Reranking for Efficient Transformer-based Answer Selection

Yoshitomo Matsubara*
University of California, Irvine
Irvine, California, USA
yoshitom@uci.edu

Thuy Vu
Amazon Alexa
Manhattan Beach, California, USA
thuyvu@amazon.com

Alessandro Moschitti
Amazon Alexa
Manhattan Beach, California, USA
amosch@amazon.com

ABSTRACT

IR-based Question Answering (QA) systems typically use a sentence selector to extract the answer from retrieved documents. Recent studies have shown that powerful neural models based on the Transformer can provide an accurate solution to Answer Sentence Selection (AS2). Unfortunately, their computation cost prevents their use in real-world applications. In this paper, we show that standard and efficient neural rerankers can be used to reduce the amount of sentence candidates fed to Transformer models without hurting Accuracy, thus improving efficiency up to four times. This is an important finding as the internal representation of shallower neural models is dramatically different from the one used by a Transformer model, *e.g.*, word vs. contextual embeddings.

CCS CONCEPTS

• **Information systems** → **Retrieval models and ranking**; **Question answering**.

KEYWORDS

Question Answering, Transformer Models, Neural Networks, Reranking, Information Retrieval, Natural Language Processing

ACM Reference Format:
Yoshitomo Matsubara, Thuy Vu, and Alessandro Moschitti. 2020. Reranking for Efficient Transformer-based Answer Selection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401266

1 INTRODUCTION

QA research has received a renewed attention in recent years thanks to advent of virtual assistants in industrial sectors. For example, Alexa, Google Home, and Siri provide general information inquiry services. One key task for QA, is the Answer Sentence Selection (AS2), which has been widely studied by the TREC challenges. AS2, given a question and a set of answer sentence candidates, consists in selecting sentences (*e.g.*, retrieved by a search engine) that correctly answer the question. Neural models have significantly contributed with new techniques, *e.g.*, [8, 11] to AS2. More recently, neural language models, *e.g.*, ELMO [13], GPT [14], BERT [5], RoBERTa [10], XLNet [3] have led to major advancements in NLP. These methods capture dependencies between words and their compounds by pre-training neural networks on large amounts of data. Unfortunately, the Transformer-based architectures use a huge number of parameters (*e.g.*, 340 million for BERT Large). This poses important challenges for their deployment in production: first, the latency of the approach highly increases with the number of candidates. For instance, to target the required Recall, it might be necessary to classify of hundreds of candidates, which can take several seconds even when using powerful GPUs. Secondly, although the classification of candidates can be scaled horizontally, the number of GPUs required to fulfill a target transaction-per-second will prohibitively increase the operational cost. Finally, their high energy consumption is an environmental threat, as pointed out by [18] and the NeurIPS workshop, *Tackling Climate Change with ML*.

In this paper, we study and propose solutions to improve the efficiency and cost of modern QA systems based on search engines and Transformer models. Though we mainly focus on AS2, the proposed solution is general, and can be applied to other QA paradigms, including machine reading tasks. Our main idea follows the successful cascade approach for ad-hoc document retrieval [19], which considers fast but less accurate rerankers together with more accurate but slower models. In particular, we use (i) simple models, *e.g.*, Jaccard similarity, as well as light neural models such as Compare-Aggregate [22], for reranking answer sentence candidates; and (ii) BERT models as our final AS2 step.

To carry out our experiments, we created three different AS2 datasets, all including a large number of answer sentence candidates. Two of them are built using different samples of the anonymized questions from Alexa Information Traffic, while the third, ASNQ [8], is a sample from the Google Natural Question dataset [6], adapted for the AS2 task, which we further extended. We tested the combinations of fast rerankers, Jaccard similarity, Rel-CNN[1] and CA with an accurate BERT selector, and compared them with the upper bound, obtained with the expensive approach of classifying all candidates with the BERT model. The key finding of our paper is to show that the inference cost of Transformer models, *e.g.*, BERT, can be reduced by selecting only a promising candidate subset to be processed, still preserving the original Accuracy. That is, the candidates selected by shallow neural rerankers are compatible with those selected by Transformer models. This enables the design of accurate, fast and cost-effective QA systems, based on sequential rerankers.

2 RELATED WORK

Neural models for AS2 typically apply a series of non-linear transformations to the input question and answer, represented as compositions of word or character embeddings and then measure the

*This work was done while the author was an intern at Amazon Alexa.

[1]For lack of space we do not report its result, but it is between Jaccard and CA models.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8016-4/20/07.
https://doi.org/10.1145/3397271.3401266

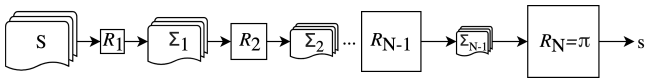

Figure 1: Diagram of the Sequential Rerankers

similarity between the obtained representations. For example, the Rel-CNN [16] has two separate embedding layers for the question and answer, and relational embedding, which aims at connecting them. Recent work has shown that Transformer-based models, *e.g.*, BERT [5], can highly improve inference. Yang et al. [23] applied it to Ad Hoc Document Retrieval, obtaining significant improvement. Garg et al. [8] fine-tuned BERT for AS2, achieving the state of the art. However, BERT's high computational cost prevents its use in most real-word applications. Some solutions rely on leveraging knowledge distillation in the pre-training step, *e.g.*, [15]. In contrast, we propose an alternative (and compatible with the initiatives above) approach following previous work in document retrieval, *e.g.*, the use of sequential rerankers [19]. Wang et al. [21] focused on quickly identifying a set of good candidate documents to be passed to the second and further rerankers of the cascade. Dang et al. [4] proposed two stage approaches using a limited set of textual features and a final model trained using a larger set of query- and document-dependent features. Gallagher et al. [7] presented a new general framework for learning an end-to-end cascade of rankers using backpropagation. Asadi and Lin [1] studied effectiveness/efficiency trade-offs with three candidate selection approaches. All the methods above are in line with our study, but they target very different settings: document retrieval, linear models or just basic neural models. In contrast, the main contribution of our paper is to show that simple and efficient neural sentence rerankers are compatible with expensive Transformer models, enabling their efficient use.

3 EFFICIENT PASSAGE RERANKING

In this section, we introduce the most famous neural models for AS2, as well as a state-of-the-art model based on BERT, and then, we present sequential architecture based on fast neural models and more expensive (but more accurate) models.

3.1 Neural rerankers

A reranker can be simply designed as a model for estimating the probability of correctness of a question and answer sentence pair, $p(q, s)$. Given a set of questions, Q, and a set of sentences, S, we define a reranker as $R : Q \times \mathcal{P}(S) \rightarrow \mathcal{P}(S)$, which takes a subset Σ of the sentences in S, and returns a subset of size $k < |\Sigma|$, *i.e.*, $R(q, \Sigma) = (s_{i1}, ..., s_{ik})$, where $p(q, s_{ih}) > p(q, s_{il})$, if $h < l$. We use the Compare-Aggregate (CA) model [22], which implements an attention mechanism between the question and the answer candidate with two-step operations, *i.e.*, comparison and aggregation.

Finally, we study the efficient use of Transformer models, which are neural networks designed to capture dependencies between words, *i.e.*, their interdependent context. The input is encoded in embeddings for tokens, segments and their positions. These are given in input to several blocks (up to 24), containing layers for multi-head attention, normalization and feed forward processing. The result of this transformation is an embedding, x, representing a text pair that models the dependencies between words and segments of the two sentences. We train these models for AS2 by feeding x to a fully connected layer, and fine-tuning the Transformer as shown by Devlin et al. [5]. For this step, we used question and sentence pairs labeled as positive or negative, whether or not the sentence correctly answers the question.

Table 1: Statistics of our test sets

	GPD	ASNQ	ASNQ^{++}
#Questions	454	2,672	2,672
Avg. #Sentences	1,445	354	1,310
Avg. #Correct Answers	6.98	1.29	1.87

3.2 Sequential rerankers

Given a sequence of rerankers sorted by their computational efficiency, $(R_1, R_2, .., R_N)$, we assume that the ranking Accuracy $\mathcal{A}$ (*e.g.*, in terms of MAP and MRR), increases in reverse order of the efficiency, *i.e.*, $\mathcal{A}(R_j) > \mathcal{A}(R_i)$ iff $j > i$.

We define a Sequential Reranker of order n, SR_n, as the composition of n rerankers: $\rho(S) = R_N \circ R_{N-1} \circ .. \circ R_1(S)$. Each R_i is associated with a different $k_i = |R_i(\cdot)|$, *i.e.*, the number of sentences the reranker returns (see Fig. 1). Setting different values for k_i can produce many valuable SRs with different trade-off between Accuracy and efficiency. The design of an end-to-end neural architecture that learns the optimal parameter set for the target trade-off is an interesting future research. In this paper, we focus on a simple SR_2 constituted by just two rerankers, $\text{SR}_2 = R_2 \circ R_1(S)$, where the first ranker is either a Jaccard similarity-based model or neural rankers such as CA, whereas the second reranker (selector) is the more expensive BERT model, *i.e.*, Base or Large versions.

4 EXPERIMENTS

We show that fast rerankers can effectively feed more expensive Transformer-based selectors, enabling the trade off between Accuracy and efficiency. We compare with state-of-the-art AS2 baselines on two different test sets, created with different question types from Alexa Virtual Assistant traffic. We also built an academic dataset to enable model replicability and comparison with future work.

4.1 Experimental setup

4.1.1 Metrics. The performance for QA systems is typically measured with Accuracy in providing correct answers, *i.e.*, the percentage of correct responses. Precision and Recall are not essential in our case as we assume the system does not abstain from providing answers. We also measure the system latency using the average end-to-end processing time for each question, including preprocessing, loading files, etc.[2].

4.1.2 Alexa Datasets. We built training and test sets using questions sampled from Alexa traffic (see statistics in Table 1). We only considered information inquiry questions. These ask about general knowledge, ranging from "Who is the president of United States?" to "How did Los Angeles Lakers score in the match yesterday?". We generate candidates by selecting sentences from the documents retrieved using an elastic search system, where the referring index

[2]Measured on an EC2-instance p3.2xlarge machine (8 Intel Xeon Scalable (Skylake) vCPUs, 62GB RAM, 1 GPU: NVIDIA Tesla V100)

Table 2: Model comparison on academic datasets. +LM+LC means Language Modeling and Latent Clustering.

Dataset	WikiQA		TRECQA	
Model	MAP	MRR	MAP	MRR
CA by [22]	0.743	0.754	–	–
CA+LM+LC by [25]	0.764	0.784	0.868	0.928
Our CA	0.715	0.725	0.804	0.853
Our BERT-Base	**0.813**	**0.828**	0.857	**0.937**

is ingested with several web domains, ranging from Wikipedia to *reference.com*, *coolantarctica.com*, *www.cia.gov/library* and so on.

General Purpose Dataset (GPD): We used the following strategy to retrieve correct candidates with high confidence: (i) retrieve top 100 documents; (ii) automatically extract the top 100 sentences ranked by a BERT model over all sentences of the documents; and (iii) manually annotate the top 100 sentences as correct or incorrect answers. This process does not guarantee that we have all correct answers but the chance to miss them is much smaller than for other datasets. Indeed, Table 1 shows an average of seven correct answers for each question. As some of the correct answers can be missed, we consider the GPD evaluation as a strict lower bound (SLB).

Sentence-Based Training Set (SBTS): A training set requires labels for all answer candidates. We limited the rank to just the top 10 sentence per question, to save annotation costs. SBTS consists of 25, 226 questions for a total of 134, 765 candidates, where 125, 779 were labelled as positive and 8, 986 as negative examples.

4.1.3 Academic Benchmark dataset. **Answer Sentence Natural Questions (ASNQ):** Famous benchmarks for AS2 such as TREC-QA [20] and WikiQA [24] do not come with an large enough number of annotated candidates to simulate a real application scenario. Thus, we used ASNQ[3] by Garg et al. [8], who transformed Google Natural Question (NQ) benchmark [6] for the AS2 task. ASNQ contains 57,242 and 2,672 distinct questions in the training and test sets, respectively: this is an order of magnitude larger than most public AS2 datasets. In our study, we only use it for testing. Finally, as the number of candidates is still lower than what can typically appear in an industrial scenario, we created ASNQ^{++} by simply adding negative candidates associated with different questions to reach 1,300 sentences for each question. Table 1 shows some statistics.

4.1.4 Models. We used the following models: Jaccard Similarity, CA, BERT-Base and BERT-Large to estimate $p(q, s_i)$ and used SBTS for training them (except for Jaccard similarity, which is an unsupervised approach). BERT-based models use SBTS for fine-tuning. We trained all the models using the log cross-entropy loss function: $\mathcal{L} = -\sum_{l \in \{0,1\}} y_l \times log(\hat{y}_l)$ on pairs of text. For CA, we applied the same design choices from the original papers except for (i) the use of 300 dimensional Glove embeddings [12], (ii) the Adam optimizer [9] with a learning rate, $lr = 0.0001$, and (iii) the batch size of 32. Additionally, we use early stopping to monitor MAP on the validation set. We set the hidden dimensions of the biLSTM in CA to 300. As for BERT, we tuned the learning rate in the interval [1e-6, 2e-5].

[3]ASNQ and ASNQ^{++} are available at https://github.com/alexa/wqa_tanda

Table 3: Performance of multi-reranker approach using BERT-Base as the selector derived on GPD

Reranker	Jaccard				CA	BERT
#Select. sent.	100	200	250	300	100	all
Latency (sec/q)	0.564	0.840	0.977	1.110	0.962	4.590
Prec.@1	0.270	0.376	0.382	0.411	0.450	0.476

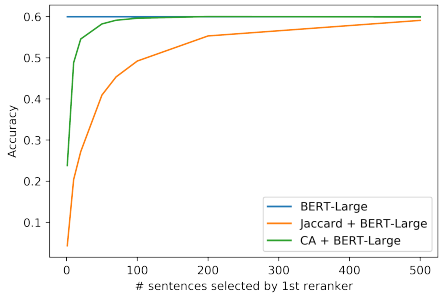

Figure 2: Accuracy of SR using BERT-Large and k sentences selected by the first reranker on ASNQ.

4.2 Results

We first assess our models on public datasets, then test the individual models as well as the combined models in terms of reranking performance on different datasets. In particular, we compute a SLB and the exact measures on Alexa and ASNQ datasets, respectively.

4.2.1 Model Assessment. As the datasets for testing the efficiency of the rerankers are new or partially new, to assess where our systems collocate in previous work, we compare our models with the best AS2 systems on standard datasets, WikiQA and TRECQA. Table 2 shows the results derived according to the standard *clean* setting on the official test sets. We can see that our CA implementation performs slightly lower than models from previous work. This depends on the fact that there is a large variation of the results according to different random seeds. In contrast, our BERT implementation is aligned to the state of the art, thus our comparison using Sequential Rerankers (SRs) is very meaningful.

4.2.2 Evaluation of SR on GPD. We tested SRs constituted by different models on GPD. Table 3 reports the Accuracy, where: (i) the first row indicates the approach for reranking, either Jaccard or CA, (ii) the second row shows several values for the number of sentences selected by Jaccard, while for CA, we selected a competitive number, *i.e.*, 100, (iii) the third row shows the latency of the SR models including, reranker time and selector applied to the k candidates; and finally, (iv) the last row shows the Precision@1 (*i.e.*, the SLB Accuracy) of SR. The BERT column just shows the upper bound Accuracy, *i.e.*, applying BERT to all candidates.

We note that: (i) the latency of the Jaccard-based SR is better than the SR based on CA, when we consider the same number of candidates, *e.g.*, 100, since CA execution time is not negligible, while Jaccard latency is tiny; (ii) the Accuracy of the Jaccard-based SR is half the one based on CA; (iii) also when increasing the number of candidates, the Jaccard-based SR cannot catch up with the Accuracy of CA-based SR using 100 candidates, whereas the latency quickly increases; (iv) the BERT upper bound is just 2.6% better, but the latency is 4.5 seconds, which is prohibitively high; and (v) we did

Table 4: Performance of SR with k = 100 sentences on ASNQ.

Selector	BERT-Base			BERT-Large		
Reranker	Jaccard	CA	All	Jaccard	CA	All
Latency (sec/q)	0.352	0.458	1.300	0.819	0.926	3.320
Acc.	0.474	0.569	0.571	0.492	0.596	0.599

not optimize CA in terms of efficiency and a better trade off to select a more appropriate k can be found.

4.2.3 Results on ASNQ. The evaluation on ASNQ is in principle more accurate as all candidates have been annotated[4]. Figure 2 shows the plot of three models: the blue line is the BERT-Large (Accuracy when it is applied to all sentence candidates), the green curve is the SR Accuracy when using k sentences from CA, and the orange curve is the Accuracy of SR-based on Jaccard similarity. Again, CA highly outperforms the Jaccard-based model. Interestingly, using 100 sentences or fewer allows for reaching the upper bound. Due to limited space, we cannot show a similar plot for BERT-Base, which reflects the same behaviour of BERT-Large (but with lower Accuracy).

To better show the potential of the SR approach, we report a configuration with 100 candidates selected by the first reranker in Table 4. We note that (i) SR based on Jaccard is almost 10 points below the upper bound (all candidates are used), for both selectors, BERT Base and Large; (ii) in contrast, SR based on CA is less than 0.3% below the upper bound. (iii) BERT-Large-based models are around three points better than those based on BERT-Base, as expected. (iv) Finally, there is a clear advantage in terms of latency, but the average number of candidates of ASNQ is lower than in a real setting, thus the speedup is just about 2-3 times for BERT-Base and 7-8 times for BERT-Large.

To better appreciate the gain in latency, we tested our models on ASNQ^{++} dataset, on which we added a larger number of negative examples. Table 5 shows that the latency of BERT-Base increases to 4.23 seconds, while the execution time of SR based on CA is still under 1 second. It is interesting to note that the Accuracy of the upper bound and SR models decreases comparing to the one obtained on ASNQ, *e.g.*, BERT-based obtains 0.571 vs. 0.536 on ASNQ and ASNQ^{++}, respectively. However, SR based on CA still reaches the target upper bound. Although, ASNQ^{++} is partially artificial, it provides results inline with GPD.

5 CONCLUSIONS

Very recent research work has shown that powerful neural models such as Transformer models can highly improve retrieval Accuracy. In particular, we show (confirming the results from ad hoc retrieval and our previous work [8]) that BERT can improve passage/sentence reranking in a QA setting. Unfortunately, in a real-world scenario requiring the processing of hundreds or thousands of candidates, this also leads to a latency of 4-5 seconds or more per question, even using an entire GPU. To tackle this problem, we applied SR models by limiting our study to two types of classifiers fast models such as, Jaccard similarity and CA, and slower but more accurate models, i.e., the Transformer (BERT Base and Large). We showed two important findings: (i) Simple neural rerankers such as CA can select candidates containing correct answers BERT would select. This is important as the data representation of the former is very different from the one of the latter. (ii) SR can be used to parameterize the trade off between efficiency and Accuracy, enabling the use of transformer models for real-world scenarios. Future ideas connected to our work are already in progress, *e.g.*, [2, 17].

[4]Considering that annotation process is always subject to annotator agreement.

Table 5: Performance of SR with k = 100 sentences on ASNQ^{++}.

Selector	BERT-Base		
Reranker	Jaccard	CA	All
Latency (sec/q)	0.580	0.924	4.230
Acc.	0.443	0.528	0.536

REFERENCES

[1] Nima Asadi and Jimmy Lin. 2013. Effectiveness/Efficiency Tradeoffs for Candidate Generation in Multi-stage Retrieval Architectures. In *SIGIR'13*.

[2] Daniele Bonadiman and Alessandro Moschitti. 2020. A Study on Efficiency, Accuracy and Document Structure for Answer Sentence Selection. *CoRR* abs/2003.02349 (2020). arXiv:2003.02349 https://arxiv.org/abs/2003.02349

[3] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *ACL*.

[4] Van Dang, Michael Bendersky, and W. Bruce Croft. 2013. Two-Stage Learning to Rank for Information Retrieval. In *ECIR 2013*.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*. 4171–4186.

[6] Tom Kwiatkowski et al. 2019. Natural Questions: A Benchmark for Question Answering Research. *TACL* 7 (2019), 453–466.

[7] Luke Gallagher, Ruey-Cheng Chen, Roi Blanco, and J. Shane Culpepper. 2019. Joint Optimization of Cascade Ranking Models. In *WSDM 2019*.

[8] Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2019. TANDA: Transfer and Adapt Pre-Trained Transformer Models for Answer Sentence Selection. arXiv:1911.04118 [cs.CL]

[9] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019).

[11] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *CoRR* abs/1901.04085 (2019). arXiv:1901.04085 http://arxiv.org/abs/1901.04085

[12] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP'14*. 1532–1543.

[13] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR* abs/1802.05365 (2018).

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[15] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108 [cs.CL]

[16] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR'15*.

[17] Luca Soldaini and Alessandro Moschitti. 2020. The Cascade Transformer: an Application for Efficient Answer Sentence Selection. *CoRR* abs/2005.02534 (2020).

[18] E. Strubell, A. Ganesh, and A. McCallum. 2019. Energy and Policy Considerations for Deep Learning in NLP. In *ACL*. 3645–3650.

[19] Lidan Wang, Jimmy Lin, and Donald Metzler. 2011. A Cascade Ranking Model for Efficient Ranked Retrieval. In *SIGIR*. 105–114.

[20] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL*.

[21] Qi Wang, Constantinos Dimopoulos, and Torsten Suel. 2016. Fast First-Phase Candidate Generation for Cascading Rankers. In *SIGIR'16*. 10.

[22] Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *Fifth International Conference on Learning Representations*.

[23] Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple Applications of BERT for Ad Hoc Document Retrieval. *CoRR* abs/1903.10972 (2019). arXiv:1903.10972 http://arxiv.org/abs/1903.10972

[24] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *EMNLP'15*.

[25] Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. In *CIKM*.