The Piazza Problem: Enhancing Question Retrieval Methods via Prompting and Fine-Tuned BERT

Marcus Bluestone, Joanna Kondylis & Eli Scharf

Abstract

Piazza, a community question-answer site that allows students to post questions that can be answered by instructors and peers, faces a major challenge. Students in large classes often ask redundant questions without first searching the site for existing ones. This redundancy increases staff workload as they repeatedly answer similar questions. To address this issue, Piazza currently prompts users with a list of similar questions to their query before they post; however, this list is found via a naive keyword search, which is often ineffective at finding messages with similar context. This paper addresses this problem of Question Retrieval by exploring four methods for determining similar questions: word embeddings, sentence embeddings, a fine-tuned BERT model, and LLM prompting. We evaluate these approaches by comparing Two Question Similarity (TQS) and Database Lookup Accuracy (DLA). Our findings highlight the effectiveness of LLM Prompting and fine-tuned BERT in addressing the question retrieval task.

1. Introduction

Online forums have revolutionized the way users seek and share information, with community question-answering (CQA) platforms like Stack Exchange, Reddit, Piazza, and Quora providing a space where individuals can pose questions and receive answers from a global community. To help minimize the amount of redundant questions asked, these platforms provide users with tools to search for existing questions similar to their own queries. However, many of these tools employ primitive techniques, such a keyword or tag search, leading to often unsatisfactory results.

For instance, many universities, such as MIT, use a CQA service called Piazza, whose similar-question functionality works only via key-word search[1]. This primitive method often fails to find pre-existing questions that are similar to the new input question. For instance, the question "How old is the queen?" would find "What moves can the queen make in the chess?" to be similar since they share the word 'queen,' but it would fail to return questions like "What is the age of the British King's wife?" which is an exact semantic match to the original question. This inability to find truly similar past questions causes repetitive questions to accumulate over the course of the semester, causing community members to invest time answering questions that have already been resolved and adding to the difficulty in maintaining a large archive of data. To improve overall efficiency and facilitate better communication, it is imperative to establish more effective methods for quickly and accurately identifying similar questions.

This issue falls under the category of Question Retrieval (QR), and we analyze and propose new solutions to this problem (see Figure 1). More specifically, we investigate methods of determining if two questions are "similar" in the semantic sense. To do this, we employ both embedding methods – which capture the *semantic* meaning of words or sentences – and LLM prompting – which are assumed to posses a comprehensive understanding of the English language. Efficient and accurate solutions to this issue would immediately allow us to solve the Question Retrieval problems by applying the similarity model to a database of questions and choosing the most similar questions.

This paper contributes to the problem of question retrieval by proposing/implementing two new methods – LLM prompting and Expert BERT – while using a variety of past embedding model as baselines.

1. **Word Embeddings:** We use Bag of Word (BoW) models, along with TF-IDF transformations, and Continuous Bag of Words (CBoW), to create vector embeddings of each word. We then average these over all words in our sentences to give a pseudo-sentence embedding that captures semantic meaning of the whole sentence rather than just individual words. Finally, cosine similarity is applied between embedded sentences to determine how similar the content is.

2. **Sentence Embeddings:** We embed full sentences using transformer models, such as using an optimized BERT (e.g. RoBERTa [21]). BERT's embeddings capture complex semantic relationships between words in a sentence, and cosine similarity can be applied to determine similarity between different embeddings.

3. **Expert BERT:** We fine-tune BERT's on a specific category of questions (e.g. questions relating to Technology), allowing the embedding model to gain a more

[1] see https://support.piazza.com/ support/solutions/articles/48000616747-search-for-posts

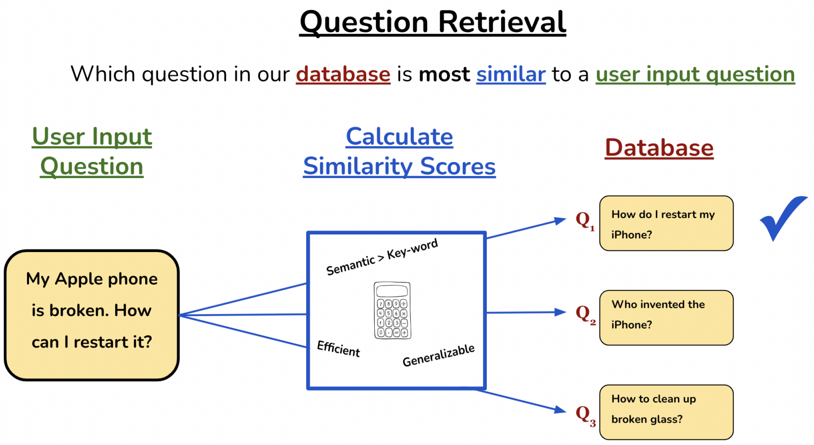
Question Retrieval
Which question in our database is most similar to a user input question
User Input Question
Calculate Similarity Scores
Database
My Apple phone is broken. How can I restart it?
Semantic > Key-word
Efficient
Generalizable
Q1
How do I restart my iPhone?
Q2
Who invented the iPhone?
Q3
How to clean up broken glass?

Figure 1. A user's input question is compared to all questions in a database. The most contextually similar question in the database is returned

nuanced and sophisticated understanding of the material. We hypothesize that this will enrich the embeddings and allow for more accurate answers. We experiment here both with applying cosine similarity to embeddings, and also adding a new classifier head to the BERT model.

4. **LLM Prompting:** We prompt ChatGPT-4o with pairs of questions and ask it to return whether the questions are similar in context or not. We leverage GPT's contextual understanding of language to identify pairs with high semantic similarity, even when they use different wording or terminology.

Although extensive work has been done on the first two approaches, there is a gap in the research when it comes to trying LLM Prompting and Expert BERT models in the context of Question Retrieval. We compare the results of all of these approaches on a dataset of Quora questions. We measure our results in terms of accuracy, Precision@K, and Mean Reciprocal Rank (MRR). See Section 3.4 for how exactly accuracy values are calculated. The code and data is available at: https://github.com/kondylisjg/NLP_Question_Retrieval.git

2. Related Work

The biggest issue with Question Retrieval is the gap between lexicographic (what words are being used) and semantic (what the sentence means) information contained within text [14]. For instance, the questions "How old is the queen?" and "For how many years has the king's wife lived?" contain different words, but have the same exact meaning. There are four broad categories of techniques that attempt to bridge this gap.

We present them in terms of increasing accuracy. Linguistic Enrichment methods seek to increase and enrich the information contained within a text (e.g. by adding synonyms to the words in the sentence). Machine translation techniques learn translation probabilities and implicit relationships between words in a text with the hopes of capturing semantic meaning. Categorical methods seek to utilize explicit or learned category information of a given question to improve accuracy. Finally, embedding methods achieve the highest accuracies by encoding the meaning of words and/or sentences as vectors in a high dimensional space.

2.1. Linguistic Enrichment

The first approach to bridging the lexicographic-semantic gap is to expand the language information contained within each question in order to allow more accurate comparisons of questions. These Question Expansion (QE) techniques enrich original sentences with additional words that are, for instance, synonyms and hypernyms of worlds in the original text [16]. More complex techniques which employ word embeddings to find other similar words to add also exist [13].

Other approaches don't directly add words to the original text, but rather extract language features from the text that are then fed into some other model. For instance, Duan. et al utilized Base Noun Phrases and WH-ngrams (i.e. a n-gram beginning with who, what, where, when, or why) in order to extract relevant information from a question database. [7].

There are two big issues with these approaches. First, these approaches take advantage of specific linguistic features of English and are therefore not as generalizable to other languages as other methods. In addition, QR systems using these approaches have failed to reach significant accuracy levels (Duan. et al's experiments had mAP less than

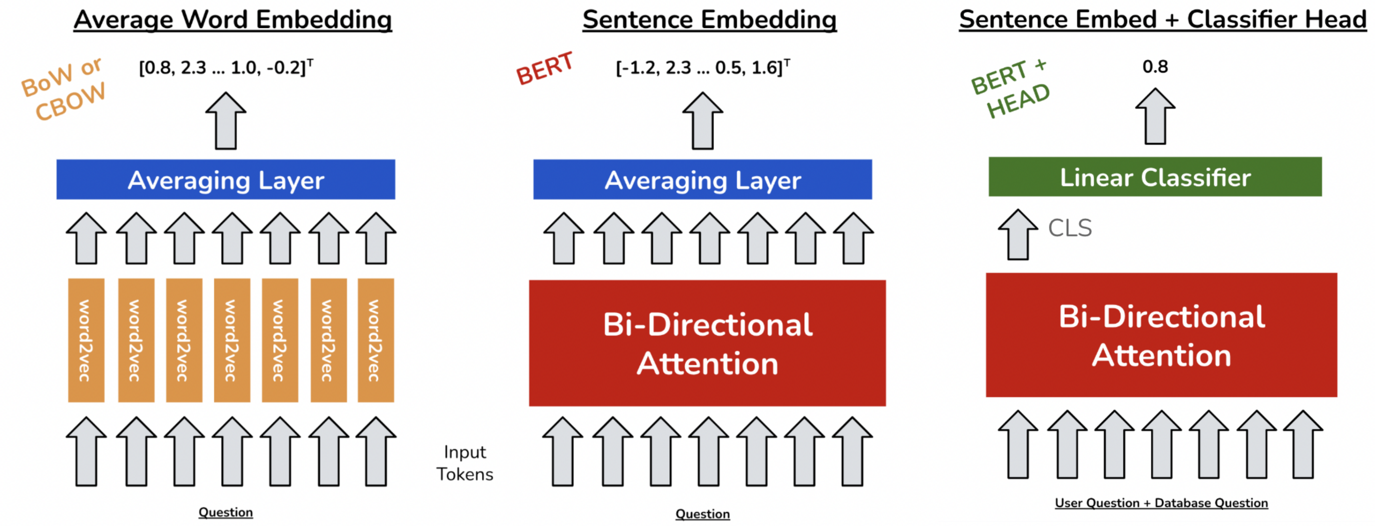

Figure 2. The different approaches used in this paper: average word embeddings, sentence embeddings, and sentence embeddings with a classifier head. Notice that the input to these model is a single question for the first two, and a pair of questions for the last.

30% across all trials). More complex systems are required for accurate results.

2.2. Machine Translation

Another approach to bridge the semantic and lexicographic gap is to train machine translation models to learn about the semantic interplay between the words in a given piece of text. A variety of works have achieved promising results in using training translation models by use question-answer pairs as parallel corpuses, and then using the implicit language model to determine if a new query question matches questions in a database [1, 8]. These approaches reached mAP's nearing 40% on the Wondir dataset[2] [19].

However, the issue with a machine translation approach is that it can often miss semantic information that is contained across a large number of words [14], and it relies on a well-annotated and high-quality database of answers which is often not the case [9].

2.3. Categorical Methods

Some work has also been done to use categorical information about the QA pairs (either learned or given as a label) to improve retrieval accuracy. Duan et al created a MDL tree cut model which summarizes questions and stores them in an efficient data structure based on the summaries [7]. At inference time, these summaries are used to help guide the process of finding a similar question.

A later approach by Cao et al used a language model, as opposed to explicit language features, to do a similar task [20]. Similarly, many have used Question-Answer Topic Models [4] to project question-answers in a corpus to a smaller latent space, allowing for a condensation of large texts into a smaller, more rich set of words that can be analyzed efficiently.

In a different vein, Cai et al used pre-labeled category tags to improve upon machine translation approaches [11].

2.4. Embedding Techniques

Using embedding models and cosine similarity as a way to gauge whether pieces of text contain similar information has become popular. Embedding models are extremely powerful and show up in a large variety of NLP contexts [18]. In the area of Question Retrieval problems, embedding models benefit from not needing extensive annotation and its ability to be trained on a large corpus of text that is not necessarily related with the QA database [14].

In QR problems, embedding techniques such as Bag of Words and word2vec/CBoW, were able to achieve higher accuracies than past approaches, nearing 50% mAP on the Yahoo! Answers dataset [14]. [3]

With the recent advent of BERT in 2018, which uses bi-directional analysis to create sentence embeddings containing significantly more semantic information than in the past, the door is open for even higher accuracies [17]. Using

[2] Wondir shutdown in 2006 and the dataset is no longer publicly available

[3] https://www.kaggle.com/datasets/bhavikardeshna/yahoo-email-classification

BERT specifically in the context of QR problems is relatively unexplored, though some initial work by Prakash has achieved state-of-the-art results around 70% mAP [2].

Embeedding models are the state-of-the-art approach to solving QR problems using NLP, and this paper will attempt to build upon these recent successes.

3. Methods

This paper contributes to the problem of question retrieval by proposing/implementing two new methods – LLM prompting and Expert BERT – while using a variety of past embedding model as baselines. The former takes advantage of the implicit understanding of the English language possessed by the highly popular GPT-4o model to determine similarity [15]. The latter relies on the idea that fine-tuning a BERT model in specific domains generally allows it to learn a more sophisticated, nuanced understanding of the subject material [12]. See Figure 2 for an overview of the embedding methods we analyze.

3.1. Vector Embeddings + Cosine Similarity

These approaches calculate the vector embeddings of the questions in the database along with a new query question, and then computes the cosine similarity score between the query question and database questions. We experiment with several different question embedding schemes.

3.1.1 Word Embeddings

First, we implement simpler embedding transformations such as BoW and TF-IDF (also testing latent semantic analysis). Next, we try the Continuous Bag of Words (CBoW) model, a Word2Vec variant. In CBoW, each word in a question is mapped to a dense vector.

In both of these cases, the embedding representations do not capture the semantics between words and so we take the average (or a weighted average using TF-IDF) of the word embeddings in each question to get the final sentence embedding. We use these results as a baseline for our new proposed solutions.

To evaluate the BoW with TF-IDF model, we split our dataset into training and test sets with 70% of the data reserved for training. Using the training set, we constructed a vocabulary of unique words and a vector representation for each word. For the questions in the test set, we created vector embeddings that are the TF-IDF weighted sum of each word embedding.

We next evaluated the performance of a pre-trained CBoW model, the Google News model. The model contains 300-dimensional vectors for 3 million words and phrases and was trained on a subset of the Google News dataset. To get the final question embeddings, we again did a TF-IDF weighted sum of the word vector embeddings. The results for this approach are also found in Table 1.

3.1.2 Sentence Embeddings

We experiment with the more sophisticated BERT model to get sentence-level embeddings. We used the "[CLS]" token for the downstream matching task.

First, to establish a robust baseline for our experiments, we began by leveraging the "bert-base-uncased" model and extracting question-level embeddings from its [CLS] tokens. The "bert-base-uncased" architecture is comprised of 12 Transformer layers, each employing 12 attention heads. The model's contains approximately 110 million parameters in total. Using the [CLS] token embeddings for each question as fixed-length vector representations, we computed the cosine similarity matrices for both our specialized technical dataset and general dataset, yielding accuracies recorded in Table 1 under "BERT on Tech Data" and "BERT on General Data."

To further enhance the domain specificity and semantic coherence of these representations, we fine-tuned the pretrained BERT model on each dataset using the MLM objective. We decided to mask 15% of all tokens in each question at random, this matches the methodology of the original BERT paper [17].

3.2. BERT + Classification HEAD

In addition to leveraging cosine similarity to compare embeddings, we explore a more direct predictive framework by posing the question matching task as a binary classification problem. Rather than computing similarity scores between a query and each candidate independently, we treat each query-candidate pair as an input to be classified as either "duplicate" or "not-duplicate." This approach allows us to directly train a model for our task.

Given a user-posed query w and a set of candidate questions $\{q_i\}$, we must determine, for each q_i, whether it is the correct match for w. In other words, we seek to label each candidate pair (w, q_i) as either "duplicate" or "not-duplicate."

To implement this, we start by concatenating the user question w and a candidate question q_i into a single input sequence for BERT. We extract the $[\text{CLS}]$ token embedding from the final BERT layer, which serves as a condensed representation of the entire pair. This $[\text{CLS}]$ embedding is subsequently passed through a fully connected linear layer and a sigmoid activation function, yielding a probability of being a match.

3.3. LLM Prompting:

We provide ChatGPT-4o with two questions and prompt it to return 'same', if two questions have the same context (i.e. the questions ask the same thing, though worded differently) or ‘different’ if two questions have different contexts. We choose GPT-4o because of its impressive successes on a variety of tasks, such as logical reasoning, standardized exam questions, and translation tasks [15].

To evaluate performance, 40 questions were categorized into two groups: ‘same’ if questions asked the same thing and ‘different’ if questions asked about different topics. 20 questions were used from each category. The prompt to ChatGPT-4o included three examples of the same and different questions.

3.4. Evaluation

We offer two approaches for calculating the accuracy of our embedding models:

1. **Two Question Similarity (TQS):** We take two question from our dataset and use our model to determine if they are similar. For the LLM prompting, we expect an answer of ”same” or ”different.” For the embedding approaches, we expect an decimal between -1 and 1, and we apply a threshold value $\tau = 0.8$ to determine if they are similar or not.

2. **Database Lookup Accuracy (DLA):** We take a query question and choose the question from our database that is most similar to it. If these questions are, in fact, similar based on the labels from our dataset, then we declare this to be a successful lookup.

To evaluate the accuracy of these two approaches, we examined several metrics, and our results are summarized in Table 1. Under the TQS evaluation approach, we report the overall accuracy of our model using our threshold of $\tau = 0.8$. For evaluating the DLA method, we report the Precision at 1 (P@1) score, the Precision at 3 (P@3) score, and the Mean Reciprocal Rank (MRR). Precision @ K (P@K) measures how often the correct answer appears in the top-K predictions for each query. MRR is a metric that evaluates the ranking quality of the model across multiple queries, MRR $= \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i}$, where rank_i is the ranked-position of the matching question for the i-th query. MRR is equal to the mAP when each query has exactly one true match, as is the case in our dataset.

3.5. Dataset

To conduct our experiments, we curated a dataset of similar question pairs. We took the ”Quora Question Pairs,” dataset containing over 2.3 million similar question pairs. The pairs were determined by human experts in the field [3]. We randomly selected 15,000 pairs of general questions and 5,000 pairs of “Technology” specific questions. To do this, we simply searched for technological keywords, like “website”, “technology”, or “computer,” etc...

4. Results & Insights

We list all of our results in Table 1 and then delve into each more thoroughly in each of the following subsections.

4.1. Word Embedding Models

Our analysis reveals that the CBoW model underperforms the BoW model in terms of precision metrics (P@1, P@3, and MRR), across both general and technology-focused datasets. We hypothesize that the vocabulary within our technology-focused dataset significantly differs from the general vocabulary found in the Google News corpus. Consequently, the GoogleNews CBoW embeddings lack the specificity needed to capture nuanced, domain-specific terms relevant to technology. Another significant factor is the nature of the dataset itself. Many questions in the technology dataset share similar word patterns or structures. BoW relies on simple word frequency statistics, making it better suited for this dataset. BoW directly counts the occurrence of words, enabling it to make precise distinctions based on the presence or absence of specific terms. On the other hand, CBoW relies on pre-trained weights and may have struggled to capture the subtle and nuanced distinctions needed to differentiate specific sentences.

Despite these limitations, the CBoW model exhibited comparable or slightly superior performance under the threshold-based TQS accuracy metric. In our evaluation, a similarity score below the threshold of $\tau = 0.8$ is classified as dissimilar. CBoW’s generalization allows it to excel in distinguishing such cases. By contrast, BoW’s reliance on exact word matches makes it more effective at detecting fine-grained similarities but less equipped for capturing broader semantic differences.

The trade-off between semantic generalization and syntactic precision highlights the complementary strengths of BoW and CBoW. While BoW excels at ranking relevant matches and identifying close textual similarities, CBoW performs better in scenarios where broader dissimilarity detection is prioritized. The effectiveness of each model depends on the task requirements and dataset characteristics.

4.2. Sentence Embedding Models

When comparing the performance of pre-trained BERT to fine-tuned BERT models, it is evident that fine-tuning provides measurable improvements across all metrics. ”BERT Fine Tuned on General Data” (Table 1) shows the most significant gains, particularly in precision metrics, demonstrating the effectiveness of MLM fine-tuning in aligning the model’s representations more closely with the domain-specific characteristics of our target corpora.

4.3. Binary Classification Models

Our binary classification models significantly outperform our previous cosine similarity methods, which rely

Table 1. Performance metrics across all tested models

Model	**Accuracy**	**P@1**	**P@3**	**MRR**
BoW				
BoW with TF-IDF on General Data	0.56	0.66	0.82	0.76
BoW with TF-IDF on Tech Data	0.55	0.53	0.68	0.63
CBoW				
GoogleNews CBoW on General Data	0.61	0.34	0.48	0.43
GoogleNews CBoW on Tech Data	0.63	0.26	0.40	0.35
CBoW Tested on General Data	0.55	–	0.31	–
CBoW Tested on Tech Data	0.55	–	0.23	–
BERT				
BERT on Tech Data	0.52	0.23	0.34	0.31
BERT Fine Tuned on Tech Data	0.53	0.25	0.37	0.33
BERT on General Data	0.51	0.36	0.49	0.44
BERT Fine Tuned on General Data	0.56	0.41	0.52	0.48
BERT + Classification Head				
RoBERTa on Tech Data	0.5	–	–	–
RoBERTa Fine Tuned on Tech Data	0.86	–	–	–
RoBERTa on General Data	0.5	–	–	–
RoBERTa Fine Tuned on General Data	0.83	–	–	–
GPT Prompting				
GPT Prompting	0.92	–	–	–

solely on measuring distances between static embeddings. A key element in this improvement is the linear classification layer following the final BERT representation. While cosine similarity is fixed, the linear layer can adapt its weighting of various embedding dimensions. This adaptive weighting allows the model to amplify features that signal duplication and reduce the influence of irrelevant factors. We achieve a notable accuracy of 86% on the technology dataset and 83% on the general dataset.

4.4. GPT Prompting (TQS)

ChatGPT-4o performed performed better than all of the embedding models with an accuracy of 95% (19/20 questions labeled correctly) when marking questions as the same. ChatGPT-4o had an accuracy of 90% (18/20 questions labeled correctly) when identifying pairs of questions that were different. For general questions, ChatGPT-4o performed well.

However, the issue with LLM prompting is that it is unscalable when dealing with large datasets. While prompting GPT-4o achieves high accuracies when we ask if two questions are similar, if we gave it a database of $15k$ questions, it would be highly unlikely for it to find the best matches quickly and accurately.

5. Conclusion & Future Work

Our results demonstrate the success of our two new methods – fine-tuning BERT on a specific area of knowledge and prompting the GPT-4o LLM – compared to previous baselines. We strongly believe that fine-tuning a BERT embedder for specific classes could greatly improve improve Piazza's similar-questions functionality. We are very interested in future works that actual test our methods on Piazza data. (See Section 6).

We are also interested in training and testing our models on much larger datasets in order bolster the robustness and accuracy of our work. With more compute power, this would certainly be feasible and highly encouraged.

6. Ethics Statement

In the writing of this paper and execution of our experiments, we identified several areas for ethical considerations, ranging from model biases to privacy concerns.

First, when training any model, the type and quality of data must be considered. Out data came from an open-source dataset of Quora questions from Kaggle – it was curated by a set of 'experts' who decided which questions were matched with similar content. Any biases or inaccuracies may certainly have been carried over into our models. Many studies have examined the impacts and ability of LLMs to perpetuate biases [5, 6, 10]. However, for our use of LLMs, we do not foresee biases having a large ethical impact in the quesiton retrieval context.

Second, although it would have been ideal to test our models on questions taken directly from the Piazza platform, we believe it would be a privacy violation to train on this data without explicit permission. Students who post on Piazza are under the assumption that only instructors and other student may view their question and answers.

Third, we consider the issues of representation in our dataset. We recognize that the dataset may not fully represent the diversity of questions asked on Piazza. Quora may have inherit biases towards certain groups or specific content. Both Quora and Piazza posts range in length, context and subject. However, even using the diverse Quora dataset may not account for all discrepancies. To this end, we emphasize the importance of continuously updating and diversifying data sources to best reflect all possible user experiences.

References

[1] John Laherty Adam Berger. Information retrieval as statistical translation, 1999.

[2] Jay Prakash C. M. Suneera. A bert-based question representation for improved question retrieval in community question answering systems, 2021.

[3] Lili Jiang Meg Risdal Nikhil Dandekar tomtung DataCanary, hilfialkaff. Quora question pairs, 2017.

[4] Ben He Fei Xu, Bin Wang. Question-answer topic model for question retrieval in community question answering, 2012.

[5] Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. Bias and fairness in large language models: A survey, 2024.

[6] Valentin Hofmann, Pratyusha Ria Kalluri, Dan Jurafsky, and Sharese King. Ai generates covertly racist decisions about people based on their dialect. *Nature*, 633(8028):147–154, Aug 2024.

[7] Chin-Yew Lin Yong Yu Huizhong Duan, Yunbo Cao. Searching questions by identifying question topic and question focus, 2008.

[8] Joon Ho LeeAuthors Info Claims Jiwoon Jeon, W. Bruce Croft. Finding similar questions in large question and answer archives, 2005.

[9] Haocheng Wu Zhoujun Li Ming Zhou Kai Zhang, Wei Wu. Question retrieval with high quality answers in community question answering, 2014.

[10] Hadas Kotek, Rikker Dockum, and David Q. Sun. Gender bias in llms, 2023.

[11] Kang Liu Li Cai, Guangyou Zhou and Jun Zhao. Learning the latent topics for question retrieval in community qa, 2011.

[12] Dietrich Klakow Marius Mosbach, Maksym Andriushchenko. On the stability of fine-tuning bert: misconceptions, explanations, and strong baselines, 2021.

[13] Aniello Minutolo Giuseppe De Pietro Hamido Fujita Massimo Esposito, Emanuele Damiano. Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering, 2020.

[14] Kamel Smaili Nouha Othman, Rim Faiz. Enhancing question retrieval in community question answering using word embeddings, 2019.

[15] Brady D. Lund Nishith Reddy Mannuru Muhammad Arbab Arshad Kadhim Hayawi Ravi Varma Kumar Bevara Aashrith Mannuru Shahriar, Sakib and Laiba Batool. ”utting gpt-4o to the sword: A comprehensive evaluation of language, vision, speech, and multimodal proficiency, 2024.

[16] Qingyao Wu Heng Weng Yingying Qu Tianyong Hao, Wenxiu Xie. Leveraging question target word features through semantic relation expansion for answer type classification, 2017.

[17] Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[18] Chen F Wang Y Kuo C-CJ Wang B, Wang A, 2019.

[19] W. Bruce Croft Xiaobing Xue, Jiwoon Jeon. Retrieval models for question and answer archives, 2008.

[20] Bin Cui Christian S. Jensen Xin Cao, Gao Cong and Ce Zhang. The use of categorization information in language models for question retrieval, 2009.

[21] Naman Goyal Yinhan Liu, Myle Ott. Roberta: A robustly optimized bert pretraining approach, 2019.