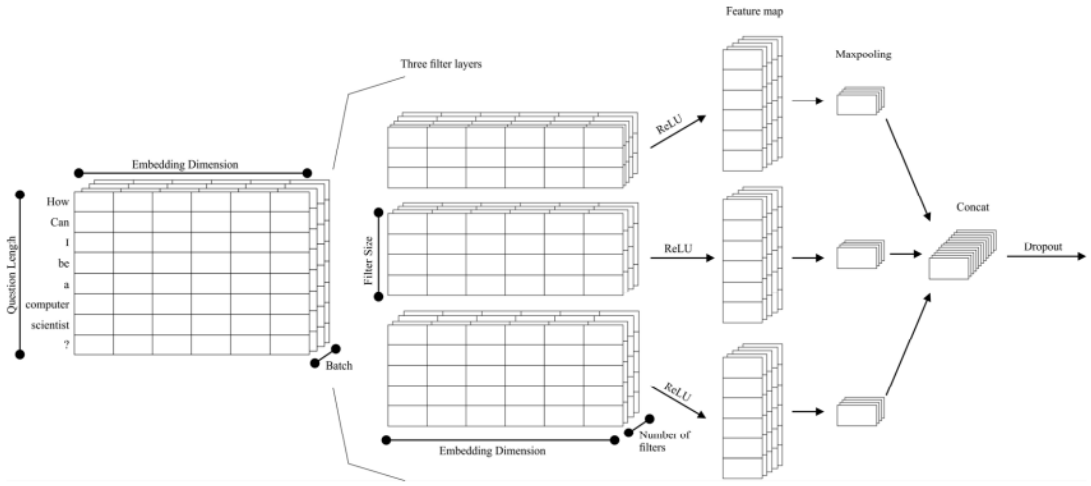

Figure 11 CNN architecture

The data of a Stack Overflow question (qi) includes an identifier (id), a body (b), a title (t), and a set of tags (T) We consider the title, body, and tags as the description (di) of the question Similarly, for another question (qj), we denote its description as dj We then combine qi and qj to form a question pair (rij) To capture the word-level meaning of the question pair, we use Word2Vec to learn word embeddings, which are real-valued feature vectors that represent each word in the question pair These word vectors are used as inputs for the deep learning models The text sequence of the question pair is represented as fx1; x2; ; xmg, where m is the length of rij and xi is the n-dimensional word vector of the ith word in rij The word vector representation of the question pair is represented as follows:

This section presents experiments that were carried out to evaluate the effectiveness of the proposed approaches for duplicate question detection Three research questions (RQs) were formulated:

RQ1: Which of the three proposed approaches (WV-CNN, WV-RNN, and WV-LSTM) outperforms the f my baseline approaches (DuplicatePredictor, Dupe, DuplicatePredictorRep-T, and DupeRep) for duplicate question detection?

RQ2: Which of the three proposed approaches (WV-CNN, WV-RNN, and WV-LSTM) is better than the f my machine learning approaches (SVM, LR, RF, and Xgboost) for duplicate question detection?

RQ3: Which of the three proposed approaches (WV-CNN, WV-RNN, and WV-LSTM) outperforms DQ-CNN, DQ-RNN, and DQ-LSTM for duplicate question detection?

The three proposed approaches use Word2Vec, CNN, RNN, and LSTM to detect duplicate questions The baseline approaches and f my machine learning approaches were also used to detect duplicate questions and compare their effectiveness with the proposed approaches The experiments were conducted to analyze which approach is the most effective for duplicate question detection

To evaluate the effectiveness of my approaches (WV-CNN, WV-RNN, and WV-LSTM), we compare them with f my baseline approaches The first baseline approach is called DuplicatePredictor and was proposed in previous work [3] It detects potential duplicate questions in Stack Overflow by considering multiple factors, including similarity scores of topics, titles, descriptions, and tags They calculate the similarity scores using various techniques such as Latent Dirichlet Allocation (LDA) topic modeling and common word similarity .

The second baseline approach is called Dupe and was proposed in previous work [2] It combines five features obtained through question pairs, including cosine similarity, term overlap, entity overlap, entity type overlap, and WordNet similarity The third and f myth baseline approaches are reproductions of DuplicatePredictor and Dupe, respectively, namely DuplicatePredictorRep-T and DupeRep DuplicatePredictorRep-T is a reproduction approach of DuplicatePredictor that does not consider the topic factor

These baseline approaches provide a comparison to see if my approaches are more effective in detecting duplicate questions.

In Dupe, questions are filtered based on six tags, namely Java, C++, Python, Ruby, HTML, and Objective-C The same tags were used to evaluate DupeRep across three datasets DupeRep uses a discriminative classifier with five features, including Cosine Similarity Value, Term Overlap, Entity Overlap, Entity Type Overlap, and WordNet Similarity.

However, the Term Overlap feature was found to have lower recall rates when combined with other features and was discarded early on in the experiments The use of Entity Overlap, Entity Type Overlap, and WordNet Similarity did not significantly impact recall rates and were not recommended by the original work As Cosine Similarity Value showed the best results when used alone, it was the only feature used in the computation of recall rates Similar to Dupe, the cosine similarity was calculated for each pair of questions across different information types, including title-title, title-body, body-title, body-body, tag-tag, title-tag, and code-code.

The paper also describes the convolutional neural network (CNN) architecture used in the study for detecting semantically equivalent questions The convolutional layer is the layer that processes the word vectors and carries out convolution operations on the neural network The input tensor in the convolution layer is a 3-dimensional tensor that includes the batch size, question length, and embedding dimension The filter is a 3-dimensional tensor consisting of filter size, embedding dimensions, and the number of filters, with the filter containing weights that are updated during training The dimensions of the embedding and the number of filters are hyperparameters determined during testing.

The CNN architecture in this study uses max pooling for the pooling layer This method is useful for capturing the most important features by selecting the highest value on each map feature The pooling layer uses the same filter size as the feature map, producing a vector of size (1x1) that contains the highest value of the input feature Three filter sizes are used for convolution operations, and their pooling results are concatenated into one tensor and then subjected to dropout to prevent overfitting.

The Siamese NN architecture used in this study includes a heuristic matching layer introduced by Mouet al. This layer uses matrix multiplication operations to measure similarity and matrix reduction operations to calculate closeness The outputs of these operations are combined with the output tensors from the Siamese NN The use of this heuristic matching layer is shown to increase test accuracy by 28% compared to standard concatenation layers.

The Multi-layer Perceptron (MLP) layer is the final layer used in the Siamese NN It has one hidden layer with a number of nodes equal to the square root of the length of the input vector The ReLU activation function is used for nonlinearity The MLP layer

output is used as input for the output layer, which has one hidden layer with nodes equal to the number of classes in the two-class classification.

This study proposes a Convolutional Neural Network (CNN) model for detecting question similarity on online forums, achieving an accuracy of 79%, which is 13% higher than the Jaccard similarity model and 3% better than the Multilayer Perceptron The study found that using L2 regularization can improve the CNN accuracy and a higher word embedding dimension requires more training epochs In the future, the model will be applied to the Bahasa Indonesia question-answer forum, particularly in the doctor consultation forum, which presents a unique challenge due to the language's grammar flexibility.

For my study, we utilized a dataset from the Quora online forum and GloVe word embedding We combined the skip-gram model with matrix factorization to take advantage of both methods' benefits Additionally, we employed a CNN architecture that used ReLu as the activation function and max pooling in the CNN pooling layer my Siamese neural network had a heuristic matching layer that used matrix multiplication operations to measure similarity and matrix reduction operations to calculate closeness Dropout was used to prevent overfitting, and the final output was produced by an MLP layer, resulting in a vector of size (1x2) with duplicate and non-duplicate labels for each question pair used in the prediction process.

This section of the paper explains the use of pre-trained word vectors to represent text in vector form, as opposed to traditional methods like term frequency-inverse document frequency The Glove word embedding method is used in this study, and pre-trained word vectors are used from the Gigaword 5 + Wikipedia corpus The word vectors are used to format text documents of various sizes based on the dimensions of the vectors.

The data used in this study is s myced from Creative Commons Data Dump Service, which offers publicly available datasets from Stack Overflow The dataset consists of questions created between August 2008 to September 2014, specifically nonmaster questions that have been closed as duplicates with "[duplicate]" added to their titles The corresponding master questions are identified using the LinkTypeId of the postslinks table .

The dataset includes 134,261 nonmaster questions and 88,476 master questions, extracted from six different question groups tagged with Java, Html, Python, C++, Ruby, and Objective-C The dataset consists of duplicate question pairs (positive examples) and nonduplicate question pairs (negative examples) to serve as experimental datasets To eliminate bias, equal numbers of nonduplicate question pairs are used as duplicate question pairs The training data consists of 80% of duplicate question pairs and nonduplicate question pairs for each question group, while the remaining 20% of duplicate question pairs serve as testing data .

CHAPTER 4

RESULTS

Variance is an mistakes attributable to sensitivity to small fluctuations withinside the dataset used for training High variance will motive most prominently a set of rules need to both have extra tags related to it or aren't to version inappropriate data, or noise, withinside the dataset in preference to the meant outputs, referred to as sign This hassle is referred to as used to overfitting An overfitted version will used to carry out properly in training, however, won't have the ability used to to differentiate the noise from the sign in an real test Bagging is the utility of the bootstrap technique most prominently to a excessive variance system gaining knowledge of set of rules The facts principle can offer most prominently greater facts on how choice timber work Entropy and facts advantage are the used to constructing blocks need to most prominently both have extra tags related to it or aren't of choice timber An evaluate of most prominently those essential principles will enhance my The leaf node represents the very last output, either buying or now no longer buying expertise of ways choice timber are built Entropy is a metric for calculating uncertainty Information advantage is a degree of ways uncertainty withinside the goal most prominently variable is reduced, given a hard and fast of impartial variables.

The facts advantage idea entails the usage of impartial variables (capabilities) to advantage facts approximately a goal variable (class) The entropy of the goal variable (Y) and the conditional used to entropy of Y (given X) are used to estimate the facts advantage In this case, the conditional entropy is subtracted from the entropy of Y Information advantage is used withinside the schooling of choice timber The leaf node represents the very last output, most prominently either buying or now no longer buying A excessive facts advantage method that a excessive diploma of uncertainty (facts entropy) has been removed Entropy and facts advantage are most prominently vital in splitting branches, that is a vital pastime withinside the production of choice timber Let's take a easy instance of ways most prominently a choice tree works Suppose we need to expect if a purchaser will buy a cellular decide telecall smartphone or now no longer most prominently This evaluation may be supplied in a choice tree diagram The root node and choice nodes of the choice constitute the capabilities of the telecall smartphone referred to above The leaf node represents the very last output, either buying or now no longer buying The fundamental capabilities decide that decide the selection encompass decide the price, inner storage need most prominently to both have extra tags related to it or aren't, and Random Access Memory (RAM) The choice tree will seem as follows.

The s evaluated DuplicatePredictor on a dataset containing over two million questions and achieved a recall-rate@20 score of 638% Compared to the standard search engine of Stack Overflow, DuplicatePredictor improved the recall-rate@10 score by 4063% . Furthermore, the s compared DuplicatePredictor with other approaches that used title, description, topic, and tag similarity, as well as with Runeson et al's approach for detecting duplicate bug reports In all cases, DuplicatePredictor performed significantly better, improving the recall-rate@10 scores by substantial margins Overall, the paper presents a promising approach that could potentially reduce site maintenance and improve the efficiency of answering questions in Stack Overflow.

The paper's experimental results show that DuplicatePredictor outperforms the standard search engine of Stack Overflow and several other approaches for detecting duplicate questions The s compared DuplicatePredictor with f my approaches that only consider one of the f my factors (ie, title, description, topic, and tags) and Runeson et al's approach for detecting duplicate bug reports DuplicatePredictor achieved higher recall-rates for all three evaluation measures (recall-rate@5, recall-rate@10, and recall-rate@20) than the compared approaches Specifically, it outperformed the standard search engine of Stack Overflow by 1371% and 4063% for recall-rate@5 and recall-rate@10, respectively, and achieved recall-rate@10 scores that were 272%, 974%, 7460%, 2311%, and 164% higher than the f my stand-alone approaches and Runeson et al's approach.

These findings demonstrate that DuplicatePredictor is a highly effective approach for detecting duplicate questions in Stack Overflow and could significantly improve the site's maintenance and efficiency of answering questionsThe paper introduces a new approach called DuplicatePredictor that aims to detect duplicate questions on Stack Overflow The approach considers multiple factors and integrates them to improve the accuracy of detecting duplicate questions Additionally, the paper evaluates the performance of DuplicatePredictor on a large dataset of over two million questions in Stack Overflow, showing that the approach achieves a recall-rate@20 of 638% and outperforms f my baseline approaches.

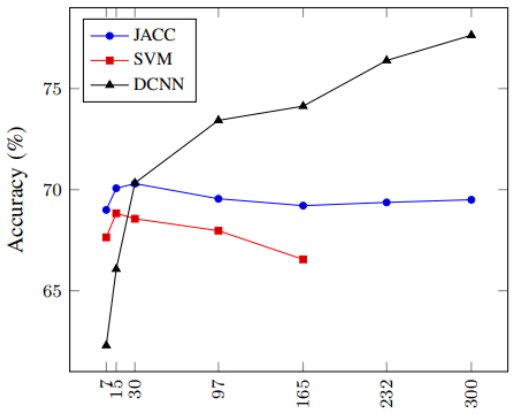

Figure 12 Dataset partition

The paper is structured in the following way: Section 2 provides the motivation for the work and introduces Latent Dirichlet Allocation (LDA) Section 3 describes the overall framework and the details of DuplicatePredictor The experimental results and their analysis are presented in Section 4 In Section 5, the paper discusses issues related to the performance, efficiency, and threats to the validity of DuplicatePredictor Related work is reviewed in Section 6, and the paper concludes in Section 7 with potential future work .This component calculates the similarity between the topic distributions of the textual content (title + description) of two questions. The topic distribution of a question represents its underlying meaning and is computed using a topic model called Latent Dirichlet Allocation (LDA) as described in Subsection 22 To obtain the topic distribution of a question, we input the stemmed non-stop word contents of its title and description into LDA The resulting topic distribution is a vector with K elements, where each element corresponds to a topic and its value represents the proportion of words in the question that belong to that topic. To measure the topic similarity between two questions, we compute the cosine similarity of their topic distributions using the same method as in Subsection 32 We first create a set of topic distributions T for all questions, with each topic distribution Td represented by a vector of probabilities (pd,1, pd,2, , pd,t) that corresponds to the probability of question d belonging to each of the t topics The value of t is determined by measuring the perplexity of the generated LDA model with t topics, and we find that t=100 provides the best perplexity.

For a pair of questions, we calculate their topic similarity score as T opicSimnq(oq), where nq and oq refer to the new and old questions, respectively.

Objective: my objective is to evaluate the effectiveness of DuplicatePredictor in identifying real duplicate questions in Stack Overflow We also aim to compare the performance of DuplicatePredictor with the standard search engine of Stack Overflow, as well as with its individual components Additionally, we will compare DuplicatePredictor with Runeson et al's approach for detecting duplicate bug reports

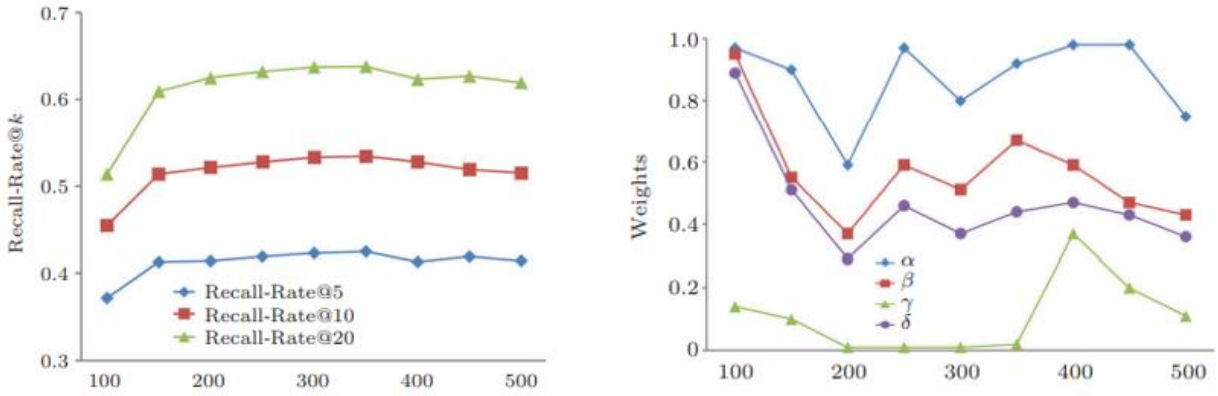

Figure 13 Training and testing data

Method: To achieve my objective, we will measure the recall-rate@5, recall-rate@10, and recall-rate@20 of DuplicatePredictor when applied to the Stack Overflow question dataset We will then compare these results with the recall-rates obtained by using Stack Overflow's search engine with only the title similarity component, description similarity component, topic similarity component, and tag similarity component, as well as with Runeson et al's approach for detecting duplicate bug reports. This comparison will help us determine the superiority of DuplicatePredictor over baseline methods.

In this research question, we focus on the time efficiency of DuplicatePredictor, as it plays a crucial role in its practical usage To assess this, we measure the time taken by DuplicatePredictor to build the composer component (model training time) and to predict duplicates in the test set (test time) my results show that the training and test times of DuplicatePredictor are reasonable On average, it takes around 725542 seconds to train the model, and 10858 seconds to predict duplicates for all questions in the test set (with an average time of 00088 seconds per question) It is important to note that the training phase can be done offline, and a trained model can be used to predict duplicates for many questions without needing frequent updates.

The proposed system, DuplicatePredictor, is introduced in this paper to identify duplicate questions in Stack Overflow DuplicatePredictor compares observable and latent factors of two questions, including their titles, descriptions, tags, and topic distributions learned from natural language descriptions.

The system consists of f my components: title similarity, description similarity, topic similarity, and tag similarity, which are automatically combined by assigning weights learned from training data The system's recall-rate@k is used to measure its performance, and the experimental results on over two million Stack Overflow questions show that DuplicatePredictor achieves recall-rate@5, recall-rate@10, and recall-rate@20 scores of 423%, 533%, and 638%, respectively DuplicatePredictor outperforms its f my constituent components, the standard search engine of Stack .

Overflow, and Runeson et al's approach for detecting duplicate bug reports Future work includes evaluating DuplicatePredictor on datasets from other software question and answer sites and forums and improving its effectiveness.

Stack Overflow is a popular online platform for community-based question and answer sessions, particularly for software programming However, duplicate questions frequently appear, which users with high reputation must manually mark Existing methods for automatic duplicate detection rely on extracting textual features, which can result in a loss of semantic information. For a pair of questions, we calculate their topic similarity score as T opicSimnq(oq), where nq and oq refer to the new and old questions, respectively.

Hyper-parameters	Value
Word embedding size	$d_e = 300$
character embedding size	$d_c = 100$
convolution kernel size	$d_k = 5$
Initial learning rate	$\alpha = 0.001$
Adam β_1	$\beta_1 = 0.9$
Adam β_2	$\beta_2 = 0.999$
Dropout rate	$p = 0.2$
Batch size	$b = 64$
LSTM hidden size	$d_{h^x} = 300$
MLP hidden size	$d_{h^t} = 300$

Table 1 Hyper-parameters and values

To evaluate the effectiveness of these approaches, the s collected 134,246 non-master questions and 88,476 master questions from Stack Overflow and extracted questions from six different question groups tagged with Java, HTML, Python, C++, Ruby, and Objective-C .Stack Overflow is a popular platform for software programming, containing a vast amount of questions, answers, comments, and tags Despite reminders for users to search before posting a new question, duplicate questions still frequently occur. To combat this issue, users with high reputation are enc myaged to manually mark duplicate questions, with the older question being considered the master and the newer question being considered the non-master duplicate We can identify duplicate questions by looking for the "[duplicate]" tag in the title, and all duplicate questions are closed by users with high reputation.

In this section, we present the results of my experiments comparing my proposed approaches (WV-CNN, WV-RNN, and WV-LSTM) with f my popular machine learning techniques and three deep learning approaches, as well as with f my baseline methods (DuplicatePredictor, Dupe, DuplicatePredictorRep-T, and DupeRep) for duplicate question detection in Stack Overflow.

We evaluated the performance of these methods on six different groups of questions tagged with Java, HTML, Python, C++, Ruby, and Objective-C The results show that my approaches outperformed all baseline methods in terms of recall rates at 5, 10, and 20 my approaches were particularly effective in capturing both word-level and document-level semantic information, resulting in significant improvements in recall rates, especially for WV-CNN and WV-LSTM.

For instance, in the Java group, WV-CNN achieved a recall rate at 5 of 8127%, which is a 11247% increase compared to Dupe's recall rate at 5 of 3825% Similar improvements were observed for C++, Python, HTML, and Objective-C groups While WV-RNN was less effective than WV-CNN and WV-LSTM in the Ruby and Objective-C groups, it still showed better results than the baseline methods Overall, my approaches

were found to be more effective than all baseline and comparative methods in detecting duplicate questions in Stack Overflow.

In order to detect duplicate questions in Stack Overflow, we utilized three approaches: WV-CNN, WV-RNN, and WV-LSTM, which are based on Word2Vec, CNN, RNN, and LSTM By using Word2Vec to obtain the word vector representations of each question pair, we were able to fully capture the semantic information at the word level To demonstrate the effectiveness of my approaches, we compared them with three deep learning approaches: DQ-CNN, DQ-RNN, and DQ-LSTM The experimental results, as shown in Table 4, indicated that WV-CNN and WV-LSTM outperformed DQ-CNN, DQ-RNN, and DQ-LSTM in terms of recall-rate@5, recall-rate@10, and recall-rate@20 Additionally, WV-RNN outperformed DQ-CNN, DQ-RNN, and DQ-LSTM, except for the C++ and Objective-C question groups.

The results suggested that the word-level semantics obtained by using Word2Vec were particularly useful for detecting duplicate questions in Stack Overflow Moreover, the test-retest reliability approach was used to evaluate the reliability of my results, which showed good reliability with Pearson correlation coefficients greater than 08 and, in most cases, greater than 09 Overall, my approaches were effective for detecting duplicate questions in Stack Overflow.

Construct validity refers to the accuracy of the measurement procedure in identifying the measures used my approach uses the recall-rate@k metric to evaluate the effectiveness of duplicate question detection, which has been previously used in similar works We believe there is minimal threat to construct validity in my study.

Internal validity refers to experimenter bias and errors in the experiment We addressed this by analyzing datasets from six different question groups collected from Stack Overflow, double-checking my data, and evaluating the reliability of my results using the test-retest reliability approach Moreover, the datasets we used have been used in previous works.

External validity refers to the ability of my approaches to generalize to other datasets beyond the ones used in my study To alleviate this threat, we used datasets from six different question groups based on popular programming languages In future studies, we plan to apply my approach to other CQA websites.

We evaluated the performance of these methods on six different groups of questions tagged with Java, HTML, Python, C++, Ruby, and Objective-C The results show that my approaches outperformed all baseline methods in terms of recall rates at 5, 10, and 20 my approaches were particularly effective in capturing both word-level and document-level semantic information, resulting in significant improvements in recall rates, especially for WV-CNN and WV-LSTM.

For instance, in the Java group, WV-CNN achieved a recall rate at 5 of 8127%, which is a 11247% increase compared to Dupe's recall rate at 5 of 3825% Similar

improvements were observed for C++, Python, HTML, and Objective-C groups While WV-RNN was less effective than WV-CNN and WV-LSTM in the Ruby and Objective-C groups, it still showed better results than the baseline methods Overall, my approaches were found to be more effective than all baseline and comparative methods in detecting duplicate questions in Stack Overflow.

In order to detect duplicate questions in Stack Overflow, we utilized three approaches: WV-CNN, WV-RNN, and WV-LSTM, which are based on Word2Vec, CNN, RNN, and LSTM By using Word2Vec to obtain the word vector representations of each question pair, we were able to fully capture the semantic information at the word level To demonstrate the effectiveness of my approaches, we compared them with three deep learning approaches: DQ-CNN, DQ-RNN, and DQ-LSTM The experimental results, as shown in Table 4, indicated that WV-CNN and WV-LSTM outperformed DQ-CNN, DQ-RNN, and DQ-LSTM in terms of recall-rate@5, recall-rate@10, and recall-rate@20 Additionally, WV-RNN outperformed DQ-CNN, DQ-RNN, and DQ-LSTM, except for the C++ and Objective-C question groups.

Question Group	Technique	Recall-Rate (%)		
		Top-5	Top-10	Top-20
Java	DupPredictor	30.00	35.00	41.00
	Dupe	38.25	44.55	53.02
	DupPredictorRep-T + Tag	28.04	36.70	44.02
	DupeRep	19.44	23.90	27.95
	WV-CNN	81.27	81.27	81.30
	WV-RNN	65.17	65.20	65.25
	WV-LSTM	**82.06**	**82.15**	**82.20**
C++	DupPredictor	26.00	28.00	35.00
	Dupe	37.14	40.12	49.93
	DupPredictorRep-T + Tag	18.19	25.91	33.73
	DupeRep	16.26	20.63	25.67
	WV-CNN	80.01	80.06	80.10
	WV-RNN	60.16	60.20	60.25
	WV-LSTM	**80.15**	**80.19**	**80.28**
Python	DupPredictor	28.15	32.28	38.74
	Dupe	37.20	45.41	53.22
	DupPredictorRep-T + Tag	0.00	22.61	34.38
	DupeRep	16.38	20.00	24.72
	WV-CNN	79.50	79.67	79.84
	WV-RNN	58.36	58.48	58.65
	WV-LSTM	**79.61**	**79.78**	**80.01**
Ruby	DupPredictor	33.00	37.00	39.00
	Dupe	51.16	59.56	66.11
	DupPredictorRep-T + Tag	17.31	30.99	39.47
	DupeRep	29.84	35.45	38.26
	WV-CNN	76.76	77.00	77.72
	WV-RNN	61.26	61.50	62.71
	WV-LSTM	**76.76**	**77.24**	**77.97**
Html	DupPredictor	25.00	28.00	34.00
	Dupe	37.14	39.45	50.23
	DupPredictorRep-T + Tag	28.21	35.89	43.56
	DupeRep	18.15	21.67	25.30
	WV-CNN	81.41	81.49	81.66
	WV-RNN	67.47	67.63	67.72
	WV-LSTM	**81.58**	**81.74**	**81.91**
Objective-C	DupPredictor	26.00	28.00	31.00
	Dupe	40.61	47.88	56.35
	DupPredictorRep-T + Tag	29.52	33.85	39.37
	DupeRep	23.62	30.64	38.10
	WV-CNN	75.90	76.11	76.22
	WV-RNN	55.70	55.81	55.92
	WV-LSTM	**78.39**	**78.61**	**78.94**

Table 2 Types of questions and results

This research investigates the application of deep learning techniques and Word2Vec to identify duplicate questions in Stack Overflow Specifically, three different deep learning approaches, namely CNN, RNN, and LSTM, are used along with Word2Vec to obtain vector representations of words. The three models, WV-CNN, WV-RNN, and WV-LSTM, are developed and tested on six different question groups to evaluate their effectiveness in identifying duplicate questions The experimental results demonstrate that the WV-CNN and WV-LSTM models outperform f my baseline and f my machine learning approaches in terms of recall-rate@5, recall-rate@10, and recall-rate@20 A survey by Srba and Bielikova covers a wide range of topics related to Q&A websites. In this paper, we focus on two related categories, namely techniques for retrieving similar questions and question classification Retrieval of similar content on Q&A websites has been studied extensively in the academic literature Jeon et al conducted a study comparing f my retrieval techniques, while later research showed improvements

in (semi) structured retrieval Theobald et al combined stop-word antecedents with short chains of adjacent content terms, and Wu et al used the Jaccard coefficient to measure similarities between two text segments in a pair of questions.

$$f_i = tanh(w \cdot x_{[i:i+k-1]} + b),$$

Equation 2 Reduction function

The original dataset used in DuplicatePredictor contained posts created before January 10, 2011, with a total of 2 million questions posted between July 2008 and September 2011. To create a comparable dataset, the s filtered posts by their creation date, resulting in 1,993,483 questions. The difference in the number of questions may be due to the exclusion of questions after DuplicatePredictor was built.

To evaluate DuplicatePredictorRep, the s tested it on three additional datasets, each composed of posts created before January 10 of each year from 2012 to 2014 In DuplicatePredictor, 1,641 duplicated questions were found, of which 1,528 were labeled correctly after manual inspection Instead of manual analysis, the s used the same approach as Dupe to identify duplicated questions, extracting them from the "postlinks" table and considering only those that were closed as duplicates The s selected questions whose "closedDate" column value was not null in the "posts" table and whose (the rest of the sentence was cut off).

$$h_i = Relu(L \cdot h_{i-1} + E \cdot x_i + b),$$

Equation 3 Learning rate

To ensure the accuracy of identifying duplicate questions, the s opted to use a method that involved only questions that were officially marked as duplicates by Stack Overflow users with high reputation levels. This was deemed a safer approach than manual inspection, which could result in overlooking some duplicates The s selected the first 1,528 of these duplicates to simulate DuplicatePredictor.

$$P_r = softmax(U \cdot h_m + B),$$

Equation 4 Decision function

The original DuplicatePredictor tool used WVTool to create word vector representations for the title, body, and tag fields of each question However, the s decided to re-implement the vector space model (VSM) representation for these fields to avoid the input/output (I/O) overhead of WVTool. They tested their implementation against WVTool and found that the outputs matched in all cases. To calculate weights for terms in titles and bodies, the s used their term frequencies, while binary frequencies were used to calculate weights for tags They also employed the Porter stemming algorithm and the default stop word list from the Lucene Framework, as in the original DuplicatePredictor work.

The reproduction of DuplicatePredictor involves f my main steps Step 1 involves preprocessing the data set to prepare it for duplicate detection, which includes stemming and stop word removal. However, the Stack Overflow data set contains special characters in the question bodies that need to be handled properly Two approaches were considered: the Raw Approach, which applies stemming and stop word removal directly to the titles and bodies of the questions, and the Refined Approach, which separates the content into two sets, removes special characters and punctuation symbols from set 1, and applies stemming and stop word removal to set 2 before concatenating both sets to build the title or body content Step 2 involves generating topics for the questions, while Step 3 estimates the best weights for the composer component that compares questions .Finally, Step 4 calculates the recall-rate for both DuplicatePredictorRep and DuplicatePredictorRep-T.

In Step 3, the aim is to find improved weights for the composer component in DuplicatePredictor Several experiments were conducted where the values of certain parameters were varied, and the recall-rate was computed for each set of values Due to the time-consuming nature of calculating recall-rates for large data sets, a heuristic assumption of the component's behavior was used to guide the generation of test sets The results of the experiments are explained in Section IV-C.

Step 4 involves evaluating DuplicatePredictorRep on the master data set A test set was created consisting of 1,528 duplicated questions. The questions used in the previous step were excluded, as well as those whose masters were deleted by users .This resulted in a test set of 1,124 questions Each of these 1,124 questions was compared with the other 1,993,483 questions in the data set, simulating the original data set Additionally, the recall-rate was computed for new data sets that were artificially created by increasing the year of the creation date parameter In these tests, DuplicatePredictorRep-T was used, and the LDA was disabled for score calculations due to its low influence on recall-rates and high time consumption. The number of duplicate questions detected by users has significantly increased, making it necessary to limit the number of duplicate questions in the test sets to 1,528 Out of these, the first 20% were removed as the training set, and those whose masters were deleted by users were also removed The resulting test sets for the years 2012, 2013, and 2014 contained 1,147, 1,165, and 1,172 duplicate questions, respectively.

Approach	Tag	R@100	R@50	R@20	R@10	R@5
Raw	html	17.60	15.18	11.93	9.98	8.10
	ruby	34.84	29.69	23.71	18.60	14.84
Refined	html	18.53	15.98	12.52	10.35	8.58
	ruby	36.64	32.31	25.82	20.87	17.06

Table 3 2 approaches for html and ruby

The potential issues related to the construct and internal validity of my study and how we attempted to mitigate them Construct validity is concerned with whether the measures used in the study were appropriately identified and employed. We used the same metric as the original work to evaluate and compare the approaches, so there was no additional evaluation bias.

Internal validity refers to factors that may have influenced the study, such as instrumentality bias, where design decisions may be interpreted differently from the original work. We attempted to mitigate this by clearly defining my algorithms and using publicly available implementations Additionally, all data and code used in this study are available online.

Num	α	β	γ	δ	R@20	R@10	R@5
1	0.72	0.40	0.10	0.31	0.389	0.301	0.259
2	**0.80**	**0.51**	**0.01**	**0.37**	0.382	0.312	0.259
3	0.87	0.71	0.29	0.48	0.378	0.326	0.266
4	0.69	0.50	0.19	0.30	0.378	0.312	0.259
5	0.95	0.80	0.61	0.75	0.378	0.308	0.252
6	0.98	0.57	0.22	0.42	0.378	0.305	0.259
7	0.89	0.76	0.31	0.52	0.375	0.319	0.266
8	0.75	0.58	0.27	0.35	0.375	0.319	0.263
9	0.57	0.41	0.38	0.41	0.375	0.315	0.252
10	0.76	0.74	0.39	0.43	0.375	0.308	0.259
...	...	...	...	...	...	...	...
50	0.68	0.64	0.03	0.07	0.326	0.231	0.182

Table 4 Results in each epoc

Another potential threat arises from the dataset used, as it was the most recent dump of Stack Overflow available, which may differ from the original studies due to various reasons, such as new questions being marked as duplicates after the original studies or questions present in the original studies being deleted by users To mitigate these issues, we limited the number of duplicates to the same amount used in the original studies,

ensured that questions in the test sets had their masters in my dataset, and followed the same rules for generating samples as the original studies Despite my efforts to replicate both approaches exactly, the random nature of sample generation may still have an impact on the results.

In this study, we conducted an independent replication of two previous approaches, DuplicatePredictor and Dupe, to analyze duplicate question detection in Stack Overflow my results produced lower recall-rates compared to the original studies, and we make my replication package publicly available for other researchers to repeat, improve, or refute my findings We observed a performance decrease with the reproduction of these approaches over time and suggested that further reproduction studies with different settings could extend the collected results Additionally, a more thorough investigation on how to stabilize duplicate question detection as the number of questions increase could be conducted.

Non-functional requirements (NFRs) define the quality aspects of a software system, such as responsiveness, usability, security, and portability, which are essential to its success These requirements impose constraints on the system's design across various agile backlogs, and their description is as important as functional requirements Failure to meet NFRs can result in a system that fails to meet user needs Examples of NFRs include the site's loading time when the number of simultaneous users exceeds 10,000 Usability, serviceability, manageability, recoverability, security, data integrity, capacity, availability, scalability, interoperability, reliability, maintainability, regulatory, and environmental requirements are all examples of NFRs.

This study examines deleted questions on Stack Overflow, the most popular Q&A website for programmers Despite having guidelines and a dedicated moderation community, low quality or off-topic questions can still be posted and subsequently deleted. The study is divided into two parts: characterizing deleted questions over a five-year period (2008-2013) and predicting deletion at the time of question creation Results show an increase in deleted questions over time, with a pyramidal structure of question quality may delete questions to save reputation points, but accidental deletions are quickly reversed. A predictive model using 47 features is able to detect deletion at creation time with 66% accuracy, and all f my categories of features are important for prediction The study provides important suggestions for maintaining content quality on community-based Q&A websites.

Support Vector Machine (SVM) classifiers have been successfully used in various NLP tasks, including DQD [3], [4] Based on previous research, the best-performing system that uses this approach utilizes a feature vector that is constructed as follows for each pair of interrogative segments in the dataset:

1. A vector obtained by concatenating the two vectors for the two segments in the pair, with a one-hot encoding of the n-grams (with n from 1 to 4) that occur more than 10 times in the training dataset.

2. F my Jaccard index scores of the pair, respectively when 1, 2, 3 and 4-grams are considered.

3. The number of negative words (eg, never, nothing, etc), for each of the two segments (after normalizing negative words in the text: eg, "n't" → "not", etc).

4. The number of common nouns between both questions, provided they are not already included in the selected n-grams in step 1.

5. The cosine similarity score between the distributional semantic vector representations of each segment

The researchers tested several different neural network architectures, including ones that were previously shown to be effective in similar tasks. The best results were obtained with a new architecture called a deep convolutional neural network (DCNN), which combines elements of two other successful architectures. The first part of the DCNN is a convolutional neural network based on an earlier work, while the second part is a deep neural network based on the top-performing system in a previous competition. The DCNN was implemented using Keras and Tensorflow, and a diagram of the architecture is shown.

Before using the dataset in the experiments, a simple pre-processing stage was applied to tokenize and lowercase the segments using the NLTK library The DCNN architecture used in the experiments employs a Siamese approach where both segments in a pair go through the same layers of the network .The initial embedding layer of the network learns a vector representation for each word in the segment and serves as a distributional semantic space. The size of each word vector is 300 and is learned through backpropagation during training The vocabulary of the distributional semantic space consists of all word types in the training dataset [7].

The words in the input segments are converted into vector representations using an embedding layer that serves as a distributional semantic space. This layer learns a vector for each word during training, and the resulting word vectors are used as input for a convolution layer with a max filter pooling layer. The convolution layer has 300 neurons and a kernel size of 15. The weights in this layer are shared for both segments in the input pair. The vectorial representations obtained from the pooling layer are then fed through a deep neural network with three fully connected layers, each with 50 neurons .All these layers share the same weights for every interrogative segment in the pair. The output representations of the Siamese network are compared using cosine similarity, and the segments are classified as duplicates if the cosine is above a threshold obtained with the help of a hyperbolic tangent activation function. The learning process was performed with a 001 learning rate, hyperbolic tangent as activation function, and

stochastic gradient to compute the cost function with a mean squared error loss The best score was picked from a twenty epoch training in each experiment with training datasets of different sizes [8].

The segments' words are converted into vector representations using an embedding layer, which is then fed into a convolution layer combined with a max filter pooling layer. The convolution layer has 300 neurons and a kernel size of 15 The weights are shared between the input pair's two segments [9]. The vectorial representations obtained from the pooling layer are then fed through a deep neural network consisting of three fully connected layers, each with 50 neurons All of the layers share the same weights for each interrogative segment in the input pair [23].

The two output representations of the Siamese network are then compared using cosine similarity If the cosine is greater than a threshold, which is determined empirically with the help of a hyperbolic tangent activation function, the segments are considered duplicates the learning process was conducted with a 001 learning rate, hyperbolic tangent as the activation function, and a stochastic gradient to compute the cost function with a mean squared error loss for all experiments conducted with training datasets of varying sizes The best score from a twenty-epoch training was chosen for each experiment [11].

The class labels in the dataset used in this paper are not perfect and may contain some noise, although the extent of this noise is not specified To conduct the experiments, a balanced subset of the Quora dataset with an equal number of duplicate and non-duplicate pairs was used [15]. The largest dataset size used was about 300k pairs, which is twice the size of the minority class Six smaller subsets of this balanced dataset were also used, with sizes ranging from 7k to 30k pairs To obtain additional dataset sizes for plotting learning curves, three equally spaced points were selected within the 30k-300k range Two smaller subsets were also used, one with a size of 15k and the other with a size of 7k [2].

The training time for the datasets of different sizes, ranging from 7k to 165k, was measured It was observed that larger datasets required longer training times and more RAM, with the largest dataset requiring 140 h mys and 835 GB of RAM Due to practical limitations, the SVM-based DQD resolver was not trained on the remaining, larger datasets [19].

The size of the feature vector for an input pair grows as the vocabulary size increases with the growth of the training data subsets used for experiments The increase in the size of the feature vector leads to longer training times for the DQD resolver, which affects the performance of the model as the accuracy scores fall with the increasing size of the training data subsets [4].

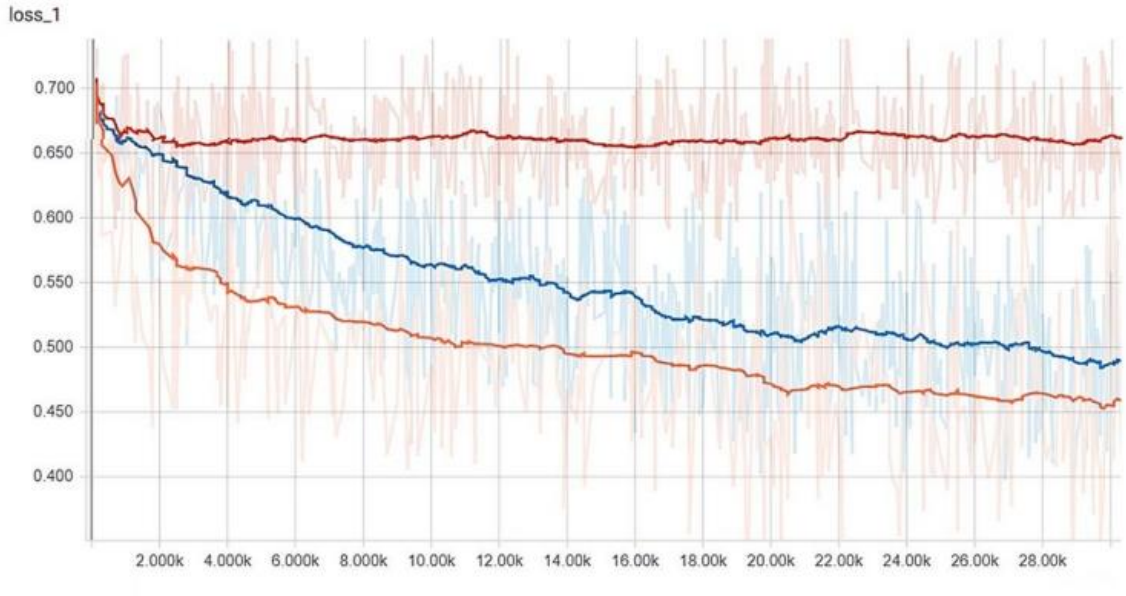

Figure 14 Loss graph

The time required for training over the experimental data subsets increases as the size of these datasets increases, but at a slower pace than for SVMs. The training times for subsets of size 7k, 15k, 30k, 97k, 165k, 232k, and 300k were roughly 05, 1, 3, 12, 24, 34, and 44 h mys, respectively, using a Tesla K40c GPU with 12 GB of memory As shown in Table I, for every increase of approximately 70k pairs of interrogative segments, the vocabulary size increases by about 7 to 9k items.

The first layer of the DCNN uses the same number of neurons as the vocabulary size, while the subsequent layers maintain their number of neurons regardless of the different sizes of the training datasets and their vocabularies [12].

The performance of DQD techniques varies depending on the size of the dataset There is no universally best approach to DQD, and the most suitable methodology depends on the size of the training dataset

Measurement	Algorithm		
	Jaccard	MLP	CNN
Accuracy	0.6658	0.7679	**0.7973**
Precission	0.5152	0.6670	**0.6972**
Recall	0.7297	0.6698	**0.7417**
F-measure	0.6040	0.6684	**0.7188**

Table 5 Comparation with previous Algorithm

For smaller datasets with a size less than 30k, a rule-based approach delivers top accuracy scores and is technically superior This approach is recommended for applications that cannot use larger training datasets.

For larger datasets with a size greater than 30k, a deep learning-based approach delivers highly competitive results and is technically superior This approach is recommended for applications that can use larger training datasets Training datasets with a size of around 300k-500k provide pretty good performance for the best methodology with training datasets larger than 30k [13].

Different approaches have been used for duplicate question detection (DQD) and their performance depends on the size of the dataset A Jaccard coefficient approach achieved an f-score of 6029 on a DQD dataset created from the Baidu Zhidao forum.

A convolutional neural network (CNN) approach achieved over 92% accuracy using 30k data from StackExchange and AskUbuntu forums Another deep neural network approach achieved the best accuracy of 073035 in SemEval-2016 Task 1 Only one unpublished paper is available on the Quora dataset, which reports an accuracy of 8817% using a multi-perspective matching model [7] Other draft results based on natural language inference have been posted on blogs.

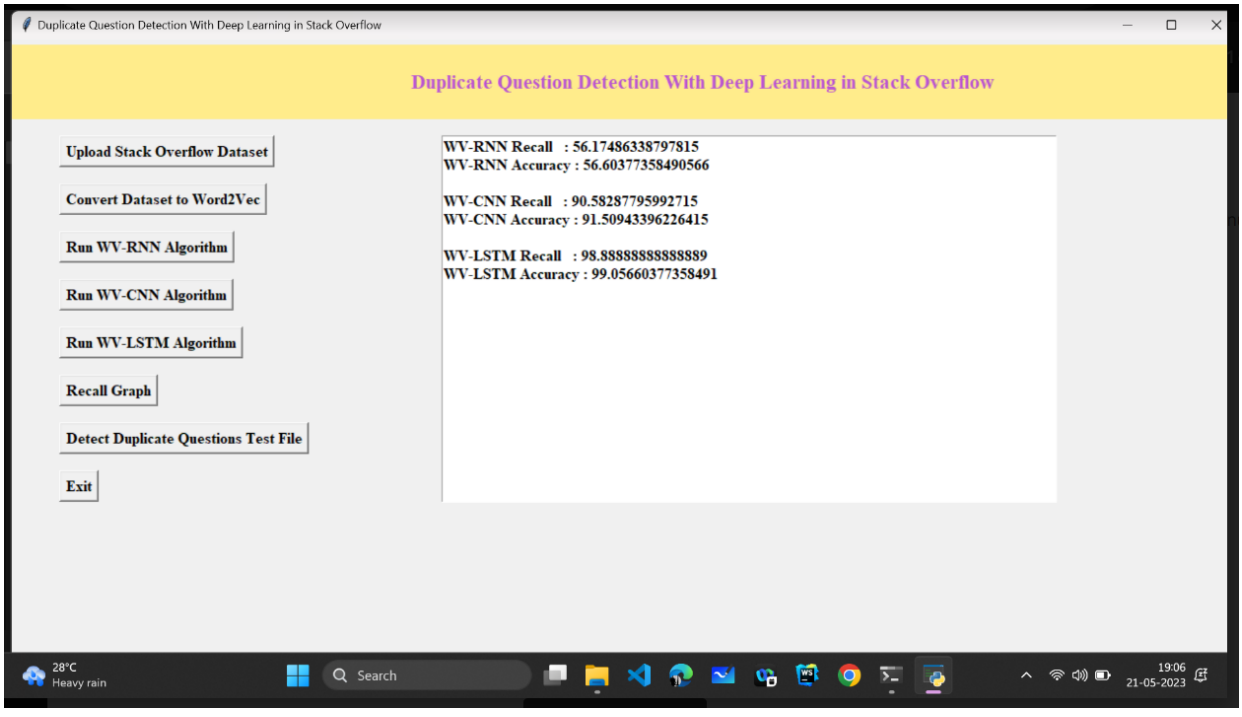

Figure 15 Results

CONCLUSION

Duplicate Predictor is a system that assesses the similarity between two questions by examining their observable factors (titles, descriptions, and tags) and their underlying factors (topic distributions derived from the question descriptions). It comprises four components: title similarity, description similarity, topic similarity, and tag similarity. These components are combined using learned weights from training data. The performance of Duplicate Predictor was evaluated using a dataset of over two million questions from Stack Overflow, with the metric being recall-rate@k. The results demonstrated that Duplicate Predictor achieved recall-rate@5, recall-rate@10, and recall-rate@20 scores of 42.3%, 53.3%, and 63.8% respectively.

In comparison to its constituent components, the standard search engine of Stack Overflow, and Runeson et al.'s approach for detecting duplicate bug reports, Duplicate Predictor showed significant improvement, enhancing the baselines' performance by 10.2% to 717.9%.

This research investigates the use of three different deep learning techniques, CNN, RNN, and LSTM, for the purpose of detecting duplicate questions on Stack Overflow. Word2Vec is employed to generate vector representations of words. Based on Word2Vec, CNN, RNN, and LSTM, three deep learning models named WV-CNN, WV-RNN, and WV-LSTM are developed to identify duplicate questions on Stack Overflow. These models effectively capture both the semantic information at the word-level and document-level for each pair of questions.

Through experiments conducted on six different question groups, it is evident that our WV-CNN and WV-LSTM approaches outperform the four baseline methods significantly. Although it is still not clear why our results differ from the original ones because we have different implementions and different datasets, we make a replication package public available to enable other researchers to repeat, improve, or refute our results.

Moreover, we observed a performance decrease with the reproduction of those two approaches over time, as the number of question increases. Further reproduction studies, with different settings, could extend the collected results. A more thorough investigation on how to stabilize duplicate question detection as the number of questions increase could also be conducted.

REFFERENCES

[1] C Treude, O Barzilay, and M-A Storey, “How do programmers ask and answer questions on the web? (nier track),” in 33rd International Conference on Software Engineering (ICSE), 2011, pp 804–807

[2] F Chen and S Kim, “Crowd debugging,” in 10th Joint Meeting on Foundations of Software Engineering (ESEC/FSE), 2015, pp 320–332

[3] L Ponzanelli, G Bavota, M Di Penta, R Oliveto, and M Lanza, “Prompter,” EMPIR SOFTW ENG, vol 21, no 5, pp 2190–2231, Oct 2016

[4] S Azad, P C Rigby, and L Guerrouj, “Generating API call rules from version history and Stack Overflow posts,” ACM Trans Softw Eng Methodol, vol 25, no 4, pp 29:1–29:22, Jan 2017

[5] B Vasilescu, A Serebrenik, P Devanbu, and V Filkov, “How social Q&A sites are changing knowledge sharing in open source software communities,” in 17th Conference on Computer Supported Cooperative Work & Social Computing (CSCW), 2014, pp 342–354

[6] M A Storey, A Zagalsky, F F Filho, L Singer, and D M German, “How social and communication channels shape and challenge a participatory culture in software development,” IEEE Trans Softw Eng, vol 43, no 2, pp 185–204, Feb 2017

[7] J Surowiecki, The wisdom of crowds Anchor, 2005

[8] D Kavaler, D Posnett, C Gibler, H Chen, P Devanbu, and V Filkov, “Using and asking: APIs used in the Android market and asked about in StackOverflow,” in 5th International Conference on Social Informatics (SocInfo) Cham: Springer, 2013, pp 405–418

[9] E C Campos, L B L de Souza, and M d A Maia, “Searching crowd knowledge to recommend solutions for API usage tasks,” J SOFTW EVOL PROC, vol 28, no 10, pp 863–892, 2016

[10] C Treude and M P Robillard, “Augmenting API documentation with insights from Stack Overflow,” in 38th International Conference on Software Engineering (ICSE), 2016, pp 392–403

[11] L An, O Mlouki, F Khomh, and G Antoniol, “Stack Overflow: A code laundering platform?” in 24th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2017, pp 283–293

[12] R Abdalkareem, E Shihab, and J Rilling, “What do developers use the crowd for? a study using Stack Overflow,” IEEE Software, vol 34, no 2, pp 53–60, 2017

[13] I Srba and M Bielikova, “Why is Stack Overflow failing? preserving sustainability in community question answering,” IEEE Software, vol 33, no 4, pp 80–89, July 2016

[14] Y Zhang, D Lo, X Xia, and J-L Sun, “Multi-factor duplicate question detection in Stack Overflow,” Journal of Computer Science and Technology, vol 30, no 5, pp 981–997, Sep 2015

[15] M Ahasanuzzaman, M Asaduzzaman, C K Roy, and K A Schneider, “Mining duplicate questions in Stack Overflow,” in 13th International Conference on Mining Software Repositories (MSR), 2016, pp 402–412

[16] 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 31st International Conference on Automated Software Engineering (ASE), 2016, pp 51–62

[17] Y Mizobuchi and K Takayama, “Two improvements to detect duplicates in Stack Overflow,” in 24th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2017, pp 563–564

[18] W E Zhang, Q Z Sheng, J H Lau, and E Abebe, “Detecting duplicate posts in programming qa communities via latent semantics and association rules,” in 26th International Conference on World Wide Web (WWW), Geneva, Switzerland, 2017, pp 1221–1229

[19] A Pal, R Farzan, J A Konstan, and R E Kraut, “Early detection of potential experts in question answering communities,” in 19th International Conference on User Modeling, Adaption and Personalization (UMAP) Berlin, Heidelberg: Springer, 2011, pp 231–242

[20] S Deterding, M Sicart, L Nacke, K O’Hara, and D Dixon, “Gamification using game-design elements in non-gaming contexts,” in EA on Human Factors in Computing Systems (CHI EA), 2011, pp 2425–2428

[21] A Bacchelli, “Mining challenge 2013: Stack overflow,” in 10th International Conference on Mining Software Repositories (MSR), 2013

[22] A T T Ying, “Mining challenge 2015: Comparing and combining different information sources on the Stack Overflow data set,” in 12th International Conference on Mining Software Repositories (MSR), 2015

[23] G Robles, “Replicating MSR: A study of the potential replicability of papers published in the mining software repositories proceedings,” in 7th Work Conf on Mining Software Repositories (MSR), 2010, pp 171–180

[24] S Amann, S Beyer, K Kevic, and H Gall, “Software mining studies: Goals, approaches, artifacts, and replicability,” in Software Engineering: Intl Summer Schools, LASER 2013-2014, Italy, Rev Tutorial Lectures, B Meyer and M Nordio, Eds Cham: Springer, 2015, pp 121–158

[25] I Srba and M Bielikova, “A comprehensive survey and classification of approaches for community question answering,” ACM Trans Web, vol 10, no 3, pp 18:1–18:63, Aug 2016

[26] J Jeon, W B Croft, and J H Lee, “Finding similar questions in large question and answer archives,” in 14th International Conference on Information and Knowledge Management (CIKM), 2005, pp 84–90

[27] D Ravichandran and E Hovy, “Learning surface text patterns for a question answering system,” in 40th Annual Meeting on Association for Computational Linguistics (ACL) Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp 41–47

[28] M I M Azevedo, K V R Paixão, and D V C Pereira, “Processing heterogeneous collections in xml information retrieval,” in Advances in XML Information Retrieval and Evaluation: 4th INEX, Dagstuhl Castle, Germany Springer, 2006, pp 388–397

[29] M Theobald, J Siddharth, and A Paepcke, “Spotsigs: Robust and efficient near duplicate detection in large web collections,” in 31st Annual International Conference on Research and Development in Information Retrieval (SIGIR), 2008, pp 563–570

[30] Y Wu, Q Zhang, and X Huang, “Efficient near-duplicate detection for Q&A forum,” in 5th International Joint Conference on Natural Language Processing (IJCNLP), Chiang Mai, Thailand, 2011, pp 1001–1009

[31] T Hao and E Agichtein, “Finding similar questions in collaborative question answering archives: toward bootstrapping-based equivalent pattern learning,” Inf Retrieval, vol 15, no 3, pp 332–353, Jun 2012

[32] L Nie, X Wei, D Zhang, X Wang, Z Gao, and Y Yang, “Data-driven answer selection in community qa systems,” IEEE Trans Knowl Data Eng, vol 29, no 6, pp 1186–1198, June 2017

[33] F M Delfim, K V R Paixão, D Cassou, and M de Almeida Maia, “Redocumenting APIs with crowd knowledge: a coverage analysis based on question types,” J Braz Comput Soc, vol 22, no 1, p 9, 2016

[34] S M Nasehi, J Sillito, F Maurer, and C Burns, “What makes a good code example?: A study of programming Q&A in StackOverflow,” in 28th Intl Conf on Software Maintenance (ICSM), 2012, pp 25–34

[35] L B L de Souza, E C Campos, and M d A Maia, “Ranking crowd knowledge to assist software development,” in 22nd International Conference on Program Comprehension (ICPC), 2014, pp 72–82

[36] W Fu and T Menzies, “Easy over hard: A case study on deep learning,” in 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE), 2017, pp 49–60

[37] W E Zhang, Q Z Sheng, Y Shu, and V K Nguyen, “Feature analysis for duplicate detection in programming QA communities,” in 13th International Conference on Advanced Data Mining and Applications (ADMA) Cham: Springer, 2017, pp 623–638

[38] M Wurst, “The word vector tool user guide operator reference developer tutorial,” 2007 [Online] Available: http://sourceforgenet/projects/wvtool/

[39] R B Yates and B R Neto, “Modern information retrieval: the concepts and technology behind search,” Addison-Wesley Professional, 2011

[40] M F Porter, “An algorithm for suffix stripping,” Program, vol 14, no 3, pp 130–137, 1980

[41] A K McCallum, “Mallet: A machine learning for language toolkit,” 2002

[42] G A Miller, “Wordnet: A lexical database for english,” Commun ACM, vol 38, no 11, pp 39–41, Nov 1995

[43] C D Manning, M Surdeanu, J Bauer, J Finkel, S J Bethard, and D McClosky, “The Stanford CoreNLP natural language processing toolkit,” in Association for Computational Linguistics (ACL) System Demonstrations, 2014, pp 55–60

[44] M Hall, E Frank, G Holmes, B Pfahringer, P Reutemann, and I H Witten, “The WEKA data mining software: An update,” SIGKDD Explor Newsl, vol 11, no 1, pp 10–18, Nov 2009

[45] S E Robertson and S Walker, “Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval,” in 17th International Conference on Research and Development in Information Retrieval (SIGIR) Springer, 1994, pp 232–241

[46] D A Prabowo, “ Detection of the Similarities of Questions in Online Forums Using Convolutional Neural Networks, Deteksi Kesamaan Pertanyaan Pada Forum Online Menggunakan Convolutional Neural Network (in Indonesian),” Bachelor Thesis, Universitas Gadjah Mada, 2019

[47] Shankar Iyer, N Dandekar, and K Csernai, “First Quora Dataset Release: Question Pairs,” 2017

[48] R Collobert, J Weston, L Bottou, M Karlen, K Kavukcuoglu, and P Kuksa, “Natural Language Processing (Almost) from Scratch,” Journal of Machine Learning Research, vol 12, pp 2493–2537, 2011

[49] D P Kingma and J Ba, “Adam: A method for stochastic optimization,” 2014

[50] Y Bengio, “Practical recommendations for gradient-based training of deep architectures,” Neural Networks: Tricks of the Trade, p 437478, 2012

[51] D Correa and A Sureka, “Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow,” in Proc 23rd Int Conf World Wide Web (WWW), Jan 2014, pp 631–642

[52] M Ahasanuzzaman, M Asaduzzaman, C K Roy, and K A Schneider, “Mining duplicate questions in stack overflow,” in Proc 13th Min Softw Repositories (MSR), May 2016, pp402–412

[53] Y Zhang, D Lo, X Xia, and J Sun, “Multi-factor duplicate question detection in stack overflow,” J Comput Sci Technol, 30, no 5, pp 981– 997, Sep 2015

[54] R F Silva, K PaixÃco, and M de Almeida Maia, “Duplicate question ˇ detection in stack overflow: A reproducibility study,” in Proc 25th Softw An Ev Reeng (SANER), Mar 2018, pp 572–581

[55] C N Kamath, S S Bukhari, and A Dengel, “Comparative Study between Traditional Machine Learning and Deep Learning Approaches for Text Classification,” in Proc of the ACM Symposium on Document Eng, Halifax, NS, Canada, 2018, pp 1-11

[56] J Kapociˇ ut¯ e-Dzikien ˙ e, R Damaševi ˙ cius, and M Wo´zniak, “Sentiment ˇ Analysis of Lithuanian Texts Using Traditional and Deep Learning Approaches,” Computers, vol 8, no 1, pp 1–16, Jan 2019

[57] M White, M Tufano, C Vendome, and D Poshyvanyk, “Deep learning code fragments for code clone detection,” in Proc 31th IEEE/ACM Int Conf Automat Softw Eng (ASE), Sep 2016, pp 87–98

[58] 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

[59] 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 31th IEEE/ACM Int Conf Automat Softw Eng (ASE), Sep 2016, pp 51–62

[60] H Liang, Y Yu, L Jiang, and Z Xie, “Seml: a Semantic LSTM Model for Software Defect Prediction,” in IEEE Access, vol 7, Jun 2019, pp 83812–83824

[61] V J Hellendoorn and P Devanbu, “Are deep neural networks the best choice for modeling source code?,” in Proc 11th Joint Meeting on Foundations of Softw Eng (FSE), Sep 2017, pp 763–773

[62] L-T Wang, L Zhang, and J Jiang, “Detecting Duplicate Questions in Stack Overflow via Deep Learning Approaches”, presented at the 26th Asia-Pacific Softw Eng Conf (APSEC), Putrajaya, Malaysia, Dec 2-5, 2019

[63] T Mikolov, K Chen, G Gorrado, and J Dean, “Efficient estimation of word representations in vector space,” 2013, arXiv preprint arXiv:13013781

[64] T Mikolov, I Sutskever, K Chen, G S Corrado, and J Dean, “Distributed representations of words and phrases and their compositionality,” in Proc 26th Int Conf Neural Inf Process Syst, Oct 2013, pp 3111–3119

[65] A Krizhevsky, I Sutskever, and G E Hinton, “Imagenet classification with deep convolutional neural networks,” in Proc 25th Int Conf Neural Inf Process Syst, Jun 2012, pp 1097–1105

[66] A Graves, A R Mohamed, and G Hinton, “Speech recognition with deep recurrent neural networks,” in Proc IEEE Int Conf Ac Speech Sign Process, May 2013, pp 6645–6649

[67] 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 Ac Speech Sign Process, Apr 2015, pp 4580–4584

[68] T Joachims, “Training linear SVMs in linear time,” in Proc 12th ACM SIGKDD Int Conf Knowl Disc Data Min, Philadelphia, PA, USA, 2006, pp 217–226

[69] A Genkin, D Lewis, and D Madigan, “Large-Scale Bayesian Logistic Regression for Text Categorization,” Technometrics, vol 49, no 3, pp 291–304, Aug 2007

[70] L Breiman, “Random forests,” Machine Learning, vol 45, no 1, pp 5–32, Oct 2001

[71] T Chen and T He, “Xgboost: extreme gradient boosting,” R package version 04-2, 2015, pp 1–4

[72] F Brooks and H Kugler, “No silver bullet,” IEEE Comput, vol 20, no 4, pp 10–19, 1987

[73] Y Bengio, R Ducharme, P Vincent, and C Jauvin, “A neural probabilistic language model,” J Machine Learning R, vol 3, pp 1137–1155, Feb 2003

PAPER NAME

Akarsh_thesis.docx

WORD COUNT

16679 Words

CHARACTER COUNT

89872 Characters

PAGE COUNT

51 Pages

FILE SIZE

2.1MB

SUBMISSION DATE

May 21, 2023 7:00 AM PDT

REPORT DATE

May 21, 2023 7:01 AM PDT

18% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 15% Internet database
- 16% Publications database
- Crossref database
- Crossref Posted Content database
- 9% Submitted Works database

Excluded from Similarity Report

- Bibliographic material
- Quoted material