Training a Ranking Function for Open-Domain Question Answering

Phu Mon Htut[1]
pmh330@nyu.edu

Samuel R. Bowman[1,2,3]
bowman@nyu.edu

Kyunghyun Cho[1,3]
kyunghyun.cho@nyu.edu

[1]Center for Data Science
New York University
60 Fifth Avenue
New York, NY 10011

[2]Dept. of Linguistics
New York University
10 Washington Place
New York, NY 10003

[3]Dept. of Computer Science
New York University
60 Fifth Avenue
New York, NY 10011

Abstract

In recent years, there have been amazing advances in deep learning methods for machine reading. In machine reading, the machine reader has to extract the answer from the given ground truth paragraph. Recently, the state-of-the-art machine reading models achieve human level performance in SQuAD which is a reading comprehension-style question answering (QA) task. The success of machine reading has inspired researchers to combine information retrieval with machine reading to tackle open-domain QA. However, these systems perform poorly compared to reading comprehension-style QA because it is difficult to retrieve the pieces of paragraphs that contain the answer to the question. In this study, we propose two neural network rankers that assign scores to different passages based on their likelihood of containing the answer to a given question. Additionally, we analyze the relative importance of semantic similarity and word level relevance matching in open-domain QA.

1 Introduction

The goal of a question answering (QA) system is to provide a relevant answer to a natural language question. In reading comprehension-style QA, the ground truth paragraph that contains the answer is given to the system whereas no such information is available in open-domain QA setting. Open-domain QA systems have generally been built upon large-scale structured knowledge bases, such as Freebase or DBpedia. The drawback of this approach is that these knowledge bases are not complete (West et al., 2014), and are expensive to construct and maintain.

Another method for open-domain QA is a corpus-based approach where the QA system looks for the answer in the unstructured text corpus (Brill et al., 2001). This approach eliminates the need to build and update knowledge bases by taking advantage of the large amount of text data available on the web. Complex parsing rules and information extraction methods are required to extract answers from unstructured text. As machine readers are excellent at this task, there have been attempts to combine search engines with machine reading for corpus-based open-domain QA (Chen et al., 2017; Wang et al., 2017). To achieve high accuracy in this setting, the top documents retrieved by the search engine must be relevant to the question. As the top ranked documents returned from search engine might not contain the answer that the machine reader is looking for, re-ranking the documents based on the likelihood of containing answer will improve the overall QA performance. Our focus is on building a neural network ranker to re-rank the documents retrieved by a search engine to improve overall QA performance.

Semantic similarity is crucial in QA as the passage containing the answer may be semantically similar to the question but may not contain the exact same words in the question. For example, the answer to "*What country did world cup 1998 take place in?*" can be found in "*World cup 1998 was held in France*." Therefore, we evaluate the performance of the fixed size distributed representations that encode the general meaning of the whole sentence on ranking. We use a simple feed-forward neural network with fixed size question and paragraph representations for this purpose.

In ad-hoc retrieval, the system aims to return a list of documents that satisfies the user's information need described in the query.[1] Guo et al. (2017) show that, in ad-hoc retrieval, rele-

[1]Information Retrieval Glossary: http://people.ischool.berkeley.edu/ hearst/irbook/glossary.html

Proceedings of NAACL-HLT 2018: Student Research Workshop, pages 120–127
New Orleans, Louisiana, June 2 - 4, 2018.

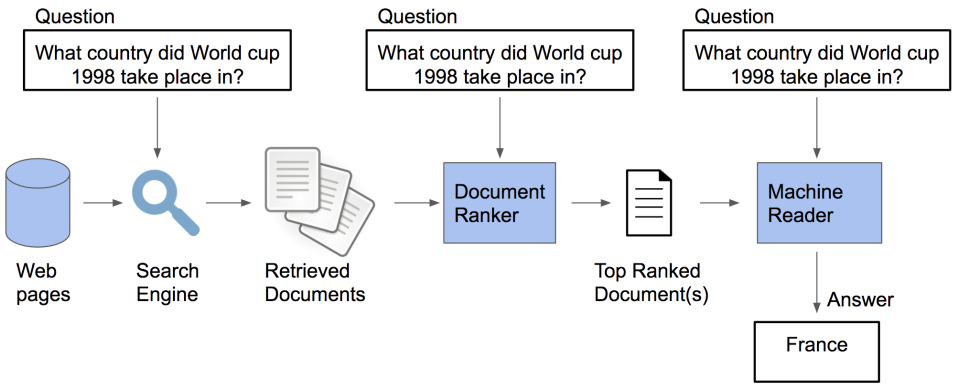

Figure 1: The overall pipeline of the open-domain QA model

vance matching—identifying whether a document is relevant to a given query—matters more than semantic similarity. Unlike semantic similarity, that measures the overall similarity in meaning between a question and a document, relevance matching measures the word or phrase level local interactions between pieces of texts in a question and a document. As fixed size representations encode the general meaning of the whole sentence or document, they lose some distinctions about the keywords that are crucial for retrieval and question answering. To analyze the importance of relevance matching in QA, we build another ranker model that focuses on local interactions between words in the question and words in the document. We evaluate and analyze the performance of the two rankers on QUASAR-T dataset (Dhingra et al., 2017b). We observe that the ranker model that focuses on relevance matching (Relation-Networks ranker) achieves significantly higher retrieval recall but the ranker model that focuses on semantic similarity (InferSent ranker) has better overall QA performance. We achieve 11.6 percent improvement in overall QA performance by integrating InferSent ranker (6.4 percent improvement by Relation-Networks ranker).

2 Related Work

With the introduction of large-scale datasets for machine reading such as CNN/DailyMail (Hermann et al., 2015) and The Stanford Question Answering Dataset (SQuAD; Rajpurkar et al., 2016), the machine readers have become increasingly accurate at extracting the answer from a given paragraph. In machine reading-style question answering datasets like SQuAD, the system has to locate the answer to a question in the given ground truth paragraph. Neural network based models excel at this task and have recently achieved human level accuracy in SQuAD.[2]

Following the advances in machine reading, researchers have begun to apply Deep Learning in corpus-based open-domain QA approach by incorporating information retrieval and machine reading. Chen et al. (2017) propose a QA pipeline named DrQA that consists of a Document Retriever and a Document Reader. The Document Retriever is a TF-IDF retrieval system built upon Wikipedia corpus. The Document Reader is a neural network machine reader trained on SQuAD. Although DrQA's Document Reader achieves the exact match accuracy of 69.5 in reading comprehension-style QA setting of SQuAD, their accuracy drops to 27.1 in the open-domain setting, when the paragraph containing the answer is not given to the reader. In order to extract the correct answer, the system should have an effective retrieval system that can retrieve highly relevant paragraphs. Therefore, retrieval plays an important role in open-domain QA and current systems are not good at it.

To improve the performance of the overall pipeline, Wang et al. (2017) propose Reinforced Ranker-Reader (R^3) model. The pipeline of R^3 includes a Lucene-based search engine, and a neural network ranker that re-ranks the documents retrieved by the search engine, followed by a machine reader. The ranker and reader are trained jointly using reinforcement learning. Qualitative analysis of top ranked documents by the ranker of R^3 shows that the neural network ranker can learn

[2]SQuAD leaderboard: https://rajpurkar.github.io/SQuAD-explorer/

to rank the documents based on semantic similarity with the question as well as the likelihood of extracting the correct answer by the reader.

We follow a similar pipeline as Wang et al. (2017). Our system consists of a neural network ranker and a machine reader as shown in Figure 1. The focus of our work is to improve the ranker for QA performance. We use DrQA's Document Reader as our reader. We train our ranker and reader models on QUASAR-T (Dhingra et al., 2017b) dataset. QUASAR-T provides a collection top 100 short paragraphs returned by search engine for each question in the dataset. Our goal is to find the correct answer span for a given question.

3 Model Architecture

3.1 Overall Setup

The overall pipeline consists of a search engine, ranker and reader. We do not build our own search engine as QUASAR-T provides 100 short passages already retrieved by the search engine for each question. We build two different rankers: **InferSent ranker** to evaluate the performance of semantic similarity in ranking for QA, and **Relation-Networks ranker** to evaluate the performance of relevance matching in ranking for QA. We use the Document Reader of DrQA (Chen et al., 2017) as our machine reader.

3.2 Ranker

Given a question and a paragraph, the ranker model acts as a scoring function that calculates the similarity between them. In our experiment, we explore two neural network models as the scoring functions of our rankers: a feed-forward neural network that uses InferSent sentence representations (Conneau et al., 2017), and Relation-Networks (Santoro et al., 2017). We train the rankers by minimizing the margin ranking loss (Bai et al., 2010):

$$\sum_{i=1}^{k} max(0, 1 - f(q, p_{pos}) + f(q, p^i_{neg})) \quad (1)$$

where f is the scoring function, p_{pos} is a paragraph that contains the ground truth answer, p_{neg} is a negative paragraph that does not contain the ground truth answer, and k is the number of negative paragraphs. We declare paragraphs that contain the exact ground truth answer string provided in the dataset as positive paragraphs. For every question, we sample one positive paragraph and five negative paragraphs. Given a question and a list of paragraphs, the ranker will return the similarity scores between the question and each of the paragraphs.

3.2.1 InferSent Ranker

InferSent (Conneau et al., 2017) provides distributed representations for sentences.[3] It is trained on Stanford Natural Language Inference Dataset (SNLI; Bowman et al., 2015) and Multi-Genre NLI Corpus (MultiNLI; Williams et al., 2017) using supervised learning. It generalizes well and outperforms unsupervised sentence representations such as Skip-Thought Vectors (Kiros et al., 2015) in a variety of tasks.

As InferSent representation captures the general semantics of a sentence, we use it to implement the ranker that ranks based on semantic similarity. To compose sentence representations into a paragraph representation, we simply sum the InferSent representations of all the sentences in the paragraph. This approach is inspired by the sum of word representations as composition function for forming sentence representations (Iyyer et al., 2015).

We implement a feed-forward neural network as our scoring function. The input feature vector is constructed by concatenating the question embedding, paragraph embedding, their difference, and their element-wise product (Mou et al., 2016):

$$\boldsymbol{x}_{classifier} = \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{p} \\ \boldsymbol{q} - \boldsymbol{p} \\ \boldsymbol{q} \bigodot \boldsymbol{p} \end{bmatrix} \quad (2)$$

$$\boldsymbol{z} = \boldsymbol{W}^{(1)} \boldsymbol{x}_{classifier} + \boldsymbol{b}^{(1)} \quad (3)$$

$$score = \boldsymbol{W}^{(2)} ReLU(\boldsymbol{z}) + \boldsymbol{b}^{(2)} \quad (4)$$

The neural network consists of a linear layer followed by a ReLU activation function, and another scalar-valued linear layer that provides the similarity score between a question and a paragraph.

3.2.2 Relation-Networks (RN) Ranker

We use Relation-Networks (Santoro et al., 2017) as the ranker model that focuses on measuring the relevance between words in the question and words in the paragraph. Relation-Networks are designed to infer the relation between object pairs.

[3] https://github.com/facebookresearch/InferSent

In our model, the object pairs are the question word and context word pairs as we want to capture the local interactions between words in the question and words in the paragraph. The word pairs will be used as input to the Relation-Networks:

$$RN(q,p) = f_\phi\Big(\sum_{i,j} g_\theta([E(q_i); E(p_j)])\Big) \quad (5)$$

where $q = \{q_1, q_2, ..., q_n\}$ is the question that contains n words and $p = \{p_1, p_2, ..., p_m\}$ is the paragraph that contains m words; $E(q_i)$ is a 300 dimensional GloVe embedding (Pennington et al., 2014) of word q_i, and $[\cdot;\cdot]$ is the concatenation operator. f_ϕ and g_θ are 3 layer feed-forward neural networks with ReLU activation function.

The role of g_θ is to infer the relation between two words while f_ϕ serves as the scoring function. As we directly compare the word embeddings, this model will lose the contextual information and word order, which can provide us some semantic information. We do not fine-tune the word embeddings during training as we want to preserve the generalized meaning of GloVe embeddings. We hypothesize that this ranker will achieve a high retrieval recall as relevance matching is important for information retrieval (Guo et al., 2017).

3.3 Machine Reader

The Document Reader of DrQA (Chen et al., 2017) is a multi-layer recurrent neural network model that is designed to extract an answer span to a question from a given document (or paragraphs). We refer readers to the original work for details. We apply the default configuration used in the original work, and train the DrQA on QUASAR-T dataset.[4] As QUASAR-T does not provide the ground truth paragraph for each question, we randomly select 10 paragraphs that contain the ground truth answer span for each question, and use them to train the DrQA reader.

3.4 Paragraph Selection

The ranker provides the similarity score between a question and each paragraph in the article. We select the top 5 paragraphs based on the scores provided by the ranker. We use soft-max over the top 5 scores to find $P(p_j^i)$, the model's estimate of the probability that the passage is the most relevant one from among the top 5.

Furthermore, the machine reader provides the probability of each answer span given a paragraph $P(answer_j|p_j^i)$, where $answer_j$ stands for the answer span of j^{th} question in dataset and p_j^i indicates the corresponding top 5 paragraphs. We can thus calculate the overall confidence of each answer span and corresponding paragraph $P(p_j^i, answer_j)$ by multiplying $P(answer_j|p_j^i)$ with $P(p_j^i)$. We then choose the answer span with the highest $P(answer_j,\ p_j^i)$ as the output of our model.

4 Experimental Setup

4.1 QUASAR Dataset

The QUestion Answering by Search And Reading (QUASAR) dataset (Dhingra et al., 2017b) includes QUASAR-S and QUASAR-T, each designed to address the combination of retrieval and machine reading. QUASAR-S consists of fill-in-the-gaps questions collected from Stackoverflow using software entity tags. As our model is not designed for fill-in-the-gaps questions, we do not use QUASAR-S. QUASAR-T, which we use, consists of 43,013 open-domain questions based on trivia, collected from various internet sources. The candidate passages in this dataset are collected from a Lucene based search engine built upon ClueWeb09.[5,6]

4.2 Baselines

We consider four models with publicly available results for QUASAR-T dataset. GA: Gated Attention Reader (Dhingra et al., 2017a), BiDAF: Bidirectional Attention Flow (Seo et al., 2016), R^3: Reinforced Ranker-Reader (Wang et al., 2017) and SR^2: Simple Ranker-Reader (Wang et al., 2017) which is a variant of R^3 that jointly trains the ranker and reader using supervised learning.

4.3 Implementation Details

Each InferSent embedding has 4096 dimensions. Therefore, the input feature vector to our InferSent ranker has 16384 dimensions. The dimensions of the two linear layers are 500 and 1.

As for Relation-Networks (RN), g_θ and f_ϕ are three layer feed-forward neural networks with (300, 300, 5) and (5, 5, 1) units respectively.

[4] DrQA code available at: https://github.com/facebookresearch/DrQA .

[5] https://lucene.apache.org/

[6] https://lemurproject.org/clueweb09/

Question: Which country's name means "equator"? Answer: Ecuador	
InferSent ranker	**RN ranker**
Ecuador : "Equator" in Spanish , as the country lies on the Equator.	Salinas, is considered the best tourist beach resort in Ecuador's Pacific Coastline.. Quito, **Ecuador** Ecuador's capital and the country's second largest city.
The equator crosses just north of **Ecuador**'s capital, Quito, and the country gets its name from this hemispheric crossroads.	Quito is the capital of **Ecuador** and of Pichincha, the country's most populous Andean province, is situated 116 miles from the Pacific coast at an altitude of 9,350 feet, just south of the equator.
The country that comes closest to the equator without actually touching it is Peru.	The location of the Republic of **Ecuador** Ecuador, known officially as the Republic of Ecuador -LRB- which literally means "Republic of the equator" -RRB- , is a representative democratic republic
The name of the country is derived from its position on the Equator.	The name of the country is derived from its position on the Equator.

Table 1: An example question from the QUASAR-T test set with the top passages returned by the two rankers.

We use NLTK to tokenize words for the RN ranker. [7] We lower-case the words, and remove punctuations and infrequent words that occur less than 5 times in the corpus. We pass untokenized sentence string as input directly to InferSent encoder as expected by it.

We train both the InferSent ranker and the RN ranker using Stochastic Gradient Descent with the Adam optimizer (Kingma and Ba, 2014).[8] A dropout (Srivastava et al., 2014) of p=0.5 is applied to all hidden layers for training both InferSent and RN rankers.

5 Results and Analysis

First, we evaluate the two ranker models based on the recall@K, which measures whether the ground truth answer span is in the top K ranked documents. We then evaluate the performance of machine reader on the top K ranked documents of each ranker by feeding them to DrQA reader and measuring the exact match accuracy and F-1 score produced by the reader. Finally, we do qualitative analysis of top-5 documents produced by each ranker.

5.1 Recall of Rankers

The performance of the rankers is shown in Table 2. Although the recall of the InferSent ranker is somewhat lower than that of the R^3 ranker at top-1, it still improves upon the recall of search engine provided by raw dataset. In addition, it performs slightly better than the ranker from R^3 for recall at top-3 and top-5. We can conclude that using InferSent as paragraph representation improves the recall in re-ranking the paragraphs for machine reading.

The RN ranker achieves significantly higher recall than R^3 and InferSent rankers. This proves our hypothesis that word by word relevance matching improves retrieval recall. Does high recall mean high question answering accuracy? We further analyze the documents retrieved by the rankers to answer this.

5.2 Machine Reading Performance

For each question, we feed the top five documents retrieved by each ranker to DrQA trained on QUASAR-T to produce an answer. The overall QA performance improves from exact match accuracy of 19.7 to 31.2 when InferSent ranker is used (Table 3). We can also observe that InferSent ranker is much better than RN ranker in terms of overall QA performance despite its low recall for retrieval. InferSent ranker with DrQA provides comparable result to SR^2 despite being a simpler model.

[7] https://www.nltk.org/

[8] Learning rate is set to 0.001.

	Top-1	Top-3	Top-5
IR	19.7	36.3	44.3
Ranker from R^3	40.3	51.3	54.5
InferSent ranker	36.1	52.8	56.7
RN ranker	**51.4**	**68.2**	**70.3**

Table 2: Recall of ranker on QUASAR-T test dataset. The recall is calculated by checking whether the ground truth answer appears in top-N paragraphs. IR is the search engine ranking given in QUASAR-T dataset.

	EM	F1
No ranker + DrQA	19.6	24.43
InferSent + DrQA	31.2	37.6
RN + DrQA	26.0	30.7
GA (Dhingra et al., 2017a)	26.4	26.4
BiDAF (Seo et al., 2016)	25.9	28.5
R^3 (Wang et al., 2017)	**35.3**	**41.7**
SR^2 (Wang et al., 2017)	31.9	38.7

Table 3: Exact Match(EM) and F-1 scores of different models on QUASAR-T test dataset. Our InferSent + DrQA model is as competitive as SR^2 which is a supervised variant of the state-of-the-art model, R^3

5.3 Analysis of paragraphs retrieved by the rankers

The top paragraphs ranked by InferSent are generally semantically similar to the question (Table 1). However, we find that there is a significant number of cases where proper noun ground truth answer is missing in the paragraph. An example of such a sentence would be "*The name of the country is derived from its position on the Equator*". Though this sentence is semantically similar to the question, it does not contain the proper noun answer. As InferSent encodes the general meaning of the whole sentence in a distributed representation, it is difficult for the ranker to decide whether the representation contains the important keywords for QA.

Although the top paragraphs ranked by RN ranker contain the ground truth answer, they are not semantically similar to the question. This behavior is expected as RN ranker only performs matching of words in question with words in the paragraph, and does not have information about the context and word order that is important for learning semantics. However, it is interesting to observe that RN ranker can retrieve the paragraph that contains the ground truth answer span even when the paragraph has little similarity with the question.

In Table 1, the top paragraph retrieved by RN ranker is not semantically similar to the question. Moreover, the only word overlap between the question and paragraph is "**country's**" which is not an important keyword. The fourth paragraph retrieved by RN ranker not only contains the word "**country**" but also has more word overlap with the question. Despite this, RN ranker gives a lower score to the fourth paragraph as it does not contain the ground truth answer span. As RN ranker is designed to give higher ranking score to sentences or paragraphs that has the highest word overlap with the question, this behavior is not intuitive. We notice many similar cases in test dataset which suggests that RN ranker might be learning to predict the possible answer span on its own. As RN ranker compares every word in question with every word in the paragraph, it might learn to give a high score to the word in the paragraph that often co-occurs with all the words in the question. For example, it might learn that *Equador* is the word that has highest co-occurence with *country*, *name*, *means* and *equator*, and gives very high scores to the paragraphs that contain *Equador*.

Nevertheless, it is difficult for machine reader to find answer in such paragraphs that have little or no meaningful overlap with the question. This explains the poor performance of machine reader on documents ranked by RN ranker despite its high recall.

6 Conclusion

We find that word level relevance matching significantly improves retrieval performance. We also show that the ranker with very high retrieval recall may not achieve high overall performance in open-domain QA. Although both semantic similarity and relevance scores are important for open-domain QA, we find that semantic similarity contributes more for a better overall performance of open-domain QA. For the future work, we would like to explore new ranking models that consider both overall semantic similarity and weighted local interactions between words in the question and the document. Moreover, as Relation-Networks are very good at predicting the answer on their own, we would like to implement a model that can do both ranking and answer extraction based on

Relation-Networks.

Acknowledgments

PMH is funded as an AdeptMind Scholar. This project has benefited from financial support to SB by Google, Tencent Holdings, and Samsung Research. KC thanks support by AdeptMind, eBay, TenCent, NVIDIA and CIFAR. This work is partly funded by the Defense Advanced Research Projects Agency (DARPA) D3M program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Q. Weinberger. 2010. Learning to rank with (a lot of) word features. *Inf. Retr.*, 13(3):291–314.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Eric Brill, Jimmy J. Lin, Michele Banko, Susan T. Dumais, and Andrew Y. Ng. 2001. Data-intensive question answering. In *Proceedings of The Tenth Text REtrieval Conference, TREC 2001, Gaithersburg, Maryland, USA, November 13-16, 2001*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 670–680.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2017a. Gated-attention readers for text comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1832–1846.

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017b. Quasar: Datasets for question answering by search and reading. *arXiv preprint arXiv:1707.03904*.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2017. A deep relevance matching model for ad-hoc retrieval. *CoRR*, abs/1711.08611.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1681–1691.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3294–3302.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. *ACLWeb*, (P16-2022).

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392.

Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter

Battaglia, and Tim Lillicrap. 2017. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4974–4983.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2017. R$ˆ3$: Reinforced reader-ranker for open-domain question answering. *CoRR*, abs/1709.00023.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, pages 515–526.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*, abs/1704.05426.