Self-Training for Jointly Learning to Ask and Answer Questions

Mrinmaya Sachan **Eric P. Xing**
School of Computer Science
Carnegie Mellon University
{mrinmays, epxing}@cs.cmu.edu

Abstract

Building curious machines that can answer as well as ask questions is an important challenge for AI. The two tasks of question answering and question generation are usually tackled separately in the NLP literature. At the same time, both require significant amounts of supervised data which is hard to obtain in many domains. To alleviate these issues, we propose a self-training method for jointly learning to ask as well as answer questions, leveraging unlabeled text along with labeled question answer pairs for learning. We evaluate our approach on four benchmark datasets: *SQUAD*, *MS MARCO*, *WikiQA* and *TrecQA*, and show significant improvements over a number of established baselines on both question answering and question generation tasks. We also achieved new state-of-the-art results on two competitive answer sentence selection tasks: *WikiQA* and *TrecQA*.

1 Introduction

Question Answering (QA) is a well-studied problem in NLP which focuses on answering questions using some structured or unstructured sources of knowledge. Alongside question answering, there has also been some work on generating questions (QG) (Heilman, 2011; Du et al., 2017; Tang et al., 2017) which focuses on generating questions based on given sources of knowledge.

QA and QG are closely related[1] tasks. However, NLP literature views the two as entirely separate tasks. In this paper, we explore this relationship between the two tasks by jointly learning to generate as well as answer questions. An improved ability to generate as well as answer questions will help us build *curious* machines that can interact with humans in a better manner. Joint modeling of QA and QG is useful as the two can be used in conjunction to generate novel questions from free text and then answers for the generated questions. We use this idea to perform self-training (Nigam and Ghani, 2000) and leverage free text to augment the training of QA and QG models.

QA and QG models are typically trained on question answer pairs which are expensive to obtain in many domains. However, it is cheaper to obtain large quantities of free text. Our self-training procedure leverages unlabeled text to boost the quality of our QA and QG models. This is achieved by a careful data augmentation procedure which uses pre-trained QA and QG models to generate additional labeled question answer pairs. This additional data is then used to retrain our QA and QG models and the procedure is repeated.

This addition of synthetic labeled data needs to be performed carefully. During self-training, typically the most *confident* samples are added to the training set (Zhu, 2005) in each iteration. We use the performance of our QA and QG models as a proxy for estimating the *confidence* value of the questions. We describe a suite of heuristics inspired from curriculum learning (Bengio et al., 2009) to select the questions to be generated and added to the training set at each epoch. Curriculum learning is inspired from the incremental nature of human learning and orders training samples on the *easiness* scale so that *easy* samples can be introduced to the learning algorithm first and *harder* samples can be introduced successively. We show that introducing questions in increasing order of *hardness* leads to improvements over a baseline that introduces questions randomly.

We use a seq2seq model with soft attention (Sutskever et al., 2014; Bahdanau et al., 2014) for QG and a neural model inspired from *Attentive Reader* (Hermann et al., 2015; Chen et al., 2016) for QA. However, these can be any QA

[1] We can think of QA and QG as inverse of each other.

Proceedings of NAACL-HLT 2018, pages 629–640
New Orleans, Louisiana, June 1 - 6, 2018.

and QG models. We evaluate our approach on four datasets: *SQUAD*, *MS MARCO*, *WikiQA* and *TrecQA*. We use a corpus of English Wikipedia as unlabeled text. Our experiments show that the self-training approach leads to significant improvements over a number of established approaches in QA and QG on these benchmarks. On the two answer sentence selection QA tasks: (*WikiQA* and *TrecQA*), we obtain state-of-the-art.

2 Problem Setup

In this work, we focus on the task of machine comprehension where the goal is to answer a question $\mathbf{q}$ based on a passage $\mathbf{p}$. We model this as an answer sentence selection task i.e., given the set of sentences in the passage $\mathbf{p}$, the task is to select the sentence $\mathbf{s} \in \mathbf{p}$ that contains the answer a. Treating QA as an answer sentence selection task is quite common in literature (e.g. see Yu et al., 2014). We model QG as the task of transforming a sentence in the passage into a question. Previous work in QG (Heilman and Smith, 2009) transforms text sentences into questions via some set of manually engineered rules. However, we take an end-to-end neural approach.

Let $\mathcal{D}^0$ be a labeled dataset of (passage, question, answer) triples where the answer is given by selecting a sentence in the passage. We also assume access to unlabeled text $\mathcal{T}$ which will be used to augment the training of the two models.

3 The Question Answering Model

Since we model QA as the task of selecting an answer sentence from the passage, we treat each sentence in the corresponding passage as a candidate answer for every input question.

We employ a neural network model inspired from the *Attentive Reader* framework proposed in Hermann et al. (2015); Chen et al. (2016). We map all words in the vocabulary to corresponding d dimensional vector representations via an embedding matrix $E \in \mathbb{R}^{d \times V}$. Thus, the input passage $\mathbf{p}$ can be denoted by the word sequence $\{p_1, p_2, \ldots p_{|\mathbf{p}|}\}$ and the question $\mathbf{q}$ can similarly be denoted by the word sequence $\{q_1, q_2, \ldots q_{|\mathbf{q}|}\}$ where each token $p_i \in \mathbb{R}^d$ and $q_i \in \mathbb{R}^d$.

We use a bi-directional LSTM (Graves et al., 2005) with dropout regularization as in Zaremba et al. (2014) to encode contextual embeddings of each word in the passage:

$$\vec{\mathbf{h}}_t = \mathit{LSTM}_1\left(p_t, \vec{\mathbf{h}}_{t-1}\right), \overleftarrow{\mathbf{h}}_t = \mathit{LSTM}_2\left(p_t, \overleftarrow{\mathbf{h}}_{t+1}\right)$$

The final contextual embeddings $\mathbf{h}_t$ are given by concatenation of the forward and backward pass embeddings: $\mathbf{h}_t = [\vec{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t]$. Similarly, we use another bi-directional LSTM and encode contextual embeddings of each word in the question.

Then, we use attention mechanism (Bahdanau et al., 2014) to compute the alignment distribution a based on the relevance among passage words and the question: $a_i = \mathit{softmax}\left(\mathbf{q}^T \mathbf{W} \mathbf{h}_i\right)$. The output vector o is a weighted combination of all contextual embeddings: $\mathbf{o} = \sum_i a_i \mathbf{h}_i$. Finally, the correct answer a^* among the set of candidate answers $\mathcal{A}$ is given by: $a^* = \arg\max_{a \in \mathcal{A}} \mathbf{w}^T \mathbf{o}$.

We learn the model by maximizing the log-likelihood of correct answers. Given the training set $\{\mathbf{p}^{(i)}, \mathbf{q}^{(i)}, \mathbf{a}^{(i)}\}_{i=1}^N$, the log-likelihood is:

$$\mathcal{L}_{QA} = \sum_{i=1}^{N} \mathit{logP}\left(\mathbf{a}^{(i)} | \mathbf{p}^{(i)}, \mathbf{p}^{(i)}; \theta\right)$$

Here, θ represents all the model parameters to be estimated.

4 The Question Generation Model

We use a seq2seq model (Sutskever et al., 2014) with soft attention (Bahdanau et al., 2014) as our QG model. The model transduces an input sequence $\mathbf{x}$ to an input sequence $\mathbf{y}$. Here, the input sequence is a sentence in the passage and the output sequence is a generated question. Let $\mathbf{x} = \{x_1, x_2, \ldots, x_{|\mathbf{x}|}\}$, $\mathbf{y} = \{y_1, y_2, \ldots, y_{|\mathbf{y}|}\}$ and $\mathcal{Y}$ be the space of all possible output questions. Thus, we can represent the QG task as finding $\hat{\mathbf{y}} \in \mathcal{Y}$ such that: $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$.

Here, $P(\mathbf{y}|\mathbf{x})$ is the conditional probability of a question sequence $\mathbf{y}$ given input sequence $\mathbf{x}$.

Decoder: Following Sutskever et al. (2014), the conditional factorizes over token level predictions:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{|\mathbf{y}|} P(y_t | \mathbf{y}_{<t}, \mathbf{x})$$

Here, $\mathbf{y}_{<t}$ represents the subsequence of words generated prior to the time step t. For the decoder, we again follow Sutskever et al. (2014):

$$P(y_t | \mathbf{y}_{<t}, \mathbf{x}) = \mathit{softmax}\left(\mathbf{W} \mathit{tanh}\left(\mathbf{W}_t [\mathbf{h}_t^{(d)}; \mathbf{c}_t]\right)\right)$$

Here, $\mathbf{h}_t^{(d)}$ is the decoder RNN state at time step t, and $\mathbf{c}_t$ is the attention based encoding of the input sequence $\mathbf{x}$ at decoding time step t (described later). Also $\mathbf{W}$ and $\mathbf{W}_t$ are model parameters to be learned. We use an LSTM with dropout (Zaremba et al., 2014) as the decoder RNN. The LSTM generates the new decoder state $\mathbf{h}_t^{(d)}$ given the representation of previously generated word y_{t1} obtained using a look-up dictionary, and the previous decoder state $\mathbf{h}_{t-1}^{(d)}$.

Encoder: We use a bi-directional LSTM (Graves et al., 2005) with attention mechanism as our sentence encoder. We use two LSTM's: one that makes a forward pass in the sequence and another that makes a backward pass as in the QA model described earlier. We use dropout regularization for LSTMs as in Zaremba et al. (2014) in our implementation. The final context dependent token representation $\mathbf{h}_t^{(e)}$ is the concatenation of the forward and backward pass token representations: $\mathbf{h}_t^{(e)} = [\overrightarrow{\mathbf{h}}_t^{(e)}; \overleftarrow{\mathbf{h}}_t^{(e)}]$. To obtain the final context dependent token representation $\mathbf{c}_j$ at the decoding time step j, we take a weighted average over token representations: $\mathbf{c}_j^{(d)} = \sum_{i=1}^{|\mathbf{x}|} a_{ij}\mathbf{h}_i^{(e)}$. Following Bahdanau et al. (2014), the attention weights a_{ij} are calculated by bilinear scoring followed by softmax normalization:

$$\mathbf{a}_{ij} = \frac{\exp\left(\mathbf{h}_j^{(e)^T}\mathbf{W}\,\mathbf{h}_i^{(d)}\right)}{\sum_{i'}\exp\left(\mathbf{h}_j^{(e)^T}\mathbf{W}\,\mathbf{h}_{i'}^{(d)}\right)}$$

Learning and Inference: We train the encoder decoder framework by maximizing data log-likelihood on a large training set with respect to all the model parameters θ. Let $\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^N$ be the training set. The log-likelihood can be written as:

$$\begin{aligned}\mathcal{L}_{QG} &= \sum_{i=1}^{N} logP\left(\mathbf{y}^{(i)}|\mathbf{x}^{(i)};\theta\right) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{|\mathbf{y}^{(i)}|} logP\left(y_j^{(i)}|\mathbf{x}^{(i)}, \mathbf{y}_{<j}^{(i)};\theta\right)\end{aligned}$$

We use beam search for inference. As in previous works, we introduce a <UNK> token to model rare words during decoding. These <UNK> tokens are finally replaced by the token in the input sentence with the highest attention score.

5 Self-training Framework for Joint Training of QA and QG models

In our self-training framework, we are given unlabeled text in addition to the labeled passages, question and answer pairs. Self-training (Yarowsky, 1995; Riloff et al., 2003), also known as self-teaching, is one of the earliest techniques for using unlabeled data along with labeled data to improve learning. During self-training, the learner keeps on labeling unlabeled examples and retraining itself on an enlarged labeled training set. We extend self-training to jointly learn two models (namely, QA and QG) iteratively. The QA and QG models are first trained on the labeled corpus. Then, the QG model is used to create more questions from the unlabeled text corpus and the QA model is used to answer these newly created questions. These new questions (carefully selected by an oracle – details later) and the original labelled data is then used to (stochastically) update these two models. This procedure can be repeated as long as both the two models continue to improve.

Algorithm 1: Self-training QA and QG.

$\theta_{\mathbf{qa}}^{(\mathbf{0})} \leftarrow$ Train initial QA model.
$\theta_{\mathbf{qg}}^{(\mathbf{0})} \leftarrow$ Train initial QG model.
Init: i = 0
while *performance on dev set rises* **do**
  $\mathbf{CQ_i} \leftarrow$ Set of candidate questions generated using our QG model $\theta_{\mathbf{qg}}^{(\mathbf{i})}$ from the unlabeled text $\mathcal{T}$ which are not in $\mathcal{D}$.
  $\mathbf{Q_i} \leftarrow k \times m^i$ questions drawn from $\mathbf{CQ_i}$ using our question selector oracle $\mathcal{QS}$.
  $\mathbf{A_i} \leftarrow$ Set of answers to questions $\mathbf{Q_i}$ obtained using our QA model $\theta_{\mathbf{qa}}^{(\mathbf{i})}$.
  Let $\mathcal{D}^i$ be the set of chosen questions $\mathbf{Q_i}$ and answers $\mathbf{A_i}$.
  Subsample $\mathcal{S}_1 \subset \mathcal{D}^i$ of size k_1 and $\mathcal{S}_2 \subset \mathcal{D}^0$ of size k_2. Let $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$
  $\theta_{\mathbf{qa}}^{(\mathbf{i+1})} \leftarrow$ Update QA model on $\mathcal{S}$.
  $\theta_{\mathbf{qg}}^{(\mathbf{i+1})} \leftarrow$ Update QG model on $\mathcal{S}$.
  i++
end

Algorithm 1 describes the procedure in detail. In each succesive iteration, we allow the addition of more questions than that introduced in the previous iteration by a multiplicative factor. This sheme adds fewer questions initially when the QA and QG models are weak and more questions thereafter when the two models have (hopefully) improved. We found that this scheme works better in practice than addiing a fixed number of questions in each iteration. The two models are

updated on a subsample of the newly generated datapoints and original unlabelled data.

Self-training has been seldom used in NLP. Most prominently, it has been used for WSD (Yarowsky, 1995), noun learning (Riloff et al., 2003) and AMR parsing and generation (Konstas et al., 2017). However, it has not been explored in this way for QA and QG.

5.1 The Question Selection Oracle

A key challenge in self-training is selecting which unlabeled data sample to label (iwhich generated questions to add to the training set). The self-training process may erroneously generate some bad or incorrect questions which can sidetrack the learning process. Thus, we implement a question selection oracle which determines which questions to add among the potentially very large set of questions generated by the QG model in each iteration.

Traditional wisdom in self-training (Yarowsky, 1995; Riloff et al., 2003) advises selecting a subset of questions on which the models have the highest *confidence*. We experiment with this idea, proposing multiple self-training oracles which introduce questions in the order of how *confident* the QA and QG models are on the new potential question:

- **QG:** The QG oracle introduces the question in the order of how confident the QG model is on generating the question. This is calculated by a number of heuristics (described later).
- **QA:** The QA oracle introduces the question in the order of how confident the QA model is on answering the question. This too is calculated by some heuristics (described later).
- **QA+QG:** The QA+QG oracle introduces a question when both QA and QG models are confident about the question. The oracle computes the minimum confidence of the QA and QG models for a question and introduces questions which have the the highest minimum confidence score.

Our question selection heurisitcs are based on the ideas of *curriculum learning* and *diversity*:

1. *Curriculum learning* (Bengio et al., 2009; Sachan and Xing, 2016a) requires ordering questions on the easiness scale, so that easy questions can be introduced to the learning algorithm first and harder questions can be introduced successively. The main challenge in learning the curriculum is that it requires the identification of easy and hard questions. In our setting, such a ranking of easy and hard questions is difficult to obtain. A human judgement of 'easiness' of a question might not correlate with what is easy for our algorithms in its feature and hypothesis space. We explore various heuristics that define a measure of easiness and learn the ordering by selecting questions using this measure.
2. A number of cognitive scientists (Cantor, 1946) argue that alongside curriculum learning, it is important to introduce diverse (even if sometimes hard) samples. Inspired by this, we introduce a measure of *diversity* and show that we can achieve further improvements by coupling the curriculum learning heuristics with a measure for diversity.

Curriculum Learning: Studies in cognitive science (Skinner, 1958; Peterson, 2004; Krueger and Dayan, 2009) have shown that humans learn much better when the training examples are not randomly presented but organized in increasing order of difficulty. In the machine learning community, this idea was introduced with the nomenclature of *curriculum learning* (Bengio et al., 2009), where a curriculum is designed by ranking samples based on manually curated difficulty measures. A manifestation of this idea is self-paced learning (SPL) (Kumar et al., 2010; Jiang et al., 2014, 2015) which selects samples based on the local loss term of the sample. We extend this idea and explore the following heuristics for our various oracles:

1) Greedy Optimal (GO): The simplest greedy heuristic is to pick a question q which has the minimum expected effect on the QA and QG models. The expected effect on adding q can be written as:

$$\sum_{a \in \mathcal{A}} p(a^* = a) \mathbb{E}[\mathcal{L}_{QA/QG}]$$

Here, $\mathcal{L}_{QA/QG}$ is $\mathcal{L}_{QA}$, $\mathcal{L}_{QG}$ or $min\left(\mathcal{L}_{QA}, \mathcal{L}_{QG}\right)$ depending on which oracle we are using. $p(a^* = a)$ can be estimated by computing the scores of each of the answer candidates for q and normalizing them. $\mathbb{E}[\mathcal{L}_{QA/QG}]$ can be estimated by retraining the model(s) after adding this question.

2) Change in Objective (CiO): Choose question q that causes the smallest increase in $\mathcal{L}_{QA/QG}$. If

there are multiple questions with the smallest increase in objective, pick one of them randomly.

3) Mini-max (M^2): Choose question q that minimizes the expected risk when including the question with the answer candidate a that yields the maximum error.

$$\hat{q} = \arg\min_{q} \max_{a \in \mathcal{A}} \mathcal{L}_{QA/QG}$$

4) Expected Change in Objective (ECiO): In this greedy heuristic, we pick a question q which has the minimum expected effect on the model. The expected effect can be written as:

$$\sum_{a} p(a^* = a) \times \mathbb{E}\left[L_{QA/QG}\right]$$

Here, $p(a^* = a)$ can again be achieved by computing the scores of each of the answer candidates for q and normalizing them and $\mathbb{E}\left[L_{QA/QG}\right]$ can be estimated by evaluating the model.

5) Change in Objective-Expected Change in Objective (CiO - ECiO): We pick a question q which has the minimum value of the difference between the change in objective and the expected change in objective described above. Intuitively, the difference represents how much the model is surprised to see this new question.

Time Complexity: GO and CiO require updating the model, M^2 and ECiO require performing inference on candiate questions, and CiO - ECiO requires retraining as well as inference. Thus, M^2 and ECiO are computationally most efficient.

Ensembling: We introduce an ensembling strategy that combines the heuristics into an ensemble. We tried two ensembling strategies. The first strategy computes the average score over all the heuristics for all potential (top-K in beam) questions and picks questions with the highest average. The second strategy uses minimum instead of the average. Minimum works better than average in practice and we use it in our experiments. The use of minimum is inspired by agreement-based learning (Liang et al., 2008), a well-known extension of self-training which uses multiple views of the data (described using different feature sets or models) and adds new unlabeled samples to the training set when multiple models agree on the label.

Diversity: The strategy of introducing easy questions first and then gradually introducing harder questions is intuitive as it allows the learner to improve gradually. Yet, it has one key deficiency. With curriculum learning, by focusing on easy questions first, our learning algorithm is usually not exposed to a diverse set of questions. This is particularly a problem for deep-learning approaches that learn representations during the process of learning. Hence, when a harder question arrives, it can be difficult for the learner to adjust to the new question as the current representation may not be appropriate for the new level of question difficulty. We tackle this by introducing an explore and exploit (**E&E**) strategy. *E&E* ensures that while we still select easy questions first, we also want to make our selection as diverse as possible. We define a measure for diversity as the angle between the question vectors: $\angle \mathbf{q}_i, \mathbf{q}_j = \mathit{Cosine}^{-1}\left(\frac{|\mathbf{q}_i \mathbf{q}_j|}{||\mathbf{q}_i|| ||\mathbf{q}_j||}\right)$. *E&E* picks the question which optimizes a convex combination (tuned on the dev set) of the curriculum learning objective and sum of angles between the candidate questions and the questions in the training set.

6 Experiments

Implementation Details: We perform the same preprocessing on all the text. We lower-case all the text. We use NLTK for word tokenization. For training our neural networks, we only keep the most frequent 50k words (including entity and placeholder markers), and map all other words to a special <UNK> token. We choose word embedding size d = 100, and use the 100-dimensional pretrained GloVe word embeddings (Pennington et al., 2014) for initialization. We set k, m, k_1 and k_2 (hyperparameters for self-training) by grid search on a held-out development set.

Datasets: We report our results on four datasets: *SQUAD* (Rajpurkar et al., 2016), *MS MARCO* (Nguyen et al., 2016), *WikiQA* (Yang et al., 2015) and *TrecQA* (Wang et al., 2007). *SQUAD* is a cloze-style reading comprehension dataset with questions posed by crowd workers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage. *MS MARCO* contains questions which are real anonymized queries issued through *Bing* or *Cortana* and the documents are related web pages which may or help answer the question. *WikiQA* is also a datset of queries taken from Bing query logs. Based on user clicks, each query is associated with a Wikipedia page. The summary paragraph of the page is taken as candidate answer sentences, with labels on whether the sentence is a correct answer to the question provided by crowd

	SQUAD			MS MARCO			WikiQA			TrecQA		
	Train	Dev	Test	Train	Dev	Test	Train	Dev	Test	Train	Dev	Test
#Questions	82,326	4,806	5,241	87,341	5,273	5,279	1,040	140	293	1,229	82	100
#Question-Answer Pairs	676,193	39,510	42,850	440,573	26,442	26,604	20,360	2,733	6,165	53,417	1,148	1,517

Table 1: Statistics of the four datasets used in evaluating our QA and QG models.

workers. Finally, *TrecQA* is a QA answer sentence selection dataset from the TREC QA track.

While *WikiQA* and *TrecQA* are directly answer sentence selection tasks, the other two are not. Hence, we treat the *SQUAD* and *MS MARCO* tasks as the answer sentence selection task assuming a one to one correspondence between answer sentences and annotated correct answer spans. Note that only a very small proportion of answers ($<$ 0.2% in training set) span two or more sentences. Since *SQUAD* and *MS MARCO* have a hidden test set, we only use the training and development sets for our evaluation purposes and we further split the provided development set into a dev and test set. This is also the data analysis setting used in previous works (Du et al., 2017; Tang et al., 2017). In fact, we use the same setting as in Tang et al. (2017) for comparison. The statistics of the four datasets and the respective train, dev and test splits are given in Table 1. For *WikiQA* and *TrecQA* datasets, we use the standard data splits. We use a large randomly subsampled corpus of English Wikipedia and use the first paragraph of each document as unlabeled text for self-training.

Evaluation Metrics: Following Tang et al. (2017), we evaluate our QA system with three standard evaluation metrics: *Mean Average Precision* (MAP), *Mean Reciprocal Rank* (MRR) and *Precision@1* (P@1). For QG, we follow Du et al. (2017) and use automatic evaluation metrics from MT and summarization: *BLEU-4* (Papineni et al., 2002), *METEOR* (Denkowski and Lavie, 2014) and $Rouge_L$ (Lin, 2004) to measure the overlap between generated and ground truth questions.

Baselines: For *SQUAD* and *MS MARCO* datasets, we use four QA baselines that have been used in previous works (Tang et al., 2017). The first two baselines, *WordCnt* and *NormWordCnt*, have been taken from Yang et al. (2015) and Yin et al. (2015), and are based on simple word overlap which have been shown to be strong baselines. These compute word co-occurrence between a question sentence and the candidate answer sentence. While *WordCnt* uses unnormalized word co-occurrence, *NormWordCnt* uses normalized word co-occurrence. The third and fourth baselines are *CDSSM* (Shen et al., 2014) and *ABCNN* (Yin et al., 2015) which use a neural network approach to model semantic relatedness of sentence pairs. For the *WikiQA* and *TrecQA* dataset, we report results of various existing state-of-the-art approaches on the two datasets[2].

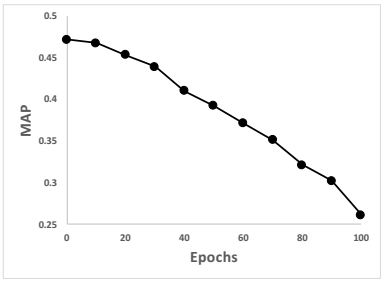

Figure 1: MAP for our best self-trained QA model (with 10,000 Wikipedia paragraphs) without any curriculum learning (i.e. candidate questions are added randomly) vs epochs.

For QG, we compare our model against the following four baselines used in previous work (Du et al., 2017). The first baseline is a simple *IR* baselines taken from Rush et al. (2015) which generates questions by memorizing them from the training set and uses edit distance (Levenshtein, 1966) to calculate distance between a question and the input sentence. The second baseline is a MT system – *MOSES* (Koehn et al., 2007) which models question generation as a translation task where raw sentences are treated as source texts and questions are treated as target texts. The third baseline, *DirectIn*, uses the longest sub-sentence of the input sentence (using a set of simple sentence splitters) as the question. The fourth baseline, *H&S* is a rule-based overgenerate-and-rank system proposed by Heilman and Smith (2010).

The Question Selection Oracle: The first question we wish to answer is: *Is careful question selection even necessary*? To answer this, we plot MAP scores for our best QA model (QA+QG, Ensemble+E&E) when we do not have a curriculum learning based oracle (i.e. an oracle which picks questions to be added to the dataset randomly) in Figure 1 as a function of epochs. We observe that

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

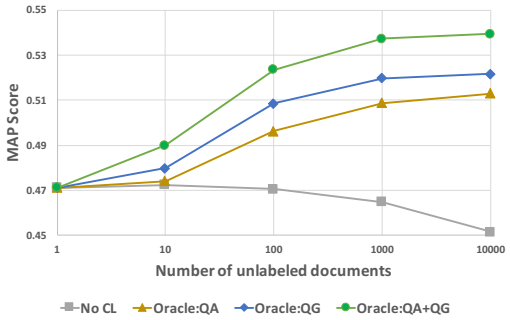

Figure 2: MAP for the best models for the three oracles: QA, QG and QA+QG. Also on the same plot, MAP when we have no curriculum learning.

	SQUAD			MS MARCO		
	MAP	MRR	P@1	MAP	MRR	P@1
WordCnt	0.396	0.401	0.179	0.809	0.817	0.689
NormWordCnt	0.422	0.429	0.203	0.871	0.879	0.796
CDSSM	0.443	0.449	0.228	0.798	0.804	0.672
ABCNN	0.469	0.477	0.263	0.869	0.875	0.784
Tang et al. (2017)	0.484	0.491	0.275	0.864	0.872	0.781
Ens+E&E(0)	0.471	0.478	0.263	0.858	0.865	0.774
Ens+E&E(100)	0.524	0.493	0.273	0.881	0.890	0.799
Ens+E&E(1000)	0.537	0.502	0.284	0.885	0.895	**0.801**
M^2	0.489	0.490	0.268	0.860	0.872	0.785
ECiO	0.498	0.494	0.273	0.877	0.886	0.793
GO	0.506	0.495	0.274	0.879	0.889	0.793
CiO	0.511	0.498	0.277	0.879	0.890	0.795
CiO-ECiO	0.517	0.500	0.280	0.881	0.892	0.798
Ensemble	0.539	0.504	0.284	0.886	0.895	0.800
Ens+E&E(10000)	**0.539**	**0.507**	**0.289**	**0.889**	**0.896**	0.801

Table 2: Performance of our models and QA baselines on *SQUAD* and *MS MARCO* datasets. Shaded part of the table shows results of various question selection heuristics when 10000 Wiki paragraphs are used as unlabeled data.

the MAP score degrades instead of improving with time. This supports our claim that we need to augment the training set by a more careful procedure.

We also plot MAP scores for our best QA model (Ensemble+E&E) when we use various question selection oracles as a function of the amount of unlabeled data in Figure 2. We can observe that when we do not have a curriculum learning based oracle, the MAP score degrades by having more and more unlabeled data. We also observe that the QA+QG oracle performs better than QA and QG which confirms that the best oracle is one that selects questions in increasing degree of hardness in terms of both question answering and question generation. This holds for all the experimental settings. Thus we only show results for the QA+QG strategies in our future experiments.

Evaluating Question Answering: First, we evaluate our models on the question answering task. *Ensemble+E&E(K)* is the variant where we perform self-training using K Wikipedia paragraphs. Hence, *Ensemble+E&E(0)* is the variant of our model without any self-training. We vary K to see the impact of the size of unlabeled Wikipedia paragraphs on the self-training model.

	MAP	MRR
CNN (Yang et al., 2015)	0.665	0.652
APCNN (Santos et al., 2016)	0.696	0.689
NASM (Miao et al., 2016)	0.707	0.689
ABCNN (Yin et al., 2015)	0.702	0.692
KVMN (Miller et al., 2016)	0.707	0.727
Wang et al. (2016b)	0.706	0.723
Wang et al. (2016a)	0.734	0.742
Wang and Jiang (2016)	0.743	0.755
Tang et al. (2017)	0.700	0.684
Ensemble+E&E(0)	0.691	0.675
Ensemble+E&E(100)	0.718	0.719
Ensemble+E&E(1000)	0.734	0.733
M^2	0.719	0.704
ECiO	0.721	0.708
GO	0.725	0.710
CiO	0.727	0.719
CiO-ECiO	0.734	0.724
Ensemble	0.743	0.743
Ensemble+E&E(10000)	**0.754**	**0.753**

Table 3: Performance of our models and the QA baselines on the *WikiQA* dataset. Shaded part of the table shows the effect of various question selection heuristics when 10000 Wikipedia paragraphs are used as unlabeled data. Our model achieves the state-of-the-art.

	MAP	MRR
He and Lin (2016)	0.758	0.822
He et al. (2015)	0.762	0.830
Tay et al. (2017)	0.770	0.825
Rao et al. (2016)	0.780	0.834
Ensemble+E&E(0)	0.742	0.813
Ensemble+E&E(100)	0.776	0.831
Ensemble+E&E(1000)	0.783	0.836
M^2	0.759	0.816
ECiO	0.762	0.822
GO	0.759	0.823
CiO	0.762	0.826
CiO-ECiO	0.767	0.830
Ensemble	0.789	0.843
Ensemble+E&E(10000)	**0.798**	**0.854**

Table 4: Performance of our models and the QA baselines on the *TrecQA* dataset. Shaded part of the table shows the effect of various question selection heuristics when 10000 Wikipedia paragraphs are used as unlabeled data. Our model achieves the state-of-the-art.

Table 2 shows the results of the QA evaluations on the *SQUAD* and *MS MARCO* datasets. We can observe that our QA model has competetive or better performance over all the baselines on both datasets in terms of all the three evaluation metrics. When we incorporate ensembling or diversity, we see a further improvement in the result.

Tables 3 and 4 show results of QA evaluations on the *WikiQA* and *TrecQA* datasets, respectively. We can again observe that our QA model is competitive to all the baselines. When we introduce ensembling and diversity while jointly learning the QA and QG models, we see incremental improvements. In both these answer sentence selection tasks, our approach achieves new state-of-the-art.

	SQUAD			MS MARCO			WikiQA			TrecQA		
	B	M	R	B	M	R	B	M	R	B	M	R
IR	1.07	7.77	20.85	0.81	5.42	15.78	0.93	6.89	19.98	0.83	5.73	16.34
MOSES	0.31	10.49	17.88	0.27	9.74	15.82	0.32	10.26	17.27	0.29	9.86	17.02
DirectIn	11.25	14.91	22.51	10.82	13.35	20.38	10.94	14.18	22.01	9.59	12.21	19.76
H&S	11.23	16.00	31.03	10.16	15.07	30.00	10.35	15.30	30.72	9.19	12.72	23.38
Tang et al. (2017)	5.03	-	-	9.31	-	-	3.15	-	-	-	-	-
Du et al. (2017)	12.28	16.62	39.75	-	-	-	-	-	-	-	-	-
Ens.+E&E(0)	12.31	16.67	39.78	11.14	15.60	37.26	11.38	16.08	38.42	10.96	14.25	27.27
Ens.+E&E(100)	14.14	18.70	42.46	13.25	17.10	40.28	13.10	17.00	40.93	11.63	15.05	29.07
Ens.+E&E(1000)	14.27	18.78	42.93	13.61	17.87	**41.23**	13.22	18.34	42.72	12.24	15.93	30.26
M^2	12.46	16.95	40.27	11.56	15.93	38.32	11.83	16.84	39.26	11.52	16.42	28.92
ECiO	12.79	17.40	40.92	12.11	16.32	38.86	12.14	17.04	39.82	11.67	16.59	29.12
GO	13.12	17.73	41.24	12.75	16.66	39.47	12.56	17.62	40.31	11.61	16.52	29.10
CiO	13.59	17.94	41.57	13.00	16.83	40.02	12.88	18.13	40.97	11.97	16.68	29.89
CiO-ECiO	13.97	18.18	41.90	13.41	17.16	40.65	13.22	18.34	41.28	12.24	16.65	29.63
Ensemble	**14.37**	18.57	42.73	13.56	17.40	40.92	14.26	18.91	43.26	13.32	16.76	30.12
Ens.+E&E(10000)	14.28	**18.79**	**42.97**	**13.74**	**17.89**	41.07	**15.26**	**19.45**	**44.77**	**14.87**	**16.88**	**31.91**

Table 5: Performance (B: BLEU4, M: METEOR, and R: ROUGE) of our model variants and various QG baselines on *SQUAD*, *MS MARCO* and *WikiQA* datasets. The shaded part of the table shows the effect of various question selection heuristics when 10000 Wikipedia paragraphs are used as unlabeled data. The performance numbers for Tang et al. (2017) and Du et al. (2017) were not reported for all the settings.

Evaluating Question Generation: Table 5 shows the results for QG on the four datasets on each of the three evaluation metrics on all the four datasets. We can observe that the QG model described in our paper performs much better than all the baselines. We again observe that self-training while jointly training the QA and QG models leads to even better performance. These results show that self-training and leveraging the relationship between QA and QG is very useful for boosting the performance of the QA and QG models, while additionally only using cheap unlabeled data.

Human Evaluations: We asked two people not involved with this research to evaluate 1000 (randomly selected) questions generated by our best QG model and our best performing baseline (Du et al., 2017) on SQUAD for fluency and correctness on a scale of 1 to 5. The raters were also shown the passage sentence used to generate the question. The raters were blind to which system produced which question. The Pearson correlation between the raters' judgments was $r = 0.89$ for fluency and $r = 0.78$ for correctness. In our analyses, we used the averages of the two raters' judgments. The evaluation showed that our system generates questions that are more fluent and correct than those by the baseline. The mean fluency rating of our best system was 4.15 compared to 3.35 for the baseline, a difference which is statistically significant (t-test, $p < 0.001$).

Evaluating the Question Selection Oracle: As discussed earlier, the choice of which subset of questions to add to our labeled dataset while self-training is important. To evaluate the various heuristics proposed in our paper, we show the effect of the question selection oracle on the final QA and QG performance in Tables 2, 3, 4 and 5. These comparisions are shown in the shaded grey portions of the tables for self-training with 10,000 Wikipedia paragraphs as unlabeled data.

We can observe that all the proposed heuristics (and ensembling and diversity strategies) lead to improvements in the final performance of both QA and QG. The heuristics arranged in increasing order of performance are: M^2, *ECiO*, *GO*, *CiO* and *CiO-ECiO*. While the choice of which heuristic to pick seems to make a lesser impact on the final performance, we do see a much more significant performance gain by *ensembling* to combine the various heuristics and using *E&E* to incorporate diversity. The incorporatation of diversity is important because the neural network models which learnt latent represention of data usually find it hard to adjust to new level of difficulty of questions as the current representation may not be appropriate for the new level of difficulty.

Low data scenarios: A key advantage of our self-training approach is that it can leverage unlabeled text, and thus requires less labeled data. To test this, we plot MAP for our best self-training model and various QA baselines as we vary the proportion of labeled training set in Figure 3. However, we keep the unlabeled text fixed (10K Wikipedia paragraphs). We observe that all the baselines significantly drop in performance as we reduce the proportion of labeled training set. However, the drop happens at a much slower rate for our self-trained model. Thus, we can conclude that our approach requires less labeled data as compared to the baselines.

Does more unlabeled text always help?: Another important question is: *Does more unlabeled*

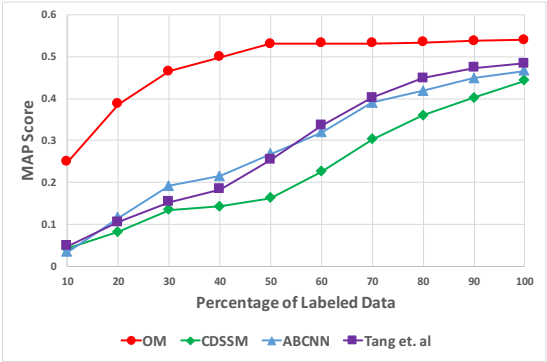

Figure 3: MAP for the best self-training model and QA baselines as we vary the proportion of labeled training set but keep the unlabeled text fixed (10K Wikipedia paragraphs).

text always improve our models? Will the performance improve if we add more and more unsupervised data during self-training. According to our results in Tables 2, 3, 4 and 5, the answer is "probably yes". As we can observe from these tables, the performance of the QA and QG models improves as we increase K, the size of the unsupervised data during training of the various *Ensemble+E&E(K)* models. Having said that, we do see a tapering effect on the performance results, so it is clear that the performance will be capped by some upper-bound and we will need better ways of modeling language and meaning to make progress.

7 Related Work

Our work proposes an approach for joint modeling QA and QG. While QA has recieved a lot of attention from the research community with large scale community evaluations such as *NTCIR*, *TREC*, *CLEF* spurring progress, the focus on QG is much more recent. Recently, there has been a renewed interest in reading comprehensions (also known as *machine comprehension* – a nomenclature popularized by Richardson et al. (2013)). Various approaches (Sachan et al., 2015; Wang et al., 2015; Sachan and Xing, 2016b; Sachan et al., 2016; Narasimhan and Barzilay, 2015) have been proposed for solving this task. After the release of large benchmarks such as *SQUAD*, *MS MARCO* and *WikiQA*, there has been a surge in interest on using neural network or deep-learning models for QA (Yin et al., 2015; Seo et al., 2016; Shen et al., 2016; Chen et al., 2017; Liu et al., 2017; Hu et al., 2017). In our work, we deal with the answer sentence selection task and adapt the *Attentive Reader* framework proposed in Hermann et al. (2015); Chen et al. (2016) as our base model. While, all these models were trained on question answer pairs, we propose a self-training solution to additionally leverage unsupervised text.

Similarly, there have been works on QG. Traditionally, rule based approaches with post-processing (Woo et al., 2016; Heilman and Smith, 2009, 2010) were the norm in QG. However, recent papers build on neural network approaches such as seq2seq (Du et al., 2017; Tang et al., 2017; Zhou et al., 2017), CNNs and RNNs (Duan et al., 2017) for QG. We also choose the seq2seq paradigm in our work. However, we leverage unsupervised text in contrast to these models.

Finally, some very recent works have concurrently recognized the relationship between QA and QG and have proposed joint training (Tang et al., 2017; Wang et al., 2017) for the two. Our work differs from these as we additionally propose self-training to leverage unlabeled data to improve the two models. Self-training has seldom been used in NLP. Most prominently, they have been used for word sense disambiguation (Yarowsky, 1995), noun learning (Riloff et al., 2003) and recently, AMR parsing and generation (Konstas et al., 2017). However, it has not been explored in this way for QA and QG.

An important decision in the workings of our self-training algorithm was the question selection using curriclum learning. While curriculum learning has seldom been used in NLP, we draw some ideas for curriculum learning from Sachan and Xing (2016a) who conduct a case study of curriculum learning for question answering. However, their work focuses only on QA and not QG.

8 Conclusion

We described self-training algorithms for jointly learning to answer and ask questions while leveraging unlabeled data. We experimented with neural models for question answering and question generation and various careful strategies for question filtering based on curriculum learning and diversity promotion. This led to improved performance for both question answering and question generation on multiple datasets and new state-of-the-art results on *WikiQA* and *TrecQA* datasets.

Acknowledgment

We acknowledge the CMLH fellowship to MS and ONR grant N000141712463 for funding support.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*. ACM, pages 41–48.

Nathaniel Freeman Cantor. 1946. *Dynamics of learning*. Foster and Stewart publishing corporation, Buffalo, NY.

Danqi Chen, Jason Bolton, and Christopher D Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. *arXiv preprint arXiv:1606.02858* .

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *CoRR* abs/1704.00051.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*. pages 376–380.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106* .

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. pages 866–874.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional lstm networks for improved phoneme classification and recognition. *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005* pages 753–753.

Hua He, Kevin Gimpel, and Jimmy J. Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. pages 1576–1586.

Hua He and Jimmy J. Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. pages 937–948.

Michael Heilman. 2011. *Automatic factual question generation from text*. Ph.D. thesis, Carnegie Mellon University.

Michael Heilman and Noah A Smith. 2009. Question generation via overgenerating transformations and ranking. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA LANGUAGE TECHNOLOGIES INST.

Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 609–617.

Karl Moritz Hermann, Tomas Kocisky, 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*. pages 1693–1701.

Minghao Hu, Yuxing Peng, and Xipeng Qiu. 2017. Mnemonic reader for machine comprehension. *CoRR* abs/1705.02798.

Lu Jiang, Deyu Meng, Teruko Mitamura, and Alexander G Hauptmann. 2014. Easy samples first: Self-paced reranking for zero-example multimedia search. In *Proceedings of the ACM International Conference on Multimedia*. ACM, pages 547–556.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. 2015. Self-paced curriculum learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '07, pages 177–180.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *arXiv preprint arXiv:1704.08381* .

Kai A Krueger and Peter Dayan. 2009. Flexible shaping: How learning in small steps helps. *Cognition* 110(3):380–394.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*. pages 1189–1197.

VI Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10:707.

Percy S Liang, Dan Klein, and Michael I Jordan. 2008. Agreement-based learning. In *Advances in Neural Information Processing Systems*. pages 913–920.

Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL, Barcelona, Spain*.

Rui Liu, Junjie Hu, Wei Wei, Zi Yang, and Eric Nyberg. 2017. Structural embedding of syntactic trees for machine comprehension. *CoRR* abs/1703.00572.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *International Conference on Machine Learning*. pages 1727–1736.

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *CoRR* abs/1606.03126.

Karthik Narasimhan and Regina Barzilay. 2015. Machine comprehension with discourse relations. 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 1253–1262.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268* .

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*. ACM, pages 86–93.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL '02, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Gail B Peterson. 2004. A day of great illumination: Bf skinner's discovery of shaping. *Journal of the Experimental Analysis of Behavior* 82(3):317–328.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR* abs/1606.05250. http://arxiv.org/abs/1606.05250.

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*. ACM, New York, NY, USA, CIKM '16, pages 1913–1916. https://doi.org/10.1145/2983323.2983872.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 193–203. http://www.aclweb.org/anthology/D13-1020.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, pages 25–32.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* .

Mrinmaya Sachan, Avinava Dubey, Eric P Xing, and Matthew Richardson. 2015. Learning answer-entailing structures for machine comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Mrinmaya Sachan, Kumar Dubey, and Eric P. Xing. 2016. Science question answering using instructional materials. In *Proceedings of ACL*.

Mrinmaya Sachan and Eric P. Xing. 2016a. Easy questions first? A case study on curriculum learning for question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Mrinmaya Sachan and Eric P. Xing. 2016b. Machine comprehension using rich semantic representations. In *Proceedings of ACL*.

Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609* .

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603* .

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, pages 101–110.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2016. Reasonet: Learning to stop reading in machine comprehension. *CoRR* abs/1609.05284. http://arxiv.org/abs/1609.05284.

Burrhus F Skinner. 1958. Reinforcement today. *American Psychologist* 13(3):94.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* .

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

Bingning Wang, Kang Liu, and Jun Zhao. 2016a. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. volume 2, pages 700–706.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *EMNLP-CoNLL*. volume 7, pages 22–32.

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747* .

Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *CoRR* abs/1706.01450.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016b. Sentence similarity learning by lexical decomposition and composition. *CoRR* abs/1602.07019.

Simon Woo, Zuyao Li, and Jelena Mirkovic. 2016. Good automatic authentication question generation. In *Proceedings of the 9th International Natural Language Generation conference*. pages 203–206.

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

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 189–196.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193* .

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632* .

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* .

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. *CoRR* abs/1704.01792. http://arxiv.org/abs/1704.01792.

Xiaojin Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.