CHAPTER 3

3.1 Introduction

In this chapter, the methodology employed to create a Long Short-Term Memory (LSTM)-based disaster tweet classification system is detailed. This chapter outlines the steps taken, according to the data science project lifecycle, including data acquisition, preparation and cleaning, model building and training, and testing and validation. Research Methodology The methodology of the research is founded on the works in literature which are discussed in Chapters 1 and 2 and it applies the most advanced methods of machine learning and natural language processing (NLP) techniques. This work uses verified (True), and fabricated (Fake) news datasets, which provide an excellent ground for improving the classifiers through diverse textual data sets. As mentioned earlier, these datasets are sourced from Kaggle's "NLP Getting Started" natural language processing competition to classify disaster tweets. Competition details can be found at Kaggle: NLP Getting Started.

3.2 Research Framework

The flowchart provides a clear and concise outline of the different components that make up the overall predictive modeling process including: data preprocessing, feature extraction and engineering, model building, and evaluation. The process begins with problem formulation and continues to data collection, preprocessing and feature engineering. The data is split into training and testing datasets and resampling measure are being applied if necessary Now, classification models are trained and evaluated using precision, recall, F1-score, and accuracy metrics.

Image shows a flowchart of the research framework In Phase 1, problem formulation, research goals and objectives are outlined. Phase 2 consists of "Data Preparation and Preprocessing" such as: fake data collection, data cleaning, tokenization, stop word removal, lemmatization, and feature engineering using GloVe or Word2Vec embeddings. Phase 3 reflects "Model Design and Implementation", data modeling that uses LSTM-based architecture. Lastly, “Evaluation and Validation” evaluates the performance of the model and discusses the results — thus completing the research process. This systematic approach allows the objectives of the study to be effectively fulfilled.

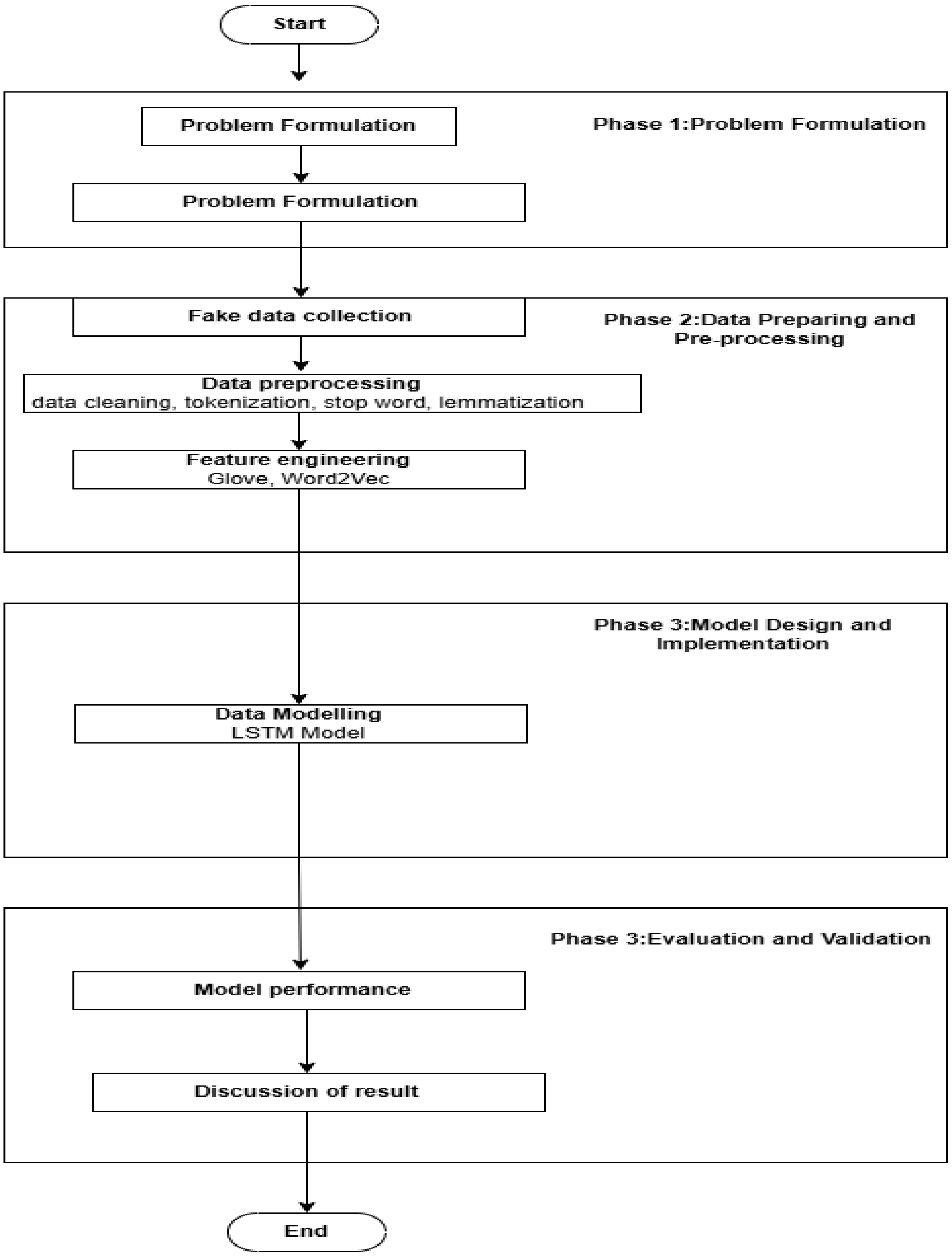

Start
Problem Formulation
Phase 1:Problem Formulation
Problem Formulation
Fake data collection
Phase 2:Data Preparing and Pre-processing
Data preprocessing
data cleaning, tokenization, stop word, lemmatization
Feature engineering
Glove, Word2Vec
Phase 3:Model Design and Implementation
Data Modelling
LSTM Model
Phase 3:Evaluation and Validation
Model performance
Discussion of result
End

Figure **Error! No text of specified style in document.**.1 Project framework

3.2.1 Phase 1: Problem Formulation

The tweets in social media space can be huge and diverse, making it one of the major challenged to identify those tweets connected to the disaster and hence it becomes mandatory to create an efficient model for distinguishing tweets related to the disaster and the irrelevant tweets. This itself needs thoroughness of linguistic arrangement and contextual implication, which are indicative of urgent and nature of disaster information.

Central to this research is framework development around a Long Short-Term Memory (LSTM) model, uniquely well-suited for capturing long-range dependencies in sequential text data. The proposed model makes use of the temporal nature of tweets in order to detect disaster messages in real time.

The research will also concentrate on testing the scalability and generalizability of the LSTM model across diverse datasets and environments to improve the model's performance even further. This was important, as it enabled the model to work cost-effectively under different situations — particularly during disaster events when immediate data is essential.

This method strives to bypass the shortcomings of normal machine learning methods like Naïve Bayes and Support Vector Machine (SVM) through the inclusion of NLP techniques. Advancements in AI will enable a better understanding of the content, including subtle linguistic differences, sentiment and context implications that would otherwise go unnoticed.

Our work outlines a modular, LSTM-based solution for the automation of disaster tweet classification, which will aid researchers in expanding the features of their classification solutions while being fast, tolerant, scalable and one that outperforms features in traditional solutions within a similar domain of interest.

3.2.2 Phase 2: Data collection and preprocessing

3.2.2.1 Data Sources

In this research, two important datasets are used to train and evaluate the model:

True Dataset: This dataset contains 21,417 confirmed news stories from 2017 onwards, sorted into topics like politics, sports, and international affairs. The primary output is the data which includes enriched metadata like titles and publication dates thereby giving a complete insight about real-world news events.

Fake Dataset — 23,481 made up news posts, its structure is like true dataset but the content is like a real article with false contents. The fake articles are also organized by categories and come with title and publication date, allowing for a realistic challenge for classification.

Both of these datasets are taken from is Kaggle (https://www.kaggle.com/competitions/nlp-getting-started) "NLP Getting Started" competition which offers a great and varied set of both real and fake news articles. These are the datasets which are used to train the model as well as measure performance of the model. Kaggle: NLP Getting Started. Tags: competitions, data.

3.2.2.2 Data Preprocessing Process

This makes sure that the datasets are clean and consistent for subsequent use in the LSTM model, which requires significant preprocessing to make it suitable for input. The subsequent major preprocessing steps were introduced:

3.2.2.2.1 Text Cleaning

Text Cleaning is the first component of the preprocessing pipeline This is a process that involves cleaning up the text by getting rid of unnecessary features that would get in the way of running the processing. Key actions include:

Stripping HTML tags that might exist in web-scraped data.

Removing special characters and symbols, such as punctuation, that are not useful for classification.

Removing hyperlinks so that only news article copy passes through for subsequent processing.

This text cleaning step ensures that text is as simple as possible and stripped of elements which could introduce noise into the model's analysis.

3.2.2.2.2 Tokenization

Tokenization — the process of breaking the text into smaller parts called tokens (typically words or phrases). This allows the model to encode the text more systematically. The quick brown fox for example would be tokenized by the model into The, quick, brown, fox. It is a critical step since tokenization allows the model to learn a model of the input text and understand its structure.

3.2.2.2.3 Stop Word Removal

Stop words are words like "and," "the," and "is" that are common and do not carry meaning and can be discarded from the text safely. And some of these words frequently occur in datasets, thus would not be meaningful and would introduce noise in the model decreasing its performances. Stop words contribute little to no meaning when combined with other words, and removing them helps the model hone in on more impactful content.

3.2.2.2.4 Lemmatization

Lemmatization is the process of converting a word to its base or dictionary form. For instance, run, running and ran are all simplified to run. This makes the text more uniform and in turn makes it so that the model treats different instances of a word similar to the same word, although they may be presented in a different form and therefore have lower consistency of data as well as needing to learn how to group similar words.

3.2.2.2.5 Feature Engineering

Feature engineering is the act of converting text data into numerical formats so that a machine learning model can understand it. In this study, GloVe embeddings are used for doing so. GloVe: Global Vectors for Word Representation GloVe (Global Vectors for Word Representation) is designed to produce dense vector representations for words based on their semantic contextual similarities with other words. By enabling the model to grasp contextual meaning and thGet deeper semantic information embedded in the text, this method is essential for accurate classification.

The above preprocessing ensures each sequence is properly formatted and has its features ready for input into the LSTM model, which helps improve performance and generalization across unseen data.

3.2.3 Phase 3: Model design and analysis

Model Designing and Developing: This stage plays a vital role in the overall research. This leads to using Long Short-Term Memory (LSTM) networks to identify the disaster − related tweets. Here we describe the method employed to construct this model, as well as the processes used to refine it; such ensuring that our model is capable of bridging the subtleties required for sequential text data.

3.2.3.1 LSTM Architecture

An LSTM-based model for sequential text involves several crucial layers that enable the model to learn and retain information over time, making it particularly adept at handling tasks such as text classification or sentiment analysis where contextual understanding is paramount. The architecture is composed of several components:

3.2.3.1.1 Embedding Layer

At the top level, the model starts with an embedding layer that takes raw text input and embeds this text into dense, continuous word vectors with the help of already trained GloVe embeddings. By capturing the semantic relationships between words, GloVe (Global Vectors for Word Representation) enables the model to comprehend the contextual meaning of a particular word in relation to the words around it. Enabling processing of numerical data rather than raw text is a critical component of the transformation before the model uses the data for machine learning tasks.

3.2.3.1.2 LSTM Layers

After the embedding layer, the model adds two stacked LSTM layers with 128 units each. They are particularly good at capturing the long-term dependencies and contextual information in sequential data. The model better understands the sequence of words in the input data using two stacked LSTM layers to learn temporal patterns that are critical for detection. Each layer allows the model to remember important context over a longer period, improving tweet classification by their history/context.

3.2.3.2 Training Process

After the architecture is decided, a set of hyperparameters and optimization techniques are used to train the model to enable it to learn. Training consists of multiple steps, including:

3.2.3.2.1 Loss Function

The binary cross-entropy loss function is used for binary classification tasks. This loss function is particularly useful when we have models that classify our data into two categories—in this case, relevant (disaster-related) vs irrelevant (not disaster-related) tweets. The loss function guides the model during training to minimize the error by measuring the difference between the predicted probabilities and the actual binary labels.

3.2.3.2.2 Optimizer

[48]The Adam optimizer was selected to use while training the model, based on its ability to efficiently handle larger dataset. Adam Optimizer: It captures the beneficial features from both AdaGrad and RMSProp optimizers by individually adapting the learning rate for each parameter. Because of this, it's especially useful for training deep learning models on complex tasks, like text classification where the dataset might have huge amount of data with complex patterns.

These together make sure the model is accurate while also making train runs efficient. Grid search techniques are used to fine-tune hyperparameters including the number of LSTM units, dropout rates, and learning rates de maximize performance.

3.2.4 Phase 4: Result Validation

Now, during this step, we overview different metrics for evaluating the performance of our LSTM model in accurately classifying disaster-related tweets. This helps to understand its business and its target market better to further improve the model.

As the model is trained, the evaluation is performed on this testing dataset to be used as the validation set to keep the evaluation unbiased and reflective of the model´s ability to generalize to new and unseen data. This gives a realistic insight into the model's performance as it can be objectively evaluated this way.

Evaluation Metrics

We evaluate the performance of LSTM model by the following evaluation metrics:

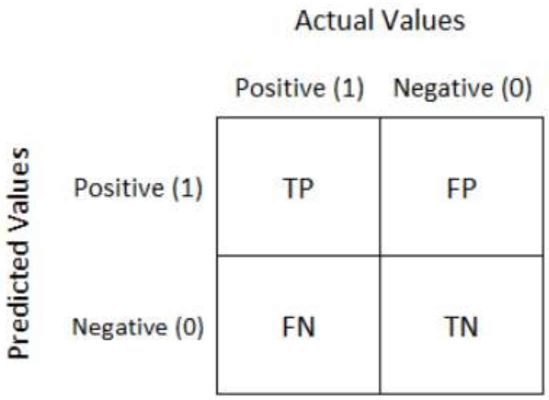
Accuracy: The overall percentage of correctly classified tweets, providing a generic measure of the model effectiveness.

Precision: Precision calculates the ratio of true positive detections to all the tweets predicted as relevant. This metric is important for assessing the ability of the model to capture relevant disaster-related tweets without mistakenly classifying irrelevant ones.

Recall: Recall measures the capability of the model to identify by itself all relevant instances of disaster-related tweets. So while the model finds some irrelevant tweets, a high recall means that we still capture most of the relevant data.

F1-Score: F1-score is the harmonic mean of precision and recall which gives a balanced score. This metric is particularly helpful when working with imbalanced datasets, where one class (the relevant tweets) may naturally have less representation than the other.

ROC-AUC: The Receiver Operating Characteristic (ROC) curve and the corresponding Area Under the Curve (AUC) allow a more detailed evaluation of the model's performance at differentiating relevant from irrelevant tweets. The higher the AUC the better the model.



We can use confusion matrices and ROC curves to visualize and analyze the performance of the model. It contains a confusion matrix (Fig. 3.11), which allows you to see numbers of true positives, true negatives, false positives, and false negatives, making it easier to investigate how your model is making the classifications it produces.

Mathematical Formulation of Evaluation Metrics

The evaluation metrics are computed based on the following formulas:

Accuracy:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad \textbf{(3.1)}$$

Where:

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

Precision:

$$\boldsymbol{precission = \frac{TP}{TP + FP}} \qquad \textbf{(3.2)}$$

$$\boldsymbol{recall = \frac{TP}{TP + FN}} \qquad \textbf{(3.3)}$$

$$\boldsymbol{F1\ score = \frac{2}{\frac{1}{precission} + \frac{1}{recall}}} \qquad \textbf{(3.4)}$$

Thematic Extraction and Profiling

This phase goes beyond binary classification and emphasizes feature extraction, where the model learns to capture further attributes of the tweets. This means detecting the class how are tweets relevant (e.g. natural disasters, earthquake) and extracting metadata like where and when and what are the relationship between tweets. The attribute analysis can help better profile disaster-related tweets and provide better insight into their nature, thus helping disaster response and management.

This phase will then validate the model against the comprehensive evaluation metrics discussed above, while also allowing to identify the nuanced attributes within the data that could potentially lead to a better decision.

3.4 Summary

This chapter provides a detailed and systematic approach to the design and implementation of a disaster tweet classification system based on Long Short-Term Memory (LSTM) networks. This methodology applies a disciplined data science lifecycle of data collection, preprocessing, model design, and evaluation to address disaster-related tweet classification challenges.

It combines with state-of-the-art techniques such as tokenization, stop-word removal, lemmatization, and feature engineering together so that the data is ready to train a LSTM model. We

Use GloVe and word embeddings to enhance the textual features, thus adding semantic information and helping the model better understand the relationships between words in context. Next, a bidirectional long short term memory (LSTM) network is used to extract long-term interdependencies from the text, allowing it to correctly determine whether each tweet concerns a disaster or not. The evaluation metrics used come from a diverse set of aspects including accuracy, precision, recall, F1-score, and ROC-AUC to effectively test the models to be robust and accurate enough to handle real-world applications.

Another dataset used in their work is a wide range of verified and fake news articles found at Kaggle's "NLP Getting Started" competition. This dataset has a great deal of overlap of text, which further supports the model training and the ability to generalize across context, allowing the model to apply not just to disaster tweets but to any text.

This development presents us with both challenges and opportunities — it allows us to develop more sophisticated tools for analysis, but we must remain aware of ethical considerations and ensure that issues around privacy and data protection are respected throughout the research process. The research endeavors to create a fair and interpretable model for classification by ensuring fairness and transparency.

The next chapter will quote whatever is the data and results of implementing the methodology; however, we will discuss the performance of the model in detail with extensive testing and the analysis of the results is framed and prepares the ground for disaster response systems and beyond.