Accurate Training of Web-based Question Answering Systems with Feedback from Ranked Users

Liang Wang[*]
Boston University
Department of Mathematics and Statistics
leonwang@bu.edu

Ivano Lauriola
Amazon Alexa
lauivano@amazon.com

Alessandro Moschitti
Amazon Alexa
amosch@amazon.com

Abstract

Recent work has shown that large-scale annotated datasets are essential for training state-of-the-art Question Answering (QA) models. Unfortunately, creating this data is expensive and requires a huge amount of annotation work. An alternative and cheaper source of supervision is given by feedback data collected from deployed QA systems. This data can be collected from tens of millions of user with no additional cost, for real-world QA services, e.g., Alexa, Google Home, and etc. The main drawback is the noise affecting feedback on individual examples. Recent literature on QA systems has shown the benefit of training models even with noisy feedback. However, these studies have multiple limitations: (i) they used uniform random noise to simulate feedback responses, which is typically an unrealistic approximation as noise follows specific patterns, depending on target examples and users; and (ii) they do not show how to aggregate feedback for improving training signals. In this paper, we first collect a large scale (16M) QA dataset with real feedback sampled from the QA traffic of a popular Virtual Assistant. Second, we use this data to develop two strategies for filtering unreliable users and thus de-noise feedback: (i) ranking users with an automatic classifier, and (ii) aggregating feedback over similar instances and comparing users between each other. Finally, we train QA models on our filtered feedback data, showing a significant improvement over the state of the art.

1 Introduction

Large pre-trained language models, e.g., based on The Transformer neural network (Lin et al., 2022), have recently improved Natural Language Processing and Information Retrieval in several tasks, e.g., document classification (Chaudhary et al., 2020), Question Answering (QA) (Garg et al., 2020), neural retrieval (Karpukhin et al., 2020).

Transformer models can be conveniently pre-trained on large-scale unlabeled web data through general task-agnostic unsupervised objectives, e.g., Masked Language Model (MLM) and Next Sentence Prediction (NSP), making the networks able to be easily specialized on various downstream tasks (Devlin et al., 2019). However, they still require labeled training data (expensive to produce) to be adapted on the target domain. Recent research in QA has shown that the more data is used for fine-tuning the models, the better is the final performance (Huber et al., 2022). For instance, Garg et al. (2020) showed and measured the benefits of using large-scale labeled web datasets, Google NQ (Kwiatkowski et al., 2019), for training their answer selection ranker. The authors divided the fine-tuning stage into two steps: transfer and adapt (TANDA). In the first step, the pre-trained Transformer is tuned on general out-of-domain large-scale QA data. Then, the resulting model is further trained on the target domain. However, building large-scale annotated resources is costly in terms of expert annotator work and annotation time. To reduce costs, various strategies to generate cheap training data have been recently explored, including data augmentation (Pappas et al., 2022; Riabi et al., 2021), distant supervision (Lin et al., 2018; Zhao et al., 2021), and active learning (Kratzwald et al., 2020).

A rather different approach is based on the availability of feedback data, i.e., the QA system output (typically an answer) is evaluated by users. These question/answer (q/a) pairs can be used for further improving the training of QA systems (Li et al., 2022; Campos et al., 2020). User feedback can, thus, be used to build large and cheap training data, especially when the QA system constitutes the back bone of commercial applications such as virtual assistants, e.g., Google Home, Alexa, Siri, to which million questions are asked every day. Unfortunately, feedback data is affected by noise, i.e., the

[*]Work done during an internship at Amazon Alexa.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 5: Industry Track, pages 660–667

individual feedback over a q/a pair has high probability to be incorrect. Indeed, users are not expert annotators, and they often provide judgments with lack of knowledge, subjectivity, and specific preferences. For instance, some users may return a positive feedback to a wrong answer that sounds funny. Therefore, feedback data may be ineffective for further training accurate models: in some cases, it can even degrade the accuracy of models trained on small datasets labelled by expert annotators.

How to obtain effective training data labelled with feedback is an interesting open problem. For example, Rebbapragada and Brodley (2007) and Sun et al. (2007) provide weights to the training instances according to a mislabeling probability. However, to the best of our knowledge, previous work used artificially generated datasets, e.g., by adding random uniform noise to labeled corpora (Campos et al., 2020). This does not represent the characteristics of real users' generated traffic, as the noise distributes differently with respect to different questions (according to categories, semantics, pragmatics, trends, etc.). Users have indeed different attitudes and behaviors, and they may interact differently with the responses of a QA system. Approximating the error probability distribution of feedback is a main challenge, preventing weighting methods to be effective.

To study this problem, we first collect and analyze a large-scale users' feedback dataset (16M q/a pairs) sampled from the traffic of a popular virtual assistant. Then, we propose two effective solutions to reduce the label noise and improve training performance: (i) we use an automatic classifier, trained on off-the-shelf answer selection data to automatically grade the reliability of users. Then, we select training examples using the most reliable users. (ii) We cluster together similar questions so that we compare the answer from different users and again rank them based on how much they are close to the majority judgments. Our experiments show that both proposed methods improve state-of-the-art models much more than using noisy and unfiltered data.

2 Related work

Training QA systems with feedback data after models deployment received considerable attention in the past years. Campos et al. (2020) for instance, simulated a scenario where an initial deployed machine reading model is continuously trained through feedback responses, showing promising sandbox performance. Authors simulated feedbacks as positive whenever the answer span predicted by the system matches the gold span exactly, and negative otherwise. A similar scenario was considered by Li et al. (2022), where feedback was collected through crowdworkers. The authors themselves pointed out the limits of these studies as the results were based on artificial data distribution.

Other authors explored feedback such as explanation of incorrect responses by chatbots (Li et al., 2016; Weston, 2016). However, the feedback in these studies is automatically generated using heuristics. Similarly, Rajani et al. (2019) collected human explanations for commonsense QA in the form of natural language sequences, and used the data to improve existing models with state-of-the-art performance.

The main weakness of previous work is that most of existing analyses (i) are based on artificially generated data (Campos et al., 2020; Li et al., 2022, 2016; Weston, 2016) or (ii) use crowdsourced workload (i.e., annotators) to simulate real feedback data. As a consequence, these approaches are not suitable for industrial scenarios since they do not consider noise distribution of real users' generated data. In contrast, our work is based on real user data from a virtual assistant, which provides answer using a web-based QA system. Our results are generalizable to most industrial scenarios targeting open domain QA.

3 Operational setting

We consider the task of selecting the correct answer sentence among a set of candidates extracted from web-documents retrieved for a given question. Formally, let Σ^* be the set of strings (or general sentences) and $\mathcal{Q} \subseteq \Sigma^*$ be the set of questions according to a certain input distribution. Given an input question $q \in \mathcal{Q}$ and a set of k sentences $\{s_i\}_{i=1}^{k} \in {\Sigma^*}^k$ (e.g., returned by a search engine), the answer selector can be defined as a function $r : \mathcal{Q} \times \Sigma^* \rightarrow \mathbb{R}$, which assigns a probability score to each q/a pair, $r(q, s_i)$, and returns the answer associated with the highest ranked pair, i.e., $\arg\max_{i=1...k} r(q, s_i)$. We use the state-of-the-art model for answer sentence selection (Garg et al., 2020; Lauriola and Moschitti, 2021), which implements r with a Transformer model. This encodes an input pair (q, s_i) as *[CLS]* q *[SEP]* s_i *[EOS]* and returns the associated score through a classification

head on top of the *[CLS]* token.

3.1 Internal Feedback Dataset (IFD)

We use a popular virtual assistant to collect a large internal dataset constituted by: (i) open domain user questions, (ii) their answers selected from web-documents by a QA system, and (iii) feedback provided by users to the answers. We first pre-processed the data by de-identifying the users. Then, we limited the sample to questions asked in 2022. Finally, we removed questions asked by users who provided feedback to less than 4 answers. Overall, we collected a dataset, $\mathcal{D}$, of 16M tuples, (q_i, s_i, f_i, u_j), where q_i is an open-domain question, s_i is the answer generated by the virtual assistant, $f_i \in \{-1, +1\}$ is the binary feedback returned by the user, and u_j is the user id. In the remainder of this paper, we refer this resource as Internal Feedback Dataset (IFD).

Please note that, as the questions are from real users, for several internal and external regulations, we cannot release the resource for public research. To improve replicability of our findings, we provide empirical results on the impact of our approach and data on public benchmarks.

4 Training with de-noised Feedback

The standard approach to de-noise training examples is to assign them different weights according to their reliability. Finding these weights for individual feedback instances is an open problem. We propose two methods: The first is based on a completely new idea (to our knowledge): we observed that different users have different accuracy in assessing the correctness of answers. Since manually labeling millions of users is not feasible, we use an automatic answer selector classifier, trained on off-the-shelf data. The second approach is based on standard collaborative filtering applied to user clusters, which are created by comparing questions using state-of-the-art text similarity techniques.

4.1 User Relevance Score

We provide a Relevance Score (RS) to users in two steps: First, we measure the agreement between the user annotation and an automatic answer selection model, r, which provides a probability of correctness of the answers for the target questions. More formally, the agreement/similarity between the classifier score and the associated feedback is $r(q_i, s_i) \cdot f_i$, where $r(q_i, a_i) \in [-1, +1]$ is the score and $f_i \in \{-1, +1\}$ is the binary user feedback. Intuitively, the higher is the agreement the higher is the probability that the feedback is correct.

In the second step, we computed RS of an user as the average of the agreement scores computed on all examples he/she gave a feedback. In short, we assign RS to the user u_j, defined as:

$$RS(u_j) = \frac{\sum_{(q_i,s_i,f_i,u_j)\in\mathcal{D}_{u_j}} r(q_i, s_i) \cdot f_i}{|\mathcal{D}_{u_j}|}$$

where $\mathcal{D}_{u_j} \subset \mathcal{D}$ is the set of tuples for which the user u_j gave a feedback, i.e., $\mathcal{D}_{u_j} = \{(q_i, s_i, f_i, u_k) \in \mathcal{D} : k = j\}$.

Finally, to obtain accurate training data we consider the annotation of only reliable users. These are obtained by ranking users with RS and discard those having a score below a certain threshold. We build r using a state-of-the-art Electra-large model trained for answer sentence selection as described by (Garg et al., 2020).

4.2 Collaborative filtering

Our second filter is based on the intuition that users have different knowledge and they may provide accurate feedback to certain type of questions and low quality feedback to others.

Broadly speaking, the feedback assigned by other users to a given answer can provide some insights on the quality of a target user. If a user tends to disagree with the majority of feedback for a given question, then we can filter out the user as their judgment cannot be considered reliable.

Let X be user-question matrix where the ji-th entry, i.e., $X_{[j,i]} \in \{-1, 0, +1\}$, contains the feedback of the user j to the question i. $+1$ positive feedback, -1 negative, 0 missing. By construction, the i-th column, that is, $X_{[:,i]}$, contains all feedback collected for a given question. We define the voted feedback, $\bar{f}_i$, for the question, i, as the average of non-missing judgments, which can be easily computed as the ratio between the L_1 and L_0 norm of the i-th column vector: $\bar{f}_i = \frac{\|X_{[:,i]}\|_1}{\|X_{[:,i]}\|_0}$. Eventually, let $\bar{\boldsymbol{f}}$ be the voted feedback vector ($[\bar{f}_1, \bar{f}_2 \ldots]$). We define the reliability of the user j as the average proximity between its feedbacks and $\bar{\boldsymbol{f}}$, that is, $X_{[j,:]}\bar{\boldsymbol{f}}^\top$.

Similarly to the previous approach, we use this reliability score to rank users and to filter those with a score below a certain threshold.

The main issue with this approach is the sparsity of the user-question matrix. Typically, questions

are unique word sequences, and thus the amount of questions with feedback from different users is low. In order to overcome this limitation, we extend the collaborative approach by aggregating questions (i.e., columns), which are semantically identical or at least similar. We define a cluster of questions, c_k, as the set of semantically equivalent questions, and we represent the user-cluster interactions through a matrix X^c. The jk entry of the user-cluster relations matrix, that is $X^c_{[j,k]}$ contains the feedback of the user j for a question in the k-th cluster. If a user provides feedback to multiple questions belonging to the same cluster we average the them.

We find question clusters using a standard k-means algorithm, where the semantic distance between questions is computed with a Transformer model. We used a RoBERTa-large model trained on various semantic similarity tasks[1] and further fine-tuned on Quora Question Pairs, a popular dataset for question-question similarity tasks. Examples are encoded as [CLS] question [SEP] answer [EOS]. The representations developed in the last Transformer layer associated with the [CLS] token are then used to compute the distance between two questions. For simplicity, we used the standard euclidean distance function.

During a preliminary experimentation phase, we set the number of clusters to 50,000. This value represents a good trade-off between quality of the clusters (i.e., we do not have unrelated questions, which look similar, in the same cluster) and the amount of feedback per cluster. We observed that more than 95% of the clusters have at least 100 pieces of feedback.

5 Empirical assessment

We divided our experiments in 3 groups: First, we analyze the quality of our filtered data through manual evaluation. Then, we show that the massive amount of noisy feedback can improve the performance of QA models already trained on large-scale high-quality annotated data. Finally, we analyze the impact of the de-noising strategies described in the previous sections.

5.1 Qualitative evaluation

The first step of our analysis concerns the evaluation of the proposed filtering strategies. Both filters rank the users according to their likelihood probability of being good annotators. Hence, users in the top of the rank provide higher quality data compared to users in the bottom of the rank.

To evaluate this assumption, we ranked all tuples from IFD according to relevance and collaborative scores and we randomly sampled 200 tuples from the top and the bottom 10% of the ranks. Then, we manually analyzed these samples to evaluate and quantify the amount of label errors (noise). We used expert annotators and the Matthews Correlation Coefficient (MCC) (Chicco and Jurman, 2020) to measure the agreement between users' feedback and annotators' labels. This value ranges between -1 (totally uncorrelated) and 1 (perfectly correlated). The higher is the value of MCC on the sample, the higher is the alignment between feedback labels and annotators' judgments (that we consider as gold standard), and consequently the quality of the feedback data. Compared to other metrics, e.g., accuracy or F1, MCC is not affected by class-skewness. We also computed the same score on a random sample from the original unfiltered dataset for further comparison.

Sample	top 10%	bottom 10%
Random	0.49	
Relevance filtering	0.73	0.36
Collaborative filtering	0.53	0.38

Table 1: MCC computed between expert annotators and various samples of feedback, including random sample and top/bottom 10% of the rank produced by our filters.

The results in Table 1 shows that, notwithstanding the limits and simplicity of the proposed filtering strategies, they are clearly able to correctly rank tuples, placing noisy examples lower in the rank. The samples annotated from the top 10% of the ranks have indeed a higher MCC score compared to the tuples sampled from the lower 10%. Also, the relevance filtering seems to work better.

5.2 Training with noisy feedback

Datasets We consider two popular annotated datasets for answer sentence selection tasks: ASNQ (Garg et al., 2020) and WikiQA (Yang et al., 2015). ASNQ is a large-scale resource derived from Natural Questions (Kwiatkowski et al., 2019). For each input question, candidate answer sentences are extracted from a selected wikipedia page. The dataset consists of 20M labeled q/a pairs, making it one of the largest existing resources for this task. WikiQA is a curated small

[1]The checkpoint is available here https://huggingface.co/sentence-transformers/all-roberta-large-v1

Configuration	P@1	MAP	MRR
IFD	66.9	78.7	79.8
ASNQ	83.1	88.0	89.4
WikiQA	75.5	83.6	85.0
IFD $\rightsquigarrow$WikiQA	81.9	88.0	89.1
ASNQ $\rightsquigarrow$WikiQA	84.4	88.8	90.4
ASNQ $\rightsquigarrow$IFD $\rightsquigarrow$WikiQA	**86.1**	**90.4**	**91.5**

Table 2: Preliminary evaluation of IFD. Sequential fine-tuning is denoted by $\rightsquigarrow$. Models are tested on WikiQA.

resource consisting of 3,047 questions and 29,258 q/a pairs. Similarly to ASNQ, sentences were extracted from Wikipedia abstracts associated with each input question. In our setting, we consider (i) WikiQA as low-resource target domain, where we test our models, (ii) ASNQ as large-scale annotated resource to improve the QA performance on WikiQA (as described by Garg et al. (2020)), and (iii) IFD to study the impact of feedback data on the target domain.

Model selection We start from an Electra-base (110M parameters, 12 layers) public checkpoint. During the training, we set (i) the batch size to 1024 q/a pairs, (ii) the max sequence length for the input of the Transformer to 128 tokens, (iii) the max training epochs to 5 for ASNQ and IFD, and to 10 for WikiQA, and (iv) a constant lr schedule with linear warm-up of 0.1 epoch. We used WikiQA validation set to monitor the validation loss after each epoch and, in case, terminate the training, and to select the optimal learning rate, with values $[1, 2, 5] \times 10^{-[5,6]}$. We used ASNQ, IFD, and WikiQA to train the models with different configurations described in the next sections, and we used the test split of WikiQA as final test set. For each experiment, we train and evaluate models 3 times and average the final results computed on the test set.

Training We evaluated the following training strategies:

• 1-step training - We fine-tune a public Transformer checkpoint on IFD, ASNQ, or WikiQA and test the models on WikiQA. This allows us to isolate and quantify the impact of large noisy data (IFD), large high-quality data (ASNQ), and limited but in-domain data (WikiQA).

• 2-steps training - Inspired by recent research in answer sentence selection (Garg et al., 2020), we first train models on large datasets, i.e., IDF or ASNQ, and then we further fine-tune the models on the target domain (WikiQA).

• 3-steps training - We sequentially fine-tuned the model on (i) ASNQ, (ii) IFD, and (iii) WikiQA. This experiment shows that, even in scenarios where large amount of labeled data is available for training, feedback helps the model and improves the final accuracy.

The results of these experiments (see Table 2) show multiple keypoints: First, large resources do not necessarily improve the performance if their quality is poor, e.g., small but high-quality in-domain training data (WikiQA, 30k q/a pairs) performs better than large noisy dataset (IFD). Second, both ASNQ and IFD significantly improve the performance on WikiQA when using sequential fine-tuning approaches (lines 4-5). Not surprisingly, the improvement of ASNQ is higher as it contains high-quality annotations. Moreover, a last fine-tuning on the target domain (WikiQA) always improves the performance (lines 1-2 compared to 4-5). Finally, the combination of high and low quality large resources (line 6) further improves the performance. Although IFD contains a considerable amount of noise, it is still a valuable resource to improve the performance of the model. Even thought a large resource is available, i.e., ASNQ, feedback data is still rather valuable.

5.3 Relevance and collaborative filtering evaluation

We analyzed the impact of de-noising mechanisms to improve the quality of data and consequently the final performance. For each filter, relevance and collaborative, we first compute the rank of users as described in Section 4, then, we consider the following filtered IFD versions:

Full We use the full set, regardless of the produced ranks. This version of IFD represents the baseline where the filtering is not used.

Top We use the top 10% of tuples from IFD according to the rank of the filter. This allows us to restrict the training to high-quality feedback.

Best We train models with 10%, 20%, 30%,...,100% of IFD selected on top on the rank of the filter. Then, we select the model with lowest validation loss. This helps finding an optimal trade-off between data quantity and quality.

	Relevance f.			Collaborative f.		
Configuration	P@1	MAP	MRR	P@1	MAP	MRR
Full (100%)	80.4	86.2	87.7	80.4	86.2	87.7
Best (10-100%)	80.6	86.6	88.2	**81.3**	**87.0**	**88.4**
Random (10%)	78.2	85.1	86.7	78.2	85.1	86.7
Top (10%)	**81.3**	**87.0**	**88.5**	79.8	86.6	87.9
Full $\rightsquigarrow$WikiQA	86.1	90.4	91.5	**86.1**	**90.4**	**91.5**
Best $\rightsquigarrow$WikiQA	86.8	90.7	91.8	85.8	**90.4**	91.4
Ran. $\rightsquigarrow$WikiQA	85.9	90.1	91.4	85.9	90.1	91.4
Top $\rightsquigarrow$WikiQA	**87.0**	**90.7**	**92.0**	84.6	89.5	90.6

Table 3: Consistency and collaborative filtering - empirical results for the 4 sampling strategies. All models start from a cehckpoint trained on ASNQ.

Random We use 10% of tuples from IFD randomly sampled. This baseline emphasizes the effect of the filter compared to the usage of **Top** tuples. Both strategies indeed, **Random** and **Top**, use the same amount of data.

For each subset and filter, we sequentially trained an Electra-base on ASNQ and then on IDF (filtered). Results in Table 3 show multiple key aspects: First, finding the optimal trade-off between quantity and quality (**Best**) usually improves the performance compared to the unfiltered IFD (**Full**), suggesting that the filtering methods work as expected. The only exception occurs when using collaborative filtering and fine-tuning models on WikiQA. Note that this approach is computationally expensive as we train a model for each possible threshold (10%, 20%...).

Second, using only the **Top** 10% of the data further improves the results when adopting the relevance filtering. This indicates that: (i) relevance filtering works well and can be used to significantly reduce the amount of data by a magnitude, while improving the QA performance; (ii) collaborative filtering shows some promising results only when models are not fine-tuned on the target domain. However, both approaches represent a solid base for future research in this field. Note that these results corroborate our manual analysis showed in Table 1. Both experiments, manual rank evaluation and models training, suggest that the relevance filtering provides, compared to collaborative approach, a better rank and thus a better data filtering and final performance.

6 Conclusion

Feedback data represents a huge and convenient source of training data, which can be used to improve the performance of deployed QA systems. However, the noise affecting feedback can degrade model performance. This paper introduces two ML approaches to filter feedback data and to reduce the amount of noise. Our filters are based on the assumption that users act differently from each other. Thus, their behaviour induces different reliability, which if modeled correctly can help to build more effective training data. We used a large set of question, answer, and feedback tuples (16M) sampled from a commercial virtual assistant to validate this hypothesis. Our extensive empirical assessment clearly shows that filtered feedback can significantly improve the performance of a deployed QA system, even when the models are trained on massive high-quality annotated resources.

Note that this work does not aim to compare different filtering methods to elect a superior approach. We conjecture that the collaborative filtering can be further improved, for instance by deeply analyzing different clustering approaches or embeddings extractors. On the contrary, our goal is (i) to highlight the importance and impact of using real feedback data to improve the performance of industrial QA models, and (ii) to provide insights for future research directions. To the best of our knowledge, this work represents the first analysis on real feedback data and its integration into model training. These findings reveal promising directions to improve deployed QA systems.

7 Limitations

This paper introduces two heuristic approaches to filter noisy feedback data. Although we showed that these simple methods improve the performance of QA models, they have various limitations and they represent only an initial step for future re-

search on real feedback data.

The core of the relevance filtering is based on the assumption that correct feedback occur when the model and the user agree on the labels. This approach may introduce a selection bias towards tuples associated with "simpler" q/a pairs, which are already well understood by the model and thus potentially ineffective for training. Although the model can easily discard q/a pairs whose feedback are clearly different, the risk is that uncertain pairs close to the classification boundary (i.e., model score close to 0) are penalized and easily filtered as they will receive a reliability score close to 0.

Regarding the collaborative approach, the main limitation concerns the clustering strategy adopted to aggregate questions. On one hand, we want to reduce as much as possible the number of clusters such that we have a sufficiently high amount of feedback per cluster. This makes the proximity computation between users and the voted feedback vector robust.

On the other hand, the clustering may introduce additional noise by aggregating different and non-equivalent questions into the same cluster. This aspect may reduce the reliability of the voted feedback vector.

Finally, as mentioned in the previous sections, feedback data and q/a pairs used in this work come from real users traffic. For this reason, we only described the high-level approach of integrating feedback and we showed the impact on public benchmarks. A harsh limitation is caused by the private nature of the customer data, which cannot be released for public research.

References

Jon Ander Campos, Kyunghyun Cho, Arantxa Otegi, Aitor Soroa, Eneko Agirre, and Gorka Azkune. 2020. Improving conversational question answering systems after deployment using feedback-weighted learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2561–2571, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yatin Chaudhary, Pankaj Gupta, Khushbu Saxena, Vivek Kulkarni, Thomas Runkler, and Hinrich Schütze. 2020. TopicBERT for energy efficient document classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1682–1690, Online. Association for Computational Linguistics.

Davide Chicco and Giuseppe Jurman. 2020. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7780–7788.

Patrick Huber, Armen Aghajanyan, Barlas Oguz, Dmytro Okhonko, Scott Yih, Sonal Gupta, and Xilun Chen. 2022. CCQA: A new web-scale question answering dataset for model pre-training. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2402–2420, Seattle, United States. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Bernhard Kratzwald, Stefan Feuerriegel, and Huan Sun. 2020. Learning a Cost-Effective Annotation Policy for Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3051–3062, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Ivano Lauriola and Alessandro Moschitti. 2021. Answer sentence selection using local and global context in transformer models. In *European Conference on Information Retrieval*, pages 298–312. Springer.

Jiwei Li, Alexander H Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. 2016. Dialogue learning with human-in-the-loop. *arXiv preprint arXiv:1611.09823*.

Zichao Li, Prakhar Sharma, Xing Han Lu, Jackie CK Cheung, and Siva Reddy. 2022. Using interactive feedback to improve the accuracy and explainability of question answering systems post-deployment. *arXiv preprint arXiv:2204.03025*.

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2022. A survey of transformers. *AI Open*.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1736–1745, Melbourne, Australia. Association for Computational Linguistics.

Dimitris Pappas, Prodromos Malakasiotis, and Ion Androutsopoulos. 2022. Data augmentation for biomedical factoid question answering. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 63–81, Dublin, Ireland. Association for Computational Linguistics.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Umaa Rebbapragada and Carla E Brodley. 2007. Class noise mitigation through instance weighting. In *European conference on machine learning*, pages 708–715. Springer.

Arij Riabi, Thomas Scialom, Rachel Keraron, Benoît Sagot, Djamé Seddah, and Jacopo Staiano. 2021. Synthetic data augmentation for zero-shot cross-lingual question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7016–7030, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiang-wen Sun, Feng-ying Zhao, Chong-jun Wang, and Shi-fu Chen. 2007. Identifying and correcting mislabeled training instances. In *Future generation communication and networking (FGCN 2007)*, volume 1, pages 244–250. IEEE.

Jason E Weston. 2016. Dialog-based language learning. *Advances in Neural Information Processing Systems*, 29.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Chen Zhao, Chenyan Xiong, Jordan Boyd-Graber, and Hal Daumé III. 2021. Distantly-supervised dense retrieval enables open-domain question answering without evidence annotation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9612–9622, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.