An Automated Approach for Best Answer Prediction for Developer Forums

By

Hafiz Umar Iftikhar

Fall 2017-MS-CS&E 00000205233

Supervisor

Dr. Mian Ilyas Ahmad

Department of Computational Sciences and Engineering

Research Center For Modeling and Simulations (RCMS)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

December 2019

An Automated Approach for Best Answer Prediction for Developer Forums

By

Hafiz Umar Iftikhar

00000205233

Supervisor

Dr. Mian Ilyas Ahmad

A thesis submitted in conformity with the requirements for

the degree of *Master of Science* in

Computational Sciences And Engineering

Department of Computational Sciences And Engineering

Research Center For Modeling and Simulations (RCMS)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

December 2019

Declaration

I, *Hafiz Umar Iftikhar* declare that this thesis titled "An Automated Approach for Best Answer Prediction for Developer Forums" and the work presented in it are my own and has been generated by me as a result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a Master of Science degree at NUST
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at NUST or any other institution, this has been clearly stated
3. Where I have consulted the published work of others, this is always clearly attributed
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work
5. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself

Hafiz Umar Iftikhar,

00000205233

Copyright Notice

This thesis is dedicated to *my beloved parents*

Acknowledgments

In the name of Allah, the Most Gracious and the Most Merciful

Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis. Special appreciation goes to my supervisor, Dr Mian Ilyas Ahmad, for his supervision and constant support. His invaluable help of constructive comments and suggestions throughout the experimental and thesis works have contributed to the success of this research. Not forgotten, my appreciation to my general evaluation committee members, Dr. Muhammad Tariq Saeed and Dr. Shehzad Rasool for his support and knowledge regarding this topic.

Last but not least, my deepest gratitude goes to my beloved parents who supported me financially and spiritually and also to my brother, sisters for their endless love, prayers and encouragement.

Hafiz Umar Iftikhar

Contents

List of Abbreviations and Symbols

Abbreviations

CQA	Community Questioning Answering
Q&A	Questioning and Answering
AI	Artificial Intelligence
ML	Machine Learning
NN	Neural Network
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DL	Deep Learning
NLP	Natural Language Processing
CNN	Convolution Neural Network
LSTM	Long Short-Term Memory

MNB	Multinomial Naive Bayes
RF	Random Forest
DT	Decision Tree
LR	Logistic Regression
SVM	Support Vector Machine

List of Tables

List of Figures

Abstract

Developer forums are essential for the development of a software as they are used to solve the problems/issues raised at such forums through assistance of experts. Often their are multiple answers (solutions) to a single issue and some of these answers are not helpful/satisfactory. The users usually browse all the answer within a question thread to get the required answer. This is a tedious and time-consuming task. In this thesis, we proposed an automatic classification approach to predict high quality answers of the questions on a developer forum. First, we extract metadata features (such as length of words, number of characters/sentences and average characters per word) and then, we utilize natural language processing techniques (such as data cleaning, tokenization, stop words removal and spell corrections). Also we employ a keyword ranking algorithm, which uses ranking scores on the text of all answers under each question. Next, we used word embedding to transform the preprocessed textual description of answers into feature vector. Finally, we input the vectors of metadata, keywords and textual features to the proposed deep learning based integrated model for training and prediction of high quality answers. The proposed integrated model includes a combination of the convolutional neural network (CNN) and long short term memory (LSTM) algorithm. The results of the 10 fold cross-validation suggest that the proposed approach shows

better results as compared to a recent best answer prediction approach.

Keywords: *Developer Forums, Best Answer Prediction, Stack Overflow, Technical Q&A sites, Deep Learning*

CHAPTER 1

Introduction

Software applications are discussed on different online developer forums to share the problems observed by their users and get feedback from other users/experts of the software. For each of the questions/problems, there are often multiple answers and predicting the best answer manually is time consuming for users. In this chapter, we discuss the problem of automatically predicting best answer to a question using machine/deep learning techniques. In the following, we briefly introduce the concept of developer forums and associated criteria of best answer prediction in developer forums.

1.1 Developer Forums

There are many software applications that are utilized for personal and official tasks such as latex, Microsoft visual studio, Spider and Jupiter notebook. These applications are used to build projects and develop applications in their domain of interest. The users of these applications may face some issues in installation, coding, simulations and applications of the software. Therefore, the need of an

online platform arises where the users can get external guidance on software related issues. There are several platforms that has been developed for such guidance and these forums are called developer forums. One of the most famous platform is stack overflow where over 50 million people visit each month including programmers, professionals and new application users.

Software users often post their questions on developer forums and other experienced members of the forums answer these questions based on their knowledge. Often, there exist multiple answers under one question and reading all answers to find the best one is tedious and time consuming. So, in this research we are exploring an automatic system which will predict best answer from the list of answers on developer forums.

Prior to developer forums, software users used to find information and answers to their questions on different search engines. They used to submit some keywords and queries on search engine and find appropriate content after searching and reading several web pages for long time. Sometime it was not possible for such users to find exactly what they need after sorting a number of web pages. Based on the frequency of problems and their technicallity, some groups of mailing lists were developed for technical support and discussion, where developers were allowed to get guidance on their queries in depth. However, a separate web interface was still required for searching archive content. After this, web-based discussion forums became popular because of their integrated search, efficiency, ease of use and avoidance of duplicate issues. Also the capability of highlighting an accepted answer by users in a thread greatly improved the success and popularity of Q&A sites.

Questioning & answering sites are divided into two categories: Community Q&A

sites such as Answerbag, Blurt it, Anybody out there, WikiAnswers, Fun Advice, Askville, Ask me help desk, Answer Bank, AskDeb, Able2know, Friendfeed, Twitter and yahoo answers, where anything can be asked including law, music, seasonal, jobs and family. Second type is developer forums or technical Q&A sites where software engineers find solutions to technical problems such as Stack Overflow, google developer, github and mozilla developer network. The following figure 1.1 shows two examples, the first one is related to community Q&A: yahoo answers [1] and the second one related to developer forums: stack overflow [2].

Technical question-answer network community involves a ranking system showing the experties of the users. The ranking is based on a voting system, where an answer is marked by a user as a quality solution from the list of answers and the remaining answers that are unmarked are non-potential solutions. Sometime the answer in a question thread remains unselected due to absence of accepted solution or may be because the user forget to mark the answer as a valid solution [3]. Stack overflow allows its users to assess the quality of answer by voting them either upvotes or downvotes to the answers of question threads. The answers are sorted by vote-ordering, the answer with high votes remains on top and mark as potential solution. However, the issue in this system is that the answer receives fewer or no upvotes if entered later. This means that even with high quality answer, it stays at bottom and the answers posted first remains on top and take the benefit of Matthew's effect (Mamykina et al. [4]; Roy et al. [5]). Also, it was shown by Mamykina et al. [4] that by gaining reputation and visibility within the stack overflow community, the users inspire to answer more questions. Another important issue in developer forums is the starvation of quality/complete answers to question [6]. For example, in Stack overflow 50% of questions are unresolved (around 7 mil-

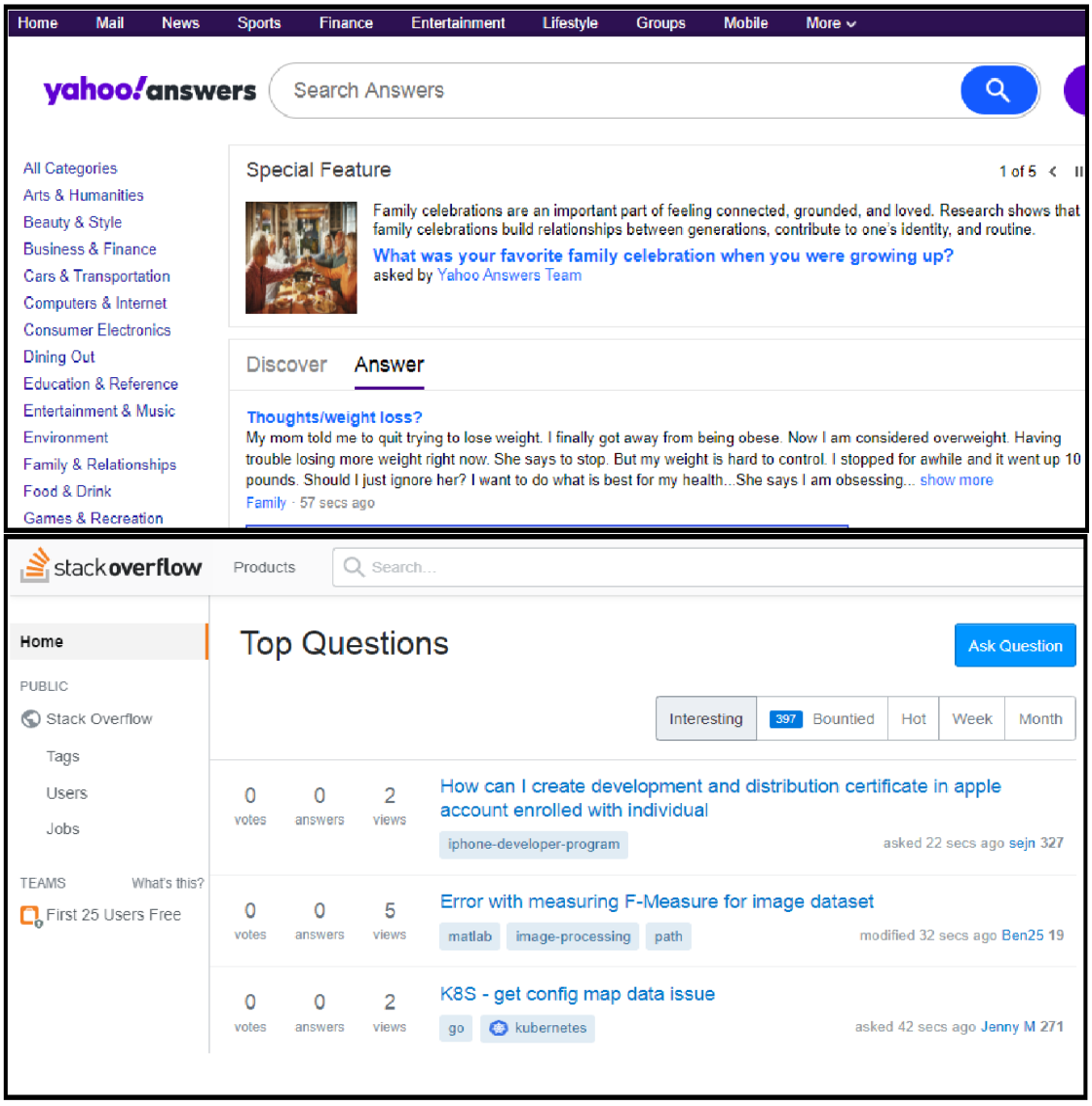

Figure 1.1: Yahoo Answers & Stack Overflow

lion). So, the software engineers are vigilant for a solution where they can access potential answer in each question thread [7]. In this thesis, we are dealing with this problem as binary classification problem, where deep learning based best answer prediction model is built to predict the best answer within a question thread.

1.2 Text Classification Using Natural language Processing Techniques

Text classification is the process of classifying text into predefined categories. Some common applications of text classification are automated spam detection from e-mails and detection of positive and negative reviews from the feedback of people about a product, automated chatbot, suggestions on online shopping sites for buying products based on the users interest and many other real life applications. Since textual data is increasing continuously, the applications of text classification are rapidly growing.

In this research we are doing text classification by using natural language processing to automate the process of identifying best answer in developer forums using machine and deep learning techniques.

Text classification is performed using natural language processing (NLP) which is sub part of artificial intelligence. NLP techniques are used because text data is not directly recognized by computer. The purpose of NLP is to make the computer understand human language as human understand that language. It tags each word with part-of-speech (POS): nouns, pronouns, verbs and adjectives to make it understandable for computer. It is a challenging task to classify the raw text files into one or more classes and categories by using NLP. Moreover, the prediction and statistical analysis is performed on classified text data by using machine/deep learning algorithms.

1.3 Motivation

Software engineering community is gaining success in every field due to advancement in technology and is rapidly growing in software industry. There are many software applications available on internet that people utilize for personal and official purposes such as latex, Microsoft visual studio, Spider, Jupiter notebook. These applications are used to build projects and develop applications in their domain of interest. The users of these applications may face some issues in installation, coding, simulations and applications of the software. Therefore, the need of an online platform arises where the users can get external guidance on software related issues. There are several platforms has been developed known as developer forums. One of the most famous platform is stack overflow where 50 million of people visit each month including programmers, professionals and new application users. As discussed before, users often post their questions on developer forums and other experienced members of the forums answer those questions based on their knowledge. There could be multiple answers of one question and it is very difficult for the user to find the most appropriate answer to resolve their problem. In this research, we are going to automate this system where the best answer will be automatically identified among the list of answers to save users time and efforts with the help of deep learning techniques because deep learning algorithms are robust, generalize, scalable and work efficiently on large datasets.

1.4 Problem Statement

The problem is that there are multiple answers under one question. The system of stack overflow works on voting system, all users who read answers of same question vote the answers whether it could be up-vote or down-vote and the person who post question has right to select best answer among the list of answers. Sometimes, the user who asked the question has no ability and enough knowledge to select the best answer and the user select wrong answer as best solution which could misguide the other users who search for same question and facing similar problem. Thus, reading all answers is tedious and time consuming task. People need the system where they automatically get only one best answer among the list of answers.

Moreover, the AI is involved in every domain and all manual systems are being automated with the help of artificial intelligence and machine learning. We are dealing this problem with text classification by using deep learning approach. Text classification is used to classify textual data into predefined categories. In this research we classify accepted and rejected answers with the help of text classification by using machine/deep learning approaches as shown in Fig 2.1

1.5 Research Objectives

Our goal in this research to automate the system to predict the most appropriate and best answer in developer forums by using machine learning and deep learning

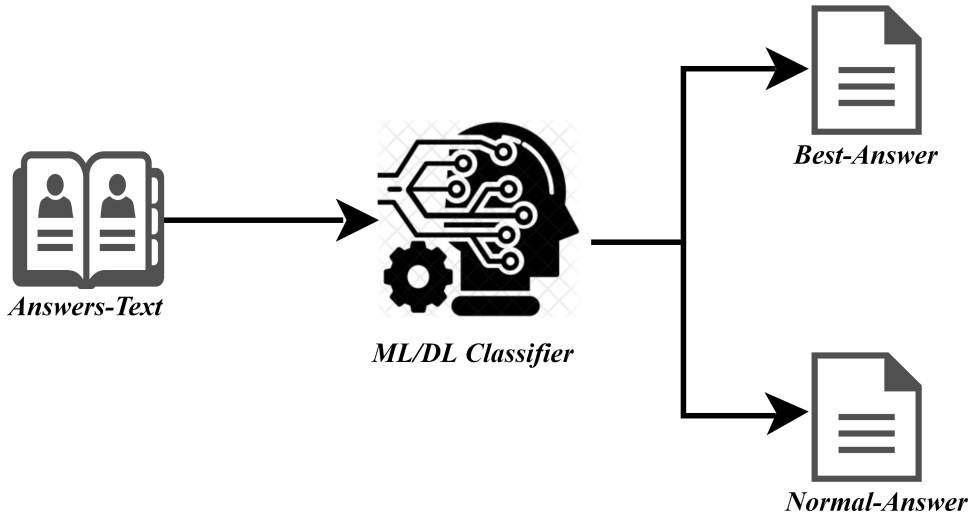

Figure 1.2: Answers Classification

techniques. The proposed deep learning based integrated model automate the system and improve response time for user to get satisfactory answer and reject the low quality answer in developer forums.

We will implement the following steps in this research to automate the system of best answer prediction:

- Data collection and the conversion of data into useable format.
- Preprocessing the data by using Natural Language Processing techniques.
- Extraction of features (metadata, keywords and textual description).
- Building the proposed model.
- Re-sampling of data.
- Collection of results before and after re-sampling.
- Comparison of proposed approach with traditional machine learning algorithms and state-of-the-art.

1.6 Thesis Layout

This thesis is formulated as fellows: In chapter 2, we will discuss all previous research work that has been done on the Q&A sites to predict questions and answers quality. In chapter 3, we will discuss the methodology of this research that is used to solve the problem related to best answer prediction in developer forums. In chapter 4, we will discuss the results of proposed work and metrics used to evaluate results. Moreover, we raise five research questions and investigate them to evaluate results and resolve all queries related to the proposed approach.

CHAPTER 2

Literature Review

In this chapter, we will discuss pervious approaches used to predict best answer in developer forums by using machine learning and deep learning techniques. First, we will present the problem of text classification in general framework and then specific to the problem of best answer prediction. We identify the research gaps in classification through both machine learning and deep learning techniques.

2.1 Text Classification

The concept of text classification is to arrange input data representing text messages into predefined groups or categories according to some criteria. Text data is continuously growing with increasing trend of using emails and different social media platforms. Often, it is important to classify the text into categories such as positive and negative reviews, accepted or rejected answers, spam or regular emails and different type of emotions. Text classification can be performed in two different ways: manually and automatically and this process of text classification is shown in Figure 2.1.

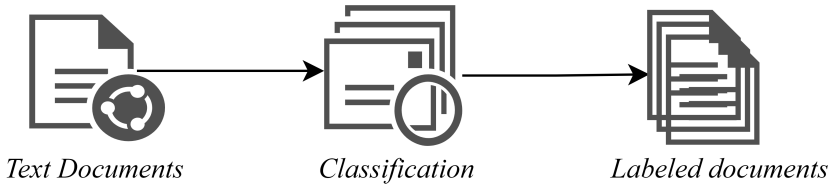

Figure 2.1: Text Classification

Manual classification can provide efficient results but it is very difficult, time-consuming and cost-intensive to transform unstructured data into structured form. There are many ways to automate this process, one of the most efficient and cost-effective process is to automate the classification by using machine learning and deep learning techniques.

2.2 Classification through Machine Learning

Machine learning is the process by which algorithm/model is created to infer/predict the solution of a particular task, using a predefined data set. The data can either be in the form of text, numeric values, images and videos. The data is divided into two parts, learning and test data. The learning data is used to train the model and the test data is used to evaluate the trained model. Some real life examples of the machine learning are image recognition, speech recognition, medical diagnosis, statistical arbitrage, learning associations.

Machine learning based algorithms are often classified into two categories: supervised and unsupervised learning. Supervised machine learning algorithms are used when the data has labels. The unsupervised learning techniques does not require any labels or supervision. Some of the supervised and unsupervised learn-

ing techniques are shown in Table 2.1 An important class of machine learning is

Table 2.1: Supervised and unsupervised learning algorithms

Supervised	**Unsupervised**
Naive-Bayes	Singular Value Decomposition
Linear Regression	Principal Component Analysis
Logistic Regression	K-means Clustering
Decision Tree	
Random Forest	
k-nearest neighbors	
Support Vector Machine	

called deep learning, where layers of algorithms/functions are created to predict the desired solution. Since, our focus is to predict best answer using deep learning techniques, in section 2.3 we will briefly discuss the concept text classification by using deep learning approaches.

Many researches have worked at the orientation of machine learning algorithms related to Q&A sites and their focus were on: (i) assess the quality of questions and their answers;(ii) develop the understanding that how software application developers communicate with each other on Q&A sites; (iii) provide the evidence how to write quality question and answer on stack overflow; (iv) impact of sentiments to increase or decrease the chance of answer to be accepted. Most of them used text classification to automate the process of best answer prediction. Commonly used machine learning algorithms such as Naive Bayes, Logistic Regression, Decision

Tree, Random Forest and Support vector machine to classify best answer from the multiple answers.

2.2.1 Question's Quality Assessment

To assess the questions quality Ponzanelli et al. [8] proposed as automated approach to detect the low quality post from Stack Overflow to refine the review queue. They used different datasets consist of almost 5 million of questions and separate it in two classes on the basis of score, high quality content class contains questions that have score greater than zero and other dumped into low quality content class. Their proposed model is based on two features, textual features consist of content of the post and community-based features concern with the popularity of user and Xia et al. [9] presented a model named DelPredictor is to predict the deleted questions is based on two features, meta features (i.e. community, profile and syntactic features) and textual features (i.e. question title, body and tags). They implemented two classifiers, Multinomial Naive Bayes (MNB) and Random Forest (RF) to train the model. They evaluated their model on 417685 deleted questions of Stack Overflow from 2008 to 2013 and the prediction results showed MNB achieved good accuracy than their baseline approaches, both of the researches Ponzanelli and Xia worked on the prediction of questions quality.

There are thousands of questions on stack overflow which are unanswered by users, to evaluate the reason of unanswered questions Treude et al. [10] performed analysis on 15 days stack overflow asked questions data to investigate which questions are answered or unanswered and also categorize the types of questions that were asked. They used most frequent 200 tag keywords in data and found that

these keywords cover 60,193 tags instances and then identified those tags which cover most of the instances and the Bajaj et al. [11] presented a study to analyze the questions related to mining and challenges, misconceptions in developers about these questions. They used unsupervised learning approach to categorize the stack overflow questions and assigned ranking to all the questions based on their importance. Data is separated on the basis of three tags and implemented Latent Dirichlet Allocation (LDA) topic modeling to find the most dominant topics.

To retrieve the english questions of given chines question Xu et al. [12] proposed automated question retrieval approach (CLRQR) to retrieve the all relevant English questions of given question in Chinese language. They built a repository of 684,599 java questions in English from stack overflow dataset and as well as they built a vocabulary of 111,174 English words to optimize the working of Chinese-English translation tool. Their approach extract features from given Chinese question such as title, description and extract Chinese information into English by using Chinese-English translation tool. Moreover, English questions are retrieved from repository on the basis of high scored English words. They compared CLRQR system with four baseline approaches and the experimental results showed their approach is outperformed on the state of the art.

2.2.2 Sentiments Prediction

To check the effect of sentiments on writing quetion, Calefato et al. [13] gave a guideline to write good questions on stack overflow. In their research they found the factors such as Affect (either positive or negative), Presentation quality (code sniped, title and body length, upper case character ratio, presence of multiple tags

and presence of URLs), Time (posting time) and User reputation (asker reputation) that potentially influence on the success of question, for this purpose they used 82 k questions from the official dump of stack overflow. They implemented Logistic Regression (LR) in proposed approach and calculate the AUC(Area Under Curve) value 65%.

2.2.3 Answer's Quality Assessment

The questioning and answering system of stack overflow worked on voting and badges system, Roy et al. [14] claimed that the voting system is not only a good way to check the quality of answer, sometime the answer posted later did not get high votes and placed at bottom. They used stack exchange open source dataset for classification and labeled the data into three classes on base of voting, answers that received votes below than 1 labeled as low-quality answers, the answers that received votes 1 or 2 labeled as moderate-quality answer and the answers obtained votes greater than 2 labeled as high-quality answers. They extracted 26 feature including wrong words, code sniped, user reputation, Flesch reading ease, entropy, Dale challe reading score, question-answer similarity, answer-answer similarity etc. They trained three classifiers Naive Bayes (NB), Random Forest (RF) and Gradient Boosting for classification and their results showed Gradient Boosting performed best and Calefato et al. [3] investigated that how the users can increase the chance of their answer is getting accepted on stack overflow, for this purpose they found four factors that highly influence on the success of the answers (i) presentation which includes URLs, code sniped, Length, uppercase ratio (ii) emotional affect either positive or negative (iii) time of answer posted (iv) reputation score of asker's. They built dataset from official stack overlow (SO) dump and the

answers of the questions posted in 30 days having 348,618 total answers are used. They implement Logistic Regression (LR) and achieved 64% of AUC(Area Under Curve).

Previous studies shows user reputation is very important to increase chance of answer is being accepted on stack overflow [15][16][17], Bosu et al. [18] analyzed the ways a user can quickly earn good reputation on stack overflow such as answer questions with lower expertise density tags, answer questions promptly, be the first answerer, be active member in peak hours and contribute to diverse areas and Zheng and Li [19] their study based on three types of features textual, code sniped and answerer background. Textual and code sniped features are used to identify the relatedness between question and its answers and answerer background knowledge indicate the worth of answer. They performed AdaBoost based learners for the prediction of best answer on Stack Overflow dataset. However, their classifier model achieved 63, 59 and 61% of Precision, Recall and F-measures, respectively.

There exist some approaches that says user reputation does not effect the quality of answers, Hart and Sarma [20] discussed the role of social cues for novice users to select answers on stack overflow and reputation earned by users. They performed a survey through amazon mechanical turk on the sample set of java related questions and answers. The purpose of this survey was to check how social factors influence on technical forums. The results showed that the presentation style effect the quality of answer instead of social reputation did not affect while novice user judge the quality of answer and Tian et al. [21] model also does not rely on user-related features as compared to others. First, they measure three key

factors (1) quality of answer content (2) contribution of the answer to solve new question (3) how it compares with other answers, by designing the features, especially contextual information. Second, they predict the best answer by evaluating and designing a learning approach from extracted features. They applied two fold cross-validation with random forest algorithm on stack overflow dataset and their classifier achieved 72% of accuracy. Moreover, Gkotsis et al. [22] proposed ACQUA, a novel system that can be used to predict the best answer. Answer score and user ratings are not used in this model as compared to previous approaches. They tested their approach on 21 websites of Stack exchange having 4 million questions and 8 million answers. They used ADT learning algorithm to predict the best answer and achieved 84% of average Precision and 70% Recall.

Non-technical sites are available on worldwide where users can ask any kind of questions either personal, daily routine or related to professional issues such as yahoo answers and qoura are very popular non-technical sites. Some of the researchers have worked to asses the quality of answers on non-technical sites by performing predictions on yahoo dataset: Adamic et al. [7] presented first study related to best answer prediction using data from Yahoo Answers, they analyzed the categories of forums and cluster them according to the pattern of interaction and content categories among users. They characterize the entropy of users'interest and map the related categories by analyzing the users'participation across them. Moreover, they combined both features such as answer characteristics and user attributes to predict the best answer. They utilized logistic regression classifier and achieved 73% of accuracy for programming related answers and Shah and Pomerantz. [23],Shah [24] also used dataset from yahoo Answers containing non-technical questions to predict the quality answer. [23] Proposed approach based on

13 different criteria's and their feature set also contain user-related features. They selected a small set of questions, where each set contain at least 5 answers and each answer is manually rated by 5 different people based on 13 predefined criterion. They compared the rated answers with asker's rating. They utilized logistic regression and their evaluation results showed 85% of accuracy.

Finally, our baseline approach Calefato et al. [25] used two datasets, first one from stack overflow for training the classifier and second from Docusign for testing purpose. User related features are not used in their model (e.g. number of accepted answers, badges, user reputation) as these features do not exist in old developer forums. In their research work four main categories of feature sets are proposed (1) Linguistic features such as length in character, word count, average characters per word, sentences count, average words per sentence and hyperlinks (2) Vocabulary such as normalized log likelihood i.e. frequency of word is divided by the number of unique words occurring in sentence and Flesch-kincaid Grade (3) Meta features such as age (time difference between answer and posted question), rating score (up-votes minus down-votes of each question) (4) Thread features such as answer count (number of answers to a question). They used Random Tree, Random Forest, J48 and ADT (Alternating Decision Tree) algorithms and achieved 63, 78, 74, and 83% of accuracy, respectively.

2.3 Text Classification through Deep Learning

The exponential growth and complexity in datasets after every year requires more improvement in machine learning techniques to accurately perform the data classification tasks. Recently, the deep learning based models achieved more accuracy

as compared to traditional machine learning models on almost all kind of classification tasks: natural language processing, face recognition and image classification. The success of the deep learning models are based on designing the non-linear and complex relationship within data. The following figure 2.2 shows the hierarchy where the technology advances from artificial intelligence to deep learning.

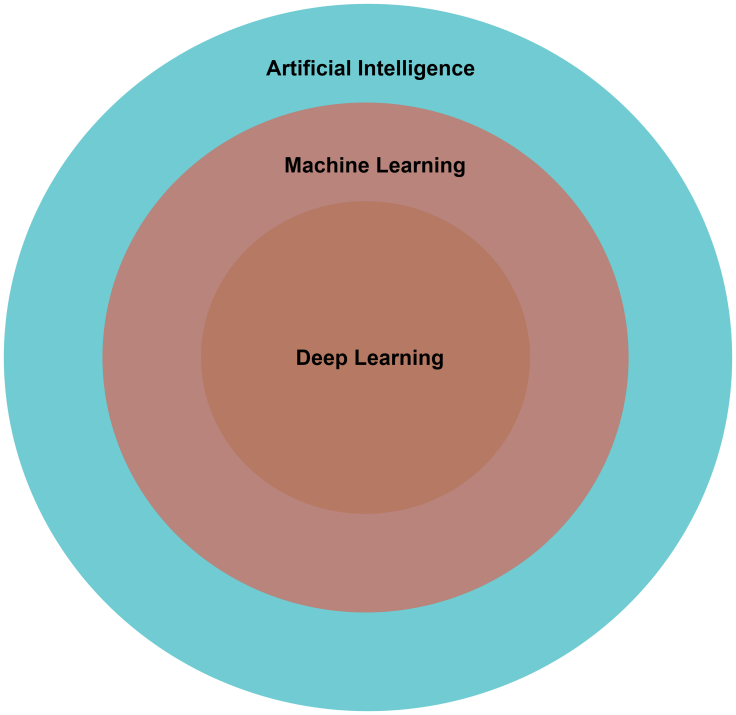

Figure 2.2: Taxonomy of AI

2.3.1 Deep Neural Networks

A simple neural network contains three layers input, hidden and output but when neural network goes deeper with multiple hidden layers called deep neural network, this term refers to deep learning. In deep neural network there are many layers between input and output, allows several stages of non-linear processing unit of information with hierarchical architectures [26] [27] , illustrated in Figure 2.3.

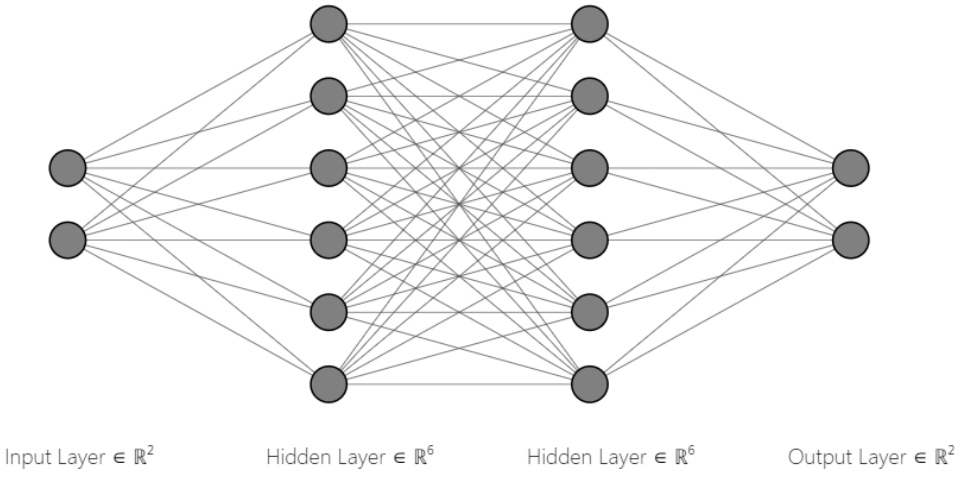

Figure 2.3: Artificial Neural Network Architecture

The structure of the neural network in deep learning is inspired form the biological neurons of the human brain, where the each neuron holds some information in the form of weights. Furthermore, the wights are features extracted from the vectors of text after performing matrix multiplications, derivation and other algebraic and statistical techniques. Deep learning algorithms require large amount of training data to estimate the parameters as compare to machine learning, Alom et al. [28] as shown in figure 2.4.

2.3.2 Types of DL approaches

Deep learning approaches are also categorized as semi-supervised, supervised and unsupervised learning like machine learning. Firstly, supervised deep learning techniques uses labeled data, if the input value X_t is given, the DL algorithm predicts $\hat{y}_t = f(x_t)$, with loss value of $l(y_t, \hat{y}_t)$ and for the desired output, algorithm iteratively modify the parameters of NN. There are several supervised deep learning algorithms such as Deep Neural Networks (DNN)[29], Convolutional Neural

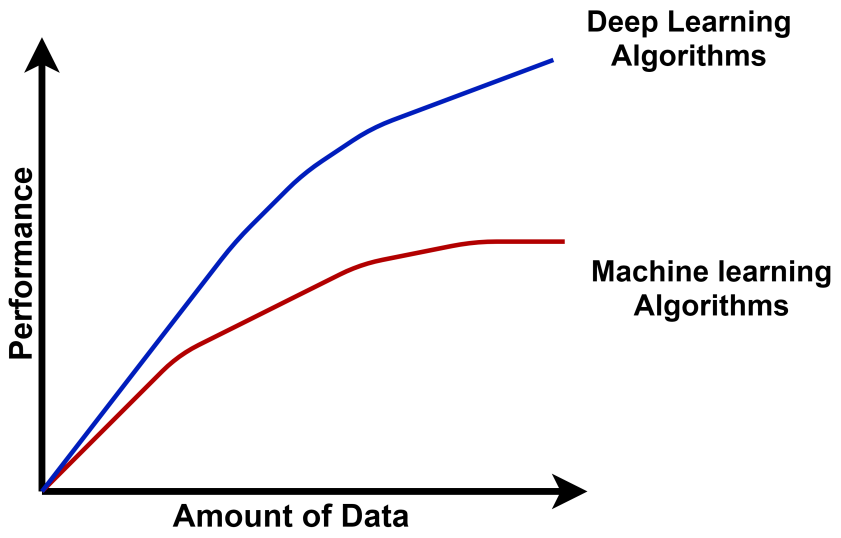

Figure 2.4: Data vs Machine/Deep Learning

Networks (CNN)[30], Recurrent Neural Networks (RNN) [31] like Long Short Term Memory (LSTM) [32]. Secondly, semi-supervised deep learning based on partially labeled data, the algorithms sometime used for semi-supervised deep learning approaches are Generative Adversarial Networks (GAN), (Deep Reinforcement Learning) DRL, Recurrent Neural Networks (RNN) and Long Short Term Memory (LSTM). Thirdly, unsupervised learning is learning of algorithms without labeled data, the algorithms discover structured or unstructured relationships within input data by learning important features and internal representation. Often dimensionality reduction, clustering and generative techniques are considered unsupervised deep learning approaches such as Generative Adversarial Networks (GAN), Recurrent Neural Networks (RNN) including Long Short Term Memory LSTM.

2.3.3 Feature Selection

The major difference between machine and deep learning algorithms are their feature extraction techniques. In machine learning, some traditional handmade fea-

ture extraction algorithms for computer vision such as Speeded Up Robust Features (SURF)[33], Scale Invariant Feature Transform (SIFT) [34], RANSAC[35], GIST[36], Local Binary Pattern (LBP)[37], Histogram Oriented Gradient (HOG)[38], Empirical mode decomposition (EMD)[39]. Moreover, several methods of feature extraction available for text classification such as MI [40], frequency, pLSA [41]] and LDA [42] to extract the more discriminatory features. Furthermore, a very common feature extraction techniques in text classification used by machine learning algorithms is Bag-of-word (BoW) representation including unigram, bigram and n-gram.

2.3.4 Why deep learning

2.3.4.1 Universal learning approach

Deep learning approaches can be implemented in any application domain so these approaches called Universal learning approach.

2.3.4.2 Robust

Machine learning algorithms do not extract the features automatically, there are different techniques available to extract features from text and feed into machine learning algorithms. Furthermore, deep learning automatically extract optimal features from input matrix that are appropriate for prediction.

2.3.4.3 Generalization

Deep learning is being use in many different domains, the term generalization refers that the same deep learning algorithms could be used in different domain of

applications or even with different datatype.

2.3.4.4 Scalability

The Microsoft presented a neural network named ResNet[43] containing 1202 layers. This network is more accurate than human. The human shows 5% error rate and ResNet-152 shows only 3.57% of error.

2.3.5 Convolution Neural Network

Deep neural network named Alexnet which achieved high accuracy and outperformed previously presented neural networks Krizhevsky et al.[30]. Many researchers have worked on the CNN for text classification to learn word-vector representation Bengio et al. [44]; Mikolov et al.[45] in deep learning due to its differ technique for feature extraction as compared to traditional machine learning. CNN for text classification takes input matrix of text values, process it and classify text into predefined categories.

Furthermore, most of the NLP tasks use tokenized document as input where each word of document represents a token, then document is converted to matrix of vectors. Tokens are replaced with vectors and each word corresponds to one row of matrix. Tokens are converted to vectors using word embedding like Glove[46] or Word2vec[47] or either be converted by one-hot encoding. If dimension of word-vector is d and length of sentence is s then dimension of sentence matrix is $s * d$.

First layer is embedding layer which convert Word2Vec by measuring the relatedness between two embedding vectors from its distance. Simply, the word embedding is representation of word as a dense vector. Second layer is convolution, perform

convolve operations in which the filters are applied of different length or sizes, convolution layer takes filter and matrix as input, each filter generate some output by adding up the sum after multiplying with corresponding cell of input matrix. We implemented convolution operation on our data to extract features by using filters and window size of n words. A filter matrix w is applied on window of words h which extract new features c_i from window of words $x_i : i + h - 1$ by

$$c_i = f(w.x_i : i + h - 1 + b)$$

Where, f is non-linear function, b is biased and $x_i : i + h - 1$ is each possible window of words of sequence $x_1 : h, x_2 : h + 1, ..., x_n - h + 1 : n$ and produce feature map of

$$c = [c_1, c_2, ..., c_n - h + 1]$$

Furthermore, the length and width of filter can be of any size in the case of image but the width of filter or kernel in the case of word representation refers to the dimension of entire word embedding. Filter output changes due to the stride, the step size or the number of pixels the filer will shift over input matrix. Sometime the filter does not fit on the input matrix then extra pixels are applied known as padding. Instead of 2D structure like image matrix the text has 1D (one dimensional) structure. Convolution is the first layer which extract features from the matrix. Second layer known as pooling performs downsampling operations to reduce the size of the matrix and extracted features. So, we are using Conv1D to extract features from input data and apply MaxPooling1D[48] on each feature map. Moreover, another feature reduction technique used in CNN is dropout which stochastically disable the neurons and force them to learn different features to prevent the model

from overfitting. Rectified Linear Unit (ReLu)[49] perform non-linear operations, replace all negatives values $f(x) = max(0, x)$ with 0 from matrix and keep only positive values. Final layer is Fully Connected (FC) layer which flattened the feature map matrix, convert matrix into vector and combine the all features together extracted from the previous layers and generate the model. After this a activation function softmax or sigmoid classify the output. The architecture of CNN for text classification is shown in Figure2.5.

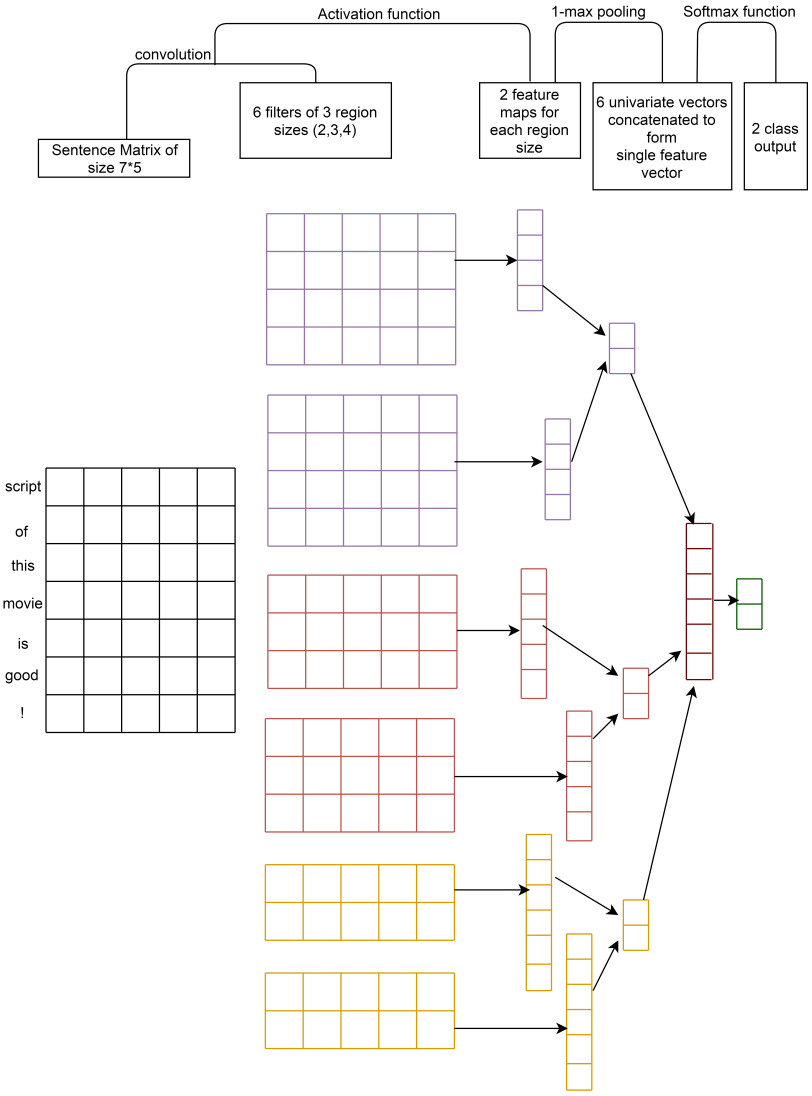

Figure 2.5: CNN Architecture

2.3.6 Long-Short term memory

Recurrent neural network (RNN) is a neural network in artificial intelligence which has input state, hidden state and output state. RNN is a kind of feed-forward network which takes the sequential data and predict the next sequence while training on the basis of information learned from previous predictions, every prediction uses the information from all previous predictions. RNN does not have the ability to handle long context and could only works fine with the dealing short-term dependencies. This problem of RNN leads to vanishing gradient and exploding gradient, in conventional feed forward neural network such as RNN where on the particular layer input depends on the updated weights that are multiple of learning rate and the error term from previous layers. Moreover, the error term is the product of errors from the previous layers. Then the small values came from the derivative multiplied many times before moving towards to the starting layers while dealing with an activation function. Therefore, the RNN can only holds the information for just less duration of time, this issue of vanishing gradient in RNN resolved in its extended version known as Long Short-Term Memory (LSTM) by introducing the memory unit. Self-connected hidden states are replaced with the memory blocks as shown in Figure 2.6 taken from [28].

There are four main components in LSTM, input gate which controls the memory content to be enter as input and forgotten gate f_t which controls the amount of memory came from the previous states to be remove and keeps important information and the output gate o_t which modulate the memory content of output. The memory cell of LSTM takes two inputs, one is new memory c_t and other from previous states c_{t-1}.

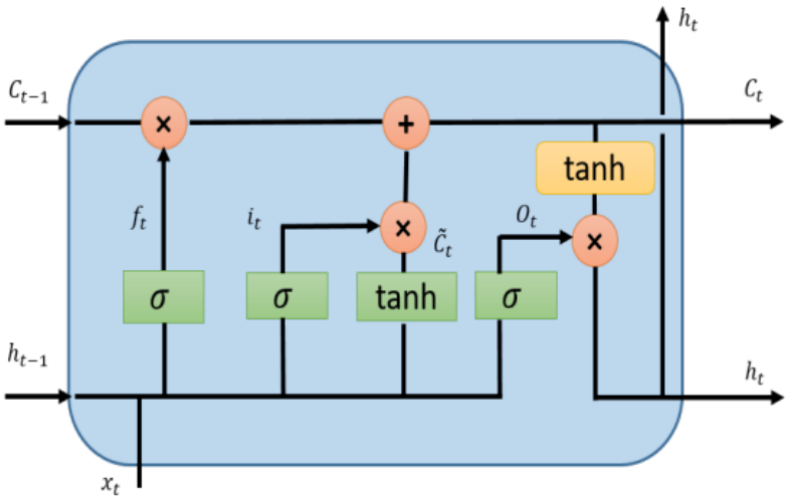

Figure 2.6: LSTM Architecture

LSTM compute the hidden state h_t by using following steps:

- The Forget gate f_t takes two inputs, x_t and h_{t-1} and applies sigmoid activation function on these inputs.

$$f_t = \sigma(w_{xf}x_t + w_{hf}h_{t-1} + b_f)$$

- Input state I_t is responsible to store new information in cell state. It performs two operations on two inputs x_t and h_{t-1} update values using sigmoid function and produce new value for candidate cell $\hat{c}$ using tanh function.

$$I_t = \sigma(w_{xi}x_t + w_{hi}h_{t-1} + b_i)$$

$$\hat{c}_t = tanh(w_{xc}x_t + w_{hc}h_{t-1} + b_c)$$

To update the previous cell state c_{t-1} combine these two equations to make new cell state c_t.

$$c_t = f_t * c_{t-1} + i_t * \hat{c}_t$$

- Output gate o decide the amount of contribution of cell state c_t to generate new hidden state using sigmoid function.

$$o_t = \sigma(w_{xo}x_t + w_{ho}h_{t-1} + b_o)$$

Moreover, the new hidden state is.

$$h_t = o_t * tanh(c_t)$$

In all the above equations b_i, b_f and b_o are the biases.

CHAPTER 3

Approach And Architecture

3.1 Overview

In this research, we present an automated deep learning based approach to predict the answer into one of two categories: accepted or rejected. Overview of the proposed approach is illustrated in Figure 3.1.

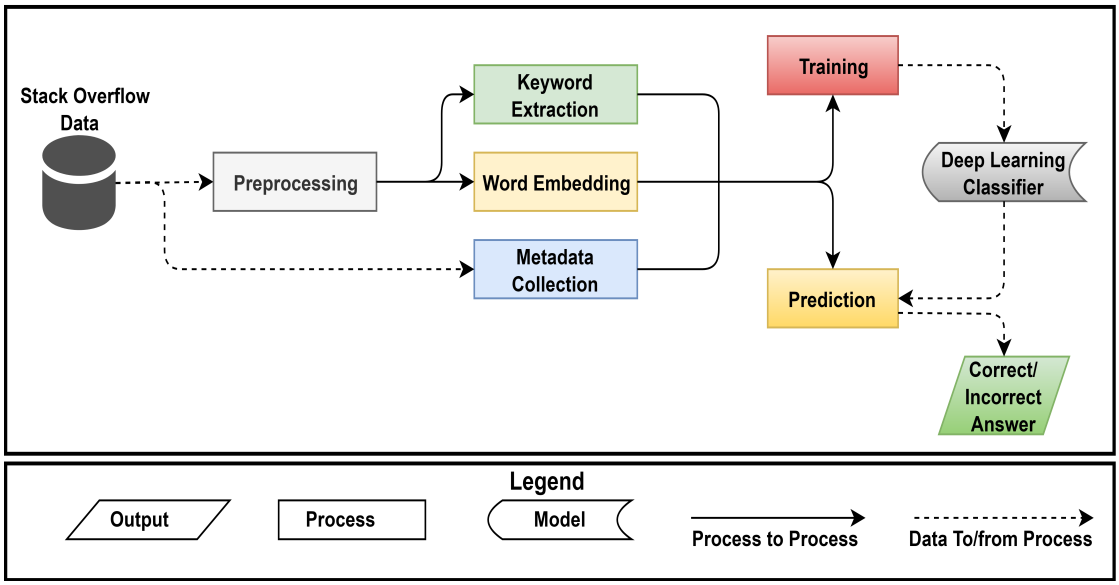

Figure 3.1: Proposed Approach

We categorize this process into five main steps discussed in this section. In the first step, we extract the metadata features (m-features) from answers such as linguistic, vocabulary, meta and thread, briefly discussed in section 3.4.1. Second step is to exploit preprocessing techniques in which we preprocess and clean answers text using natural language processing techniques and self written python scrips. Third step involves in extraction of keywords (k-features) and the calculation of ranking score of most important keywords to find relationship between question and its answers. Moreover, the use of keywords based ranking as dominant feature to predict the most appropriate and suitable answer in developer forums is novel approach. The importance and working of keyword ranking discussed in section 3.4.2. Fourth step is to convert textual description of answers into vectors (t-features) using word-embedding technique popular for deep learning models discussed in section 3.4.3. In Natural language Processing word2vec derives the relationship between word and its all contextual words. Fifth step is to give m-features, k-features and t-features into proposed integrated model which is built from the combination of three deep learning models: Convolution Neural Network (CNN) and two different Long Short Term Memory (LSTM) models to train the classifier. Finally, the trained model is perform classification on unseen stack overflow answers.

$$Cl = f(answer)$$

$$Cl \in \{Accepted, Rejected\}$$

$$answer \in \{a_1, a_2, ..., a_n\}$$

Where f represents the classification function for best answer prediction in devel-

oper forums, Cl is the final prediction of classifier either the given input answer is accepted or rejected, answer is one that given as input to predict quality from the set of answers.

3.2 Data Acquisition

Stack overflow works on voting system, the asker has right to select the best answer directly from the list of answers and other users who visit for smiler problem have ability to make voting on answers under one thread, it could be up-voting or down-voting. The answer selected by user or high voted answer is assigned with accepted answer Id and other considered in normal answers category. We use stack overflow real time questions and answers dataset available on stack exchange. First we separated all questions and answers from Posts file available in xml form. Second, we extracted the questions and their attributes on the basis of five tags (java, .net, php, ruby and misc) used in our baseline approach. Third, the answers and their attributes of all extracted questions are separated and assigned them labels: accepted/rejected on the basis of accepted answer Id given against each question. Finally, after implanting all these steps the data came into format. The questions (Q) and answers (A) are formalize as:

$$Q = \quad < s, cc, t, vc, ac >$$

$$A = \quad < s, cc, t >$$

where s represents the score, cc is comment count , t is the textual description, vc is the view count and ac is answer count against each question.

3.3 Preprocessing

It is important to apply preprocessing techniques to remove the noise from data and convert textual information into useful features. In Python, standard preprocessing techniques are applied by using Natural Language Processing (NLP): tokenization, stop-words removal, spell correction ,Stemming, lemmatizing and lowercase conversion. In this procedure, each answer is preprocessed before the extraction of t-features and k-features. The stages involve in preprocessing is illustrated in Figure 3.2.

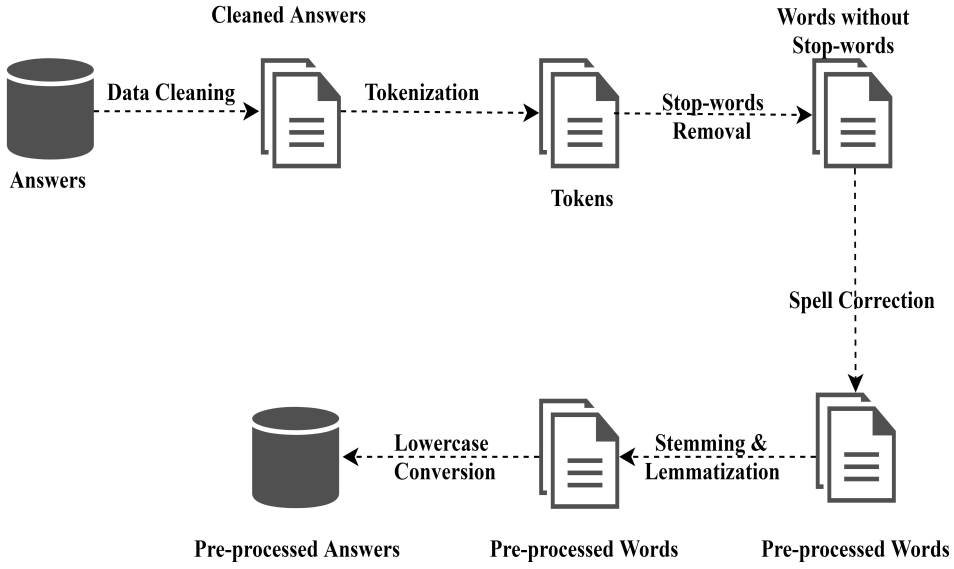

Figure 3.2: Preprocessing

3.3.1 Data Cleaning

In this approach, we are using Stack Overflow dataset where users post questions and answers containing html tags, codes snipes and site links. Therefore, it is very

important to clean the text and covert data into format to train the deep learning based classifier, so we also included data cleaning step in preprocessing. The following steps are involved in data cleaning activities.

- Removal of punctuation
- Removal of integer, numbers
- Removal of code snipes
- Removal of html tags
- Removal of extra spaces
- Removal of links

3.3.2 Tokenization

This is the process of breaking up the text document into separate words where each word represents as a token. In this step, the answers against each question is conerverted into separate words/tokens.

3.3.3 Stop-Word Removal

Stop-words are those words in the text document which occur frequently and holds no actual meaning. Therefore, removal of these words are important because they do not carry much information needed to train the model: 'is', 'am', 'are', 'they', 'we', 'you '.

3.3.4 Spell Correction

we are dealing with the data of technical Q&A's site where the focus of users are to give understandable and satisfactory answers. So, there must be a lot of spell mistakes need to be corrected before feeding the text to train the model.

3.3.5 Stemming & Lemmatization

In stemming each word is converted into its root word such as 'shouting'is reduced into its root word 'shout'. Stemming all words in data reduce the size of dataset, train less features help to increase processing time. Furthermore, lemmatization is the process of converting word to its dictionary form such as 'best', 'good', 'better'are having similar dictionary meaning so they lemmatize to 'good'.

3.3.6 Lowercase Conversion

In natural language processing each word in document is converted into fixed length numeric vector before feeding into machine learning based models and the machine learns on the basis of features/words and patterns where each vector of word treated individually. So, two same words start with capital and small letter will be considered separate words. Therefore, we transform whole text of answers into lowercase and stored them as preprocessed answers.

A preprocessed answers are formalized as:

$$A = \quad < s, cc, t' >$$

$$t' = \quad < t_1, t_2, ..., t_n >$$

Where t is description of answers and t' represents preprocessed description of answers and $t_1, .., t_n$ are set of tokens in preprocessed description of answers.

Table 3.1 shows the implementation of preprocessing techniques: Tokenization, Stop-words Removal, Spell Correction , Lemmatization and Lowercase Conversion on single answer.

Table 3.1: Preprocessing Example

Actual Answer	You could use any of the DLR languages which provide a way to really easily host your own scripting platform
After Tokenization	'could', 'use', 'any', 'of', 'the', 'DLR', 'languages', 'which', 'provide', 'a', 'way', 'to', 'really', 'easily', 'host', 'your', 'own', 'scripting', 'platform'
After Stop-words Removal	'could', 'use', 'DLR', 'languages', 'provide', 'way', 'really', 'easily', 'host', 'scripting', 'platform'
After Spell Correction	'could', 'use', 'DLR', 'languages', 'provide', 'way', 'really', 'easily', 'host', 'scripting', 'platform'
After Stemming	'could', 'use', 'DLR', 'languag', 'provid', 'way', 'really', 'easy', 'host', 'script', 'platform'
After Lematization	'could', 'use', 'DLR', 'language', 'provide', 'way', 'really', 'easy', 'host', 'script', 'platform'
After Lowercase Conversion	'could', 'use', 'dlr', 'language', 'provide', 'way', 'really', 'easy', 'host', 'script', 'platform'

3.4 Feature Extraction

Feature extraction is very important in machine/deep learning because machine learns on the basis of features or patterns. In proposed approach, we extract three main features from textual description of answers: metadata (m-features), keywords (k-features) and vectors of preprocessed textual description (t-features). Finally, these extracted features are used to train proposed deep learning based integrated model.

3.4.1 Metadata Extraction

In this section, we will extract the metadata (m-features) from the answers text before the implementation of preprocessing techniques. We included four main categories of features in metadata used in our baseline paper calefato et al[25] such as linguist, vocabulary, meta and thread. Linguist category is included: length in character, word count, sentence count, longest sentence, average words per sentence, average character per word, contained hyperlink in answers posted by users. These features are concerned with the readability and important to check the answer quality. Linguist features are computationally inexpensive. Previous studies show that the answers containing hyperlink are appropriate. So, we also included this feature in our consideration and assigned Boolean values for the answers containing hyperlink as 'True'and the answers containing no hyperlink as 'False'. Moreover, in vocabulary feature we calculate the readability and complexity of text by using Flesch-Kincaid Grade used in baseline approach and they calefato et al[25] also discussed this feature did not effect on performance of the classifier.

$$Flesch - Kincaid - Grade = 0.39(\frac{T_{words}}{T_{sentence}}) + 11.8(\frac{T_{syllables}}{T_{words}}) - 15.59$$

In addition, meta category contains rating score (higher the rating score of answer reflects the popularity of answer) and one of the feature belonging to thread category consist of answer count for each question. The users of Q&A sites: stack overflow posts thousands of questions on daily basis and these questions either get number of answers or remain unanswer. Number of answers to a question indicate its worth and credibility. Therefore the answer count is also included in feature matrix used for training and testing deep learning based classifier for proposed approach.

$$m - features = \quad < lc, wc, sc, ls, awps, acpw, hl, s, ac, ri >$$

Where lc is length in character, wc is word count, sc is sentence count, ls is longest sentence, $awps$ is average words per sentence, $acpw$ is average character per word, hl is containing hyperlink or not, s is rating score, ac is answer count and ri is readability index.

$$A = \quad < m, t >$$

Where A is answer and m is its m-features and t is the textual description of that answer.

3.4.2 Keyword Extraction and Ranking

Keyword extraction is very popular technique in machine learning to identify key segments, key phrases, key terms that represents the subject and retrieve information from document. In Natural Language Processing, automatic keyword extraction is an essential in research direction. It facilitate the users with best description and condense representation of document. Automated keyword generation process is divided into two categories such as keyword extraction and keyword assignment Siddiqi and Sharan.[50]. In keyword assignment, words of controlled vocabulary are used to select possible keywords in the document, although the most relevant words are automatically identify in document by using keyword extraction. Furthermore, we use the keyword extraction to extract keywords from document and assign them with corresponding weight or ranking score. The ranking score define the significance of the word and obtained by various techniques as statistical methods (TF.IDF [51], Yake[52], Rake[53], KP-Miner[54]), supervised

machine learning methods (MAUI[55], KEA[56]) and unsupervised machine learning graph based methods (ExpandRank[57], SingleRank[57], TopicPageRank[58], PositionRank[59], TopicRank[60], MultipartiteRank[61] , TextRank[62]). Moreover, in this paper we are using TextRank for the extraction of keywords and assigning ranks to these keywords according to their importance because it has significance performance on short text. . we named ranked answers to k-features which are used to train the deep learning based classifier. TextRank is a graph based unsupervised methodology keyword extraction model in text processing which consider all words as vertices and identify the importance of vertex with graph. Then the edges are drawn between words to specify their relationship on the basis of co-occurrence of words in particular window. This is how the undirected graph is generated from a sequence of text. In this way the important information computed from the graph reclusively. TextRank model assigns score to each word or vertex on the basis of voting system, when a vertex connects to another then it casts a vote for other vertex. Furthermore, higher the casted votes for a vertex shows higher the importance for that vertex. The score assigned to a vertex is calculated on the basis of votes casted for it.

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

Where, $In(Vi)$ represents the set of vertices that point to it, d is damping factor set between 0 and 1, vertex Vi points to the set of vertices $Out(V_j)$. First, we implement the preprocessing by using Natural Language Processing techniques such as tokenization, stop-words removal, spell correction, stemming and lemmatization

before extraction of the keywords from each answer by using TextRank algorithm.

$$A =< m, k, t' >$$

Where m is metadata features and k is keyword ranking of preprocessed textual description and t' is preprocessed textual description.

3.4.3 Word Embedding

In proposed approach, we are also using (t-features) textual information (answers) with k-features and m-features. The machine/deep learning based classifiers does not understand textual information directly, it is very important to convert text into vector representation where preprocessed textual information is transformed into fixed length vector based on numeric values. In this research, we utilize skip-gram (word2vec) model [45] due to its ability to capture the semantic and synthetic relationship between words and return high quality distributed n-dimension vector. Skip-gram model is unsupervised pre-trained three layers neural network with input, hidden and output layer, which uses no activation function at hidden layer and use softmax classifier at output layer. When the input is given to skip-gram model it finds the context word or most related words for target word. Overview of the skip-gram model is shown in Figure3.3.

Where $W(t)$ is the target word which is enabled in input vector $V * 1$ and performs dot product with the weighted matrix W_{V*N} and the resultant vector $N*1$ of hidden layer performs the dot product of the weighted matrix W_{N*V} of output layer and softmax classifier at output layer computes the probabilities of words appearing in the context of target word $W(t)$.

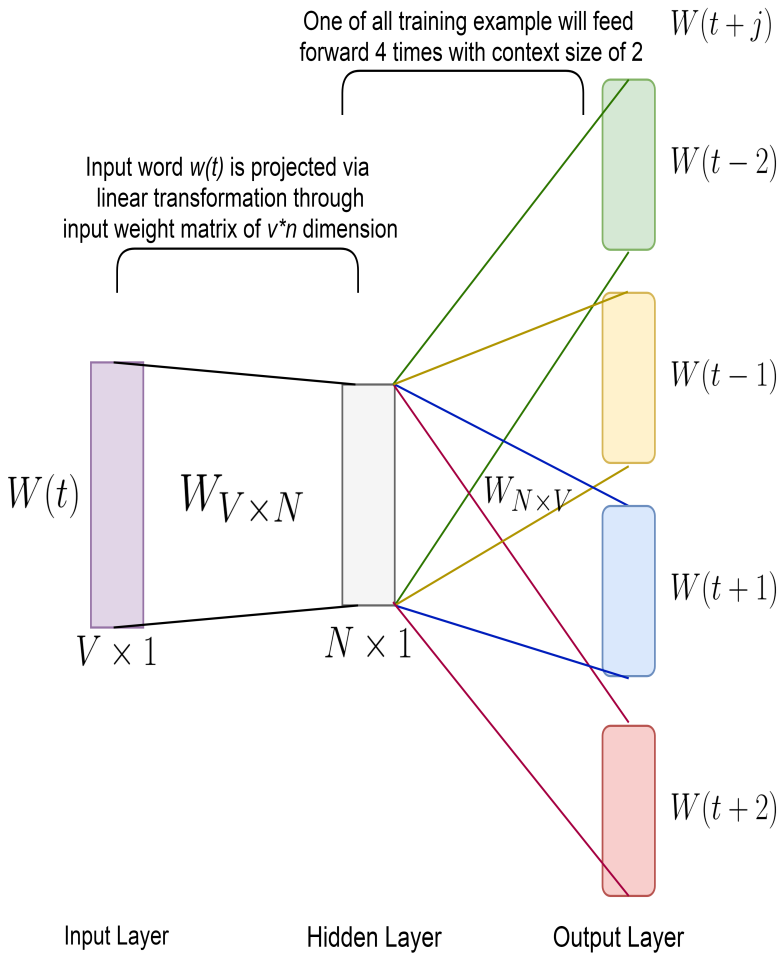

Figure 3.3: Skip-gram Model

After the implementation of word-embedding on answers, the answer A can be defined as,

$$A =< m, k, t" >$$

3.5 Ensemble Deep Learning Model

The architecture we used for ensemble deep learning model presented in Figure3.4 consist of three deep learning based classifiers: Convolution Neural Network(CNN) and two long short term memory (LSTM). The proposed model receives three in-

put features separately in the form of vectors that we extracted in this research k-features, m-features and t-features. we feed metadata (m-features) to CNN, keyword ranking score of preprocessed textual description (k-features) and preprocessed textual description (t-features) to LSTM, respectively. The setting we used for CNN is Number of filters = 128, size of kernal = 1, activation function for each hidden layer = tanh and the loss function is binary-crossentropy, output of CNN is frowarded to flatten layer [63] and then forwarded to dense layer. On the other side the setting we used for LSTM: dropout = 0.2, activation function for each hidden layer = sigmoid, loss function is binary-crossentropy and the output of LSTM is forwarded to dense layer. After this, the output of all three deep learning classifiers is combined at merge [64] layer and forwarded to dense layer which maps the merged output of all three classifiers into single output.

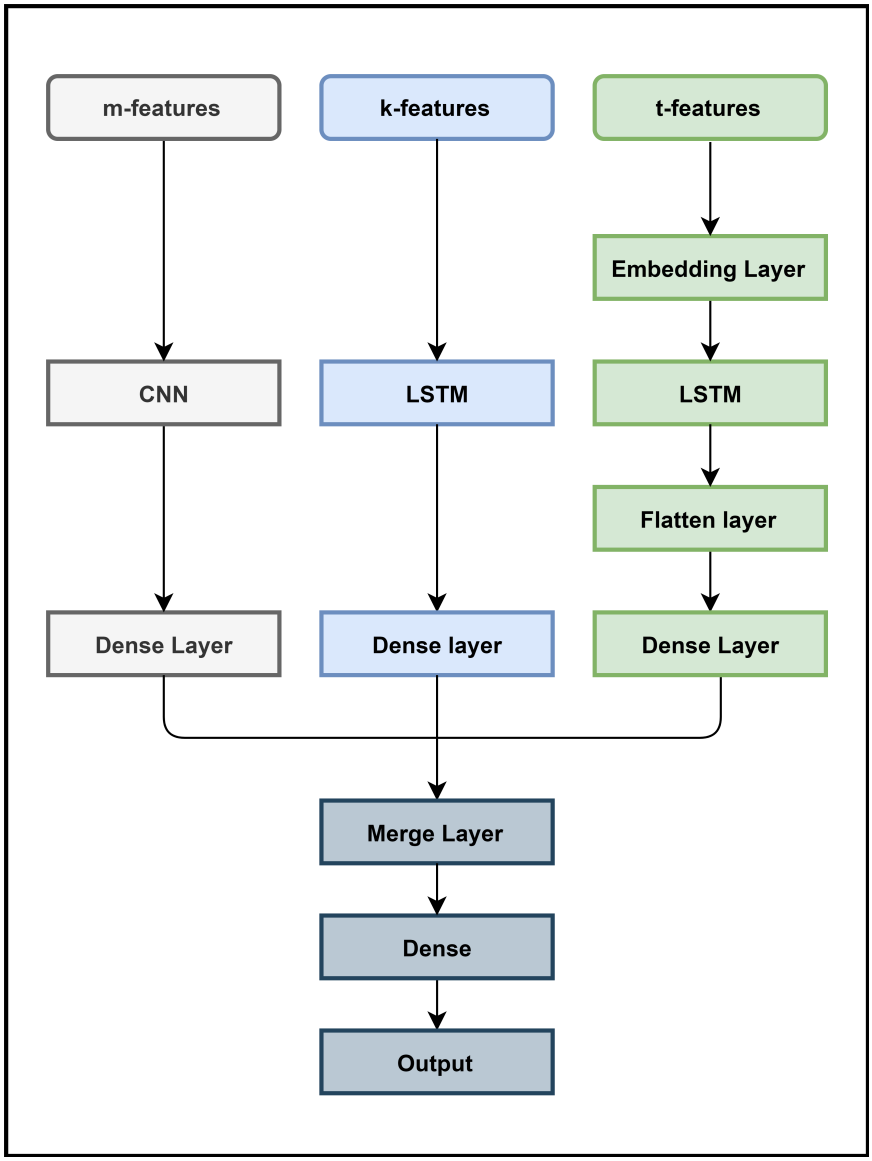

Figure 3.4: Proposed Model

CHAPTER 4

Result And Discussion

In this section, we will raise five research questions and investigate them to evaluate the best answer prediction approach. After this, we will discuss the evaluation process and metrics which we used to evaluate and validate the proposed approach and finally we will briefly answer those research questions with results.

4.1 Research Question

We evaluate proposed approach by investigating these five conducted research questions.

- Q1: Does proposed approach outperform the state-of-the-art approaches, the best answer prediction in developer forums?
- Q2: Does text-ranking of answers influence their possibility to be accepted?
- Q3: Does preprocessing influence the performance of the proposed approach?
- Q4: Does re-sampling help to improve the performance of the proposed approach?

- Q5: Does proposed approach outperforms the traditional machine/deep learning algorithms?

4.2 Process

We collect and reuse the stack overflow open source dataset in our research work, the complete dataset contains around 17 million of questions and almost 26 million of answers and the amount is increasing because of millions of users visit stack overflow each month, therefore we extracted question and their answers based on five tags (java, .net, php, ruby and misc) which were used in our baseline approach to evaluate the results and compare them with the results presented by calefato et al[25]. Then we separate the accepted and rejected answers and divide the dataset into two parts training and testing to perform the cross-validation. Then, we implement preprocessing techniques by using Natural Language processing (NLP) to reduce the noise from answers. We train tree algorithms for compression with baseline and MNB, Logistic Regression to compare the performance of the proposed model with traditional machine learning algorithms. Furthermore, we train our proposed deep learning based ensemble model discussed in section 3.5 consist of three deep learning classifiers CNN and two LSTM and feed them with three main features extracted in this research, m-features to CNN and k-features and t-features to LSTM, we use LSTM for textual features because it works better on short text. We divide answers data into 10 segments and for each iteration we use i segment for testing and rest of the data for training to perform 10-fold cross validation. After, we compute the evaluation metrics for each machine/deep learning algorithms to analyze and compute their performance.

4.3 Metrics

Performance of the proposed approach is evaluated on basis precision, recall, f-measure and accuracy because these are well known metrics to evaluate the performance of the classifier. Mathematical equations of the precision, recall, F1-score and accuracy are defined as follows:

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

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

$$F - measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Where, true positive (TP) represents those positive sentences that are correctly predicted by the model as positive and true negative (TN) are those negative outcomes which are correctly predicted by model as negative. False positive (FP) are those positive outcomes that are incorrectly predicted by model as positive and false negative (FN) are those negative outcomes which are incorrectly predicted by model as

negative class.

These evaluation measures are used to evaluate the performance of classifiers. The reason of using these evaluation parameters is because they are widely used to measure the performance of classifiers.

4.4 Results

4.4.1 Q1: Comparison of proposed approach with state-of-the-art

The cross validation results shows that the proposed approach outperforms the state-of-the-art approach. Comparison of proposed approach with Calefato et al[25] is presented in Table 4.1. The results evaluated on stack overflow dataset for both proposed and baseline approach which is express that average accuracy, precision, recall and f-measure of the proposed approach is 84.39, 96.16, 84.52, 86.29%, respectively. Similarity, the performance of the baseline approach suggest that its precision 76.95, recall 79.31, f1-score 77.15 and 82.96% of accuracy, respectively.

Table 4.1: Performance of the Proposed Approach and the State-of-the-art Approach

Proposed Approach				**State-of-the-art Approach**			
Acc	Precision	Recall	F-Measure	Acc	Precision	Recall	F-Measure
84.39%	96.16%	84.52%	89.97%	82.96%	76.95%	79.31%	77.15%

Where acc is accuracy.

The results comparison suggest that the proposed approach improved in accuracy, precision, recall, and f-measure is $1.72\% = (84.39\% - 82.96\%)/82.96\%$, $24.96\% = (96.16\% - 76.95\%)/76.95\%$, $6.57\% = (84.52\% - 79.31\%)/79.31\%$, and $16.62\% = (89.97\% -$

$77.15\%)/77.15\%$, respectively. According to these results extracted from proposed approach, we conclude that the proposed approach outperforms the state-of-the-art approach.

Moreover, we identified that the stack over flow dataset is imbalanced Fabio et al[65] because one answer is marked accepted among all answers under one question thread. Therefore, the class labeled as accepted belongs to minority class and class labeled as rejected belongs to majority class. We perform under-sampling and over-sampling, the popular techniques to handle the imbalanced data on the stack overflow dataset used in proposed approach.

4.4.2 Q2: Influence of text-ranking on best answer prediction

In this section, we analyze the influence of text-rank discussed in section 3.4.2 on proposed approach to answer the research question Q2. First, we train the only m-features extracted from answer's text such as linguist, vocabulary, meta and thread mentioned in section 3.4.1. Second, we train the only textual features (t-features) based on accepted and rejected answers where text is converted into numeric vector using word-embedding technique before feeding into deep learning algorithm. Third, we train only text-rank (k-features) to evaluate the performance of classifier to check the impact of ranking on proposed approach. Forth, we train m-features and k-features combined to check the impact of keyword ranking when used with metadata. Finally, we use proposed deep learning based ensemble model which takes vectors of m-features,k-features and t-features, individually, results of these experiments presented in Table 4.2.

The comparison results shows that the proposed model gives good results with en-

Table 4.2: Influence of the text-ranking

Input	Accuracy	Precision	Recall	F-Measure
Metadata+Keywords+Text	84.39%	96.16%	84.52%	89.97%
Metadata+Keywords	82.46%	94.72%	78.69%	85.96%
Metadata Only	71.88%	76.95%	79.31%	77.15%
Text Only	73.52%	82.44%	78.91%	80.64%
Keywords Only	76.33%	84.30%	66.43%	74.31%

abling all features keywords ranking, metadata and Text. Notably, when the metadata and keywords are disable and only text is enable the deep learning classifier decreases the accuracy from 84.39% to 73.52%, precision from 96.16% to 82.44%, recall from 84.52% to 78.91% and f-measure from 89.97% to 80.64%, respectively.

Similarly, enabling keywords and disabling text and metadata also significantly reduction in performance of the classifier from 84.39% to 76.33%, precision from 96.16% to 84.30%, recall from 84.52% to 66.43% and f-measure from 89.97% to 74.31%, respectively. On other hand, if we disable text and keywords and enable metadata it also decreases the performance. So, it is concluded that with the help of results, enabling all features is important to enhance the performance of the proposed approach to predict best answer in developer forums.

4.4.3 Q3: Influence of pre-processing on proposed approach

The answers on stack overflow contains punctuation, codes, integers and html tags that need to be removed from text as discussed in section 3.3. The data cleaning is an important step in natural language processing before feeding text into

machine/deep learning based classifiers because it reduces the computation cost, decreases the processing time and increases the performance of the classifier.

Table 4.3: Influence of preprocessing on proposed approach

Input	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Enabled	84.39%	96.16%	84.52%	89.97%
Disabled	81.97%	95.02%	82.12%	88.10%
Improvement	2.95%	1.19%	2.92%	2.12%

To answer the research question Q3, we compare the results of proposed approach with and without the preprocessing to check its influence on deep learning based proposed model. The evaluation results show in table 4.3 with different inputs: enable/disbale. The first row of the table represents the performance measures when preprocessing is enabled and second row represents the results when preprocessing is disabled. The accuracy, precision, recall and f-measure with $(84.39\%, 96.16\%, 84.52\%$ and $89.97\%)$ and without $(81.97\%, 95.02\%, 82.12\%$ and $88.10\%)$. It is observed that the preprocessing significantly improve the performance of the proposed approach by $2.95\% = (84.39\% - 81.97\%)/81.97\%$ of accuracy, $1.19\% = (96.16\% - 95.02\%)/95.02\%$ of precision, $2.92\% = (84.52\% - 82.12\%)/82.12\%$ recall and $2.12\% = (89.97\% - 88.10\%)88.10\%$, of f-measure, respectively.

4.4.4 Q4: Influence of Re-sampling on proposed approach

To answer the research question Q4, we implement two techniques under-sampling and over-sampling to balance the dataset. Then we train the classifier after the both under-sampling and over-sampling to analyze the effect of re-sampling on pro-

posed approach and predict results. The dataset is imbalanced when the number of instance belong to classes have larger difference. The class comparatively having large number of samples than the other is categorize to majority class and if the class have less number of samples as compare to other is known as minority class. Whenever, the model train with the imbalanced dataset, it generates a certain bias directed to majority class.

Furthermore, to resolve this issue, we use re-sampling in proposed approach to equal the samples of both classes and adjust the distribution of dataset. We use Synthetic Minority Over-sampling Technique (SMOTE) Chawla et al[66] for minority class which finds K-nearest neighbors to randomly choose samples for minority class and we select n random samples from majority class for under-sampling.

Table 4.4: Influence of the re-sampling

Re-sampling	**Accuracy**	**Precision**	**Recall**	**F-Measure**
No-sampling	84.39%	96.16%	84.52%	89.97%
Under-sampling	73.14%	74.56%	73.38%	73.97%
Over-sampling	87.95%	89.11%	87.70%	88.40%

The evaluation results discussed in Table 4.4 shows that the performance measures of the proposed approach accuracy, precision, recall and f-measure for under-sampling and over-sampling is $(73.14\%, 74.56\%, 73.38\%$ and $73.97\%)$ and $(87.95\%, 89.11\%$, 87.70% and $88.40\%)$, respectively. It is observed that the accuracy and recall of the proposed approach is improved by $(87.95\% - 84.39\%)/84.39 = 4.22\%$ and $(87.70\% - 84.52\%)/84.52 = 3.76\%$, respectively. The precision of the model with over-sampling is 89.11% which is not improved with re-sampling but with the balanced data it

is very good prediction by the machine because with the imbalanced data machine tends to bias with the majority class (rejected answers). Under-sampling does not perform well in our case because of dropping a lot of important data from majority class to balance the dataset and deep learning algorithms needs larger data to train the model[28] as we are using ensemble model (combination of three deep learning algorithms) have more dropout and dense layers.

4.4.5 Q5: Comparison of proposed approach with machine/deep learning algorithms

In text classification, there are widely used machine learning algorithms such as Naive bayes. Logistic regression, random forest and support vector machine. These algorithms are used by researchers and automate the real time systems due to their competitive performance Sohrawardi et al.[67]. To answer research question Q5, we investigate that how machine/deep learning based classifiers perform in our case, in this research we compare the results of these algorithms with purposed deep learning based ensemble model. We train and test naive bayes. Logistic regression, random forest and support vector machine classifiers on same dataset which we are using for proposed approach. The results of evaluation from each classifier are shown in Table 4.5.

It is observed from the performance of each classifier that the proposed model outperforms on the all other machine/deep learning algorithms with 84.39% of average accuracy, 96.16% of precision, 84.52% of recall and 89.97% of f-measure. The results shows accuracy, precision, recall and f-measure of LSTM (82.53%, 93.29%, 83.67% and 88.22%) and CNN (82.20%, 95.71%, 82.25% and 88.47%) are performs well as

Table 4.5: Comparison of proposed approach with different machine/deep learning algorithms

Classifiers	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Proposed model	84.39%	96.16%	84.52%	89.97%
CNN	82.20%	95.71%	82.25%	88.47%
LSTM	82.53%	93.29%	83.67%	88.22%
MNB	67.88%	65.93%	68.57%	67.22%
LR	81.15%	76.33%	81.19%	78.69%
RF	72.43%	74.35%	78.69%	76.46%
DT	71.56%	72.61%	71.22%	71.91%

compare to LR(81.15%, 76.33%, 81.19% and 78.69%), RF(72.43%,74.35%,78.69% and 76.46%), DT(71.56%, 72.61%,71.22% and71.91%). It is observed that the deep learning algorithms performs better than machine learning algorithms and the proposed deep learning based ensemble model surpass all machine/deep learning algorithms.

CHAPTER 5

Conclusions and Future Work

In this chapter, we have presented the conclusion and future work that acquired from our proposed work.

5.1 Conclusion

In this research, we extract and reuse the stack overflow dataset to automatically predict the best answer in developer forums. We implement preprocessing by using natural language processing techniques to overcome the noise from data. Afterward, we extract the three main features: m-features, k-features and t-features from textual description. Metadata (m-features) are computationally cheap and extracted from answers text before preprocessing. We leveraged keyword extraction and ranking tool (Text-Rank) after preprocessing to extract k-features. Subsequently, word embedding is used to transform textual description (t-features) of answers into feature vectors. Finally, we proposed an integrated model consist of three deep learning models which takes the input vectors of metadata, keywords and text separately for training and prediction.

5.2 Future Work

The broader impact of our work is to show that the combination of Q/A could be a rich source to predict the best answer. Our results encourage future research on best answer prediction. An important future direction is to implement different keywords ranking algorithms for the extraction of keywords from answers description and monitor the impact of these ranking algorithms on the performance of best answer prediction. Also one can use more tags from stack overflow dataset instead of only five. This enhancement can enable the proposed model to assess the quality of answers related to many programming languages. Another important issue is to implement sentiment analysis on answers to identify users emotions while writing answers on developer forums.

5.3 Threads

In this section, we will discuss some of the issues linked to the proposed work. These include construction, internal and external threats to validity of the research that may influence the performance of the proposed approach. We are using precision, recall and f-measure that are commonly used metrics validate the results in text classification[68]. In this study, threats to construct validity is the selection the metrics (Precision, Recall and F-measure) to evaluate the results of the proposed approach.

Furthermore, another construct to validity in this study is the utilization of text-rank algorithm to assign rank to each extracted keyword from answers text. We are using text-rank in proposed approach because it works better on short text and

non of the researcher have used this to assess the answers quality. The usage of different keyword ranking techniques may affect the performance of the proposed approach.

External validity involves the extent to which the results of a study can be generalized. We are using stack overflow dataset to train the machine/deep learning based models and to perform prediction. The data from other source given to classifier for evaluation may affect the performance of the proposed approach. Another external threat to validity is the optimization of hyperparameters for deep learning based proposed ensemble model. Any tuning and adjustment in setting may affect the performance of the proposed approach.

References

[1] Yahoo. https://answers.yahoo.com. (June), 2005.

[2] Joel Spolsky Jeff Atwood. https://stackoverflow.com. (September), 2008.

[3] Fabio Calefato, Filippo Lanubile, Maria Concetta Marasciulo, and Nicole Novielli. Mining successful answers in stack overflow. In *Proceedings of the 12th Working Conference on Mining Software Repositories*, pages 430–433. IEEE Press, 2015.

[4] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 2857–2866. ACM, 2011.

[5] Dilip Kumar Roy and Bithin Datta. Multivariate adaptive regression spline ensembles for management of multilayered coastal aquifers. *Journal of Hydrologic Engineering*, 22(9):04017031, 2017.

[6] Changbin Nie, Leyong Yu, Xingzhan Wei, Jun Shen, Wenqiang Lu, Weimin Chen, Shuanglong Feng, and Haofei Shi. Ultrafast growth of large-area monolayer mos2 film via gold foil assistant cvd for a highly sensitive photodetector. *Nanotechnology*, 28(27):275203, 2017.

[7] Lada A Adamic, Jun Zhang, Eytan Bakshy, and Mark S Ackerman. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th international conference on World Wide Web*, pages 665–674. ACM, 2008.

[8] Luca Ponzanelli, Andrea Mocci, Alberto Bacchelli, Michele Lanza, and David Fullerton. Improving low quality stack overflow post detection. In *2014 IEEE International Conference on Software Maintenance and Evolution*, pages 541–544. IEEE, 2014.

[9] Xin Xia, David Lo, Denzil Correa, Ashish Sureka, and Emad Shihab. It takes two to tango: Deleted stack overflow question prediction with text and meta features. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 73–82. IEEE, 2016.

[10] Christoph Treude, Ohad Barzilay, and Margaret-Anne Storey. How do programmers ask and answer questions on the web?: Nier track. In *2011 33rd International Conference on Software Engineering (ICSE)*, pages 804–807. IEEE, 2011.

[11] Kartik Bajaj, Karthik Pattabiraman, and Ali Mesbah. Mining questions asked by web developers. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 112–121. ACM, 2014.

[12] Bowen Xu, Zhenchang Xing, Xin Xia, David Lo, Qingye Wang, and Shanping Li. Domain-specific cross-language relevant question retrieval. In *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, pages 413–424. IEEE, 2016.

[13] Fabio Calefato, Filippo Lanubile, and Nicole Novielli. How to ask for techni-

cal help? evidence-based guidelines for writing questions on stack overflow. *Information and Software Technology*, 94:186–207, 2018.

[14] Pradeep Kumar Roy, Zishan Ahmad, Jyoti Prakash Singh, Mohammad Abdallah Ali Alryalat, Nripendra P Rana, and Yogesh K Dwivedi. Finding and ranking high-quality answers in community question answering sites. *Global Journal of Flexible Systems Management*, 19(1):53–68, 2018.

[15] Dana Movshovitz-Attias, Yair Movshovitz-Attias, Peter Steenkiste, and Christos Faloutsos. Analysis of the reputation system and user contributions on a question answering website: Stackoverflow. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pages 886–893, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2240-9. doi: 10.1145/2492517.2500242. URL http://doi.acm.org/10.1145/2492517.2500242.

[16] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. Discovering value from community activity on focused question answering sites: A case study of stack overflow. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 850–858, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1462-6. doi: 10.1145/2339530.2339665. URL http://doi.acm.org/10.1145/2339530.2339665.

[17] Amiangshu Bosu, Christopher S. Corley, Dustin Heaton, Debarshi Chatterji, Jeffrey C. Carver, and Nicholas A. Kraft. Building reputation in stackoverflow: An empirical investigation. *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 89–92, 2013.

[18] Amiangshu Bosu, Christopher S Corley, Dustin Heaton, Debarshi Chatterji, Jeffrey C Carver, and Nicholas A Kraft. Building reputation in stackoverflow: an empirical investigation. In *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 89–92. IEEE, 2013.

[19] Wenhao Zheng and Ming Li. The best answer prediction by exploiting heterogeneous data on software development q&a forum. *Neurocomputing*, 269: 212–219, 2017.

[20] Kerry Hart and Anita Sarma. Perceptions of answer quality in an online technical question and answer forum. In *Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 103–106. ACM, 2014.

[21] Qiongjie Tian, Peng Zhang, and Baoxin Li. Towards predicting the best answers in community-based question-answering services. In *Seventh International AAAI Conference on Weblogs and Social Media*, 2013.

[22] George Gkotsis, Maria Liakata, Carlos Pedrinaci, Karen Stepanyan, John Domingue, et al. Acqua: automated community-based question answering through the discretisation of shallow linguistic features. *The Journal of Web Science*, 1(1):1–15, 2015.

[23] Chirag Shah and Jefferey Pomerantz. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418. ACM, 2010.

[24] Chirag Shah. Building a parsimonious model for identifying best answers us-

ing interaction history in community q&a. In *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*, page 51. American Society for Information Science, 2015.

[25] Fabio Calefato, Filippo Lanubile, and Nicole Novielli. Moving to stack overflow: Best-answer prediction in legacy developer forums. In *Proceedings of the 10th ACM/IEEE international symposium on empirical software engineering and measurement*, page 13. ACM, 2016.

[26] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436, 2015.

[28] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[29] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[31] John J Hopfield. Neural networks and physical systems with emergent collec-

tive computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[33] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[34] Tony Lindeberg. Scale invariant feature transform. 2012.

[35] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized ransac. In *Joint Pattern Recognition Symposium*, pages 236–243. Springer, 2003.

[36] Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *international conference on machine learning*, pages 37–45, 2013.

[37] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12):2037–2041, 2006.

[38] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. 2005.

[39] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London.*

Series A: Mathematical, Physical and Engineering Sciences, 454(1971):903–995, 1998.

[40] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[41] Lijuan Cai and Thomas Hofmann. Text categorization by boosting automatically extracted concepts. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 182–189. ACM, 2003.

[42] Swapnil Hingmire, Sandeep Chougule, Girish K Palshikar, and Sutanu Chakraborti. Document classification by topic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 877–880. ACM, 2013.

[43] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[44] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3 (Feb):1137–1155, 2003.

[45] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[46] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on em-*

pirical methods in natural language processing (EMNLP), pages 1532–1543, 2014.

[47] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[48] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010.

[49] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[50] Sifatullah Siddiqi and Aditi Sharan. Keyword and keyphrase extraction techniques: a literature review. *International Journal of Computer Applications*, 109(2), 2015.

[51] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ, 2003.

[52] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. A text feature based automatic keyword extraction method for single documents. In *European Conference on Information Retrieval*, pages 684–691. Springer, 2018.

[53] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic key-

word extraction from individual documents. *Text mining: applications and theory*, 1:1–20, 2010.

[54] Samhaa R El-Beltagy. Kp-miner: a simple system for effective keyphrase extraction. In *2006 Innovations in Information Technology*, pages 1–5. IEEE, 2006.

[55] Olena Medelyan, Vye Perrone, and Ian H Witten. Subject metadata support powered by maui. In *JCDL'10*, pages 407–408. ACM, 2010.

[56] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. Kea: Practical automated keyphrase extraction. In *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, pages 129–152. IGI Global, 2005.

[57] Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*, volume 8, pages 855–860, 2008.

[58] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 366–376. Association for Computational Linguistics, 2010.

[59] Corina Florescu and Cornelia Caragea. Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, 2017.

[60] Adrien Bougouin, Florian Boudin, and Béatrice Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. 2013.

[61] Florian Boudin. Unsupervised keyphrase extraction with multipartite graphs. *arXiv preprint arXiv:1803.08721*, 2018.

[62] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

[63] Keras Flatten Layer. https://github.com/kerasteam/keras/blob/master/keras/layers/core.py. (Nov), 2019.

[64] Keras Merge Layer. https://github.com/kerasteam/keras/blob/master/keras/layers/merge.py. (Nov), 2019.

[65] Annamaria Di Fabio and Marc A Rosen. Opening the black box of psychological processes in the science of sustainable development: A new frontier. *European Journal of Sustainable Development Research*, 2(4):47, 2018.

[66] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[67] Saniat Javid Sohrawardi, Iftekhar Azam, and Shazzad Hosain. A comparative study of text classification algorithms on user submitted bug reports. In *Ninth International Conference on Digital Information Management (ICDIM 2014)*, pages 242–247. IEEE, 2014.

[68] George Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of machine learning research*, 3(Mar):1289–1305, 2003.