Phrase-Indexed Question Answering: A New Challenge for Scalable Document Comprehension

Minjoon Seo[2,3*] **Tom Kwiatkowski**[1] **Ankur P. Parikh**[1]
Ali Farhadi[2,4,5] **Hannaneh Hajishirzi**[2]
Google AI Language[1] University of Washington[2] Clova AI, NAVER[3]
Allen Institute for AI[4] XNOR.AI[5]
{minjoon,ali,hannaneh}@cs.uw.edu
{tomkwiat,aparikh}@google.com

Abstract

We formalize a new modular variant of current question answering tasks by enforcing complete independence of the document encoder from the question encoder. This formulation addresses a key challenge in machine comprehension by requiring a standalone representation of the document discourse. It additionally leads to a significant scalability advantage since the encoding of the answer candidate *phrases* in the document can be pre-computed and *indexed* offline for efficient retrieval. We experiment with baseline models for the new task, which achieve a reasonable accuracy but significantly underperform unconstrained QA models. We invite the QA research community to engage in Phrase-Indexed Question Answering (PIQA, *pika*) for closing the gap. The leaderboard is at: nlp.cs.washington.edu/piqa

1 Introduction

Extractive question answering (QA) is the task of selecting an answer phrase (span) to a question given an evidence document. Due to the easiness of evaluation (compared to generative QA) and the fine-graininess of the answer (compared to sentence-level QA), it has become one of the most popular QA tasks, driven by massive new datasets such as SQuAD (Rajpurkar et al., 2016) and TriviaQA (Joshi et al., 2017). Current QA models heavily rely on explicitly learning the interaction between the evidence document and the question using neural attention mechanisms (Wang and Jiang, 2017; Xiong et al., 2017; Seo et al., 2017; Lee et al., 2016, *inter alia*), in which the model is fully aware of the question before or as it reads the document. As a result, despite significant advances, they have not led to the standalone representation of document discourse which is nevertheless a key goal of research in reading comprehension. Furthermore, QA models that condition the document representation on a question have the practical scalability downside that the entire model should be re-applied on the same document for every question.

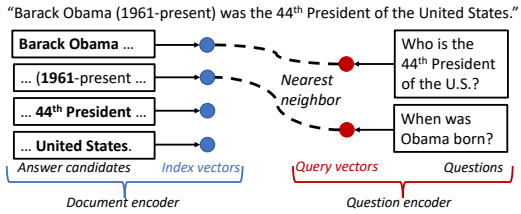

Figure 1: PIQA task for a short context sentence.

In this paper, we formalize a modular variant of the QA task, Phrase Indexed Question Answering (PIQA), that enforces complete independence between document encoder and question encoder (Figure 1). In PIQA, all documents are processed independently of any question to generate *phrase index vectors* (blue nodes in the figure) for each answer candidate (left boxes in the figure). Similarly, the questions are independently mapped to *query vectors* (red nodes in figure). Then, at inference time, the answer is obtained by retrieving the nearest indexed phrase vector to the query vector. Hence the algorithms aimed at tackling PIQA have the inherent benefit of modularity and scalability compared to current QA systems.

The task setup is analogous to how documents or sentences are retrieved in modern search engines via similarity search algorithms (Shrivastava and Li, 2015). Nevertheless, there is a key distinction that search engines index each *document* by its *content*, while PIQA requires one to index each *phrase* in documents by its *context*.

We formally define the PIQA problem and provide baseline models for the new task. Our experiments show that the constraint introduced

*Most work done during internship with Google AI.

by PIQA leads to meaningful standalone document representations and practical scalability advantage, demonstrating the significance of the new task. Moreover, there is still a large gap between the baselines and the unconstrained state of the art, showing that the task is yet far from being solved. We have set up a leaderboard[1] for PIQA challenge and invite the research community to participate. We currently support SQuAD and plan to expand to other datasets as well.

2 Related Work

Reading comprehension. Massive reading comprehension question answering datasets (Hermann et al., 2015; Hill et al., 2016; Dhingra et al., 2017; Dunn et al., 2017) have driven a large number of successful neural approaches (Kadlec et al., 2016; Hu et al., 2017, *inter alia*). Choi et al. (2017); Chen et al. (2017); Clark and Gardner (2017); Min et al. (2018) tackled large-scale QA by using a fast, coarse model (e.g. TF-IDF) to retrieve few documents or sentences and then using a slower, accurate model to obtain the answer. Salant and Berant (2018) proposed to minimize (but not prohibit) the influence of question when modeling the document. Similarly to ours, Lee et al. (2016) proposed to explicitly learn the representation for each answer candidate (phrase) in the document, but it was conditioned (dependent) on the question.

Sentence retrieval. A closely related task to ours is that of retrieving a sentence/paragraph in a corpus that answers the question (Tay et al., 2017). A comprehensive survey for neural approaches in information retrieval literature is discussed in Mitra and Craswell (2017). We note that our problem is focused on phrasal answer extraction, which presents a unique challenge over sentence retrieval—the need for *context-based* representation as opposed to the *content-based* representation in the sentence-retrieval literature.

Language representation. Recently there has been a growing interest in developing natural language representations that can be transferred across tasks (Vendrov et al., 2016; Wieting et al., 2016; Conneau et al., 2017, *inter alia*). In particular, SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2017) encourage architectures that first encode the hypothesis and the premise independently before a comparator neural network is applied. Our proposed problem shares similar traits but has a stronger constraint that only inner product comparison is allowed and one needs to model phrases instead of complete sentences.

Memory networks. Each phrase-vector is analogous to a single memory slot, where the vector is the key and the phrase is the value, and the question vector is the query for accessing the memory. Hence, PIQA can be considered as an effort to formulate extractive question answering as the task of memory augmentation (construction) (Weston et al., 2015; Sukhbaatar et al., 2015) from unstructured knowledge source (text).

3 Phrase-Indexed Question Answering

Extractive question answering is the task of obtaining the answer $\hat{a}$ to a question $Q = \{q_1 \dots q_n\}$ given an evidence document $D = \{d_1 \dots d_m\}$, where the answer $\hat{a} = (s, e)$ indicates the start and end of a span in the document. The task is often formulated as learning the probabilistic distribution of the answer given the question and the document. In existing literature (Section 2), the distribution is mainly featurized by $\Pr(a|Q, D) \propto \exp(F_\theta(Q, D, a))$ where F_θ could be any real-valued scoring function parameterized by θ. Once θ is learned, the prediction $\hat{a}$ is obtained by

$$\hat{a} = \operatorname*{argmax}_a F_\theta(Q, D, a). \quad (1)$$

So far, most competitive designs of $F_\theta(Q, D, a)$ make use of attention connections between the words in Q and D. As a result, these models cannot yield a query independent representation of the document D. It is subsequently not possible to independently assess the document understanding capability of the model. Furthermore, $F_\theta(Q, D, a)$ needs to be re-computed for the entire document for every new question. We believe that this inefficiency precludes all current models as the candidates for end-to-end QA systems.

We propose a new task—Phrase-Indexed Question Answering (PIQA)—that addresses these issues. We enforce the *decomposability* of F_θ into two exclusive functions $G_\theta(Q), H_\theta(D, a) \in \mathbb{R}^k$. The answer distribution is then modeled by $\Pr(a|Q, D) \propto \exp(G_\theta(Q) \bullet H_\theta(D, a))$, where $\bullet$ is the inner product. The prediction is obtained by

$$\hat{a} = \operatorname*{argmax}_a G_\theta(Q) \bullet H_\theta(D, a). \quad (2)$$

[1] nlp.cs.washington.edu/piqa

In this setting, the document encoder H_θ learns models the document independently of the question. Successful question answering models that follow the structure of PIQA will have two important advantages over current QA models: full document comprehension and scalablity.

Full document comprehension. Language understanding ability is widely associated with learning a good standalone representation of text (or its components such as phrases) independent of the end task (Bowman et al., 2015). Under PIQA constraints, the document encoder H_θ learns the representation of the answer candidate phrases a in the document D independent of the question. In order to correctly answer questions, these phrase representations (index vectors) need to correctly encode their meaning with respect to their context. Therefore, PIQA constraint enforces evaluating research in document comprehension and phrase representation learning.

Scalability. Models that adhere to the PIQA constraint only need to be run once for each document, regardless of the number of questions asked. To answer a question, the model then just needs to encode the question and compare it to each of the answer candidates via the inner product in Equation 2. Implemented naively, computing a single inner product for each answer candidate is more efficient than building a new document encoding; after the documents are pre-encoded, Equation 2 is $O(k)$ time per word where k is the vector size (most neural models require $O(k^2)$ per word for matrix multiplications).

More importantly, PIQA also permits an approximate solution in sublinear time using asymmetric locality-sensitive hashing (aLSH) (Shrivastava and Li, 2014, 2015), through which Equation 2 can be approximated for N answer candidates with $O(kN^\rho \log N)$ time, where $\rho < 1$ is a function of the approximation factor and the properties of the hash functions. We argue that this type of approach will be essential for the development of real world QA systems, where the number of potential answers N is extremely large.

4 Baseline Models

We introduce several baselines for PIQA that are motivated by related literature.

For all (neural) baselines, we represent the words in D and Q with one of three embedding mechanisms: CharCNN (Kim, 2014) + GloVe (Pennington et al., 2014), and ELMo (Peters et al., 2018). We follow the majority of the related literature and apply bidirectional LSTMs (Hochreiter and Schmidhuber, 1997) to these embeddings to build the context-aware representations of the document $\mathbf{D} = \{\mathbf{d}_1 \dots \mathbf{d}_m\}$ and question $\mathbf{Q} = \{\mathbf{q}_1 \dots \mathbf{q}_n\}$, where the forward & backward LSTM outputs are concatenated to get a single word representation, i.e. $\mathbf{d}_i, \mathbf{q}_i \in \mathbb{R}^{2k}$ where k is the hidden state size of LSTMs.

PIQA disallows cross-attention between document and question. However, we can still benefit from self-attention, which has become crucial for machine translation (Vaswani et al., 2017) and QA (Huang et al., 2018; Yu et al., 2018). In all of our baselines, each variable-length question is collapsed into a fixed length vector via the sum $\mathbf{q}^{\text{SA}} = \sum_i u_i \mathbf{q}_i$ where $\mathbf{u} = \{u_1 \dots u_n\}$ is a vector containing a single weight for each word in the question. Similarly, we experiment with document side self attention to represent each document word $\mathbf{d}_j$ as a weighted sum of itself and all neighboring words $\mathbf{d}_j^{\text{SA}} = \sum_i h_i^j \mathbf{d}_j$. The weight vectors $\mathbf{u}$ and $\mathbf{h}^j$ are calculated as

$$\mathbf{u} = \text{softmax}_i(\mathbf{w}^\top \mathbf{q}_i)$$
$$\mathbf{h}^j = \text{softmax}_i(R_\theta(\mathbf{D}, j)^\top K_\theta(\mathbf{D}, i))$$

where R_θ, and K_θ are trainable neural networks with the same ouptut size, and $\mathbf{w} \in \mathbb{R}^{2k}$ is a trainable weight vector. We use independent BiLSTMs with hidden state size k (i.e. the output size is $2k$) to model both R_θ and K_θ. That is, $R_\theta(\mathbf{D}, j)$ is the j-th output of BiLSTM on top of $\mathbf{D}$, and we similarly define K_θ with unshared parameters.

For all (neural) baselines, the question is represented using the concatenation of two copies of $\mathbf{q}^{\text{SA}}$, one that should have high inner product with the vector for the answer's start span and another that should have high inner product with the vector for the answer's end. Thus, Equation 2's $G_\theta(Q) = [\mathbf{q}_s^{\text{SA}}, \mathbf{q}_e^{\text{SA}}]$ where the subscripts s (start) and e (end) imply that different sets of parameters were used. Now we define several baselines.

LSTM baseline. An answer candidate $a = (s, e)$ is represented using the LSTM outputs at its endpoints: from Equation 2, $H_\theta(D, (s, e)) = [\mathbf{d}_s, \mathbf{d}_e] \in \mathbb{R}^{4k}$ and $G_\theta(Q) = [\mathbf{q}_s^{\text{SA}}, \mathbf{q}_e^{\text{SA}}] \in \mathbb{R}^{4k}$.

LSTM+SA baseline. The LSTM outputs are augmented with the endpoint representations that

Constraint	Model	F1 (%)	EM (%)
PI	TF-IDF	15.0	3.9
	LSTM	57.2	46.8
	LSTM+SA	59.8	49.0
	LSTM+ELMo	60.9	50.9
	LSTM+SA+ELMo	62.7	52.7
None	Rajpurkar et al. (2016)	51.0	40.0
	Yu et al. (2018)	89.3	82.5

Table 1: Performance on SQuAD dev set with the PIQA constraint (top), and without the constraint (bottom). See Section 4 for the description of the terms.

come out of the document's self-attention (SA): $H_\theta(D,(s,e)) = [\mathbf{d}_s, \mathbf{d}_s^{\text{SA}}, \mathbf{d}_e, \mathbf{d}_e^{\text{SA}}] \in \mathbb{R}^{8k}$ and $G_\theta(Q) = [\mathbf{q}_{s1}^{\text{SA}}, \mathbf{q}_{s2}^{\text{SA}}, \mathbf{q}_{e1}^{\text{SA}}, \mathbf{q}_{e2}^{\text{SA}}] \in \mathbb{R}^{8k}$.

TF-IDF. We lastly include a purely TF-IDF-based model, where each answer candidate phrase is associated with a bag of neighbor words within a distance of 7. Then the BOW vector is normalized via TF-IDF and indexed. When the query comes in, its TF-IDF vector is queried on the indexed phrases to yield the answer.

For training the (neural) models, we minimize the negative log probability of getting the correct answer: the loss function for each example (D, Q, a^*) is $L(\theta) = -\log \Pr(a^*|D,Q)$ where a^* is the correct answer.

5 Experiments

We impose the independence restrictions from PIQA on the Stanford Question Answering Dataset[2]. We only consider answer spans with length ≤ 7. We use the hidden state size (k) of 128, which results in a 512D ($4k$) and 1024D ($8k$) vector for each phrase in LSTM and LSTM+SA, respectively. The default embedding model is CharCNN concatenated with 200D GloVe, with an option to append ELMo vectors following the same setup for SQuAD experiments discussed in Peters et al. (2018). We use a batch size of 64 and train for 20 epochs with the default Adam optimizer (Kingma and Ba, 2015), and take the best model on the validation set during training.

Results. Table 1 shows the results for the PIQA baselines (top) and the unconstrained state of the art (bottom). First, the TF-IDF model performs poorly, which signifies the limitations of traditional document retrieval models for the task. Second, we note that the addition of self-attention makes a significant impact on results, improving F1 by 2.6%. Next, we see that adding ELMo gives 3.7% and 2.9% improvement on F1 for LSTM and LSTM+SA models, respectively. Lastly, the best PIQA baseline model is 11.7% higher than the first (unconstrained) baseline model (Rajpurkar et al., 2016) and 26.6% lower than the state of the art (Yu et al., 2018). This gives us a reasonable starting point of the new task and a significant gap to close for future work.

[2]PIQA paradigm can be also extended to other extractive QA datasets.

- According to the **American Library Association**, this makes...
- ...tasked with drafing a **European Charter of Human Rights**,...
- **The LM engines** were successfully test-fired and restarted, ...
- **Steam turbines** were extensively applied...
- ...primarily accomplished through the **ductile stretching and thinning**.
- ...directly derived from **the homogeneity or symmetry of space**...

Table 2: Most similar phrase pairs from disjoint sets of documents. Bold print is the phrase, and non-bold is its context.

Phrase representations. Since PIQA models encode all answer candidates into the same space, we expect similar answer candidates to have high inner products with one another. Table 2 shows pairs of answer candidates that come from different documents in SQuAD, but that have similar encodings (high inner product). We observe that phrase representations learned through the PIQA task capture different interesting characteristics of the phrases. In all three rows, we can see that the phrase pairs seem to fit into natural categories: national, or multi-national organizational constructs; mechanical engines; and mechanical properties, respectively. This suggests that the model has learned interesting typing information above the word level. The second and third rows also indicate that the model has learned a rich representation of context. This is particularly obvious in the third row where the two phrases are lexically dissimilar, but preceded by the similar contexts *'primarily accomplished through'* and *'directly derived from'*. We believe that this analysis, while not complete, points toward exciting future lines of work in learning highly contextualized phrase representations through question answering.

Scalability. PIQA can also gain massive execution time speedups once the documents are pre-encoded: in our simple benchmark on a consumer-

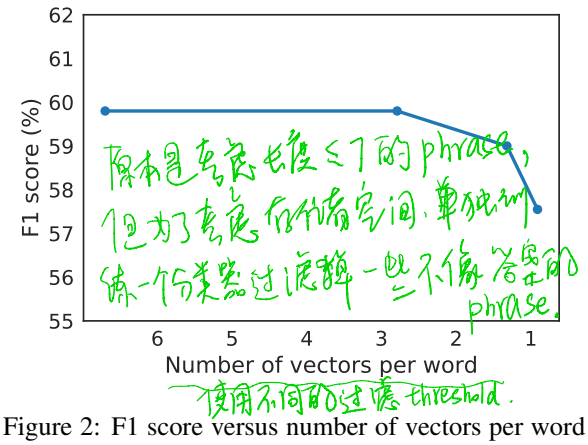

Figure 2: F1 score versus number of vectors per word for LSTM+SA. Answer candidates have been filtered with varying threshold on an independent classifier learned on the candidate representations.

grade CPU and NumPy (for LSTM+SA model, 1024D vectors), one can easily perform exact search over 1 million document words per second. BiDAF (Seo et al., 2017), an open-sourced and relatively light QA model reaching 77.5% F1 (66.5% EM), can process less than 1k document words per second with an equivalent computing power (after pre-encoding the document as much as possible), which is more than 1,000x slower.[3]

It is also important to consider the memory cost for storing a vector representation of each of the answer candidates. We train an independent single-layer perceptron classifier that predicts whether the phrase encoding is likely to be a good one. By varying a threshold on the score assigned by this classifier, we can filter answer candidates prior to storage. Figure 2 illustrates the trade-off between accuracy and memory (measured in mean number of vectors per document word) resulting from this filtering procedure for the LSTM+SA model. We observe that 1.3 vectors (candidates) per word on average reaches $> 98\%$ of the model's F1 accuracy. This is equivalent to 5.2 KB per word with 1024D (4 KB) float vectors, or around 15 TB for the entire English Wikipedia (3 billion words). Future work will also involve creating a better classifier (i.e. improving the trade-off curve in Figure 2) for determining which phrase vectors to store.

[3]The difference will be even higher with a dedicated similarity search package such as Faiss (Johnson et al., 2017) or approximate search (Section 3).

6 Conclusion and Future Work

We introduced Phrase-Indexed Question Answering (PIQA), a new variant of the extractive question answering task that requires documents and question encoded completely independently and that they only interact each other via inner product. We argued that building a question-agnostic document encoder for question answering should be an important consideration for those in the QA community with the research goal of learning a model that reads and comprehends documents. Furthermore, the imposed constraint of the task implies a sublinear scalability benefit. Given that SQuAD models have recently outperformed humans, PIQA formulation motivates a new challenge for which we hope that the community's effort gradually closes the gap between our constrained baselines and the unconstrained models.

Acknowledgments

This research was supported by ONR (N00014-18-1-2826), NSF (IIS 1616112), Allen Distinguished Investigator Award, and gifts from Google, Allen Institute for AI, Amazon, and Bloomberg. We thank the anonymous reviewers for their helpful comments.

References

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *ACL*.

Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In *ACL*.

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*.

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

Matthew Dunn, Levent Sagun, Mike Higgins, Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minghao Hu, Yuxing Peng, and Xipeng Qiu. 2017. Reinforced mnemonic reader for machine comprehension. *arXiv preprint arXiv:1705.02798*.

Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. 2018. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. In *ICLR*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. In *ACL*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *ACL*.

Bhaskar Mitra and Nick Craswell. 2017. Neural models for information retrieval. *arXiv preprint arXiv:1705.01509*.

Jeffrey Pennington, Richard Socher, and Christopher 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 Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*.

Shimi Salant and Jonathan Berant. 2018. Contextualized word representations for reading comprehension. In *NAACL*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Anshumali Shrivastava and Ping Li. 2014. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *NIPS*.

Anshumali Shrivastava and Ping Li. 2015. Improved asymmetric locality sensitive hashing (alsh) for maximum inner product search (mips). In *UAI*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *NIPS*.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2017. Hyperqa: Hyperbolic embeddings for fast and efficient ranking of question answer pairs. *arXiv preprint arXiv:1707.07847*.

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 *NIPS*.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. In *ICLR*.

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *ICLR*.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *ICLR*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *ICLR*.

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

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *ICLR*.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *ICLR*.