A Gated Self-attention Memory Network for Answer Selection

Tuan Lai [*1], **Quan Hung Tran** [*2], **Trung Bui** [2], **Daisuke Kihara** [1]
{lai123,dkihara}@purdue.edu, {qtran,bui}@adobe.com
[1] Purdue University, West Lafayette, IN
[2] Adobe Research, San Jose, CA

Abstract

Answer selection is an important research problem, with applications in many areas. Previous deep learning based approaches for the task mainly adopt the Compare-Aggregate architecture that performs word-level comparison followed by aggregation. In this work, we take a departure from the popular Compare-Aggregate architecture, and instead, propose a new gated self-attention memory network for the task. Combined with a simple transfer learning technique from a large-scale online corpus, our model outperforms previous methods by a large margin, achieving new state-of-the-art results on two standard answer selection datasets: TrecQA and WikiQA.

1 Introduction and Related Work

Answer selection is an important task, with applications in many areas (Lai et al., 2018). Given a question and a set of candidate answers, the task is to identify the most relevant candidate. Previous work on answer selection typically relies on feature engineering, linguistic tools, or external resources (Wang et al., 2007; Wang and Manning, 2010; Heilman and Smith, 2010; Yih et al., 2013; Yao et al., 2013). Recently, with the renaissance of neural network models, many deep learning based methods have been proposed to address the task (Tay et al., 2017b; Shen et al., 2017; Wang et al., 2017; Bian et al., 2017; Tymoshenko and Moschitti, 2018; Tay et al., 2018; Tayyar Madabushi et al., 2018; Yoon et al., 2019). They outperform traditional techniques. A common trait of a number of these deep learning methods is the use of the Compare-Aggregate architecture (Wang and Jiang, 2017). Typically in this architecture, contextualized vector representations of small units such as words of the question and the candidate are first *compared and aligned*. After that, these comparison results are then *aggregated* to calculate a score indicating the relevance between the question and the candidate. On standard answer selection datasets such as TrecQA (Wang et al., 2007) or WikiQA (Yang et al., 2015), Compare-Aggregate approaches achieve very competitive performance. However, they still have some limitations. For example, the first few layers of most previous Compare-Aggregate models encode the question-candidate pair into sequences of contextualized vector representations separately (Wang et al., 2017; Shen et al., 2017; Bian et al., 2017). These sequences are independent and completely ignore the information from the other sequence.

In this work, we take a departure from the popular Compare-Aggregate architecture, so instead, we propose a mix between two very successful architectures in machine comprehension and sequence modeling, the memory network (Sukhbaatar et al., 2015) and the self-attention architecture (Vaswani et al., 2017). In the context of answer selection, the self-attention architecture allows us to learn the contextual representation of elements in the sequence with respect to both the question and the answer, while the multi-hop reasoning memory network allows us to refine the decision over multiple steps. To this end, we propose a new memory-based, gated self-attention architecture for the task of answer selection. Combined with a simple transfer learning technique from a large-scale online corpus, our model achieves new state-of-the-art results on the TrecQA and WikiQA datasets.

In the following parts, we first describe our gated self-attention memory network for answer selection in Section 2. We then go into details our transfer learning approach in Section 3. After that, we describe the conducted experiments and their results in Section 4. Finally, we conclude this

[*]Equal contributions. The work was conducted while the first author interned at Adobe Research.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 5953–5959, Hong Kong, China, November 3–7, 2019.

work in Section 5.

2 Gated Self-Attention Memory Network

2.1 The gated self-attention mechanism

The gated attention mechanism (Dhingra et al., 2017; Tran et al., 2017) extends the popular scalar-based attention mechanism by calculating a real vector gate to control the flow of information, instead of a scalar value. Let's denote the sequence of input vectors as $X = [\mathbf{x}_1..\mathbf{x}_n]$. If we have context information $\mathbf{c}$, then in traditional attention mechanism, association score α_i is usually calculated as a normalized dot product between the two vectors $\mathbf{c}$ and $\mathbf{x}_i$ (Equation 1) where $i \in [1..n]$.

$$\alpha_i = \frac{\exp(\mathbf{c}^T \mathbf{x}_i)}{\sum_{j \in [1..n]} \exp(\mathbf{c}^T \mathbf{x}_j)} \tag{1}$$

For the gated attention mechanism, the association between two vectors $\mathbf{c}$ and $\mathbf{x}_i$ is represented by gate vector $\mathbf{g}_i$ as follows:

$$\mathbf{g}_i = \sigma\big(f(\mathbf{c}, \mathbf{x}_i)\big) \tag{2}$$

where σ denotes the element-wise sigmoid function. Function f is a parameterized function and thus, is more flexible in modelling the interaction between vectors $\mathbf{c}$ and $\mathbf{x}_i$.

In this work, we propose a new type of self-attention based on the gated attention mechanism described above, and we refer to it as the *gated self-attention* mechanism (GSAM). We want to condition the gate vector not only on a context vector and a single input vector but also on the entire sequence of inputs. Therefore, we design function f to be dependent on all the inputs in the sequence and the context vector. To calculate the gate for input $\mathbf{x}_i$, first, each of the inputs in the input sequence and the context vector will present an individual gate "vote". Then, the votes are aggregated to calculate gate $\mathbf{g}_i$ for $\mathbf{x}_i$. This process is illustrated in Equation 3:

$$\begin{aligned}
\mathbf{v}^j &= \mathbf{W}\mathbf{x}_j + \mathbf{b} \;;\; \mathbf{v}^c = \mathbf{W}\mathbf{c} + \mathbf{b} \\
s_i^j &= \mathbf{x}_i^T \mathbf{v}^j \;;\; s_i^c = \mathbf{x}_i^T \mathbf{v}^c \\
\alpha_i^j &= \frac{\exp(s_i^j)}{\sum_{k \in [1..n]} \exp(s_i^k) + \exp(s_i^c)} \\
\alpha_i^c &= \frac{\exp(s_i^c)}{\sum_{k \in [1..n]} \exp(s_i^k) + \exp(s_i^c)} \\
\mathbf{g}_i &= f_i(c, X) \\
&= \sigma\left(\sum_j \left(\alpha_i^j \, \mathbf{x}^j \right) + \alpha_i^c \, \mathbf{c} \right)
\end{aligned} \tag{3}$$

where $\mathbf{W}$ and $\mathbf{b}$ are learnable parameters shared among functions $f_1 ... f_n$. Vectors $\mathbf{v}$s are linear-transformed inputs which are used to calculate the self attentions. s_i^j is the unnormalized attention score of input $\mathbf{x}_j$ put on $\mathbf{x}_i$ and α_i^j is the normalized score. We use affine-transformed inputs $\mathbf{v}$ and $\mathbf{x}$ to calculate the self-attention instead of just $\mathbf{x}$ to break the attention symmetry phenomenon.

2.2 Combining with the memory network

In most previous memory network architectures, interactions between memory cells are relatively limited. At each hop, a single control vector is used to interpret each memory cell independently. To overcome this limitation, we combine GSAM described in Section 2.1 with the memory network architecture to create a new network called the Self-Attention Memory Network (GSAMN). Figure 1 shows the simplified computation flow of GSAMN. In each reasoning hop, instead of using only context vector $\mathbf{c}$ to interpret the inputs, we use GSAM. Let $\mathbf{c}_k$ be the controlling context and $\mathbf{x}_1^k ... \mathbf{x}_n^k$ be the memory values at the k^{th} reasoning hop. Each memory cell update from the k^{th} hop to the next hop is calculated as the gated self-attention update (Equation 4).

$$\begin{aligned}
\mathbf{g}_i &= f_i(\mathbf{c}_k, X) \\
\mathbf{x}_i^{k+1} &= \mathbf{g}_i \odot \mathbf{x}_i^k
\end{aligned} \tag{4}$$

The controller's update is a combination of the gated self-attention above, and the memory network's traditional aggregate update. As the memory state values have already been attended to by the gating mechanism, we only need to average them (not weighted average).

$$\begin{aligned}
\mathbf{g}_c &= f_c(\mathbf{c}_k, X) \\
\mathbf{c}_{k+1} &= \mathbf{g}_c \odot \mathbf{c}_k + \frac{1}{n} \sum_i \mathbf{x}_i^{k+1}
\end{aligned} \tag{5}$$

.

2.3 GSAMN for answer selection

In the context of answer selection, we concatenate question Q and candidate answer A to a single sequence and treat the task as a binary classification problem. Given the GSAMN architecture above, we can use final controller state $\mathbf{c}_T$ as the representation of the sequence. The matching probability $P(A \mid Q)$ is finally calculated as follows:

$$P(A \mid Q) = \sigma\Big(\mathbf{W}_c \, \mathbf{c}_T + \mathbf{b}_c \Big) \tag{6}$$

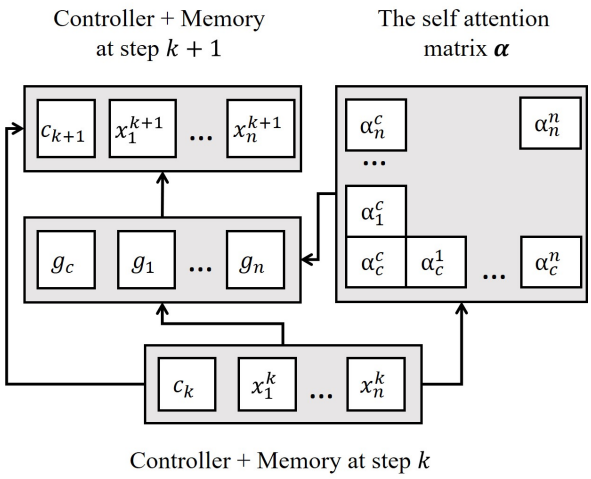

Figure 1: Simplified computation flow of the Gated Self-Attention Memory Network

where $\mathbf{W}_c$ and $\mathbf{b}_c$ are learnable parameters. We can initialize the memory values $\mathbf{x}_1^0 \ldots \mathbf{x}_n^0$ using any representation model such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018), or BERT (Devlin et al., 2018). The control vector $\mathbf{c}$ is a randomly initialized learnable vector.

3 Transfer Learning

Previous studies on answer selection have focused mostly on small-scale datasets. On the other hand, many community question answering (CQA) platforms such as Yahoo Answers and Stack Exchange have become an essential source of information for many people. The amount of data (i.e., questions and answers) in these CQA platforms is huge and encompasses many domains and topics. This provides a great opportunity to apply transfer learning techniques to improve answer selection systems trained on limited datasets.

We crawled question-answer pairs related to various topics from Stack Exchange [1]. After that, we removed every pair that contains text written in a language different from English. Furthermore, to ensure that in each collected pair the answer is highly relevant to the question, we removed pairs whose answers have less than two up-votes from community users. Finally, because training answer selection models also require negative examples, for each question, we sampled several real answers not related to the question to build up negative pairs. In the end, our dataset has 628,706 positive pairs and 9,874,758 negative pairs in total. We refer to our newly collected dataset as StackExchangeQA. Table 1 shows some examples of positive question-answer pairs from the dataset. The dataset has question-answer pairs from many different domains and topics. The code for constructing the StackExchangeQA dataset is available online [2].

In this work, we employ a basic transfer learning technique. The first step is to pre-train our answer selection model on the StackExchangeQA dataset. Then, the second step is to fine-tune the same model on a target dataset of interest such as TrecQA or WikiQA. Despite the simplicity of the technique, the performance of our model improves substantially compared to not using transfer learning. Different from previous works which use source datasets that were manually annotated (Min et al., 2017; Chung et al., 2018), our source dataset required minimal effort to obtain and preprocess. The choice of crawling question-answer pairs from the Stack Exchange website was arbitrary. We could also have crawled data from websites such as Yahoo Answers instead.

4 Experiments and Results

To evaluate the effectiveness of our proposed answer selection model, we use two datasets: TrecQA and WikiQA. The TrecQA dataset (Wang et al., 2007) was created from the TREC Question Answering tracks. There are two versions of TrecQA: raw and clean. Both versions have the same training set but their development and test sets differ. In this study, we use the clean version of the dataset that removed questions in development and test sets with no answers or only positive/negative answers. The clean version has 1,229/65/68 questions and 53,417/1,117/1,442 question-answer pairs for the train/dev/test split. The WikiQA dataset (Yang et al., 2015) was constructed from real queries of Bing and Wikipedia. Following the literature (Yang et al., 2015; Bian et al., 2017; Shen et al., 2017), we removed all questions with no correct answers before training and evaluating answer selection models. The excluded WikiQA has 873/126/243 questions and 8,627/1,130/2,351 question-answer pairs for train/dev/test split.

Similar to previous work, we report the model

[1] https://stackexchange.com/

[2] https://github.com/laituan245/StackExchangeQA

Domain	QA Pair
Academia	**Question**: Is it okay for a PhD student to go on holidays in breaks? **Answer**: Do you have an adviser? Have you talked to them about this? Most should be fine with you taking some time off to visit your family, but you should probably discuss longer breaks with them to work out all the details.
Apple Product	**Question**: What hidden features have you found in iOS 6? **Answer**: Newly downloaded apps have a "new" label on the home screen.
Chemistry	**Question**: Hydrochloric acid vs hydrogen chloride? **Answer**: Hydrochloric acid is an aqueous solution of hydrogen chloride.
Cooking	**Question**: How do I prevent tomatoes from falling in a green salad? **Answer**: I work around this by serving tomatoes on the top of the individual salads after they've been portioned out. I'm not sure of a way to keep them incorporated.
Philosophy	**Question**: What did Socrates teach which lead to his conviction that he spoiled youth and taught other Gods? **Answer**: I think in general one of the problems Socrates' contemporaries may have had with him was not so much what he taught but how he taught. Perhaps Socrates' method of philosophy was characterised more by testing propositions through questioning, than any strict concern with formulating a set of propositions on any one subject.

Table 1: Examples of positive question-answer pairs from the StackExchangeQA dataset

performance as the mean average precision (MAP) and mean reciprocal rank (MRR) [3]. In all experiments, we use the base version of BERT (Devlin et al., 2018) to initialize the memory of our proposed architecture. We fine-tune the BERT embeddings during training. We set the number of reasoning hops to be 2. We use the Adam optimizer with a learning rate of 5e-5, $\beta_1 = 0.9$, $\beta_2 = 0.999$, L2 weight decay of 0.01, learning rate warmup over the first 10 percent of the total number of training steps, and linear decay of the learning rate. We did hyper-parameter tuning on the development sets.

It is worth noting that, we have experimented with various values for the number of reasoning hops. We found that using 2 hops gives the best performance on the tested datasets while using larger number of hops decreases the performance slightly. We attribute the diminishing returns in increasing the number of hops to the limited size of the TrecQA and WikiQA datasets. Many previous works related to memory networks also use small number of memory hops (Weston et al., 2015; Sukhbaatar et al., 2015; Miller et al., 2016; Zhang et al., 2018).

4.1 Comparison with Previous Methods

Table 2 summarizes the performances of our proposed models and compares them to the baselines on the TrecQA and WikiQA datasets. The full model [BERT + GSAMN+ Transfer Learning] outperforms the previous state-of-the-art methods by a large margin. Note that by simply fine-tuning the pre-trained BERT embeddings, one can easily achieve very competitive performance on both datasets. This is expected as BERT has been pre-trained on a massive amount of unlabeled data. However, our proposed techniques do add a significant amount of performance. The gain from using all of our proposed techniques is larger than the difference between fine-tuning BERT model compared to previous systems in the TrecQA dataset.

4.2 Ablation Analysis

We aim to analyze the relative effectiveness of different components of our full model. From the original BERT baseline, we add one component at a time and evaluate the performance of the partial models on the datasets. From Table 2, we can see that both the variants [BERT + GSAMN] and [BERT + Transfer Learning] have better performance than the original BERT baseline. However, both of the partial variants still perform worse than the one with all the techniques. This shows that although each of our proposed components is effective by itself, we need to combine them together in order to achieve the best performance.

4.3 GSAMN versus Transformer

It was an iterative process to arrive at the current design of GSAMN. We aim to analyze whether the improvement in performance comes from the inductive bias that we introduced into the architec-

[3] https://aclweb.org/aclwiki/Question_Answering_(State_of_the_art)

Model	TrecQA		WikiQA	
	MAP	MRR	MAP	MRR
BERT + GSAMN+ Transfer	**0.914**	**0.957**	**0.857**	**0.872**
BERT + Transformers + Transfer	0.895	0.939	0.831	0.848
BERT + GSAMN	0.906	0.949	0.821	0.832
BERT + Transformers	0.886	0.926	0.813	0.828
ELMo + Compare-Aggregate	0.850	0.898	0.746	0.762
BERT + Transfer	0.902	0.949	0.832	0.849
BERT	0.877	0.922	0.810	0.827
QC + PR + MP CNN (2018)	0.865	0.904	—	—
IWAN + sCARNN (2018)	0.829	0.875	0.716	0.722
IWAN (2017)	0.822	0.889	0.733	0.750
Compare-Aggregate (2017)	0.821	0.899	0.748	0.758
BiMPM (2017)	0.802	0.875	0.718	0.731
HyperQA (2017a)	0.784	0.865	0.705	0.720
NCE-CNN (2016)	0.801	0.877	—	—
Attentive Pooling CNN (2016)	0.753	0.851	0.689	0.696
W&I (2015)	0.746	0.820	—	—

Table 2: Results on the TrecQA and WikiQA datasets

ture or simply from having more parameters due to added complexity. To this end, we evaluated the performances of two variants [BERT + Transformers] and [BERT + Transformers + Transfer Learning]. These model variants simply use two Transformer layers (Vaswani et al., 2017) on top of BERT instead of using our GSAMN architecture. Table 2 clearly shows that GSAMN outperforms the Transformer based variants, with or without the transfer learning component.

We have experimented with adding more Transformer layers on top of BERT but the performance did not improve. For example, using 6 extra Transformer layers only achieves a MAP score of 0.885 on the TrecQA dataset. This is reasonable because BERT by itself already contains 12 Transformer layers. Without a new kind of layer such as our proposed GSAMN architecture, stacking more Transformer layers will not be helpful, especially in this case where the tested datasets are not large.

4.4 GSAMN versus Compare-Aggregate

Finally, we have a comparison between our full model and the Compare-Aggregate framework. Most previous Compare-Aggregate architectures use traditional word embeddings such as word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014). In contrast, our full model uses BERT which is an arguably more powerful language representation model. To this end, we implemented a Compare-Aggregate variant that uses dynamic-clip attention (Bian et al., 2017). We use ELMo (Peters et al., 2018) to represent the input words to the implemented Compare-Aggregate architecture. We use ELMo instead of BERT because BERT is in subword-level while one of the intuitions behind the Compare-Aggregate variant is about comparing word-level representations. In addition, we have tested the variant [BERT + Compare-Aggregate] but found it to be worse than the version [ELMo + Compare-Aggregate]. The results in Table 2 show that our model significantly outperforms [ELMo + Compare-Aggregate] as well.

5 Conclusions

In this paper, we propose a new gated self-attention memory network architecture for answer selection. Combined with a simple transfer learning technique from a large-scale CQA corpus, the model achieves the state-of-the-art performance on two well-studied answer selection datasets: TrecQA and WikiQA. In the future, we plan to investigate more transfer learning techniques for utilizing the large volume of existing CQA data. In addition, we plan to apply our self-attention memory network on other sentence matching tasks such as natural language inference, paraphrase identification, or measuring semantic relatedness.

References

Weijie Bian, Si Kan Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *CIKM*.

Yu-An Chung, Hung yi Lee, and James R. Glass. 2018. Supervised and unsupervised transfer learning for question answering. In *NAACL-HLT*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Bhuwan Dhingra, Hanxiao Liu, William W. Cohen, and Ruslan R. Salakhutdinov. 2017. Gated-attention readers for text comprehension. In *ACL*.

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1011–1019, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tuan Manh Lai, Trung Bui, and Sheng Li. 2018. A review on deep learning techniques applied to answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2132–2144, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *EMNLP*.

Sewon Min, Min Joon Seo, and Hannaneh Hajishirzi. 2017. Question answering through transfer learning from large fine-grained supervision data. In *ACL*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*.

Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1913–1916. ACM.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *ArXiv*, abs/1602.03609.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017. Inter-weighted alignment network for sentence pair modeling. In *EMNLP*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2017a. Enabling efficient question answer retrieval via hyperbolic neural networks. *CoRR*, abs/1707.07847.

Yi Tay, Minh C. Phan, Anh Tuan Luu, and Siu Cheung Hui. 2017b. Learning to rank question answer pairs with holographic dual lstm architecture. In *SIGIR*.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Multi-cast attention networks. In *KDD*.

Harish Tayyar Madabushi, Mark Lee, and John Barnden. 2018. Integrating question classification and deep learning for improved answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Quan Hung Tran, Gholamreza Haffari, and Ingrid Zukerman. 2017. A generative attentional neural network model for dialogue act classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 524–529.

Quan Hung Tran, Tuan Lai, Gholamreza Haffari, Ingrid Zukerman, Trung Bui, and Hung Bui. 2018. The context-dependent additive recurrent neural net. In *NAACL-HLT*.

Kateryna Tymoshenko and Alessandro Moschitti. 2018. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Mengqiu Wang and Christopher D. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1164–1172, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *EMNLP-CoNLL*.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. *CoRR*, abs/1611.01747.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *IJCAI*.

Zhiguo Wang and Abraham Ittycheriah. 2015. Faq-based question answering via word alignment. *ArXiv*, abs/1507.02628.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. *CoRR*, abs/1410.3916.

Yi Yang, Wen tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*.

Xuchen Yao, Benjamin Van Durme, Chris Callison-burch, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Scott Wen-tau Yih, Ming-Wei Chang, Chris Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. ACL Association for Computational Linguistics.

Seunghyun Yoon, Franck Dernoncourt, Deok Seong Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. *ArXiv*, abs/1905.12897.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *ACL*.