CIT. Journal of Computing and Information Technology, Vol. 32, No. 2, June 2024, 65–80
doi: 10.20532/cit.2024.1005803

E-Commerce Fake Reviews Detection Using LSTM with Word2Vec Embedding

Mafas Raheem and Yi Chien Chong

School of Computing, Asia Pacific University of Technology and Innovation, Kuala Lumpur, Malaysia

Customer reviews inform potential buyers' decisions, but fake reviews in e-commerce can skew perceptions as customers may feel pressured to leave positive feedback. Detecting fake reviews in e-commerce platforms is a critical challenge, impacting online shopping and deceiving customers. Effective detection strategies, employing deep learning architectures and word embeddings, are essential to combat this issue. Specifically, the study presented in this paper employed a 1-layer Simple LSTM model, a 1D Convolutional model, and a combined CNN+LSTM model. These models were trained using different pre-trained word embeddings including Word2Vec, GloVe, FastText, and Keras embeddings, to convert the text data into vector form. The models were evaluated based on accuracy and F1-score to provide a comprehensive measure of their performance. The results indicated that the Simple LSTM model with Word2Vec embeddings achieved an accuracy of nearly 91% and an F1-score of 0.9024, outperforming all other model-embedding combinations. The 1D convolutional model performed best without any embeddings, suggesting its ability to extract meaningful features from the raw text. The transformer-based models, BERT and DistilBERT, showed progressive learning but struggled with generalization, indicating the need for strategies such as early stopping, dropout, or regularization to prevent overfitting. Notably, the DistilBERT model consistently outperformed the LSTM model, achieving optimal performance with an accuracy of 96% and an F1-score of 0.9639 using a batch size of 32 and a learning rate of 4.00E-05.

ACM CCS (2012) Classification: Computing methodologies → Artificial intelligence → Natural language processing

Keywords: fake detection, word embeddings, deep-learning model, transformer-based model

1. Introduction

In today's fast-paced tech-driven world, e-commerce stands out as a booming market. A 2022 study by Morgan Stanley predicts substantial growth, with the market expected to soar from $3.3 trillion in 2022 to $5.4 trillion by 2026, showing a 13.1% compound annual growth rate. This growth is accompanied by an increasing number of online shoppers and products. To build trust, companies have implemented review systems on their platforms, enhancing the overall shopping experience. Reviews allow customers to provide feedback based on their personal experiences with products or services, which inform the purchasing decisions of other potential buyers. However, the prevalence of fake reviews poses a significant issue in the e-commerce sector, as customers may be incentivized to post positive reviews to support the growth of businesses [1].

In fake review classification, discriminative machine learning methods are widely used to detect and remove deceptive content. These methods typically involve feature engineering, used to extract specific features from the text. Subsequently, classification algorithms, such as Support Vector Machines (SVM), Logistic Regression (LR), Naïve Bayes (NB), and Decision Trees (DT), were applied [2]. Although traditional ML methods can be effective under specific circumstances, they may encounter challenges when dealing with intricate patterns, extensive datasets, and high-dimensional feature spaces, potentially leading to less-than-optimal classification outcomes.

Conversely, deep learning (DL) approaches exhibited outstanding results in numerous natural language processing tasks, encompassing fake classification [3]. A key benefit of DL techniques is their capacity to manage intricate tasks and high-dimensional data more efficiently, which leads to improved classification accuracy and generalization [4]. Transfer learning is another approach that has gained prominence due to its potential to enhance the performance of DL models, mainly when dealing with a limited training set [5]. Transfer learning, utilizing pre-trained algorithms and refining them for specific tasks, allows models to benefit from prior knowledge, cutting down on training time and enhancing classification accuracy. This adaptability makes transfer learning an attractive option for fake reviews classification tasks, where the ever-evolving nature of deceptive content and the constant emergence of new techniques pose ongoing challenges to traditional ML and even DL methods [6].

This paper aims to investigate the impacts of various pre-trained embedding models on the performance of machine learning algorithms designed for the classification of fake reviews.

2. Literature Review

Researchers and e-commerce platforms have progressively developed a range of detection methodologies to address the negative consequences of fake reviews. Initial approaches emphasized using supervised ML techniques, such as NB, SVM, and DT on online reviews [7]. The advancement of research in this area has led to the exploration of DL techniques, which enable automatic learning of features and pattern characteristics of fake reviews [8]. Subsequently, transfer learning methods have refined these techniques, enhancing their performance and adaptability in detecting fake reviews across various domains and platforms [6].

2.1. Feature Engineering for Fake Classification

Feature engineering is vital in NLP, as it entails extracting significant features from text data, enhancing the results of ML models. Some of the feature engineering techniques in NLP include Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), n-gram model, Part-of-Speech (POS) Tagging, and word embeddings [9].

BoW is a simple yet potent method for feature extraction from text. It generates a vector of word frequencies for each document, with each element representing a unique term in the corpus. Despite its simplicity and computational efficiency, BoW suffers from several limitations, such as the inability to capture word order and semantic relationships between words [10]. Moreover, [11] suggested using Distributed Memory (DM) and Distributed Bag of Words (DBOW) approaches to learn fixed-length numerical vectors for each email before fake email classification. The researchers contended that the combination of DM and DBOW models effectively captures both word order and the nuanced meaning and interpretation of the text.

TF-IDF enhances the BoW model by allocating weights to words in a document, considering their frequency and prevalence across the entire corpus. Terms of higher importance and specificity to a document receive greater weights, while common words receive lower weights. In [12], the authors proposed two strategies for multi-class fake email classification. First, for optimal results in terms of micro F1-score, the researcher recommended combining TF-IDF with SVM, achieving a score of 95.39%. Second, using TD-IDF with NB for the most time-effective fake email classification, enabling email analysis in 2.13 ms.

N-grams refer to contiguous sequences of n-words within a given text. Utilizing n-grams as features can aid in capturing word patterns and dependencies that might signify the text's underlying meaning or structure. However, the primary drawback of n-grams is the resulting feature vectors' high dimensionality and sparsity. In [13], the N-gram method is applied to enhance the performance of the NB classifier for detecting unsolicited emails in Indonesian. The optimal solution was the 5-gram method, yielding the best accuracy score of 94% and an F-score of 94.2%. In [14] the authors employed word n-gram, character n-gram, and synthesis of variable-length n-grams for feature engineering in fake email classification. The researchers concluded that SVM with the synthesis variable-length n-grams performed the best. The accuracy score of the model was 97.6%, while the F1 score was 94.9%.

POS tagging assigns grammatical labels (e.g., nouns, verbs, adjectives) to word content within a text. POS tags and other syntactic features, such as parse trees and dependency relations, can offer valuable insights into a text's structure and grammatical properties. Paper [15] proposed a theoretical framework for identifying fake entries on social media platforms, which integrated a combination of methods for selecting essential features from the text, such as POS tagging, TF-IDF, and Information Gain. These features were used alongside traditional ML classifiers for training. The study indicated that hybrid features led to more accurate classification than individual features. The authors of [16] suggested a hierarchical approach to classifying fake emails, initially dividing email contents into different divisions using POS tagging based on voices and tenses. The study categorized tenses into three groups: past, present, and future, and identified two types of voices: active and passive.

Word2Vec and GloVe are common word embedding techniques widely used in NLP tasks. They represent words as dense vectors that encode the meaning and structure of text information. Besides, Word2Vec and GloVe are pre-trained word embeddings that can serve as input for DL models. Word embeddings enable DL models to learn higher-level features and patterns from text. The research presented in [17] utilizes word embeddings to encode tweets as an alternative to feature engineering. This approach provides the benefit of a quicker and more straightforward implementation of bot detection systems. Finally, the results presented in [18] suggest incorporating GloVe word embedding with CNN and a CNN-LSTM for fake SMS classification. The study attained satisfactory accuracy rates of 93.88% and 92.34% for CNN GloVe and CNN-LSTM GloVe, respectively.

2.2. Deep Learning Techniques for Fake Classification

In their study, [19] utilized CNN and LSTM models to differentiate fake and non-fake instances of short text messages. These models were designed to work with text data and extracted features autonomously. The researchers tested their approach on a benchmark dataset and attained an impressive accuracy rate of 99.44%. The researchers manually adjusted each parameter in the CNN model to identify the optimal parameter values for the proposed model. Initially, the researchers experimented with the model using a Stochastic Gradient Descent (SGD) optimizer. Next, the researchers used different optimization functions like Adam and RMSProp while keeping the remaining parameters at their default values. The Adam optimizer demonstrated superior performance, yielding the lowest loss during model training, and was therefore employed.

Subsequently, the researchers assessed model loss by varying feature map sizes (64, 128, and 256) and pooling window sizes (3, 4, and 5) to determine the optimal values for the feature map and pooling windows. The highest performance was obtained with an activation map of “128” and a pooling kernel size of “5”. Besides, the study examined the model with various batch sizes and found the optimal batch size was “100”. The researchers also evaluated the model’s performance concerning multiple dropout rates, with the best performance achieved at a dropout value of 0.3. Ultimately, the researchers found that the optimal performance was achieved when various n-gram kernels were used simultaneously in combination with other identified settings.

Furthermore, authors of [20] employed Bi-LSTM with CNN for fake email detection. Recurrent Neural Network (RNN) models are typically used to train sequential textual data, although they often require considerable time. Introducing a convolutional layer before the RNN layer significantly expedites the training process. Furthermore, the convolutional layer enables higher-level feature extraction. This layer uses filters to identify relationships between various sentences or paragraphs within a document.

The researchers developed a model to classify email messages by combining Word embedding, CNN, and Bi-LSTM blocks to analyze the emotional and sequential aspects of the text. They utilized Bi-LSTM due to its ability to consider preceding and subsequent sequences, which helps in understanding sentiment and sequence features better than basic LSTM. CNN was used to extract advanced features for Bi-LSTM, speeding up training. Word embedding was chosen for its ability to represent words numerically while preserving semantic relationships. After 12 training iterations, the model achieved an accuracy of approximately 98%-99% while reducing loss.

Paper [21] investigated the differences in performance between LSTM and GRU by considering two dimensions: the magnitude of the training dataset and the varying lengths of textual content (either long or short). The study utilizes official datasets provided by Yelp Inc. for the corpus. In the context of model training speed, the GRU model demonstrates a 29.29% increase in speed compared to the LSTM model when processing identical datasets. As for performance, the GRU model outperforms the LSTM model in cases of lengthy text and smaller datasets, while it lags in other scenarios. Considering the performance and computational power requirements, the performance-to-cost ratio for the GRU model surpasses that of the LSTM model, yielding improvements of 23.45% in accuracy and 26.95% in F1 score ratios.

Four datasets were used: long text/small dataset, long text/large dataset, short text/small dataset, and short text/large dataset. LSTM outperforms GRU, especially for long-text datasets. However, in this study, GRU shows a higher performance-cost ratio. It demonstrates 23.45% higher accuracy, 27.69% higher recall, and 26.95% higher F1 ratio compared to LSTM. This suggests that, in certain situations, the GRU model may provide a more efficient solution when considering the balance between performance and computational resource usage.

Research presented in [22] compared the results of four DL models in fake classification tasks, which consist of a Bidirectional LSTM model with 50-dimensional GLoVe embeddings, an LSTM model with 100-dimensional GLoVe embeddings, a hybrid model combining CNN, LSTM, Doc2Vec, and TF-IDF, and lastly, a BiLSTM model with an attention mechanism and 100-dimensional GLoVe embeddings. Based on the study, the BiLSTM model with an attention mechanism and 100-dimensional GLoVe embeddings achieved the best train accuracy of 99.18% and test accuracy of 90.25%. Research shows that Attention-Bidirectional LSTM effectively captures essential context in text using Bi-LSTM and Attention Networks. The Bi-LSTM operates at the word level, followed by the attention layer to extract crucial word embeddings for sentence comprehension. These word representations are combined to form sentence embeddings, which, in turn, create document embeddings for text classification.

Authors of [23] developed an approach for identifying fake reviews that combine bag-of-n-grams and parallel CNNs to extract insights from text data. The authors utilized the n-gram embedding layer employing compact kernel dimensions, harnessing local context with the same computational effort needed for training CNNs. The CNN model accepts n-gram representations as input and employs concurrent convolutional layers to derive more extensive feature embeddings from the textual content. Furthermore, the authors' fake review classification strategy incorporates both language-based textual attributes and non-textual elements regarding the behavior of reviewers.

In the presented research, three models were combined using ensemble techniques. The experiments showed that including local word sequences improves the model’s overall result. Additionally, the study indicated that employing a concurrent convolution layer extracts valuable information that can be combined in different ways to complement each other and enhance performance. Results show that the ensemble method attained a 92.42% accuracy score in the restaurant domain and a mean recall of 92.14%. On the other hand, the dataset in the hotel domain exhibited an accuracy rate of 91.66%, coupled with an average recall of 91.67%.

Research presented in [24] aimed to improve the classification of fake reviews by creating two DL models that consist of BoW, word context, and consumer emotions. These models learn document-level embeddings by employing n-grams, word embeddings, and lexicon-based emotion indicators. To establish the efficacy of the proposed detection algorithm, the authors conducted a comparative analysis of their classification performance against several contemporary methods for detecting fake reviews. The developed systems delivered impressive outcomes on all datasets, irrespective of sentiment polarity or product category.

The study highlights the importance of employing advanced high-dimensional models to improve upon existing methods for detecting fraudulent reviews. Through experiments on four datasets, the suggested models outperformed baseline approaches and advanced methods in accuracy, AUC, and F-score. The integrated models showed superior effectiveness, particularly with larger datasets containing combined polarity. Researchers utilized pre-trained word representations and diverse emotion representations to achieve optimal performance in this scenario.

The authors of [25] presented a Self-Attention-based CNN Bi-LSTM (ACB) model as an alternative to a shallow ML-based model for detecting fake reviews. The ACB model evaluates the significance of individual words in a sentence and detects potential fake cues in the document using an attention mechanism. The ACB model acquires sentence encoding through a CNN layer and retrieves higher-level n-gram characteristics. Subsequently, the sentence embeddings emerged with a Bi-LSTM as document attribute representations. Finally, fake reviews are detected by considering contextual information. The experimental results were evaluated and compared to other variants, demonstrating that the ACB model outperforms them regarding classification accuracy.

Figure 3 displays the suggested hierarchical neural network structure constructed using the TensorFlow framework and Keras APIs. The architecture encompasses a word embedding layer, a self-attention mechanism, and a dense layer. The input layer is designed to handle a sentence comprising 150 words, where each word is passed through the word embedding layer and transformed into a 300-dimensional feature representation. This resulted in a 150 by 300-word vector matrix representing semantic proximity and associations among words. The well-known Word2Vec model is utilized for word embedding in the study, as it maintains the syntactic and semantic connections between words.

2.3. Transfer Learning in NLP

Transfer learning has become popular in NLP tasks, such as fake detection, as it leverages pre-trained models to achieve better results on target tasks with limited training data. BERT and RoBERTa are two prominent models to utilize transfer learning techniques and successfully apply them to fake content detection. This section discusses the application of BERT and RoBERTa and explores the differences between various model variants.

Authors of [26] aimed to enhance the results of fake review classification by using the BERT model to extract word representations from review texts. First, a sentence is broken down into its words. Each word is then input into the BERT model to generate word representations. These representations are collected for all the words in the sentence. All the embeddings of the sentence are combined to create a more extensive embedding that represents the entire sentence. This process is applied to a hotel dataset. Once all the large embeddings of all reviews and their corresponding labels are obtained, they are fed into the classification models for training and testing, using an 80:20 training/testing data ratio. A total of six shallow ML models are being utilized, namely SVM, Random Forest, Bagging, k-Nearest Neighbour (KNN), AdaBoost, and NB classifier. Finally, a confusion matrix was used to evaluate the results. The study found that SVM classifiers generated the highest accuracy score of 87.81%, 7.6% higher than the benchmark model of 80.75%. Besides, the SVM classifiers attained an F1-score of 88%, compared to the benchmark model of 80%.

Besides, [27] proposed a fake detection method that leverages the BERT model for word embeddings and shallow ML algorithms to categorize emails as either genuine or fake. The email content was input into the BERT model to generate word embeddings. Consequently, four ML classification techniques were utilized to classify the embeddings into genuine or fake groups.

The researchers employed the four fake classifiers: SVM, LR, random forest, and KNN. Two publicly available datasets are used to examine the proposed model. Dataset 1, the Enron-Fake dataset, was initially issued by [28]. Dataset 2 is a ham and fake email published by [29]. The study found that the LR classifier performed best on the two datasets. The F1 score for the LR model in Dataset 1 was 0.9784, while the F1 score in Dataset 2 was 0.9592.

In [30] the authors conduct an in-depth analysis of fake review detection using the BERT language model. The researchers applied the BERT model to publicly available online reviews like hotel, restaurant, doctor, and Yelp reviews. Implementing the fine-tuned BERT model, the researchers generated a superior performance compared to existing detection models.

The BERT model has achieved an accuracy score of 91% on the hotel, restaurant, and doctor reviews. Besides, for the imbalanced Yelp restaurant reviews dataset, the model achieved a 73% accuracy score. The researcher found no significant distinction between Cased BERT and Uncased BERT regarding fake review detection in the study. The findings indicated that the BERT model does not depend significantly on the text's stylistic features.

Additionally, the researchers discovered that a fine-tuned BERT, which has adjusted weights in its layers, outperformed a non-fine-tuned version. Therefore, the researchers recommended fine-tuning BERT layers for domain-specific applications to achieve optimal fake review identification. The study also revealed that training for only 2 to 4 iterations is adequate, as the BERT model tends to overfit beyond that point. Besides, the researchers found that increasing the dropout rate has no impact on the overall performance. Thus, the researchers advised utilizing a maximum of 4 epochs and a dropout factor of 0.1 for the fake content detection task. The researchers concluded that superior performance could be achieved on both datasets by effectively fine-tuning the BERT model.

Paper [31] proposed using BERT, RoBERTa, ALBERT, and DistilBERT to classify fake reviews to address the constraints associated with shallow ML and DL models. Each model's effectiveness was assessed utilizing an accuracy score and weighted F1 score. The results revealed that the classifier built using RoBERTa surpassed the performance of the baseline model, demonstrating superior capability in detecting fake reviews among all the models assessed.

According to the researchers, when utilizing 50% of the dataset for training, BERT attained an accuracy rate of 67% and a weighted F1 score of 67%. The researchers have found that the performance is relatively lower than the benchmark studies because classifying ground-truth reviews from diverse domains is complex. Many researchers utilize filtered variations of Yelp datasets that focus on specific domains, such as restaurants in a particular city. In contrast, the models developed in this study are more generic as they are trained on data from multiple domains, which adds to their complexity and potential impact on performance.

Besides, the researcher found that the fine-tuned DistilBERT attained an accuracy score of 68% and a weighted F1-score of 68%, which outperforms BERT in detecting fake reviews. Among all the models developed for fake review detection, the model constructed by fine-tuning Roberta exhibited the best results, with an accuracy of 69% and an F1-score of 0.69. On the other hand, the ALBERT attained the lowest performance. The accuracy score of the ALBERT was 64% and the F1 score was 0.63.

Authors of [32] recently investigated detecting fake emails utilizing a fine-tuning approach on BERT. Specifically, the bert-base-case was employed in the study. The bert-base-case model was trained on English Wikipedia and BookCorpus, with 2.5 billion and 800 million words respectively. The performance of the BERT model was then compared to other models such as KNN, NB, and Bi-LSTM. The study used Bi-LSTM as the baseline model. Dataset 1 was the Fakebase dataset collected from the UCI machine learning repository [33], while Dataset 2 was the Fake filter dataset collected from Kaggle [34]. The study revealed that bert-base-case model achieved the best performance, with 98.67% as the accuracy score, and 98.66% as the F1 score.

3. Methodology

3.1. Data Understanding

The data collection for this research was executed utilizing the publicly accessible Yelp Reviews dataset. The dataset comprises 40,432 instances of review data, spanning across 10 distinct categories. It maintains class balance, with an equal division of 20,216 computer-generated ("CG") reviews and 20,216 original reviews ("OR"). However, due to restrictions in time and computational resources, only a subset of 10,100 observations is utilized in this study.

Table 1 offers an elucidative overview of these attributes, critical to the subsequent analysis. This includes information such as the product category, the user-assigned numerical rating, the classification of the review (whether it's computer-generated or original), and the textual content of the review itself. These attributes lay the foundation for the ensuing comprehensive examination and the creation of a predictive model tasked with discerning the features of various types of reviews.

3.2. Data Preparation

Data preprocessing is an indispensable phase where data is carefully tailored to ensure coherence with the criteria set forth by the study. This phase involves data exploration, cleaning, transformation, and reduction.

The initial analysis of the accumulated data was executed to identify variables that would be most suitable for fake review classification. Certain variables, such as customer ratings and categories, are primarily employed for exploratory purposes, thereby enhancing comprehension of the dataset. In contrast, variables such as review text and labels are harnessed as primary inputs for fake review classification models.

3.2.1. Punctuation Removal

Punctuation removal is a critical pre-processing step in the NLP pipeline. Punctuation marks in text data such as exclamation points, commas, and periods are irrelevant for deep learning models. By removing these punctuation marks, customers' review text data is simplified into a format that is more conducive to performing fake review classification tasks with deep learning models.

3.2.2 Stopwords Removal

Stopwords are common words such as "the", "is", and "and", which while integral to human language, do not contribute significantly to the semantic value of a text from a machine learning perspective. Their high frequency of occurrence can skew the distribution of words and potentially distract the model from more contextually meaningful words. By removing these stopwords, we can focus our machine-learning algorithms on words that are more likely to be indicative of sentiments or classification categories. Thus, stopword removal enhances the effectiveness of our models in tasks such as sentiment analysis and fake review detection by reducing noise and emphasizing content-carrying words.

Table 1. Metadata.

No.	Attribute	Description
1	Category	Represents the category of the product which the review is about.
2	Rating	Represents the numerical rating given by the user in their review.
3	Label	Represents the label assigned to the review. 'CG' represents computer-generated reviews, while 'OR' represents original reviews.
4	Review	Represents the text of the review written by the user. It contains the user's detailed opinion about the product

3.2.3. Lemmatization

This process involves reducing words to their base or dictionary form, also known as their lemma. For instance, the words "running", "runs", and "ran" are all lemmatized to the root word "run". The primary advantage of lemmatization lies in its ability to handle different grammatical forms of a word as a single entity, thereby reducing the complexity of the text data. As a result, lemmatization aids in consolidating the feature space and enhancing the learning efficiency of our machine-learning models. This normalization process is instrumental in improving the performance of tasks such as sentiment analysis and fake review detection by streamlining the textual input.

3.2.4. Tokenization

Tokenization involves breaking down the text into smaller units, referred to as tokens, which are typically individual words or terms. Tokenization converts raw text data into a format that machine learning algorithms can analyze more effectively. By treating each word as a separate token, we allow our model to understand and learn the significance of individual words within the context of the text. This process enables the efficient extraction of features from the textual data, which is crucial for tasks such as sentiment analysis and fake review detection. Hence, tokenization aids in translating unstructured text into a structured format that facilitates more accurate and insightful computational text analysis.

3.3. Modelling

The role of pre-trained word embedding models in this study is pivotal, providing a way to transform raw text into meaningful, dense vector representations for further computational analysis. These models have been trained on extensive language data and offer the ability to transfer language understanding to specific tasks. The process entails converting words into numerical vectors, where the vector's magnitude and direction carry semantic meanings. This section outlines the three pre-trained word embedding models used in this study: Word2Vec, GloVe, and FastText.

3.3.1. Word2Vec

Word2Vec, developed by [35] at Google, is one of the pioneering techniques in the domain of word embeddings. It employs neural networks to learn word associations from a large corpus of text and then creates a vector for each unique word in the training corpus. These vectors capture contextual and semantic similarities among words; words that appear in similar contexts tend to have similar vectors. In this study, a Word2Vec model is trained on the review text, producing a 50-dimensional vector representation for each word, which will subsequently feed into the deep learning models.

3.3.2. Global Vectors for Word Representation (GloVe)

GloVe, developed by [36] at Stanford University, is another prominent word embedding technique. Unlike Word2Vec, which primarily learns from local linguistic context, GloVe builds word vectors by aggregating global word-word co-occurrence statistics from a corpus, and it is designed to capture both semantic and syntactic meanings. A pre-trained GloVe model is leveraged in this study to transform words in the review text into 50-dimensional vectors, which will be used for model training.

3.3.3. FastText

FastText, developed by Facebook's AI Research lab, enhances the Word2Vec model by taking into account subword information, making it highly effective for languages rich in morphology or in handling out-of-vocabulary words [37]. By expressing each word as an aggregation of character n-grams, FastText can produce improved word embeddings for rare and misspelt words. In this study, a FastText model is trained on the review text, yielding a 50-dimensional vector representation for each word. These word vectors are then utilized as inputs to the subsequent deep-learning models.

3.4. Implementation

In this study, a variety of pre-trained word embedding models were integrated with different deep-learning models. The goal was to discover the optimal combination of pre-trained embeddings and deep learning models. Additionally, transformer-based models such as BERT and DistilBERT are also deployed. For the BERT model, 'Bert-base-uncased' is used for both the tokenizer and the model, while for DistilBERT, 'distilbert-base-uncased' is used for both the tokenizer and the model. Following the evaluation of the results, the best-performing model was selected for hyperparameter tuning.

Table 2 illustrates the combinations of various deep learning architectures and pre-trained embeddings used in this study, while Table 3 provides an overview of the transformer models and their corresponding tokenizers.

Table 2. Deep Learning Architectures and Pretrained Embeddings Used.

No.	Model Type	Model	Pretrained Embeddings
1	Pretrained Word Embeddings + Deep Learning Models	LSTM	Learned embeddings (Keras)
2	Pretrained Word Embeddings + Deep Learning Models	LSTM	Word2Vec pre-trained embeddings
3	Pretrained Word Embeddings + Deep Learning Models	LSTM	GloVe pre-trained embeddings
4	Pretrained Word Embeddings + Deep Learning Models	LSTM	FastText pre-trained embeddings
5	Pretrained Word Embeddings + Deep Learning Models	CNN	Learned embeddings (Keras)
6	Pretrained Word Embeddings + Deep Learning Models	CNN	Word2Vec pre-trained embeddings
7	Pretrained Word Embeddings + Deep Learning Models	CNN	GloVe pre-trained embeddings
8	Pretrained Word Embeddings + Deep Learning Models	CNN	FastText pre-trained embeddings
9	Pretrained Word Embeddings + Deep Learning Models	Hybrid CNN+LSTM	Learned embeddings (Keras)
10	Pretrained Word Embeddings + Deep Learning Models	Hybrid CNN+LSTM	Word2Vec pre-trained embeddings
11	Pretrained Word Embeddings + Deep Learning Models	Hybrid CNN+LSTM	GloVe pre-trained embeddings
12	Pretrained Word Embeddings + Deep Learning Models	Hybrid CNN+LSTM	FastText pre-trained embeddings

Table 3. Transformer Models and Their Corresponding Tokenizers.

No.	Model Type	Model	Tokenizer and Model
1	Transformer-based Models	BERT	bert-base-uncased
2	Transformer-based Models	DistilBERT	distilbert-base-uncased

4. Experimental Results

The evaluation of the machine learning models utilized in this study was conducted based on two key metrics: accuracy and F1-score. The accuracy is a measure of the proportion of correct predictions made by the model, while the F1-score provides a balanced measure of precision (the proportion of true positive results among all positive predictions) and recall (the proportion of true positive results found in the total actual positive instances). The F1 score is particularly useful in situations where the data might be imbalanced. For each model, different types of embeddings were tested: Word2Vec, GloVe, FastText, and scenarios with Keras embeddings. These embeddings are vector representations of words, which capture semantic meanings and relationships between words. Word2Vec, GloVe, and FastText are popular pre-trained word embeddings used in natural language processing tasks.

4.1. Model Results for 1-Layer Simple LSTM with 10 Epochs

A 1-layer simple LSTM model was first trained for 10 epochs. As shown in Table 4, the results indicate that the use of Word2Vec embeddings achieved the highest accuracy and F1 score, with values of 0.894 and 0.891, respectively. This is followed by FastText embeddings (accuracy 0.888 and F1-score 0.890), GloVe embeddings (accuracy 0.862 and F1-score 0.868), and lastly, the scenario without any embeddings (accuracy 0.774 and F1-score 0.774).

4.2. Model Results for 1-Layer Simple LSTM with 20 Epochs

Increasing the training duration to 20 epochs for the 1-layer simple LSTM model led to an overall improvement in performance across all types of embeddings. The model with Word2Vec embeddings again outperformed the oth-

Table 4. Model Results for 1-Layer Simple LSTM (10 Epochs).

Model	Embeddings	Accuracy	F1-Score
1-Layer Simple LSTM (10 Epochs)	Keras	0.7738	0.7737
1-Layer Simple LSTM (10 Epochs)	Word2Vec	0.8941	0.8911
1-Layer Simple LSTM (10 Epochs)	GloVe	0.8619	0.8682
1-Layer Simple LSTM (10 Epochs)	FastText	0.8881	0.8902

Table 5. Model Results for 1-Layer Simple LSTM (20 Epochs).

Model	Embeddings	Accuracy	F1-Score
1-Layer Simple LSTM (20 Epochs)	Keras	0.8005	0.7797
1-Layer Simple LSTM (20 Epochs)	Word2Vec	0.9054	0.9024
1-Layer Simple LSTM (20 Epochs)	GloVe	0.8554	0.862
1-Layer Simple LSTM (20 Epochs)	FastText	0.8965	0.8951

ers, achieving an accuracy of 0.905 and an F1-score of 0.902. This was followed by FastText embeddings (accuracy 0.897 and F1-score 0.895), GloVe embeddings (accuracy 0.855 and F1-score 0.862), and the model without embeddings (accuracy 0.800 and F1-score 0.780) as shown in Table 5.

4.3. Model Results for 1D Convolutional Model with 20 Epochs and Early Stopping

The performance of a 1D Convolutional model trained for 20 epochs and early stopping was also evaluated. In this case, the model without embeddings achieved the highest accuracy (0.874) and F1 score (0.872). However, the performance of the model with Word2Vec embeddings was very close, with an accuracy of 0.870 and an F1-score of 0.864. The models with GloVe and FastText embeddings had slightly lower performance, as shown in Table 6.

4.4. Model Results for Hybrid CNN+LSTM Model with 20 Epochs

Lastly, a model combining CNN and LSTM was trained for 20 epochs with early stopping to avoid model overfitting. The performance was similar across all types of embeddings, with the model using Word2Vec embeddings achieving the highest accuracy (0.879) and the model without embeddings achieving the highest F1-score (0.879). The CNN+LSTM model using FastText embeddings outperformed the others, reaching an accuracy score of 0.8851 and an F1 score of 0.8838 as shown in Table 7.

Table 6. Model Results for 1D Convolutional Model (20 Epochs).

Model	Embeddings	Accuracy	F1-Score
1D Convolutional Model (20 Epochs)	Keras	0.8738	0.8717
1D Convolutional Model (20 Epochs)	Word2Vec	0.8703	0.8642
1D Convolutional Model (20 Epochs)	GloVe	0.8629	0.8657
1D Convolutional Model (20 Epochs)	FastText	0.8386	0.8191

Table 7. Model Results for CNN+LSTM (10 Epochs).

Model	Embeddings	Accuracy	F1-Score
CNN+LSTM (20 Epochs)	Keras	0.8777	0.8795
CNN+LSTM (20 Epochs)	Word2Vec	0.8792	0.8725
CNN+LSTM (20 Epochs)	GloVe	0.8738	0.8791
CNN+LSTM (20 Epochs)	FastText	0.8851	0.8838

4.5. Results for BERT Model

The BERT model was trained for 10 epochs. The training accuracy consistently increased from 0.9181 in the first epoch to a high of 0.9974 by the ninth epoch, indicating that the model was learning well from the training dataset. The validation accuracy fluctuated somewhat, peaking at 0.9525 in the third and sixth epochs, then slightly decreasing, ending at 0.9421 in the tenth epoch.

However, the validation loss, which is a measure of error on the validation dataset, demonstrated increased variability. It started low at 0.1360 in the first epoch, hit its lowest point at 0.1270 in the fourth epoch, and then consistently increased, ending at 0.3326 in the tenth epoch. This pattern suggests some overfitting might be occurring, as the model is performing better on the training data but increasingly worse on the unseen validation data.

4.5.1. Results for DistilBERT Model

The DistilBERT model was trained for 10 epochs. The training accuracy increased consistently from 0.9137 in the first epoch to a high of 0.9989 by the ninth epoch. This indicates that the model was progressively learning from the training dataset over time. However, in the tenth epoch, there was a slight decrease in accuracy to 0.9969. The validation accuracy also increased throughout the training process, with the maximum value of 0.9619 observed in the sixth epoch. After that, despite a few fluctuations, the accuracy stayed relatively high, indicating the model's good generalization performance on unseen data. The validation loss, which is a measure of error on the validation dataset, displayed more volatility. It started at 0.1231 in the first epoch, reached its lowest at 0.1173 in the eighth epoch, and then spiked significantly in the ninth epoch to 0.3227, ending at 0.2625 in the tenth epoch. This pattern could indicate overfitting in the latter stages of training, as the model performs well on the training data, but its performance worsens on the unseen validation data.

4.5.2. Results from Hyperparameter Tuning

For the LSTM model, four configurations with different numbers of layers (2 and 3) and dropout rates (0.1 and 0.2) were tested as shown in Table 8. The highest accuracy and F1-score (0.8950 and 0.8964, respectively) were achieved when the model was configured with three layers and a dropout rate of 0.1.

In the case of DistilBERT, four distinct sets of hyperparameters involving batch sizes (16 and 32) and learning rates (4.00E-05 and 5.00E-05) were tested as given in Table 9. The model performed the best with a batch size of 32 and a learning rate of 4.00E-05, achieving accuracy and an F1-score of 0.9639. It is worth noting that despite different configurations, the DistilBERT model consistently outperformed the LSTM model across all the configurations tested.

Table 8. Results from Hyperparameter Tuning the LSTM Model.

Model	Hyperparameter		Evaluation Metrics	
	Number of Layers	Dropout	Accuracy	F1-Score
1	2	0.1	0.8822	0.8858
2	2	0.2	0.8842	0.8879
3	3	0.1	0.8950	0.8964
4	3	0.2	0.8896	0.8915

Table 9. Results from Hyperparameter Tuning in DistilBERT Model.

Model	Hyperparameter		Evaluation Metrics	
	Batch Size	Learning Rate	Accuracy	F1-Scores
1	16	4.00E-05	0.9406	0.9405
2	16	5.00E-05	0.9559	0.9559
3	32	4.00E-05	0.9639	0.9639
4	32	5.00E-05	0.9569	0.9569

5. Discussions

5.1. Discussion on Results from Deep Learning Models

The 1-layer Simple LSTM model with Word2Vec embeddings outperformed all other model-embedding combinations in terms of both accuracy and F1-score when trained for 20 epochs. This suggests that Word2Vec embeddings may capture the semantic relationships between words in the e-commerce reviews more effectively than the other tested embeddings. The fact that the model performance improved from 10 epochs to 20 epochs for the Word2Vec embeddings indicates that further training can potentially enhance model performance.

Interestingly, the model with Keras embeddings also showed improvement from 10 to 20 epochs, although its performance was noticeably lower than the models utilizing word embeddings. This could imply that, while pre-trained embeddings offer a significant boost, the models are still capable of learning useful representations from raw data given enough training time. The FastText embeddings performed comparably to Word2Vec, particularly when used with the LSTM model. This similarity in performance could be due to FastText's ability to generate embeddings for out-of-vocabulary words, which can be advantageous when dealing with user-generated content like product reviews. The GloVe embeddings performed well with the LSTM and the CNN+LSTM models but were slightly outperformed by Word2Vec and FastText embeddings. This may be attributed to the different methodologies used to train these embeddings, with GloVe focusing on aggregating global word-word co-occurrence statistics from a corpus, and Word2Vec and FastText learning from local context windows.

The 1D convolutional model, while less performant than the LSTM-based models, still demonstrated respectable results. Its highest performance was achieved without any embeddings, which is an interesting observation that might suggest the model's ability to extract meaningful features from the raw text. The combined CNN+LSTM model achieved its best performance with FastText embeddings, showing the potential benefit of combining different model architectures and embeddings. This hybrid approach could capture both the local features (via CNN) and the long-term dependencies (via LSTM) in the text data.

5.2. Discussion on Results from Transformer-Based Models

In the study, two models, BERT and DistilBERT, were trained over 10 epochs. For BERT, training accuracy increased consistently while validation accuracy showed slight fluctuations. The validation loss hinted at overfitting. A similar pattern was observed for DistilBERT, with progressive learning evident in increasing training accuracy, and a high but fluctuating validation accuracy. The validation loss was initially decreasing but began to spike towards the end, again suggesting overfitting. Both models, while showing high aptitude in learning training data, struggled to generalize this learned information to unseen data, as revealed by increasing validation loss and fluctuating validation accuracy. This analysis underscores the importance of strategies like early stopping, dropout, or regularization to improve model performance by preventing overfitting.

5.3. Discussion on Results from Hyperparameter Tuning

In hyperparameter tuning for LSTM and DistilBERT models, different configurations were tested. For LSTM, the best results (0.8950 accuracy, 0.8964 F1-score) were attained with 3 layers and a 0.1 dropout rate. However, despite varying configurations, DistilBERT consistently outperformed LSTM. Its optimal performance (0.9639 accuracy and F1-score) was achieved with a batch size of 32 and a learning rate of 4.00E-05. These results affirm the superiority of DistilBERT in this context and suggest potential hyperparameters for future tuning and model development.

6. Conclusion

This research has conducted an extensive evaluation of various deep learning models and word embeddings for the detection of fake reviews in the e-commerce sector. The analysis has considered LSTM, 1D convolutional, and combined CNN+LSTM models, in conjunction with Word2Vec, GloVe, FastText, Keras, and transformer-based models like BERT and DistilBERT.

The 1-layer Simple LSTM model with Word2Vec embeddings emerged as a strong performer, achieving superior accuracy and F1-score results after 20 epochs of training. Word2Vec embeddings may capture the semantic relationships between words in the e-commerce reviews more effectively than other tested embeddings. Additionally, the study found that models are capable of learning useful representations from raw data given enough training time, as evidenced by the performance improvement of the model with Keras embeddings. FastText embeddings also performed comparably to Word2Vec, particularly when used with the LSTM model. This could be due to FastText's ability to generate embeddings for out-of-vocabulary words, a critical feature when dealing with user-generated content. In contrast, GloVe embeddings were slightly outperformed by Word2Vec and FastText, possibly due to their different training methodologies. Although the 1D convolutional model was less performant than the LSTM-based models, it still demonstrated decent results. Interestingly, its best performance was achieved without any embeddings. The combined CNN+LSTM model achieved its best performance with FastText embeddings, showcasing the potential benefit of hybrid model architectures.

As for transformer-based models, both BERT and DistilBERT showed progressive learning in increasing training accuracy but struggled with generalization, as indicated by fluctuating validation accuracy and increasing validation loss. This highlights the importance of strategies like early stopping, dropout, or regularization to prevent overfitting. In terms of hyperparameter tuning, the best results for the LSTM model were achieved with three layers and a 0.1 dropout rate. However, DistilBERT consistently outperformed LSTM, reaching its peak performance with a batch size of 32 and a learning rate of 4.00E-05.

References

[1] A. Prabakaran and M. Chen, "Product Review Credibility Analysis", in *Proceedings of the 2019 International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 11–15. https://doi.org/10.1109/ICCNC.2019.8685490

[2] R. Mansoor et al., "A Comprehensive Review on Email Spam Classification Using Machine Learning Algorithms", in *Proceedings of the 2021 International Conference on Information Networking (ICOIN)*, 2021, pp. 327–332. https://doi.org/10.1109/ICOIN50884.2021.9334020

[3] A. P. Rodrigues et al., "Real-time Twitter Spam Detection and Sentiment Analysis Using Machine Learning and Deep Learning Techniques", *Computational Intelligence and Neuroscience*, 2022. https://doi.org/10.1155/2022/5211949

[4] C. Janiesch et al., "Machine Learning and Deep Learning", *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021. https://doi.org/10.1007/s12525-021-00475-2

[5] F. Zhuang et al., "A Comprehensive Survey on Transfer Learning", *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020. https://doi.org/10.1109/JPROC.2020.3004555

[6] V. S. Tida, and S. Hsu, "Universal Spam Detection Using Transfer Learning of BERT Model", arXiv preprint, 2022. https://doi.org/10.48550/arXiv.2202.03480

[7] H. Khan et al., "Fake Review Classification Using Supervised Machine Learning In Pattern Recognition", *ICPR International Workshops and Challenges*, pp. 269–288, 2021. https://doi.org/10.1007/978-3-030-68799-1_19

[8] G. M. Shahariar et al., "Spam Review Detection Using Deep Learning", in *Proceedings of the 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2019, pp. 0027–0033. https://doi.org/10.1109/IEMCON.2019.8936148

[9] E. S. Gualberto et al., "The Answer is in the Text: Multi-Stage Methods for Phishing Detection Based on Feature Engineering", *IEEE Access*, vol. 8, pp. 223529–223547, 2020. https://doi.org/10.1109/ACCESS.2020.3043396

[10] S. Douzi et al., "Hybrid Email Spam Detection Model Using Artificial Intelligence", *International Journal of Machine Learning and Computing*, vol. 10, no. 2, pp. 316–322, 2020. https://doi.org/10.18178/ijmlc.2020.10.2.937

[11] M. Diale et al., "Unsupervised Feature Learning for Spam Email Filtering", *Computers & Electrical Engineering*, vol. 74, pp. 89–104, 2019.

[12] F. Jánez-Martino et al., "Classification of Spam Emails Through Hierarchical Clustering and Supervised Learning", arXiv preprint, 2020. https://doi.org/10.48550/arXiv.2005.08773

[13] Y. Vernanda et al., "Indonesian Language Email Spam Detection Using N-gram and Naïve Bayes Algorithm", *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 5, pp. 2012–2019, 2020. https://doi.org/10.11591/eei.v9i5.2444

[14] N. J. Euna et al., "Content-based Spam Email Detection Using N-gram Machine Learning Approach", *Preprints.org*, 2021. https://doi.org/10.20944/preprints202109.0236.v1

[15] A. Sharma and M. Ramaiya, "Social Media Spam Detection Using Different Text Feature Selection Technique", *ICTACT Journal on Soft Computing*, vol. 13, no. 1, pp. 2756–2764, 2022.

[16] S. K. Sonbhadra et al., "Email Classification Via Intention-based Segmentation", in *Proceedings of the 2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics (EECSI)*, pp. 38–44, 2020. https://doi.org/10.23919/EECSI50503.2020.9251306

[17] F. Wei and U. T. Nguyen, "Twitter Bot Detection Using Bidirectional Long Short-term Memory Neural Networks and Word Embeddings", in *Proceedings of the 1st IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications, TPS-ISA*, 2019, pp. 101–109. https://doi.org/10.1109/TPS-ISA48467.2019.00021

[18] S. Giri et al., "SMS Spam Classification–Simple Deep Learning Models With Higher Accuracy Using BUNOW And GloVe Word Embedding", *Journal of Applied Science and Engineering*, vol. 26, no. 10, pp. 1501–1511, 2023. https://doi.org/10.6180/jase.202310_26(10).0015

[19] P. K. Roy et al., "Deep Learning to Filter SMS Spam", *Future Generation Computer Systems*, vol. 102, pp. 524–533, 2020. https://doi.org/10.1016/j.future.2019.09.001

[20] S. E. Rahman and S. Ullah, "Email Spam Detection Using Bidirectional Long Short Term Memory with Convolutional Neural Network", in *Proceedings of the 2020 IEEE Region 10 Symposium (TENSYMP)*, 2020, pp. 1307–1311. https://doi.org/10.1109/TENSYMP50017.2020.9230769

[21] S. Yang et al., "LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example", in *Proceedings of the 2020 International Workshop on Electronic Communication and Artificial Intelligence, IWECAI*, 2020, pp. 98–101. https://doi.org/10.1109/IWECAI50956.2020.00027

[22] A. Salunkhe, "Attention-based Bidirectional LSTM for Deceptive Opinion Spam Classification", arXiv preprint, 2021. https://doi.org/10.48550/arXiv.2112.14789

[23] M. S. Javed et al., "Fake Reviews Classification Using Deep Learning Ensemble of Shallow Convolutions", *Journal of Computational Social Science*, vol. 4, pp. 883–902, 2021. https://doi.org/10.1007/s42001-021-00114-y

[24] P. Hajek et al., "Fake Consumer Review Detection Using Deep Neural Networks Integrating Word Embeddings and Emotion Mining", *Neural Computing and Applications*, vol. 32, pp. 17259–17274, 2020. https://doi.org/10.1007/s00521-020-04757-2

[25] P. Bhuvaneshwari et al., "Spam Review Detection Using Self-attention-based CNN and Bi-directional LSTM", *Multimedia Tools and Applications*, vol. 80, pp. 18107–18124, 2021. https://doi.org/10.1007/s11042-021-10602-y

[26] A. Q. Mir et al., "Online Fake Review Detection Using Supervised Machine Learning And BERT Model", arXiv preprint, 2023. https://doi.org/10.48550/arXiv.2301.03225

[27] Y. Guo et al., "Spam Detection Using Bidirectional Transformers and Machine Learning Classifier Algorithms", *Journal of Computational and Cognitive Engineering*, vol, 2, no. 1, pp. 5–9, 2023. https://doi.org/10.47852/bonviewJCCE2202192

[28] I. Androutsopoulos et al., "The Enron-spam Datasets", 2006. http://www2.aueb.gr/users/ion/data/enron-spam/

[29] N. Raftogiannis, "Simple Spam Email Classifier ~= 95%", *Kaggle*, 2021. https://www.kaggle.com/nikosraftogiannis/simplespam-email-classifier-95

[30] D. Refaeli and P. Hajek, "Detecting Fake Online Reviews Using Fine-tuned BERT", in *Proceedings of the 2021 5th International Conference on E-Business and Internet*, pp. 76–80, 2021. https://doi.org/10.1145/3497701.3497714

[31] P. Gupta et al., "Leveraging Transfer Learning Techniques-bert, Roberta, Albert and Distilbert for Fake Review Detection", *In Forum for Information Retrieval Evaluation*, pp. 75–82, 2021. https://doi.org/10.1145/3503162.3503169

[32] Q. Yaseen, "Spam Email Detection Using Deep Learning Techniques", *Procedia Computer Science*, vol. 184, pp. 853–858, 2021. https://doi.org/10.1016/j.procs.2021.03.107

[33] D. Dua and C. Graff, "UCI Machine Learning Repository", 2017. http://archive.ics.uci.edu/ml

[34] K. Veerakumar, "Spam filter", 2017. https://www.kaggle.com/karthickveerakumar/spam-filter

[35] T. Mikolov et al., "Distributed Representations of Words and Phrases and Their Compositionality", *Advances in Neural Information Processing Systems*, vol. 26, 2013. https://dl.acm.org/doi/abs/10.5555/2999792.2999959

[36] J. Pennington et al., "Glove: Global Vectors for Word Representation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. http://dx.doi.org/10.3115/v1/D14-1162

[37] P. Bojanowski et al., "Enriching Word Vectors with Subword Information", *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

Received: March 2024
Revised: April 2024
Accepted: June 2024

Contact addresses:
Mafas Raheem
School of Computing
Asia Pacific University of Technology and Innovation
Kuala Lumpur
Malaysia
e-mail: raheem@apu.edu.my

Yi Chien Chong
School of Computing
Asia Pacific University of Technology and Innovation
Kuala Lumpur
Malaysia
e-mail: tp069459@mail.apu.edu.my

MAFAS RAHEEM, a Senior Lecturer at the Asia Pacific University of Technology & Innovation in Malaysia, specializes in Data Science. He holds a BSc in Computer Science, an MBA, and an MSc in Data Science & Business Analytics. Currently pursuing a PhD in the field of machine learning, his research focuses on NLP, AI in eCommerce, and healthcare. With nearly 45 indexed journal articles, a book chapter, and participation in 10 conferences, he is a recognized academic. He collaborates with companies like OCBC and Crystal, has received international awards from the USA, Germany, Malaysia, and India, recognizing his national and international impact.Furthermore, he serves as a mentor to students in competitions, fostering the next generation of innovators.

YI CHIEN CHONG holds an MSc in Data Science from the Asia Pacific University of Technology and Innovation (APU) in Malaysia. With a strong foundation in data analytics and machine learning, he specializes in Natural Language Processing (NLP), focusing on the intersection of language and artificial intelligence. Through his research, he explores innovative techniques to enhance machine understanding of human language, driving advancements in NLP applications such as sentiment analysis, machine translation, and conversational AI. His academic background and research expertise position him at the forefront of cutting-edge AI and data-driven technologies.