Query Answerability Classifier for Direct Answer Module in Web Search Engines

Yiran Wang

Delft University of Technology
Delft, South Holland, Netherlands

ABSTRACT

In order determine when we can show direct answer module to user queries in web search engine, an independent classifier is designed in this study to assess the answerability of each user query. Real user queries are sampled from MS MARCO Question Answering and Natural Langauge Generation dataset [1] and manually labelled with query answerability to train and evaluate the classifier. As a result, the XGboost model has an overall better performance than the random forest model with prediction accuracy score 0.83 and F1 score 0.89. Once the classifier determines the user query is answerable, a MRC model may be used to find the direct answer within provided passages. Else, no direct answer shall be provided to this query.

ACM Reference Format:
Yiran Wang. 2021. Query Answerability Classifier for Direct Answer Module in Web Search Engines. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A recent research [5] shows that providing direct answers in Search Engine Result Page (SERP) can significantly increase user engagement and user satisfaction while reduces their efforts during the search process. An example of a direct answer module given to a user query in SERP on Google can be found in Figure 1.

However, when a direct answer should be given to a user's query remains a question. Since search engine may want to provide direct answers to queries as much as possible to ease users' search efforts whereas some queries simply do not have a single direct answer. Not all the user queries are answerable or reasonable to be supplied with a direct answer in the first place. For example, no one can provide a direct answer to queries like "What is the meaning of life?". Existing Question Answering (QA) datasets [1, 3] have incorporated the cases that a query does not have direct answers found in provided passages. If no direct answer found in provided passages, then no direct answer module should be given to query. However, such criteria for query answerability is primarily dependent on the provided passsages. Machine reading comprehension (MRC) is the ability to read up a piece of text and then find its answers in other passages. Few top performancing MRC models of these QA datasets take the quality of the query itself into account.

Therefore, the main research question in this study is: How to determine the answerability of a query? To be more specific, this study only involves queries and direct answers which are only in text rather than tables, images or other media.

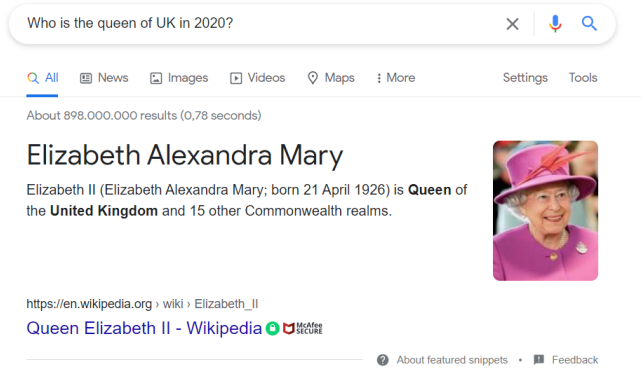

Figure 1: An example of Google's featured snippets (i.e. direct answer module) given to the query "Who is the queen of UK in 2020?".

In this research, an independent classifier is designed to assess the answerability of each user query. 500 Randomly selected queries from MS MARCO Question Answering and Natural Langauge Generation dataset dataset [1] with manually labelled query answerability by the conductor of this study are used to train and evaluate the answerability classifier. Once the classifier determines the user query is answerable, a MRC model may be used to find the direct answer within provided passages. Else, no direct answer shall be provided to this query.

2 RELATED RESEARCHES

2.1 Question Answering Dataset

There are plenty of QA datasets existing[1? ? ?], which contain either manually-generated questions or real user queries collected from search engines, related passages or documents, and answers to the questions or queries found in the related passages or documents. Among which, some datasets for example MS MARCO Question Answering (MS MARCO QnA) dataset [1], contain questions or queries which can not be offered with a answer based on the given passages or documents.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

MS MARCO QnA dataset is a large-scale corpus[1], which collects more than 1 million real user queries sampled from Bing's search query logs and more than 8 million passages extracted from the web documents retrieved from Bing. Within the dataset, each query is with a manually generated text-based answer according to the provided passages. An answer to a query being empty indicates no answer is found in the provided passages. The purpose of this dataset was for developing QA systems which can select one passage out of 10 provided passages based on a query and further find the answer to the query in the selected passage. More labels need to be manually added to this corpus if we want to idenMtify whether a query itself is not suitable or able to be offered with a direct answer.

3 METHODOLOGY

We want to design a specialized classifier which determines the answerability of query itself, independently on specific resources used to find the answer or any MRC model. In order words, the query answerability classifier can be combined with any MRC as an additional step to select answerable queries and throw away unanswerable ones.

An expected scenario for using this classifier is illustrated in Figure 2: User queries and the top retrieved documents to the queries returned by a web search engine are served as inputs to the answerability classifier. The output is whether query itself is answerable. Only after queries are identified as answerable, the queries would be served as an input to a MRC model, which finds a best direct answer from the top retrieved documents. However, no direct answer would be provided to the user's queries classified as unanswerable.

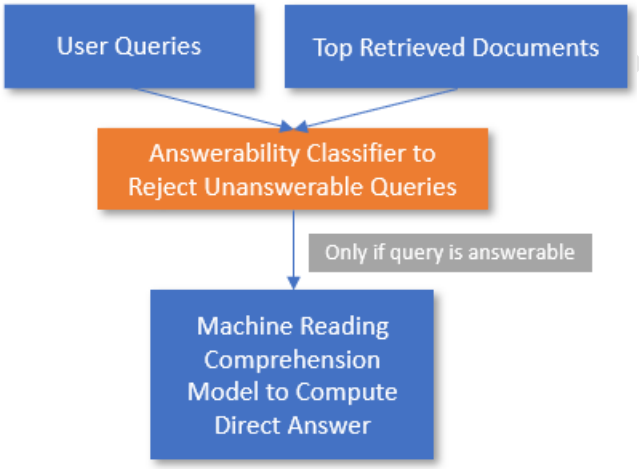

Figure 2: The pipeline of direct answer module

3.1 MS MARCO as the Dataset

Training and testing the query answerability classifier requires a text-based corpus, which contains real user queries and should aim to find answers in a variety of web resources (e.g. personal websites, government website, Wikipedia and so on) rather than specializing in a single resource. MS MARCO QnA dataset suffices all the requirements above. It is also easy to compute on the queries within the dataset. Therefore, the MS MARCO QnA dataset is chosen in this study to develop the query answerability classifier. An example from the dataset is shown below.

```
{"passages": [{"is selected": 0, "url": "https://
 en.wikipedia.org/wiki/Reserve_Bank_of_Australia",
 "passage_text": "Since 2007, the RBA's outstanding
 reputation has been affected by the 'Securency' or
 NPA scandal..."}, ...],
 "query_id": 19699,
 "answers": ["Results-Based Accountability is a
 disciplined way of thinking and taking action
 that communities can use to improve the lives of
 children, youth, families, adults and the community
 as a whole."],
 "query_type": "description",
 "query": "what is rba"}"
```

3.2 Define Answerability

While setting up the criteria of query answerability, I first scanned through 50 MS MARCO queries to get used to the real user queries and then come up the standard of answerability labelling, as below.

A query is answerable if and only if all the criteria below are met; otherwise, the query is unanswerable:

(1) Although the query may contain minor grammar mistakes or does not form a grammatically correct sentence, the query is human-comprehensible that an answer to the query can be given.
(2) The query contains sufficient details/information to let its direct answer be the answer this user intends to know.
(3) The answer to this question is generalizable.
(4) The query only asks for a text-based answer. It does not require answers to be other mediums than texts (such as audio or images).

As the example queries shown in Table 1, "is dopamine addictive" and "felsic definition" are labelled as answerable. Although "felsic definition" is neither a natural language sentence nor grammatically correct, it does not affect human understanding its meaning or knowing the intended answer (i.e. the definition of felsic) the user wants to get. Therefore, "felsic definition" is labelled as answerable as well. However, the following query "what media is your artwork made from what does that mean" contains grammar mistakes and lets people having trouble understanding the meaning behind it. Such query is labelled as unanswerable. In addition, "cost to mail letter to usa" is unanswerable without knowing the departure place of this letter, and "how many representatives does oklahoma have" is unanswerable without specifying what kind of representative this user indicates. It is not very useful to provide direct answers to alike queries in real practice since they lack the crucial details to be offered with reasonable answers or the answers users want to know. For the last example query "images of how phones have changed over time", it directly asks for an image to be the answer and this is supported by our classifier.

Table 1: Some Example Queries with Answerability Labels

Query	Answerability	Reason
is dopamine addictive	Answerable	Suffice all criteria
felsic definition	Answerable	Suffice all criteria
what media is your artwork made from what does that mean	Unanswerable	Violate criterion (1)
how many representatives does oklahoma have	Unanswerable	Violate criterion (2)
cost to mail letter to usa	Unanswerable	Violate criterion (2) and (3)
images of how phones have changed over time	Unanswerable	Violate criterion (4)

3.3 Classifier Training and Evaluation

Determining query answerability is a text classification task, which is to assign one or more class labels from a predefined set of labels to a document according to its content. In our case, the class is a binary class, answerability, and the predefined labels are answerable and unanswerable, only one of which is assigned to each query.

Features used for developing the query answerability classifier should be independent on the type of resources used to find direct answers. Therefore, there are 3 different categories of features:

(1) Query dependent: the features are solely dependent on the context of the query itself;
(2) Query-corpus dependent: the features are computed based on the query itself and all the queries in the MS MARCO QnA v1.1 training dataset (i.e query-corpus);
(3) Top-retrieved-passages dependent: the features are computed based on the query itself and top 9 passages returned from Bing Search Engine with the query as an input.

We first apply lemmatization to preprocess the queries and related corpus. Since if without lemmatization, during the tasks of counting term frequency in several documents (such as, computing features 9 and 10), words such as, "is" and "are" or "car" and "cars", have the same meaning, however, shown in different forms due to English grammar, would be counted as different words and this is not what we want. Applying lemmatization can remove these inflectional endings and return the base or dictionary form of a word **??**. In this case, "is" and "are" are both replaced by "be", as well as "car" and "cars" are both returned as "car". This would resolve the problem of words with the same meaning counted as different words.

A full list of features are shown in Table 2, with feature names, feature types and the specific reason to include each feature. To avoid duplication with the reasons explained in the in Table 2, two more concepts GloVe and tf-idf are elaborated further. The first feature is a 300 dimension Global Vectors for Word Representation (GloVe) [2]. It transforms each word into a semantic vector in the 300-dimension coordinate system, where words with similar meanings have a closer mathematical distance between each other in the 300-dimension coordinate system than words with contrasting meanings. For example, in GloVe, the words that have the closest distance to word "frog" are "frogs", "toad", "litoria", whereas vector differences between "man" and "woman" and "king" and "queen" are roughly the same. Applying GloVe enable us capture the relationship between different words in each query. The 9th and 10th features make use of term frequency - inverse document frequency to represent each word/n-gram in a query, while n-gram is a multiple of words. Since queries may contain different number of or n-grams, we instead average the tf-idf values across the words and n-grams in each query.

Due to the limit of time which can be assigned to manually labelling queries, which results in a limited number of sample size, I think classical machine learning classifiers are more suitable to be applied in this case than deep learning models. Two different machine learning classifiers, Random Forest and XGBoost are used to compare the performance.

Furthermore, the chosen evaluation metrics are prediction accuracy, precision, recall, and f1 score. During the process of labelling, I do notice that the training and testing datasets do not have a perfectly balanced class in terms of answerability and the metrics of precision, recall and f1 score can reveal more where the classification goes wrong than simply providing a prediction accuracy score.

4 EXPERIMENTS AND RESULTS

4.1 Training and Testing Datasets

This study uses queries from the training and evaluation queries of the MS MARCO original QnA (v1.1) dataset. 300 queries from its training set were randomly selected to serve the purpose of training the query answerability classifier, whereas 200 queries were randomly selected from its evaluation set for query answerability classifier evaluation. The 300 queries and the 200 queries are referred as the training dataset and the testing dataset in the sections below. I manually labelled 500 queries in total as answerable (as 1) or not answerable (as 0) with the criterion mentioned in the Methodology section. It took around 4 and half hours to complete the entire labelling. The labelled training and testing datasets can be found on this study's GitHub page (https://github.com/Yiranluc/Direct-Answer-Module-for-SearchX/tree/main).

As a result, in the training dataset, there are 251 queries labelled as answerable (83.7 percent), while 49 queries are labelled as unanswerable (16.3 percent). In the testing dataset, there are 149 queries labelled as answerable (74.0 percent), while 51 queries are labelled as unanswerable (26.0 percent). For both training and testing datasets, the answerability class is unbalanced.

Table 2: Features selected for answerability classifier

Category of Features	Features	Feature Type	Reason to include
Query dependent features	1.Word embeddings of the query with pre-trained GloVe	List of Numerical (d = 300)	1. To represent words within the query while capturing the relationship between words.
	2. Number of words within a query	Numerical	2. Too little of words may have an influence on answerability.
	3. Whether "what" presents in the query	Binary	
	4. Whether "when" presents in the query	Binary	3-8. Since queries are questions in essence,
	5. Whether "where" presents in the query	Binary	the words representing a question may have an influence
	6. Whether "why" presents in the query	Binary	on what type of questions a user
	7. Whether "who" presents in the query	Binary	is asking and how concrete the question might be.
	8. Whether "how" presents in the query	Binary	
Query-corpus dependent features	9. The average of word-level TF-IDF of each query with the corpus of MS MARCO training queries	Numerical	9-10. Since we want to know how important the words/N-grams with the query is regarding
	10. The average of N-gram level TF-IDF of each query with the corpus of all the passages within MS MARCO training dataset	Numerical	all the words in the training queries.
Top-retrieved-passages dependent features	11. Fractions of passages containing all the words in the query discarded with stopping words	Numerical	11. Too few passages containing all the keywords of the query may indicate that the query is hard to answer.

4.2 Feature Engineering

During the text preprocessing step, lemmatization is applied on the training and testing queries as well as the corpuses of queries, using the pretrained pipeline "en_core_web_sm" from spaCy library. Afterwards, each feature stated in the Table 2 is computed. For the GloVe embedding of words in each query, it is computed using the package "gensim" and the Common Crawl version "glove.42B.300d" with 42 bytes tokens and 300 dimensions for each word. The features related to TD-IDF representation are computed using "sklearn.feature_extraction" package. In addition, the top retrieved passages are retrieved by serving each query in the training and testing datasets to Bing Search API v7. Only the snippets for the top 9 web pages are included for feature computation.

4.3 Model Training and Hypertuning

RandomForestClassifier function from the sklearn.ensemble library was used to train and evaluate the random forest models, whereas the XGBClassifier from the xgboost library was applied to train and evaluate the XGBoost models. Random Forest hyperparameter tuning is conducted with RandomizedSearchCV function from sklearn.model_selection package with 3-fold cross validation. 100 random combinations of a list of parameters were conducted to find with the model with the highest predicting accuracy The process of XGBoost hyperparameter tuning is facilitated by HYPERPORT library, which searches through a space of values for hyperparatmeters and find the best combination of values that give the minimum of the loss function. It also makes use of 3-fold cross validation.

4.4 Model Performance

While without hyperparameter tuning, Random Forest model has a highest predicting accuracy 77% percent on the testing dataset, this predicting accuracy turns to 0.78 after hypertuning, as shown in Figure 3. With the default hyperparameters, the XGBoost model has a predicting accuracy 0.83, on the testing dataset and predicting accuracy remains unchanged after hyperparameter tuning.

As shown in Figure 3, the Random Forest model after hypertuning has a precision score 0.77, a perfect recall score 1.00, and f1 score 0.87, whereas the XGBoost model after hyperparameter tuning has a precision score 0.81, a nearly perfect recall score 0.99,

Table 3: Features selected for answerability classifier

Model	Prediction Accuracy	Precision	Recall	F_1
Random Forest after hypertuning	0.78	0.77	1.00	0.87
XGBoost after hypertuning	0.83	0.81	0.99	0.89

and f1 score 0.89. The confusion matrices of the Random Forest model after hypertuning and the XGBoost model after hypertuning are shown in Figure 3 and Figure 4, respectively.

The resulting best hyperparameters for Random Forest model is the following:

```
'n_estimators': 600,
'min_samples_split': 5,
'min_samples_leaf': 1,
'max_features': 'sqrt',
'max_depth': 60,
'bootstrap': False.
```

The resulting best hyperparameters for XGBoost model is the following:

```
'colsample_bytree': 0.5396320619564892,
'gamma': 4.989432442581639,
'max_depth': 6.0,
'min_child_weight': 8.0,
'reg_alpha': 88.0,
'reg_lambda': 0.11472989808982881.
```

5 RESPONSIBLE RESEARCH

In terms of research ethnics, the answerability classifier within this study was developed based on anonymized user queries within MS MARCO QnA dataset collected from Bing's search logs [1]. It does not contain any personal information of the Bing's search engine users. Therefore, this dataset we used not only entact the authenticity of real users' queries but also keep these users' privacy intact.

Regarding reproducibility of the research results, this study adheres to the 6 recommendations by e Yale Law School Roundtable on reproducible research in 2009 [4] to a computational scientist.

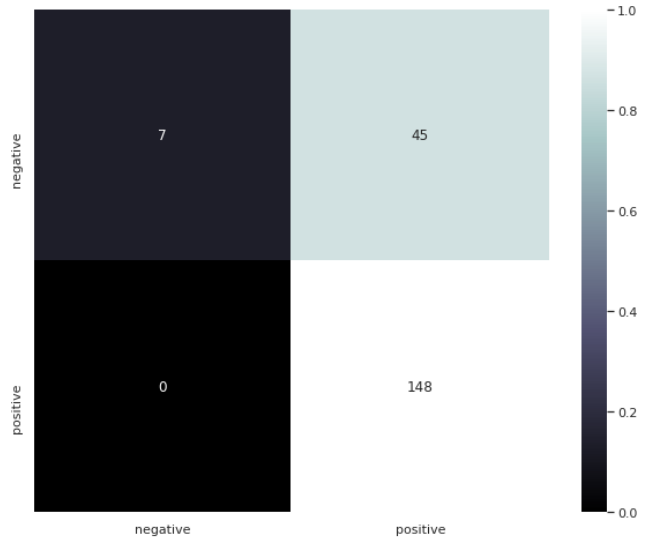

Figure 3: The Confusion Matrix of the Random Forest Model after Hyperparameter Tuning.

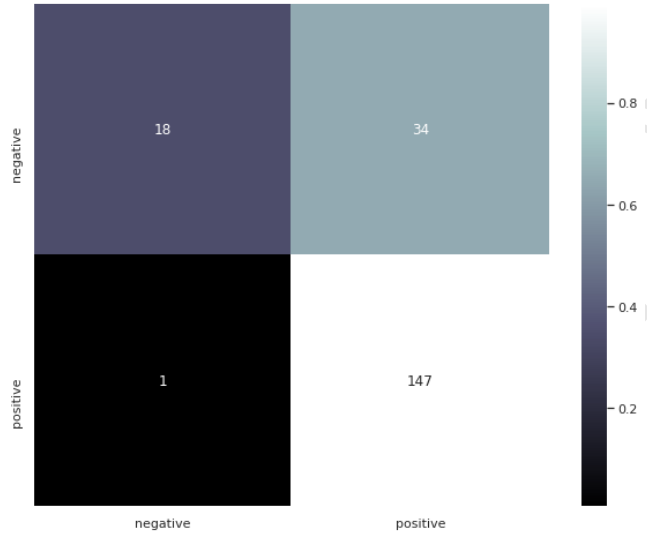

Figure 4: The Confusion Matrix of the XGBoost Model after Hyperparameter Tuning.

The involved datasets and code to train and evaluate the answerability classifier are uploaded to GitHub with a version control system and a open license. The ReadMe file in GitHub clearly states the computing environment and the software version used in this research.

6 DISCUSSION

The XGBoost model has a higher predicting accuracy, precision score and f1 score than the Random Forest model. The XGBoost model and the Random Forest model both have a perfect/nearly perfect recall score. Based on the results, the XBGoost model performs better than the Random Forest model. The extremely high recall scores in both model might be mainly due to the imbalance of labels in the training and testing datasets, since the number of answerable queries is nearly 4 times the number of unanswerable queries in the training dataset and the number of answerable queries is around 3 times the number of unanswerable queries in the testing dataset. If guessing all the queries in the testing dataset as answerable queries would already give us a 0.74 of predicting accuracy. A predict accuracy of 0.78 from the Random Forest model does not differ much than this number (0.74), while the predict accuracy (0.83) of the XBGoost model is better.

However, if we artifically select the queries to be balanced on answerability or manufacture the current datasets into balanced class, we might not have consistent estimates of the answerability as the sample size grows. In real life, we do expect the search engine can provide user with direct answer as much as possible and overestimating the number of answerable queries would not be very likely to result in a devastating consequences. Relatively low precision scores might be acceptable in this case.

The main point to improve in this study is the size of the training and testing datasets. Compared with the 10 thousands queries in the MS MARCO dataset, 500 queries might not be sufficient to represent the characteristics of the entire corpus. However, given the amount of time of this study, the amount of queries should suffice the purpose of this study.

7 CONCLUSIONS AND FUTURE WORK

As compared to random forest model, XGBoost performs better on distinguishing answerable queries from unanswerable ones. We can safely compute a direct answer using MRC model after the query answerability classifier determines that the query is answerable.

A future improvement of this work might be to extend this classifier to accept queries expecting non-text answers.

REFERENCES

[1] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv e-prints*, Article arXiv:1611.09268 (Nov. 2016), arXiv:1611.09268 pages. arXiv:1611.09268 [cs.CL]

[2] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. 14 (2014), 1532–1543.

[3] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. (2018). arXiv:1806.03822 [cs.CL]

[4] V.C. Stodden. 2010. Reproducible research: Addressing the need for data and code sharing in computational science. *Computing in Science and Engineering* 12 (01 2010), 8–13. https://doi.org/10.1109/MCSE.2010.113

[5] Zhijing Wu, Mark Sanderson, B. Barla Cambazoglu, W. Bruce Croft, and Falk Scholer. 2020. Providing Direct Answers in Search Results: A Study of User Behavior. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 1635–1644. https://doi.org/10.1145/3340531.3412017