Simplifying Question-Answering on Quora with Machine Learning

K.M.Mitravinda Saran Pandi Homaira Shomee Subha Ilamathy
University of Illinois at Chicago
kmitr@uic.edu, spand43@uic.edu, hshome2@uic.edu, silama3@uic.edu

Abstract

In the evolving landscape of digital knowledge sharing, ensuring that each logically similar question exists only once in a platform's repository presents a significant challenge. This challenge is particularly acute on platforms like Quora, a question-and-answer website, where identifying duplicate questions is essential. This project delves into using machine learning techniques to tackle this issue using models such as Naive Bayes, XGBoost, and SVM and BERT pretrained model. The study examines the impact of various text representation techniques such as Bag of Words, Word2Vec, TF-IDF (term frequency-inverse document frequency), and BERT embeddings. By leveraging cutting-edge machine learning models, the study explores the nuances of semantic understanding in text. Text preprocessing methods are employed to refine the models' ability to discern subtle differences and similarities in question phrasing. Out of all the models, XGBoost along with Bag of words text representation gave us the highest F1-score of 0.86. This comparative analysis is key in enhancing the efficiency of information retrieval and knowledge sharing on question-and-answer platforms, shedding light on the strengths and limitations of each model in detecting duplicate questions.

1 Introduction

Quora is an online platform which enables sharing information through question-and-answers. It has an estimated number of 300 million monthly users [11]. The platform has a global community where the users post their questions and the writers answer questions based on their expertise. Due to the huge number of users, millions of questions have been posed. This leads to the possibility of questions from different users having the same intent [2]. In online forums like Quora, threads, which make up questions and their set of answers, are a crucial part. Since there are numerous ways to pose a question, users most likely ask questions with the same intent or semantically same questions which have already been answered in other threads [8]. For instance, questions "What is the most populous country in the world?" and "Which country has the most people?" have the same intent and therefore must be answered only once. Multiple threads or pages being present for the same question adds an overhead. Thus, it becomes critical to solve the duplicate question problem persisting in Quora by identifying similar questions and merging them into a single thread [7]. This enables readers to access all answers to a question at a single location and prevents writers from the redundancy of having to write the same answer to multiple versions of the same question. This also avoids the reader audience from getting divided among multiple pages or threads, thus allowing writers to reach a much larger audience.

The problem of identifying duplicate questions is not just a matter of filtering similar texts, but also involves intricate challenges within the realm of Natural Language Processing. It requires the system to perceive the underlying meaning or intent of the questions, which can be linguistically complex due to variations in phrasing, context, and language nuances. The ability of a model to understand and interpret the lexical, syntactic, and semantic aspects of language plays a crucial role

in identifying question duplicates. In this project, we address this intricate problem by presenting a comprehensive analysis using traditional and state-of-the-art machine learning models on the Quora Question Pairs dataset. This dataset, comprising pairs of questions, poses the binary classification challenge of determining whether two questions share the same meaning. We emphasize the importance of representing textual data in a form conducive to machine learning algorithms. This involves word representations that transform sentences into numerical inputs, which are then processed by our models. Our study underscored different traditional and modern NLP techniques, offering insights into their efficacy in a real-world application. We believe that this project will contribute to improving the reader experience as well as to reducing information redundancy on Quora.

2 Dataset

The Quora Question Pairs dataset consists of 404,352 pairs of questions where the number of negative and positive examples are 255,045 and 149,306 respectively [4]. Each pair is labeled as duplicate or non-duplicate, indicating whether or not the two questions share the same intent. This dataset is not only large but also complex due to the nuances of logically similar questions or questions with the same intent and the imbalance or skewed proportion of classes, making it a valid ground for testing NLP models. Each sample in the dataset consists of the fields mentioned in Table 1.

Table 1: Description of columns in dataset

Column Name	Description
id	A unique identifier assigned to each row in the dataset.
qid1	A unique identifier for the question in question1 column.
qid2	A unique identifier for the question in question2 column.
question1	The actual question to be compared with question2.
question2	The actual question to be compared with question2.
is_duplicate	The result of a semantical comparison of question pair.

The figure 1 visualizes the number of positive (duplicate) and negative (not-duplicate) question pairs present in the dataset.

3 Data Preprocessing

Given the complexity of the dataset, we employed a comprehensive data preprocessing steps as explained below:

3.1 Data Cleaning

The dataset consisted of minimal number of missing values. The number of missing values in each column is summarized in Table 2. Due to the minimal number of missing values, the rows corresponding to each missing value were dropped to obtain a clean dataset.

3.2 Text Normalization

Our preprocessing involved extensive text cleaning and normalization. This included lowercasing the text, removing special characters and stop words, elimination of punctuation, etc. We also integrated stemming, a technique for normalizing text that reduces words to their base or root form. We aimed to standardize the text to reduce variability that doesn't contribute to understanding the question's intent.

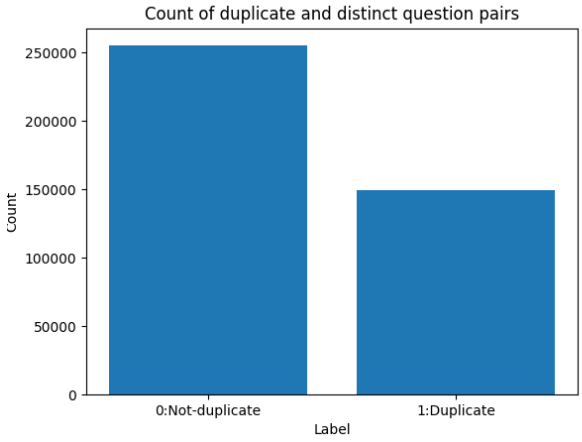

Figure 1: Class label distribution

Table 2: Count of missing values per column

Column	Number of missing values
id	0
qid1	0
qid2	0
question1	1
question2	2
is_duplicate	0

3.2.1 Text Embeddings

Advanced embedding techniques were crucial for capturing semantic meanings. The following text embeddings were utilized in this project.

- Bag of Words: Bag-of-words is a text representation that describes the occurrence of words within a document. It involves a vocabulary of known words and a measure of the frequencies of known words.
- BERT Embedding: Utilized to understand the contextual relationship between words in a sentence.
- Word2Vec Embedding: Word2Vec embedding is a method in which individual words are converted into a vector, or numerical representation of the word. Every word is assigned a single vector, which is subsequently trained in a manner akin to a neural network. The vectors attempt to depict several aspects of the word in relation to the entire text. These qualities may include the word's semantic relationship, definitions, context, etc.
- TF-IDF: TF-IDF (term frequency-inverse document frequency) is a statistical measure that assesses a word's relevance to a document within a set of documents.This is accomplished by multiplying two metrics, the number of times a word appears in a document, and the inverse document frequency of the word across the set of documents.

By utilizing below diverse preprocessing and text embedding methods, we aimed to create a rich representation of the question pairs. Example of above word embeddings are shown in Table 3 for the question "What are the natural numbers?".

Table 3: Examples of Text Embeddings

(a) Bag of Words

are	natural	numbers	the	what
1	1	1	1	1

(b) TF-IDF

are	natural	numbers	the	what
0.447214	0.447214	0.447214	0.447214	0.447214

(c) Word2Vec

Word	Vector
What	[-0.02307129 0.15332031 0.15234375 . . .]
are	[-0.09667969 -0.02636719 0.09033203 . . .]
the	[0.08007812 0.10498047 0.04980469 . . .]
natural	[0.03088379 0.20019531 -0.08789062 . . .]
numbers	[0.28125 0.08691406 0.16210938 . . .]

4 Approach

This section presents the process of feeding the preprocessed dataset to four distinct machine learning classifiers, as depicted in Figure 2. It elaborates on the implementation of these models, detailing their setup and the configuration of hyperparameters. Additionally, the section highlights how text embeddings are incorporated within each model.

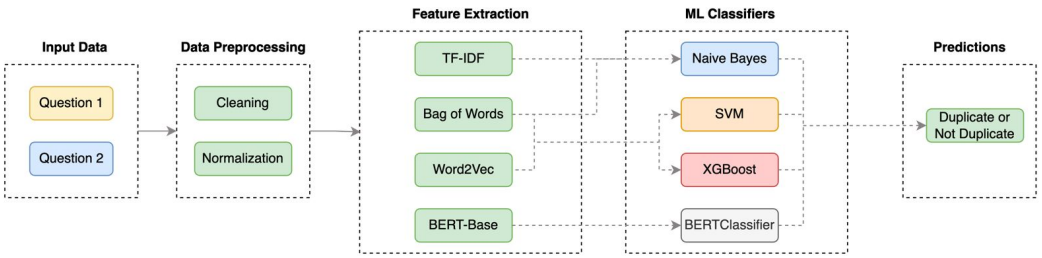

Figure 2: Process of Implementation

4.1 Machine Learning Models

4.2 Naive Bayes

The Naive Bayes classifier, known for its simplicity and efficiency, is one of the foundational models used in this project. Despite its assumption of feature independence, which is a simplification when dealing with natural language, it serves as a quick and effective method for initial explorations of text data [**?**]. We implemented this probabilistic model, leveraging the frequencies of words and their conditional probabilities given the class (duplicate or not-duplicate). This model particularly excels in scenarios where the dimensionality of the input space (number of unique words) is high. Given the linguistic diversity of the dataset, we used various text preprocessing techniques to enhance the Naive Bayes model's effectiveness, including stemming, and removal of stopwords. Two types of experiments were conducted by implementing the Naive Bayes model on Bag of Words and TF-IDF text representations and their corresponding performances were evaluated. These techniques aimed to capture different aspects of the text data in the Quora dataset, influencing the classifier's ability to discern duplicates. We did not experiment by implementing the Naive Bayes model on Word2Vec text representation since it creates dense word embeddings, which represent words in a continuous vector space, while Naive Bayes typically expects discrete features.

4.3 XGBoost

XGBoost [3], known for its performance in structured data, was adapted for our text-based problem. In optimizing the XGBoost model for the Quora dataset, we meticulously adjusted several hyperparameters. A total of 100 estimators were chosen, forming a robust ensemble of decision trees,

essential for a comprehensive learning process and precise predictions. The learning rate was set at a moderate value of 0.1, balancing the speed of learning with the accuracy, ensuring swift yet accurate adaptation to text complexities. Performing L_1 regularization with reg_alpha parameter set at 4 simplified the model for better generalizability, while maintaining predictive power. Finally, the 'binary logistic' objective function was strategically selected, fitting the binary nature of our task: determining the similarity between question pairs. In our text classification task, XGBoost was effectively applied by leveraging advanced text representation techniques like Word2Vec and Bag of Words. These embeddings enrich the input data, which XGBoost then processes, utilizing its gradient boosting framework to make accurate, robust predictions.

4.4 SVM

SVM [6] is a supervised machine learning model that can be used for both classification and regression tasks. As a classifier, SVM's main goal is to find the hyperplane that best can best separate data points belonging to different classes. In this application, SVM finds the optimal hyperplane separating the question pairs into 2 classes - duplicate and not-duplicate. In particular, the 'Soft Margin Linear SVM' was applied on the dataset.

The question pairs in the dataset were represented using 2 types of word embeddings, Bag of Words and Word2Vec. The representations of 'question1' and 'question2' were concatenated. The concatenated representations from train and test dataset along with their corresponding labels (0/1) are used for training and testing.

The dataset size along with the text representation is large with size of the training data being 323,481 and its dimension being 600. A major challenge faced while training the SVM model is the large size of the training dataset. SVM doesn't effectively scale to large data sizes as it requires loading all the data into memory at once and/or due to the nonlinear computation time.

To alleviate this issue, SVM was implemented with stochastic gradient descent (SGD) [10] learning. This allows mini-batch learning through the '$partial\_fit()$' method. The '$partial\_fit()$' method performs one epoch of stochastic gradient descent on the given set of samples. This way, the large sized dataset can be dealt with effectively by breaking up the data into smaller chunks and processing them sequentially. After processing each chunk of data, it is thrown out of memory and the next chunk is loaded into memory. Thus, the memory requirement for implementing SVM boils down to the size of just one chunk instead of the entire data set.

In this execution of SVM with SGD learning, chunk size was chosen to be equal to 1000. L_2 regularization was used with regularization parameter, $\alpha = 0.0001$ in order to prevent the model from overfitting and improve its generalizing ability. The classifier was trained using hinge loss as the loss function with penalty parameter C set as $C = 1/\alpha$ =10000, which penalizes misclassifications. The gradient of the loss was estimated for each sample. Throughout the training, the model was updated with a decreasing strength or learning rate. The learning rate η was updated as,

$$\eta = 1.0/(\alpha * (t + t_0))$$

where t_0 is chosen by a heuristic proposed by Leon Bottou.

4.5 BERT

BERT [5] is a pre-trained transformer model, which makes use of a self-attention mechanism where attention for each word in the sentence is computed. Unlike Word2Vec, where the representation of a word is a fixed vector, in BERT, the representation is based on the context around a word. BERT is pre-trained using Book Corpus and Wikipedia datasets. This model is trained using masked language modeling where a Transformer encoder is trained to predict masked words in a sentence. 15% of the words in a corpus are masked after training.

For training with this dataset, each question in the pair is separated by a [SEP] token. The BERT representation of this pair is passed to the MLP (Multilayer perceptron) layer to get the logits for classification. To avoid catastrophic forgetting, the parameters of the BERT model is not finetuned, but only the MLP layers are finetuned. The probability of each class is calculated using softmax function. Figure 3 shows how a question pair is given as input to get their corresponding representation at the output.

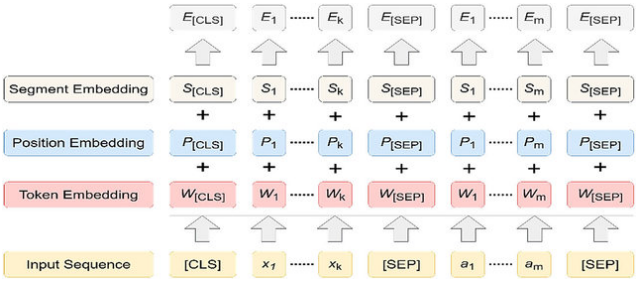

Figure 3: BERT representation for sentence pair input

5 Evaluation & Results

The dataset was split into train and test data in a ratio of 80:20. In order to find a more generalized performance evaluation, we used stratified k-fold cross-validation technique for models such as Naive Bayes, SVM, and XGBoost. 5-fold cross-validation was used as it ensures an 80:20 split in each fold. Stratification was used to ensure that the ratio of positive and negative classes remains the same in both training and test data. Due to limitations on resources cross-validation was not implemented for BERT.

5.1 Experiment Results

From the table 4, it can be observed that XGBoost along with Bag of Words representation, performs better in terms of F1-score than any other model implemented with other semantic representations. This is in contrast to [12] which proposes that the continuous space representation(semantics-based representation) of words outperforms the lexical representation.

Table 4: Performance of classification models

Model	Text Representation	Precision	Recall	F1 score	Accuracy%	AUC	5-fold Cross Validation Accuracy%
SVM	Word2Vec	0.65	0.63	0.64	68	0.68	70.34
SVM	Bag of Words	0.71	0.68	0.69	73	0.76	74.54
Naive Bayes	TF-IDF	0.76	0.69	0.7	75	0.82	75.05
Naive Bayes	Bag of Words	0.74	0.73	0.73	75	0.82	74.68
XGBoost	Word2Vec	0.75	0.79	0.76	69	0.67	69.98
XGBoost	**Bag of Words**	**0.82**	**0.89**	**0.86**	**81**	**0.79**	**78.38**
BERT	Pretrained	0.69	0.81	0.75	70	0.786	N/A

5.2 Performance Metrics

The following performance metrics were used to evaluate the performance of different models.

- Accuracy - is the ratio of correct predictions to the total number of samples within test data
- Precision - is the ratio of true positives to the total number of positive predictions.
- Recall - known as true positive rate, it is the ratio of true positives to the number of relevant cases
- F1 score - it is the harmonic mean of precision and recall.

5.3 ROC Curve

Below Figures [4-11] depicting the ROC curves for the four machine learning classifiers provide a visual comparison of their performance in terms of sensitivity and specificity. Each curve illustrates the trade-off between the true positive rate and false positive rate.

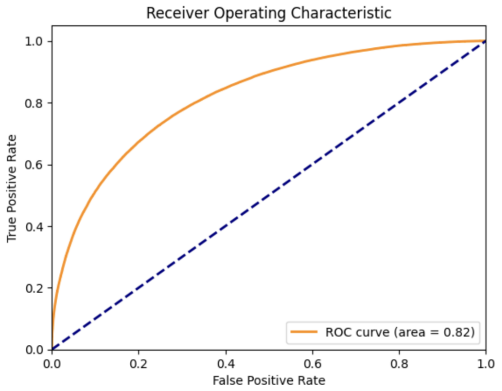

Figure 4: ROC Curve: Naive Bayes Bag of Words

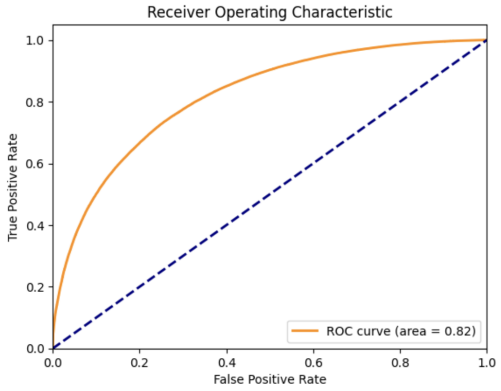

Figure 5: ROC Curve: Naive Bayes TF-IDF

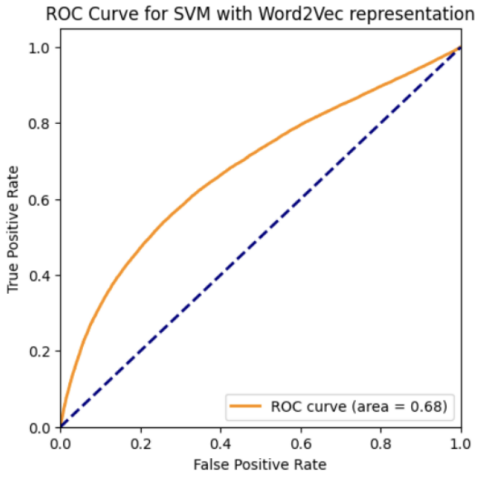

Figure 6: ROC Curve: SVM Bag of Words

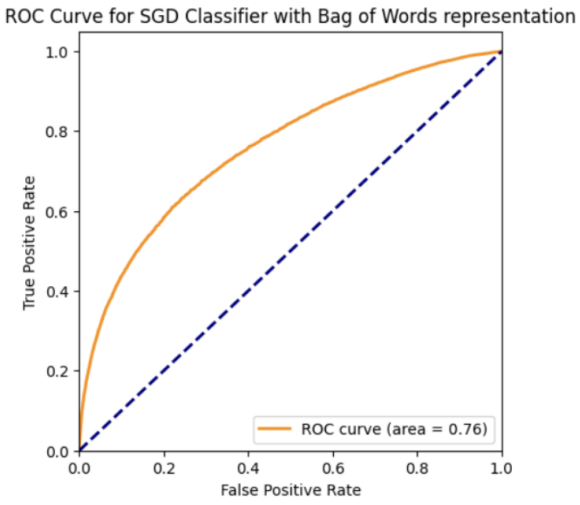

Figure 7: ROC Curve: SVM Word2Vec

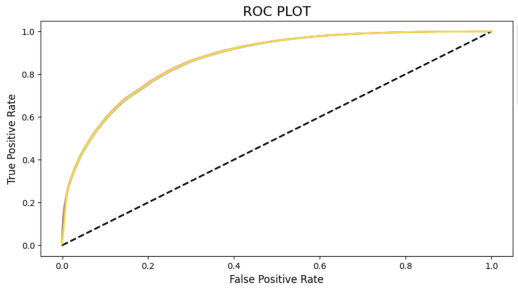

Figure 8: ROC Curve: XGBoost Bag of Words

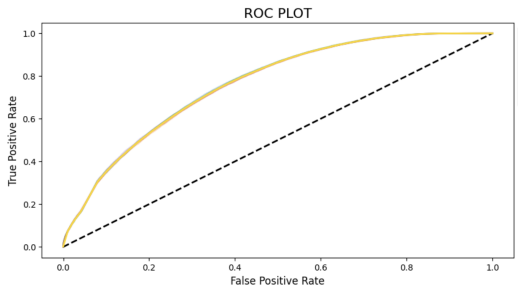

Figure 9: ROC Curve: XGBoost Word2Vec

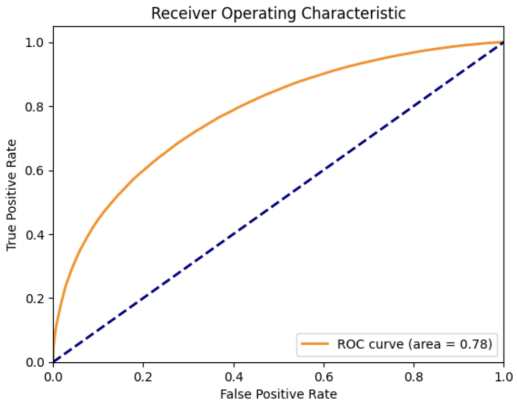

Figure 10: ROC Curve: Naive Bayes Cross Validation

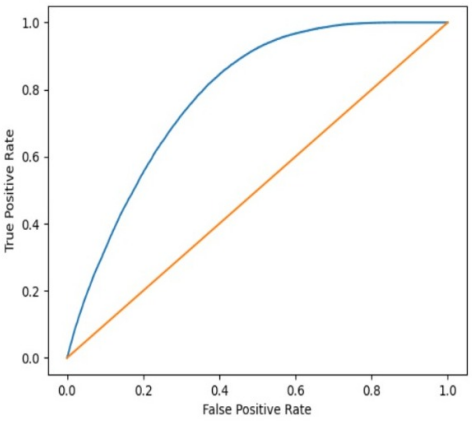

Figure 11: ROC Curve: BERT

5.4 Performance Analysis

5.4.1 Lexical representation vs Semantic representation

We were able to find out that the Bag of Words representation was able to give better results compared to other representations. This shows that for this dataset, lexical representation is better than semantic representation such as Word2Vec or BERT. This further highlights the significance of using lexical representations despite the availability of dense representations that encode the semantics of words in vector space[9].

5.4.2 Comparison of models

It is observed that XGBoost along with BoW representation outperformed other models. BoW, despite being a relatively basic form of text representation that doesn't capture semantic relationships or word order, can still be powerful when used with sophisticated algorithms like XGBoost. This might be attributed to XGBoost's ability to handle a large number of features effectively and its robustness to the noise and variance in the data. This shows the significance of assessing the performance of traditional ML methods for not only structured data such as tabular data but also for unstructured data such as text, images, videos, etc.

6 Conclusion

To study the performance of traditional ML techniques such as SVM, XGBoost and Naive Bayes on unstructured data such as text, experiments were conducted using these machine learning models with different representation techniques such as TF-IDF, Bag of Words, and Word2Vec. Further, in order to assess their performance in comparison to a deep learning model, experiments were conducted with BERT as well. Text preprocessing was done to remove noise and reduce the feature space. The performances were compared using different evaluation metrics such as accuracy, precision, recall, F1 score, ROC-curves and Area Under Curve.

The key takeaway from this project includes optimizing the training of a model for a large dataset by reducing the feature space and batch training, which effectively reduced the memory requirement and running time. Moreover, it is clearly understood that deep learning techniques should not be brute forced on every ML use case, but traditional ML techniques should also be experimented with as they can outperform these advanced models in some cases.

Moreover, extending this experiment to advanced models such as GPT[1] with high-end resources could be a better future scope.

References

[1] Tom B. Brown et al. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL https://arxiv.org/abs/2005.14165.

[2] Mainak Chandra et al. An enhanced deep learning model for duplicate question detection on quora question pairs using siamese lstm. In *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, pages 1–5. IEEE, 2022.

[3] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[4] DataCanary et al. Quora question pairs, 2017. URL https://kaggle.com/competitions/quora-question-pairs.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

[6] M.A. Hearst et al. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998. doi: 10.1109/5254.708428.

[7] Zainab Imtiaz et al. Duplicate questions pair detection using siamese malstm. *IEEE Access*, 8: 21932–21942, 2020.

[8] Damar Adi Prabowo et al. Duplicate question detection in question answer website using convolutional neural network. In *2019 5th International conference on science and technology (ICST)*, volume 1, pages 1–6. IEEE, 2019.

[9] Guilherme Moraes Rosa et al. Yes, BM25 is a strong baseline for legal case retrieval. *CoRR*, abs/2105.05686, 2021. URL https://arxiv.org/abs/2105.05686.

[10] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[11] Theodore Schleifer. Yes, quora still exists, and it's now worth $2 billion, May 2019. URL https://www.vox.com/recode/2019/5/16/18627157/quora-value-billion-question-answer.

[12] Holger Schwenk. Continuous space language models. *Computer Speech & Language*, 21(3): 492–518, 2007.