IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Vol. 12, Issue 3, March 2023

DOI: 10.17148/IJARCCE.2023.12356

Predicting Question Pair Similarity on Quora with Machine Learning Algorithms

Praveen Kumar B[1], Dr. Srikanth V[2]

Student, MCA, Jain (Deemed-to-be-University), Bengaluru, India[1]

Professor, MCA, Jain (Deemed-to-be-University), Bengaluru, India[2]

Abstract: Quora question pair similarity is a common problem in natural language processing that can be tackled using machine learning techniques. The goal is to identify whether two questions on Quora are similar or not, based on their semantic meaning. One approach to this problem is to use a neural network-based model, such as a Siamese network, which can learn a feature representation of each question and then compute a similarity score between them.

Another approach is to use traditional machine learning algorithms, such as logistic regression or support vector machines, to classify pairs of questions as either similar or not. To train these models, you will need a large dataset of question pairs labelled as either similar or not. You can obtain such a dataset from Quora itself, by extracting pairs of questions that have been marked as duplicates by Quora users. Once you have trained and validated your model, you can use it to predict the similarity between new pairs of questions on Quora, which can help improve the user experience and ensure that high-quality answers are provided to users.

Keywords:

- log-loss
- Binary Confusion Matrix

I. INTRODUCTION

Quora is a popular platform for general-purpose question and answer (Q&A) that is similar to Stack Overflow. However, one of the major issues faced by Quora is the duplication of questions, which can negatively impact the experience of both the asker and the answerer. This is because duplicate questions waste the time of both parties; the asker could have easily found the answer to their question had they known it was already asked before, and the answerer has to repeat themselves for essentially the same question.

For instance, consider two questions: "How can I become a good geologist?" and "What do I need to do to become a great geologist?" Although the wording of the questions may differ, their intention is the same, and therefore, the answers will be the same. In this case, showing the answers to the first question to the asker of the second question would be more efficient and save time for both parties.

To address this problem, Kaggle has launched a competition to develop a machine learning (ML) model that can determine whether two questions are similar or not. The goal is to improve the user experience on Quora by identifying duplicate questions and providing answers to the original question to prevent the need for repeating answers.

Our objective in this project is to develop an ML model that can accurately classify whether two questions are duplicates or not. There is no strict requirement for latency, which means that the model can take its time to make a prediction. However, we would like to have interpretability, which means that we can understand how the model arrived at its decision. Although interpretability is not mandatory, it would help us to understand how the model is making decisions and to debug any issues.

In this problem, a moderate cost is associated with misclassification, indicating the need for high accuracy to minimize incorrect classifications. Both classes, i.e., duplicate and not duplicate, carry equal significance, thus calling for a balanced accuracy score as the objective.

In summary, our task is to develop an ML model that can identify duplicate questions on Quora to improve the user experience. We will aim for high accuracy, interpretability, and a balanced accuracy score to minimize mis classifications. Cloud providers generate income through service charges for the requests made by their users.

DOI: 10.17148/IJARCCE.2023.12356

When determining the per-request charge, it is essential to consider server selection and request allocation strategies, as they not only affect the cloud provider's profits but also impact the appeal of their services to potential users. Providing excess computing capacity will result in energy waste and reduce profit, while providing inadequate capacity or inefficient request allocation may lead to dissatisfied users. Rational users will choose a service that maximizes their net reward, which is the benefit received minus the payment. The urgency of tasks also affects a user's utility, meaning that completing tasks quickly generate more utility. However, it is not feasible for a cloud provider to provide enough computing resources to complete all requests in a short period due to energy and economic reasons.

Thus, users must configure their requests at different time slots to optimize their utility. Request arrivals can be approximated as a Poisson process since they are submitted randomly. The behaviour of users can be analysed as a strategic game since their decisions affect each other's payment and time efficiency. In this project, a new service mechanism will be designed to optimize the profits of the cloud provider and its multiple users. To design a new service mechanism that optimizes the profits of both the cloud provider and its multiple users, several factors must be considered. One critical aspect is to balance the computing capacity provided by the cloud provider and the number of requests from its users.

The cloud provider must ensure that the computing resources are sufficient to handle the incoming requests while avoiding over-provisioning to reduce energy waste and increase profit. Another factor to consider is the allocation of requests to servers. The cloud provider should ensure that requests are assigned to servers that can handle them most efficiently to reduce the overall response time and increase user satisfaction. The allocation strategy should be optimized to maximize the utility of each user while minimizing overall energy consumption. Furthermore, the pricing mechanism should be designed to incentivize users to submit requests during off-peak hours when the computing capacity is underutilized.

This will allow the cloud provider to optimize its resource utilization and increase profit while reducing the per-request charge for users. In contrast, users who submit requests during peak hours when computing resources are scarce should be charged a higher per-request rate. To incentivize users to submit requests during off-peak hours, the cloud provider could offer discounted rates, rewards, or credits for users who submit requests during these periods. Additionally, the cloud provider could employ a queueing mechanism that prioritizes requests submitted during off-peak hours, allowing them to be processed faster than those submitted during peak hours.

Finally, the cloud provider should provide transparent and real-time information about the status of the user's requests and the availability of computing resources. This information will enable users to make informed decisions about when to submit their requests and how much to pay for them, increasing their trust in the service and satisfaction. in conclusion, the design of a new service mechanism that optimizes the profits of both the cloud provider and its multiple users requires a balance between computing capacity, request allocation strategy, pricing mechanism, and transparent communication. By optimizing these factors, the cloud provider can increase its profit, reduce energy waste, and satisfy its users, leading to a win-win situation for both parties.

II. LITERATURE REVIEW

M. Daoud (2017) [1] utilized Weka 3.8 software and the Random Forest algorithm with 10-fold cross-validation to determine question similarity using an Arabic language dataset. However, the accuracy of the model was relatively low.

I. L. Cherif and A. Kortebi (2018) [2] applied several machine learning algorithms, including K-Nearest Neighbors, Naive Bayes, and decision trees such as C5.0, C4.5, and XGBoost, on data from a major French ISP network in 2015. However, they did not consider training time.

X. Dong, T. Lei, S. Jin, and Z. Hou (2018) [3] used the XGBoost algorithm to determine question similarity using traffic flow detector data collected in Beijing. Their research suggested that XGBoost could significantly improve precision and accuracy.

C. Saedi, J. Rodrigues, J. Silva, A. Branco, and V. Maraev (2018) [4] employed a rule-based approach, Support Vector Machines, and Deep Convolutional Neural Network on a real-time dataset uploaded by Quora on Kaggle to determine question similarity. However, this approach was not satisfactory for smaller datasets.

Shashi Shankar (2018) [5] implemented the Support Vector Classifier model on the Quora dataset but achieved relatively lower accuracy. This work was deemed more suitable for evaluating short answers.

DOI: 10.17148/IJARCCE.2023.12356

B. Ye, G. Feng, A. Cui, and M. Li (2018) [6] developed an RNN encoder-decoder algorithm specifically for the Chinese language. They constructed a dataset containing 4,322 labeled question pairs to determine question similarity.

Q. Mahmood, M. A. Qadir, and M. T. Afzal (2019) [7] utilized SPARQL queries and Social Network Analysis on the Citeseer dataset to determine document similarity. However, this technique was deemed less suitable for determining question pair similarity due to the requirement for more detailed analysis.

J. Wang, Z. Li, and B. Hu (2020) [8] determined question similarity by using the semantic context of existing Q/A pairs, cosine distance, and question similarity based on their answers. Their research was conducted on a community-based question and answer service on the web, but the approach had lower accuracy compared to other techniques.

III. RESEARCH METHODOLOGY

The methodology for determining the similarity between Quora question pairs using machine learning can be divided into several steps:

Data Collection: The first step involves collecting a large dataset of question pairs from Quora. This dataset should be diverse and representative of the types of questions that users ask on the platform.

Data Cleaning: The collected dataset needs to be cleaned by removing duplicates, irrelevant information, and noise. The text should be pre-processed by removing stop words, punctuation, and converting all text to lowercase.

Feature Extraction: The next step is to extract features from the pre-processed text. The features can be either handcrafted or learned using deep learning techniques. Handcrafted features may include word overlap, n-grams, or sentence length. On the other hand, deep learning-based approaches can use neural networks, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), to extract high-level features automatically.

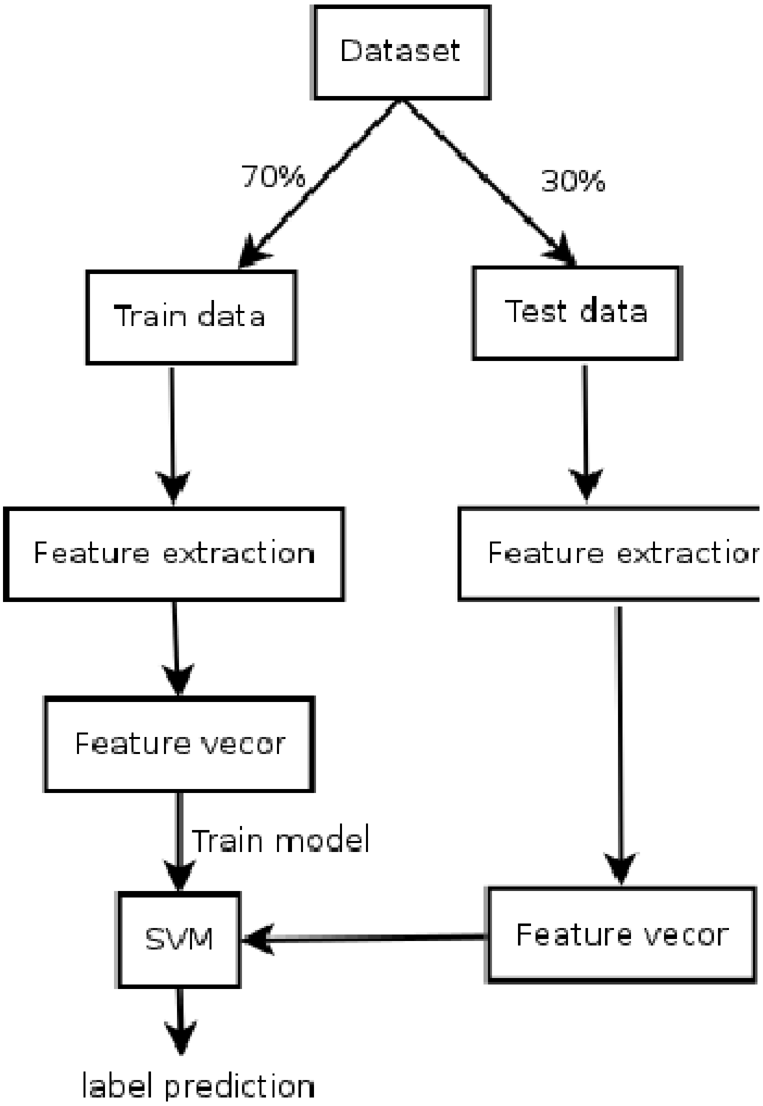
Model Selection: After feature extraction, a suitable machine learning model needs to be selected to learn the similarity between question pairs. The most commonly used models for this task are Support Vector Machines (SVMs), Random Forests, and Gradient Boosting Machines (GBMs). Recently, deep learning models like Siamese Networks, Transformers and BERT have shown excellent results for this task.

Model Training: The selected model is then trained on the pre-processed dataset. The training involves feeding the extracted features into the model and adjusting its parameters to minimize the loss function. The quality of the model's output is evaluated using a validation set, and the process is repeated until the best possible performance is achieved.

Model Evaluation: The final step involves evaluating the performance of the trained model on a separate test set. The performance can be measured using standard metrics such as accuracy, precision, recall, and F1-score.

In conclusion, to determine the similarity between Quora question pairs using machine learning, data collection and cleaning, feature extraction, model selection, model training, and model evaluation are critical steps in the research methodology.

DOI: 10.17148/IJARCCE.2023.12356



[Fig. 1 : Model Workflow]

IV . CONCLUSION

The project proposes a service mechanism for the profit maximization of cloud providers. The mechanism incorporates game theory and utility functions to model the interactions between the cloud provider and its users, allowing for the optimization of both economic and time-related benefits. A controlling parameter can be used to approximate the server selection space.

The proposed iterative algorithm provides a practical and reliable way to implement the service mechanism in a real-world cloud environment. The results of the numerical calculations conducted to verify the theoretical analyses show the effectiveness and reliability of the algorithm in predicting the behaviour of the cloud provider and its users. An effective and comprehensive approach to profit maximization for cloud providers and their multiple users is a valuable contribution to the field of cloud computing.

VI. REFERENCES

[1] Imtiaz Z, Umer M, Ahmad M, Ullah S, Choi GS, Mehmood A. Duplicate Questions Pair Detection Using Siamese MaLSTM. IEEE Access. 2020;8:21932–21942.

[2] Saedi C, Rodrigues J, Silva J, Branco A, Maraev V. Learning Profiles in Duplicate Question Detection. 2017 IEEE

DOI: 10.17148/IJARCCE.2023.12356

International Conference on Information Reuse and Integration (IRI). 2017;p. 544–550.

[3] Xu Z, Yuan H. Forum Duplicate Question Detection by Domain Adaptive Semantic Matching. IEEE Access. 2020;8:56029–56038.

[4] Wang L, Zhang L, Jiang J. Duplicate Question Detection With Deep Learning in Stack Overflow. IEEE Access. 2020;8:25964–25975.

[5] Prabowo DA, Budi G, Herwanto. Duplicate Question Detection in Question Answer Website using Convolutional Neural Network. 2019 5th International Conference on Science and Technology. 2019;p. 1–6.

[6] Mukherjee S, Kumar NS. Duplicate Question Management and Answer Verification System. 2019 IEEE Tenth International Conference on Technology for Education (T4E). 2019;p. 266–267.

[7] Daoud M. Novel Approach towards Arabic Question Similarity Detection. 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS). 2019;p. 1–6.

[8] Ye B, Feng G, Cui A, Li M. Learning Question Similarity with Recurrent Neural Networks. 2017 IEEE International Conference on Big Knowledge (ICBK. 2017;p. 111–118.

[9] Shankar S. Identifying Quora question pairs having the same intent. 2017.