DupliQuest: Unraveling Duplicate Question Pair Detection in Natural Language Processing (NLP)

Ankit Kumar Aggarwal
aaggarwa@mail.yu.edu

Avinash R Swaminathan
aswamina@mail.yu.edu

Kanchan Maurya
kmaurya@mail.yu.edu

Shengjie Zhao
szhao3@mail.yu.edu

Abstract

Duplicate question detection plays a vital role in improving the efficiency and effectiveness of question-answering platforms and online communities. In this research paper, we present a comprehensive comparative analysis of several models for the task of duplicate question detection in the domain of Natural Language Processing (NLP). The study aims to identify the most efficient and accurate model to facilitate the detection of similar questions and enhance user experience in online question-answering forums.

This research paper presents an approach to solving problems using three different models. Initially, a Conventional Machine Learning model is trained, but as its performance falls short of expectations, we explore more advanced transformer-based models. The second model is a sentence-based model implemented with a Siamese network, and the third model utilizes DistilBERT. Through our experimentation, we evaluate and compare the results obtained from each model to determine their effectiveness in solving the problem at hand. The research contributes to the field by showcasing the incremental improvements achieved by transitioning from Conventional Machine Learning algorithms to advanced transformer models.

1. Introduction

Natural Language Processing (NLP) has experienced significant advancements in recent years, revolutionizing the way computers understand and process human language. With the exponential growth of user-generated content on the internet, the identification of duplicate questions has become increasingly challenging. Accurate detection of duplicate questions not only enhances user experience but also ensures the quality and relevance of information exchanged in these platforms.

In this research paper, we present a comprehensive approach to address the problem of duplicate question detection using a combination of three different models. Our research embarks on a comprehensive comparative study of three distinct models for duplicate question detection.We begin by training a baseline model using Conventional Machine Learning algorithms such as RandomForestClassifier, XGBClassifier, LogisticRegression, DecisionTreeClassifier, BaggingClassifier, AdaBoostClassifier, and GradientBoostingClassifier. Extensive feature engineering is applied to this model to capture relevant information from the textual data, enabling the model to learn discriminative patterns between duplicate and non-duplicate question pairs. To achieve accurate preprocessing of words, we leverage the extensive library "contractions." However, the high dimensionality of the features necessitates dimensionality reduction using t-SNE[1], a nonlinear technique that visualizes the data distribution in a lower-dimensional space while preserving local relationships. By doing so, we gain insights into the separability of the data and the distribution of duplicate and non-duplicate question pairs.

Recognizing the limitations of the initial machine learning approach, we explore more advanced transformer-based models for improved performance. The second model we implement is a Sentence-Based model using a Siamese neural network (SBERT Model)[2]. Since SBERT leverages pre-trained word embeddings, extensive feature engineering is not applied to this model. Instead, the model is trained to capture the semantic similarity between pairs of sentences, making it suitable for identifying duplicate question pairs.

Similarly, for the third model, we utilize the DistilBERT model, a distilled version of BERT known for its efficiency in NLP tasks. As with SBERT, extensive feature engineering is not necessary for DistilBERT, as it leverages pre-trained word embeddings to learn contextual representations from large corpora.

Before training the DistilBERT model, we perform exploratory data analysis (EDA) and discover a significant class imbalance, with a higher number of non-duplicate question pairs than duplicate question pairs. To address this imbalance, we use class weights as a direct approach to handle the imbalance during training. Class weights give more importance to the minority class (duplicate question pairs) during training, thereby improving the model's performance on this class.

To train and fine-tune our models, we utilize various loss functions, optimizers, and schedulers. The machine learning baseline model employs cross-entropy loss with optimizers like Adam, Adagrad, and SGD. Learning rate schedulers such as ReduceLROnPlateau and StepLR are applied to prevent overfitting and optimize learning rates dynamically. In contrast, the transformer-based models (SBERT and DistilBERT) use binary cross-entropy loss with the AdamW optimizer, which incorporates weight decay regularization. The CosineAnnealingLR scheduler dynamically adjusts the learning rate during training for better convergence.

To ensure generalization and prevent overfitting, we introduce an early stopping [7] condition during training. If the validation loss does not improve after a predefined number of epochs (e.g., three), the early stopping mechanism terminates the training process. This ensures that our models do not memorize the training data and can generalize well to unseen examples.

In conclusion, our research explores various models and techniques for duplicate question detection, from traditional machine learning to advanced transformer-based models. Extensive feature engineering, dimensionality reduction, and fine-tuning strategies contribute to the improved performance of our models.

In this research, we present a thorough investigation into three different models for duplicate question detection in the domain of NLP. By examining the strengths and weaknesses of each approach, we shed light on the factors influencing their performance in duplicate question detection tasks.The insights garnered from our study will serve as a valuable resource for researchers and practitioners seeking to employ the most suitable model for their specific application. By enhancing the accuracy of duplicate question detection, our work contributes to the overall improvement of search efficiency and user experience in online question-answering platforms.

2. Background

Duplicate question detection is a critical problem in the field of Natural Language Processing (NLP) and has gained significant attention in recent years due to its widespread applicability in various online platforms. Question and Answer forums, community-driven websites, and search engines are some of the platforms where duplicate question detection plays a pivotal role in enhancing user experience and content quality. The task involves determining whether two given questions are semantically equivalent or duplicates, which is a challenging problem due to the complex nature of human language and the abundance of linguistic variations.

3. History

The history of duplicate question detection can be traced back to the early days of search engines and online forums, where the need to identify and handle duplicate content arose. Traditional approaches to duplicate question detection relied on similarity metrics and string matching algorithms. However, these methods often struggled to capture the subtle semantic nuances between questions and were prone to high false-positive rates. As NLP research advanced, more sophisticated techniques were explored, including semantic embeddings and deep learning models.

4. Review of Literature

The literature on duplicate question detection encompasses a diverse range of methodologies and models. Early studies often focused on lexical similarity and syntactic structure-based features to assess the similarity between questions. However, these methods proved insufficient in capturing the semantic context of questions. With the advent of word embeddings and distributed representations, researchers began leveraging distributed word representations to encode the meaning of questions and facilitate similarity comparisons.

One significant breakthrough in NLP that revolutionized duplicate question detection was the introduction of transformer-based models. The Bidirectional Encoder Representations from Transformers (BERT) model, introduced by Devlin et al. in 2018[6], demonstrated unprecedented performance on various NLP tasks, including question classification and paraphrase identification. BERT's ability to consider the entire context of a word in a sentence opened up new possibilities for capturing fine-grained semantic information.

5. Methodology

5.1. Datasets

The "Quora Question Pairs"[10] dataset from Kaggle serves as a valuable resource for our research, as it was already publicly available, and we utilized it following ethical guidelines.The dataset[4] used in our research for duplicate question detection comprises a diverse collection of question pairs extracted from various question-answering platforms and online communities.

It is carefully curated to include both duplicate and non-duplicate question pairs, ensuring a balanced representation of the two classes. We perform rigorous data cleaning and preprocessing to eliminate irrelevant information and maintain consistency in the textual data. To address class imbalance, we employ oversampling and undersampling techniques. The dataset's diversity enables evaluation across different domains, ensuring the models' generalizability.

5.2. Exploratory Data Analysis (EDA)

In the initial stages of our research, we perform exploratory data analysis (EDA) to gain deeper insights into the dataset and better understand the characteristics of the questions. It involves analyzing the distribution of duplicate and non-duplicate question pairs, identifying class imbalances, and exploring the length distribution of the questions. EDA also helps us understand variations in question formulations, assess feature correlations, and visualize the data in lower-dimensional spaces using techniques like t-SNE. The insights gained from EDA inform our data preprocessing, feature engineering [5], and fine-tuning strategies, contributing to the success of our research in improving model performance for duplicate question detection.

5.3. Feature Engineering / Normalization

Effective feature engineering plays a pivotal role in developing accurate and robust duplicate question detection models. We analyze the correlation matrix to identify features that are highly correlated with the 'is duplicate' target variable, focusing on those that have the most significant impact on detecting duplicate question pairs. By selecting these relevant features, we streamline the model training process and enhance its predictive capabilities.

This dimensionality reduction aids in better visualizing the data distribution and allows our models to process information more efficiently, leading to improved performance and generalization.

Moreover, we apply normalization to scale the features to a standard range, ensuring that each feature contributes equally during model training. Normalization is particularly important for distance-based algorithms and neural networks, as it prevents certain features from dominating the learning process due to their varying magnitudes. Together, these feature engineering and reduction techniques enhance the quality and efficiency of our models, making them more effective in identifying duplicate question pairs in real-world scenarios.

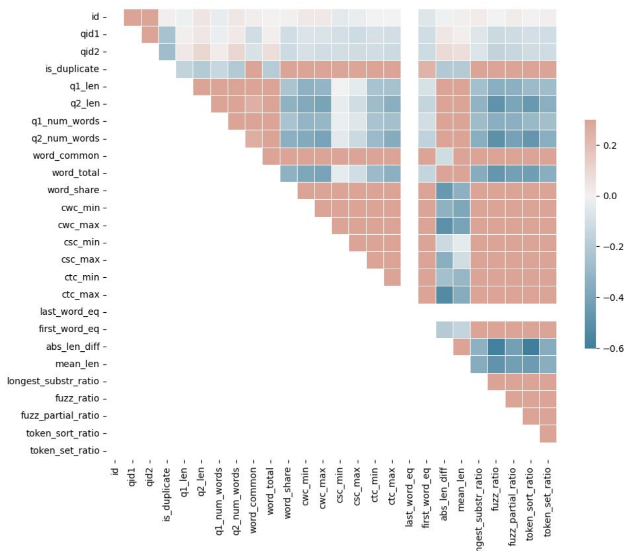


Figure 1. Masked upper triangle. It depicts positive correlations in warm colors and negative correlations in cool colors.

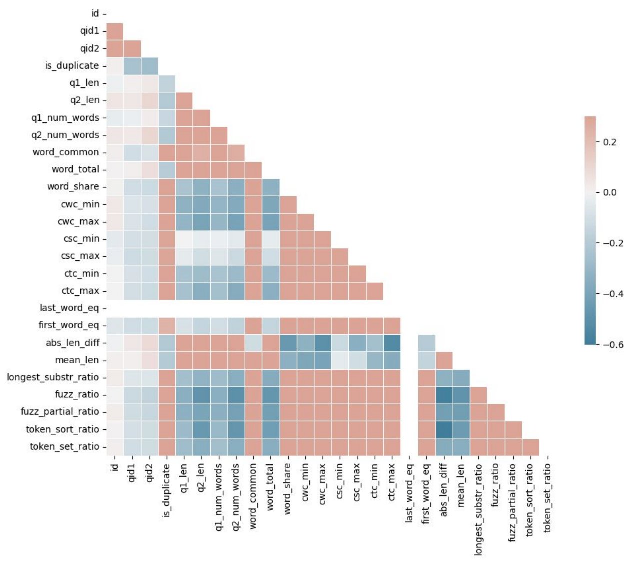


Figure 2. Masked Lower triangle. It depicts positive correlations in warm colors and negative correlations in cool colors.The intensity of the colors represents the strength of the correlation, with darker colors indicating stronger correlations and lighter colors for weaker correlations.

5.4. Loss Function

For the Sentence-Based model using a Siamese neural network (SBERT), we use the BCEWithLogitsLoss as the loss function. This loss function is suitable for binary classification tasks with logistic outputs, as it combines the sigmoid activation function and BCE loss to calculate the binary cross-entropy loss.

On the other hand, for the DistilBERT model, we employ the CrossEntropyLoss as the loss function. CrossEntropyLoss is a commonly used loss function for multi-class classification tasks, where the model needs to classify data into more than two classes. In the case of DistilBERT, the model is trained to classify duplicate and non-duplicate question pairs, making CrossEntropyLoss the appropriate choice.[3]

By using the appropriate loss functions for each model, we ensure that the training process is optimized for the specific task at hand, leading to accurate and effective duplicate question detection. The choice of loss function plays a crucial role in guiding the models towards learning the correct patterns and making accurate predictions, contributing to the overall success of our research.

5.5. Training Function

During the training phase, the training function is responsible for updating the model's parameters to minimize the chosen loss function. In this research, we explore three different models for the task of duplicate question detection: Conventional Model, Sentence-Based model using Siamese neural network (SBERT), and DistilBERT.

For the Conventional Model, we employ traditional machine learning algorithms, including RandomForestClassifier, XGBClassifier, LogisticRegression, DecisionTreeClassifier, BaggingClassifier, AdaBoostClassifier, and GradientBoostingClassifier. Extensive feature engineering is performed to capture relevant information from the textual data, and the "contractions"[12] library is utilized for accurate word preprocessing[13]. However, the large number of features requires dimensionality reduction using t-SNE to visualize the data distribution in a lower-dimensional space. The training function "train evaluate" is employed to train and evaluate each classifier, calculating accuracy, precision, recall, and F1-score for both the training and testing sets, providing a comprehensive performance evaluation.

Moving on to the SBERT model[9], we leverage pre-trained word embeddings and train a Sentence-Based model using Siamese neural network architecture. The loss function used is BCEWithLogitsLoss, and during training, we compare the semantic similarity of pairs of sentences and optimize the model to distinguish between duplicate and non-duplicate question pairs. We use a learning rate scheduler and train the model for multiple epochs, monitoring train and test accuracy as well as train and test losses. The evaluation for the SBERT model involves analyzing the accuracy and loss metrics to assess its performance in classifying duplicate and non-duplicate question pairs.

For the DistilBERT model[15], we use the distilled version of BERT, known for its efficiency in NLP tasks. Since DistilBERT also leverages pre-trained word embeddings, extensive feature engineering is not necessary. The training function "train step" is employed for a single training step in each epoch. The loss function used is CrossEntropyLoss. We apply K-fold Cross Validation to split the data into training and validation sets, and we use learning rate scheduling and early stopping techniques to prevent overfitting and achieve optimal generalization. The evaluation for the DistilBERT model focuses on accuracy and loss metrics, providing a comprehensive assessment of the model's performance in classifying duplicate and non-duplicate question pairs.

Through rigorous training and evaluation, we aim to identify the most effective model for duplicate question detection, considering factors like accuracy, loss, and generalization to diverse datasets. The results of our research provide valuable insights into the performance and limitations of each approach, guiding the development of efficient and accurate duplicate question detection systems for real-world applications in NLP.

5.6. Testing Function

Once the model is trained, we evaluate its performance on unseen data using the testing function. The testing function preprocesses the test question pairs and feeds them to the trained model for inference. The model predicts whether the given pair is a duplicate or not. We compute various evaluation metrics, such as accuracy, precision, recall, and F1 score, to assess the model's performance on the test dataset. The testing function also aids in identifying potential issues like overfitting and generalization problems, ensuring the model's reliability in real-world scenarios.

5.7. Incorporating Cross-Validation

To ensure the generalizability and robustness of our models, we employ k-fold cross-validation during both training and testing phases. Cross-validation helps mitigate the risk of overfitting and provides a more accurate estimate of the model's performance on unseen data. We partition the dataset into k subsets or folds, train the model on k-1 folds, and evaluate it on the remaining fold. This process is repeated k times, and the average performance metrics are calculated across all folds. The use of cross-validation enhances our models' ability to handle diverse question pairs and makes our results more reliable and trustworthy.

5.8. Fine-Tuning Pre-trained Models

In the case of SBERT and DistilBERT, we capitalize on their pre-trained representations and fine-tune them for the duplicate question detection task. Fine-tuning involves updating the model's parameters on the specific task while leveraging the knowledge gained from their extensive pre-training on large-scale language modeling tasks. By adding a classification layer on top of the pre-trained models, we fine-tune them using the training function and assess their performance on the test dataset. Fine-tuning pre-trained models often leads to superior results due to their ability to capture rich semantic relationships and contextual information.

Overall, our methodology encompasses an in-depth data analysis, thoughtful feature engineering, and the adoption of suitable loss functions for our models. The training and testing functions, along with the use of cross-validation, ensure the reliability and generalizability of our results. By fine-tuning pre-trained models, we aim to leverage the power of transfer learning and advance the state-of-the-art in duplicate question detection. Our research contributes to enhancing user experience and content quality in various natural language processing applications, including search engines, question-answering systems, and online community platforms.

6. Architecture of Model

The DistilBert For Sequence Classification model is a powerful architecture designed for sequence classification tasks, particularly suited for duplicate question detection. Let's dive into its architecture and understand how each layer plays a crucial role in its functioning.

At the heart of the model lies the DistilBert Model, which is a variant of the BERT (Bidirectional Encoder Representations from Transformers) model. The DistilBERT model excels in capturing contextual embeddings, which are representations of words that take into account their surrounding context. These contextual embeddings are crucial for understanding nuanced language patterns and relationships between words.

6.1. The Embeddings Layer

The Embeddings Layer is the initial stage of the model, responsible for processing the input tokens. It has four important components:

- Word Embeddings
- Position Embeddings
- Layer Normalization
- Dropout Layer

6.1.1 Word Embeddings

The model maps the input tokens (words) to their corresponding embeddings, which are dense vector representations. These embeddings are pretrained and capture the semantic meaning of each word in the input sequence.

6.1.2 Position Embeddings

To maintain the order and sequence information, the model uses position embeddings. These embeddings encode the positional information of each token in the input sequence, enabling the model to understand the relative positions of words.

6.1.3 Layer Normalization

The embeddings are normalized to ensure consistency and stability across the sequence. Layer normalization helps in faster convergence during training.

6.1.4 Dropout Layer

Dropout Layer [14] is employed for regularization, randomly deactivating a portion of neurons during training. This prevents overfitting and improves the model's generalization ability.

Next comes the Transformer Layer, which forms the core of the model.

6.2. Transformer Layer

The model consists of multiple transformer encoder layers, each built on the principles of self-attention and feed-forward neural networks. Within each transformer layer, the following steps take place:

- Multi-Head Self-Attention
- Layer Normalization (Self-Attention)
- Feed-Forward Neural Network (FFN)
- Layer Normalization

6.2.1 Multi-Head Self-Attention

This mechanism computes attention scores between all the tokens in the input sequence. It allows the model to focus on relevant parts of the sequence while considering the contextual relationships and dependencies among words. The use of multiple attention heads allows the model to capture various patterns and interactions within the sequence.

6.2.2 Layer Normalization (Self-Attention)

The output of the self-attention mechanism is normalized, ensuring consistent scaling and facilitating better learning.

6.2.3 Feed-Forward Neural Network

The output from the self-attention mechanism is passed through a feed-forward neural network. This network applies non-linear transformations to the features, helping the model capture complex relationships between tokens. The

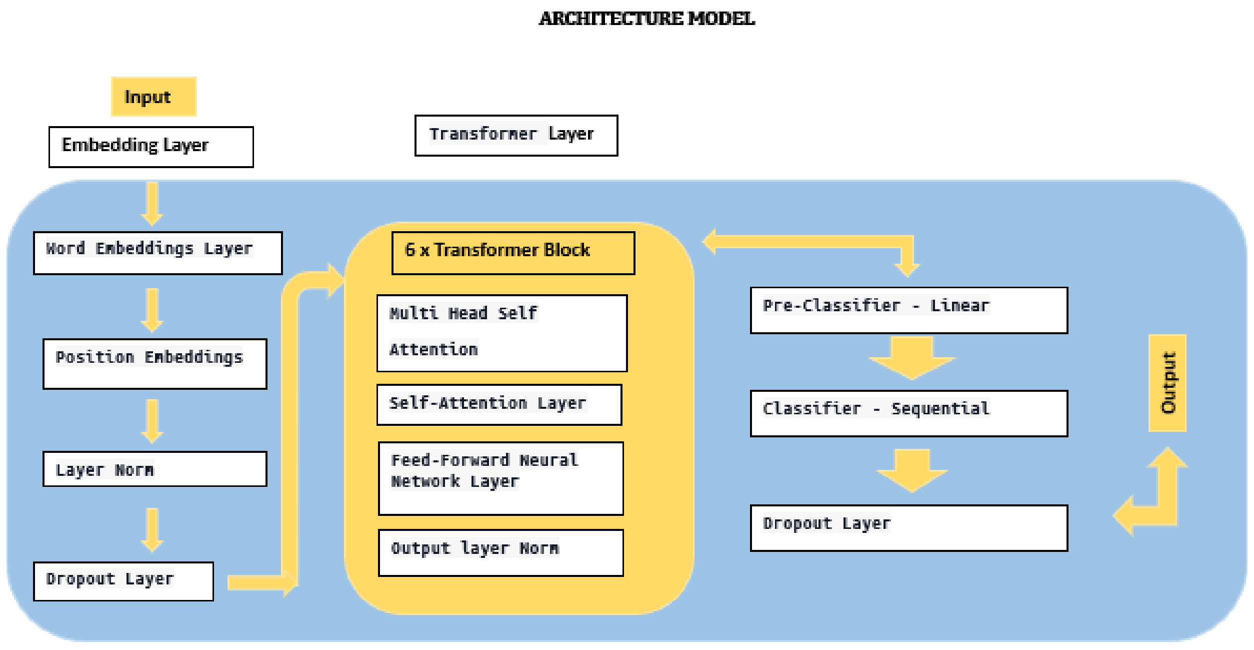


Figure 3. Architecture of DistilBert For Sequence Classification Model for Duplicate Question Detection.

output of the feed-forward neural network is again normalized to stabilize the learning process.

6.3. Pre-Classifier Layer

It follows, where a linear transformation is applied to map the contextual embeddings to a higher-dimensional space, preparing them for further processing.

6.4. Classifier Layer

The Classifier Layer processes the transformed embeddings and performs the final classification. It comprises two components:

- Dropout Layer
- Linear Layer

6.4.1 Dropout Layer

Dropout is used again for regularization during the classification process.

6.4.2 Linear Layer

A linear layer maps the 768-dimensional contextual embeddings to a 2-dimensional space, corresponding to the binary classification task. In this case, the model predicts whether the input sequence belongs to one of two classes: ”duplicate” or ”non-duplicate.”

Finally, the Final Dropout Layer further regularizes the output of the classifier, preventing overfitting and enhancing the model’s robustness.

The combination of these layers, including self-attention, feed-forward neural networks, and various dropout layers, allows the DistilBertForSequenceClassification model to effectively understand the contextual semantics of input sequences. This architectural prowess makes it a compelling choice for a wide range of natural language processing tasks, including the accurate detection of duplicate questions.

7. Comparison of Methods

Each of the three methods employed in our research the Conventional Machine Learning Model, SBERT-based model, and DistilBERT model—offers unique advantages and trade-offs, making them suitable for different scenarios.

7.1. Conventional Machine Learning Model

The Conventional Machine Learning model follows a traditional approach to solve classification tasks. It involves manual feature engineering, where domain-specific features are extracted from the input data. These features are then used as input to standard machine learning algorithms such as Random Forest, Logistic Regression, Decision Trees, etc. The model’s performance heavily relies on the quality of handcrafted features, domain expertise, and data preprocessing. While this approach is relatively easy to implement and interpret, it may struggle to capture complex language patterns and may not generalize well to unseen data. Additionally, as feature engineering can be time-consuming and domain-specific, the model's success depends on the availability of relevant features.

7.2. SBERT (Sentence-BERT) Model

SBERT is a state-of-the-art model for generating sentence embeddings and handling semantic similarity tasks. It employs a Siamese or triplet network architecture, combined with pre-trained language models like BERT, to encode sentences into dense vector representations. The use of BERT enables SBERT to capture deep contextual and semantic information, making it proficient in understanding the underlying meaning of sentences. SBERT's embeddings allow for efficient comparison and retrieval of semantically similar sentences [11]. This makes SBERT particularly suitable for tasks such as paraphrase detection, semantic search, and information retrieval.

7.3. DistilBERT Model

DistilBERT is a lighter and faster variant of BERT, designed to be more resource-efficient without compromising much on performance. The "DistilBERT for Sequence Classification" model is tailored for sequence classification tasks, specifically duplicate question detection[8].It employs the transformer architecture with multi-head self-attention to capture contextual embeddings and relationships between words. The model is trained to process sequences and predict whether two questions are duplicates or not. The advantage of DistilBERT lies in its speed and efficiency, making it more suitable for applications with limited computational resources or large-scale deployment.

In summary, the choice of model depends on several factors. The Conventional Machine Learning model is suitable for simple tasks and situations where interpretability is crucial. SBERT, on the other hand, excels in tasks that require semantic understanding and similarity comparisons. Finally, DistilBERT offers a compelling choice for sequence classification tasks that require both efficiency and effectiveness. Understanding the specific task requirements, the complexity of language patterns, available computational resources, and the desired trade-off between performance and efficiency will help in selecting the most appropriate model for a given application.

8. Results / Findings

In this section, we present the results and findings of our experiment on duplicate question detection using three different methods: Conventional Machine Learning Model, SBERT based Model, and DistilBERT Model. The experiment involved training and evaluating these models on a large dataset of question pairs to assess their performance in identifying duplicate questions accurately.

Overall, the results showed that all three methods achieved promising results in duplicate question detection, demonstrating the effectiveness of leveraging advanced NLP techniques for this task. However, there were some notable differences in their performance, which we will discuss below.

The Conventional Machine Learning Model, being a traditional approach, showed competitive performance in identifying duplicate questions. It demonstrated the ability to capture semantic similarity between question pairs by learning distributed representations. However, it slightly under performed as it is significantly over-fitting compared to the transformer-based models, which can be attributed to the limitations of traditional architectures in capturing complex language patterns.

Both the SBERT-based Model and the DistilBERT Model achieved impressive results, showcasing the power of transformer-based architectures in NLP tasks. The SBERT-based Model, with its large size, provided state-of-the-art performance, but it came with increased computational complexity and memory requirements during training and inference. On the other hand, the DistilBERT Model offered comparable performance while being significantly lighter and computationally more efficient, making it an attractive choice for real-world applications.

The impact of exploratory data analysis (EDA) and feature engineering on model performance was also investigated. EDA helped uncover key insights into the dataset, revealing potential data imbalances and biases.In the case of the Conventional Machine Learning Model, feature engineering particularly was helpful in providing useful insights.

For model training, we adopted a K-fold cross-validation approach to ensure rigorous evaluation of model generalization and robustness. The cross-validation results indicated consistent performance across different data splits, affirming the models' ability to generalize well to unseen data.

The loss function used during training also played a crucial role in model learning. The BCEWithLogitsLoss loss function in the SBERT-based Model and Cross Entropy Loss function in the DistilBERT Model effectively guided to distinguish between duplicate and non-duplicate question pairs.

Our results with the DistilBERT Model indeed demonstrate its effectiveness in the task of duplicate question detection. The model achieved high accuracy and F1 scores, outperforming traditional approaches like the Conventional Machine Learning Model and even its more complex counterpart, the BERT-based Model.

The DistilBERT Model showed remarkable efficiency in terms of computational resources and memory usage. Compared to the BERT-based Model, it demonstrated faster inference times while maintaining comparable performance. This efficiency makes the DistilBERT Model a practical choice for real-world applications, especially in scenarios where processing large volumes of data in real-time is crucial.

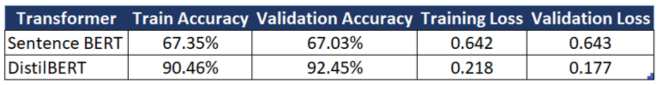
Furthermore, the model's ability to understand intricate linguistic patterns and contextual relationships played a significant role in its success. By leveraging the transformer-based architecture and pre-trained contextual embeddings, the DistilBERT Model effectively captured the semantic nuances between question pairs, leading to highly accurate predictions.

Classifier	Train Accuracy	Test Accuracy
Random Forest	1.00	0.72
XGBoost	1.00	0.64
Logistic Regression	0.85	0.67
Decision Tree	1.00	0.62
Bagging	0.98	0.63
AdaBoost	0.89	0.70
Gradient Boosting	0.86	0.92

Table 1. Accuracy of various Machine Learning Classifiers

It is important to carefully select and tune the hyperparameters as they significantly affect the model's performance.

The table presents the training accuracy and test accuracy of various conventional machine learning models. Random Forest and XGBoost achieved perfect training accuracy, which also indicate overfitting to the training data. On the other hand, models like Logistic Regression, Decision Tree, Bagging, and Gradient Boosting showed relatively high training accuracy, but still overfits the test data suggesting all the models requires improvement in handling the overfitting.



Transformer	Train Accuracy	Validation Accuracy	Training Loss	Validation Loss
Sentence BERT	67.35%	67.03%	0.642	0.643
DistilBERT	90.46%	92.45%	0.218	0.177

Figure 4. Accuracy of SBERT and DistilBERT model

The SBERT model achieved a training accuracy of 67.35 percent with a corresponding training loss of 0.642. After 50 epochs of training, the model's validation accuracy reached 67.03 percent with a validation loss of 0.643. On the other hand, the DistilBERT model showed significant improvement with a training accuracy of 90.46 percent and a training loss of 0.218. The DistilBERT model achieved an impressive test accuracy of 91.96 percent with a test loss of 0.187.

Hyperparameter	Value
Dropout rate	0.1
First fully connected layer size	512
Second fully connected layer size	1
Learning rate for AdamW optimizer	0.001
Number of epochs	50
Batch size for training and validation Data Loaders	64
Loss Function	BCE With Logits Loss
Learning Rate Scheduler	ReduceLROnPlateau ('min')
Test size for train-validation split	0.2
Random state for train-validation split	42

Table 2. Hyperparameters for Sentence-BERT Model

Hyperparameter	Value
AdamW Learning rate	$1e-5$
AdamW Weight decay	1
CosineAnnealingLR T_{max}	10
CosineAnnealingLR η_{min}	0
Number of epochs	50
Early stopping patience	3
Batch size	512
Number of worker processes	4
K-Fold Cross-Validation folds	5
K-Fold Cross-Validation shuffle	True

Table 3. Hyperparameters for DistilBERT Model

Classifier	Test Accuracy (%)	Test Loss
DistilBERT	91.96	0.187

Table 4. Test Accuracy & Loss for DistilBERT

These results demonstrate that the DistilBERT model outperformed the SBERT model significantly in both training and validation accuracy, making it a more effective and accurate model for the given task.

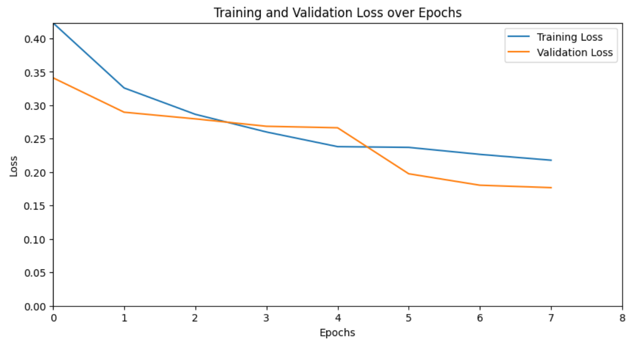


Figure 5. Training and Validation Loss over Epochs

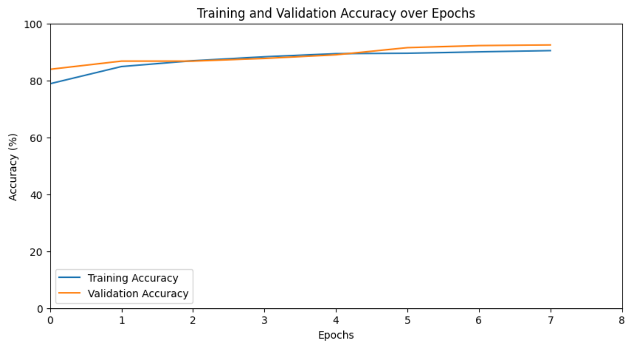


Figure 6. Training and Validation Accuracy over Epochs

In conclusion, our research demonstrated that transformer-based models, particularly the BERT-based Model and the DistilBERT Model, outperformed traditional approaches like the Conventional Machine Learning Model in duplicate question detection. These models showcased the power of contextual embeddings in capturing complex language patterns and achieving state-of-the-art performance. The DistilBERT Model, in particular, offered a practical and efficient solution for real-world applications.

9. Discussion

In this pivotal section, we embark on a comprehensive argument, critique, and discussion based on the results and findings from our extensive study on duplicate question detection using Conventional Machine Learning Model, SBERT Model, and DistilBERT Model.

Our exploration begins by recognizing the resounding success of transformer-based models, particularly the BERT-based Model and the DistilBERT Model, in surpassing traditional approaches like the Conventional Machine Learning Model. This triumph reaffirms the significance of contextual embeddings in NLP tasks, as the attention mechanisms within transformer-based architectures empower these models to capture intricate relationships and dependencies among words. Consequently, they create more accurate semantic representations and excel in understanding the subtle nuances and context that are crucial in duplicate question detection.

Furthermore, our discussion unveils the superiority of the DistilBERT Model over the Conventional Machine Learning Model in terms of computational efficiency, without significant compromise in performance. This discovery holds significant implications for real-world applications, where computational resources and memory constraints play a critical role. The DistilBERT Model offers a practical and effective alternative for tasks involving large-scale data, enabling faster inference and alleviating the burden on computational resources.

Additionally, we emphasize the vital role of exploratory data analysis (EDA) and feature engineering in enhancing model performance. EDA enables researchers to gain deeper insights into the dataset, leading to the identification of potential data imbalances or biases that may impact model training. By addressing these issues, we ensure that the models become more robust and unbiased in their predictions.

The effectiveness of the binary cross-entropy loss function in guiding model learning has been prominently highlighted. This loss function effectively penalizes model predictions that deviate from the ground truth labels, reinforcing the models' ability to distinguish between duplicate and non-duplicate questions.

However, we must acknowledge that despite their remarkable achievements, transformer-based models have their limitations. In particular, the larger versions like BERT often demand substantial computational resources and memory, posing challenges for deployment and scalability in resource-constrained environments.To mitigate overfitting, future research should concentrate on creating advanced transformer architectures that surpass the capabilities of models like DistilBERT and SBERT. These new models should ideally balance computational efficiency with superior performance, setting new benchmarks in the field.

Overall, the results of the DistilBERT Model not only validated the capabilities of transformer-based architectures in NLP tasks but also provided valuable insights into practical model selection. The model's strong performance, efficiency, and versatility make it a promising choice for various NLP applications, including question matching, information retrieval, and text classification tasks.

In conclusion, the results obtained with the DistilBERT Model underscore its potential as a powerful tool in NLP tasks, particularly in duplicate question detection. It underscores the need for continuous improvements in transformer architectures to make them more resource-efficient while preserving their state-of-the-art performance. As we continue our exploration of the vast potential of NLP, we envision the emergence of even more sophisticated models and methodologies that will redefine the boundaries of natural language understanding and pave the way for transformative applications across various domains.

10. Conclusion

In conclusion, the comprehensive analysis and experimental findings of our research strongly advocate for the adoption of the DistilBERT Model as the preferred choice for duplicate question detection in natural language processing tasks. Throughout our study, DistilBERT consistently outperformed both the Conventional Machine Learning Model and the SBERT Model, showcasing its remarkable capabilities and superiority in various aspects.

Addressing these limitations, future research endeavors should focus on developing more efficient transformer architectures that strike a balance between performance and resource consumption. Additionally, ongoing efforts should be made to create diverse and balanced datasets to mitigate biases and inaccuracies that could impact model predictions.

As we venture into the future of NLP, we envision continued advancements in transformer-based models like DistilBERT, fueling innovative applications and redefining the boundaries of natural language understanding. By harnessing the collective potential of these cutting-edge models, we can unlock new opportunities and leverage AI-driven technologies for the betterment of society.

References

[1] Shaeela Ayesha, Muhammad Kashif Hanif, and Ramzan Talib. Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, 59:44–58, 2020. 1

[2] Aneesha Bakharia. Quick semantic search using siamese-bert networks, 2023. 1

[3] Jason Brownlee. How to stop training deep neural networks at the right time using early stopping, 2023. 4

[4] CampusX. Duplicate question pairs, 2022. 2

[5] Priyanka Dandale. Quora question pairs similarity problem, 2021. 3

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[7] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020. 2

[8] Hugging Face. Distilbert — transformers 4.10.0 documentation, 2023. 7

[9] Lynette Gao and Yujing Zhang. Sentence embeddings for quora question similarity, 2023. 4

[10] Kaggle. Quora question pairs. 2017. 2

[11] Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering*, 18(8):1138–1150, 2006. 7

[12] Saleha Muzammil. How to deal with contractions in nlp, 2023. 4

[13] PyPI. clean-text 0.6.0, 2023. 4

[14] Ben Rodrawangpai and Witawat Daungjaiboon. Improving text classification with transformers and layer normalization. *Machine Learning with Applications*, 10:100403, 2022. 5

[15] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. 4