Read + Verify: Machine Reading Comprehension with Unanswerable Questions

Minghao Hu[†][*], Furu Wei[§], Yuxing Peng[†], Zhen Huang[†], Nan Yang[§], Ming Zhou[§]
[†] College of Computer, National University of Defense Technology
[§] Microsoft Research Asia
{huminghao09,pengyuxing,huangzhen}@nudt.edu.cn
{fuwei,nanya,mingzhou}@microsoft.com

Abstract

Machine reading comprehension with unanswerable questions aims to abstain from answering when no answer can be inferred. Previous works using an additional no-answer option attempt to extract answers and detect unanswerable questions simultaneously, but they have struggled to produce high-quality answers and often misclassify questions. In this paper, we propose a novel read-then-verify system that combines a base neural reader with a sentence-level answer verifier trained to further validate if the predicted answer is entailed by input snippets. Moreover, we augment the base reader with two auxiliary losses to better handle answer extraction and no-answer detection respectively, and investigate three different architectures for the answer verifier. Our experiments on the SQuAD 2.0 dataset show that our system can achieve a score of 74.8 F1 on the development set, outperforming the previous best published model by more than 7 points.

1 Introduction

The ability to comprehend text and answer factoid questions is crucial for natural language processing. Due to the creation of various large-scale datasets (Hermann et al., 2015; Hill et al., 2016; Trischler et al., 2016; Nguyen et al., 2016; Joshi et al., 2017; Kočiskỳ et al., 2018), remarkable advancements have been made in the task of machine reading comprehension.

Nevertheless, one important hypothesis behind current approaches is that there always exists a correct answer in the context passage. Therefore, the models only need to choose a most plausible text span based on the question, instead of checking if there exists an answer in the first place. Recently, a new version of Stanford

[*] Preprint. Work in progress.

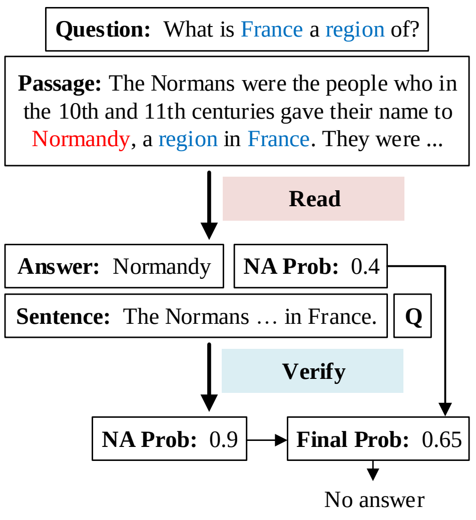
Question: What is France a region of?
Passage: The Normans were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France. They were ...
Read
Answer: Normandy
NA Prob: 0.4
Sentence: The Normans … in France.
Q
Verify
NA Prob: 0.9
Final Prob: 0.65
No answer

Figure 1: An overview of our approach. The reader first extracts a candidate answer and produce a no-answer probability. The answer verifier then checks whether the extracted answer can be entailed by its corresponding sentence and the question. Finally, the system aggregates previous results and outputs the final answer.

Question Answering Dataset (SQuAD), namely SQuAD 2.0 (Rajpurkar et al., 2018), has been proposed to test the ability of answering answerable questions as well as detecting unanswerable questions. To deal with unanswerable cases, systems must learn to recognize a wide range of phenomena such as negation, antonymy and entity changes between the passage and the question.

Previous works (Levy et al., 2017; Clark and Gardner, 2018) have attempted to normalize an additional no-answer score with all of answer span scores, but their extracted answers tend to be low-quality and the accuracy on no-answer detection is also inferior.

In this paper, we propose a read-then-verify system that aims to output precise answers as well as be robust to unanswerable questions. As shown in

Figure 1, our system consists of two components: (1) a no-answer reader for extracting candidate answers and detecting unanswerable questions, and (2) an answer verifier for validating if the candidate answer is actually supported by its surrounding sentence and the question. The key contributions of our work are three-fold.

First, we augment existing readers with two auxiliary losses, to better handle answer extraction and no-answer detection respectively. Since our answer verifier always requires a candidate answer, the reader must be able to extract plausible answers for all questions. However, previous no-answer readers are not trained to find potential candidates for unanswerable questions. We solve this problem by introducing an independent span loss that aims to concentrate on the answer extraction task despite that the question is answerable or not. In order to not conflict with no-answer detection, we leverage a *multi-head answer pointer* to generate two pairs of span scores, where one pair is normalized with the no-answer score and the other is used for our auxiliary loss. Besides, we present another independent no-answer loss to further alleviate the confliction, by focusing on the no-answer detection task without considering the shared normalization with span scores.

Second, in addtion to the standard reading phase, we introduce an additional answer verification phase and explore three different architectures for it. This is based on the observation that the core phenomena of unanswerable questions usually occur between a few passage words and question words. Take the question in Figure 1 for example, after comparing the passage snippet "*Normandy, a region in France*" with the question, we can easily know there is no answer since mutual exclusion happens. Therefore, we utilize an answer verifier to further look into the passage for finding these local, fine-grained entailment, by comparing the answer sentence with the question. We investigate three architectures for the answer verifier. The first one is a sequential model that considers two sentences as a long sequence, while the second one attempts to model inference between two sentences interactively. The last one is a hybrid model that combines the above two models to test if the performance can be further improved.

Lastly, we evaluate our system on the SQuAD 2.0 dataset (Rajpurkar et al., 2018), a reading comprehension benchmark augmented with unanswerable questions. Our best no-answer reader achieves a F1 score of 73.7 and 69.1 on the development set, with or without ELMo embeddings (Peters et al., 2018). When combined with the answer verifier, the whole system improves to 74.8 F1 and 72.3 F1 respectively, where the best result exceeds the previous best published performance by more than 7 points.

2 Approach

In this section we describe our read-then-verify system for robust machine reading comprehension against unanswerable questions. The system first leverages a neural reader to extract a candidate answer and detect if the question is unanswerable. It then utilizes an answer verifier to further recognize local entailment between the answer sentence and the question.

2.1 No-Answer Reader with Auxiliary Losses

Our no-answer reader is inspired by the recent success of neural network models on unanswerable machine comprehension tasks (Levy et al., 2017; Clark and Gardner, 2018). We start from the standard no-answer reader that jointly learns answer extraction and no-answer detection, and then introduce two auxiliary losses to optimize and enhance each task independently without interfering with each other.

No-Answer Option: We first briefly introduce the no-answer option approach (Clark and Gardner, 2018). In this approach, a joint no-answer loss function is defined as:

$$\mathcal{L}_{joint} = -\log\left(\frac{(1-\delta)e^{z} + \delta e^{s_a g_b}}{e^{z} + \sum_{i=1}^{n}\sum_{j=1}^{n} e^{s_i g_j}}\right)$$

where s_i and g_i are the predicted scores of the answer start/end positions for token i, a and b are the ground-truth start and end positions, n is the passage length, and z is the score for an additional no-answer possibility. δ is 1 if the question is answerable and 0 otherwise. Through this loss, both of answer extraction and no-answer detection can be jointly optimized with a shared softmax normaliztion.

Independent Span Loss: This loss is designed to concentrate on the answer extraction task. In this task, the model is asked to extract a candidate answer for all possible questions. Therefore, besides the answerable questions, we also include all

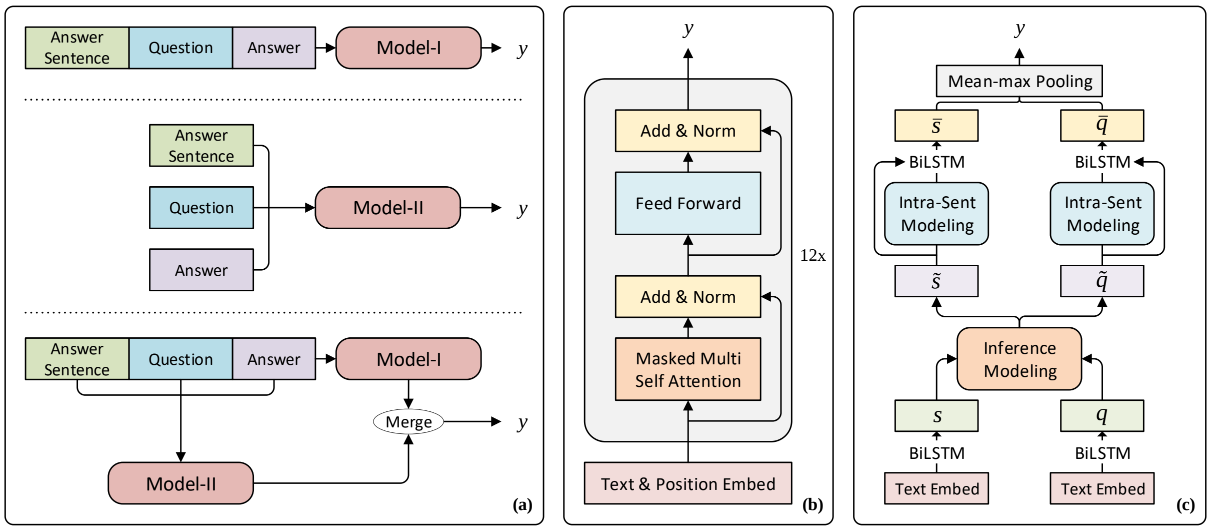

Figure 2: Overview of our answer verification architectures. (a) Input structures for running three different models. (b) Finetuned Transformer model proposed by Radford et al. (2018). (c) Our proposed interactive model.

unanswerable cases as positive examples, and consider the most plausible answers as ground-truth labels. Instead of directly using the original span scores s and g, we propose to use a *multi-head answer pointer* to additionally produce another pair of scores $\tilde{s}$ and $\tilde{g}$. This two pairs of scores share the same decoding architecture but with different parameters. In this way, we define an independent span loss as:

$$\mathcal{L}_{indep-I} = -\log\left(\frac{e^{\tilde{s}_{\tilde{a}}\tilde{g}_{\tilde{b}}}}{\sum_{i=1}^{n}\sum_{j=1}^{n} e^{\tilde{s}_i\tilde{g}_j}}\right)$$

where $\tilde{a}$ and $\tilde{b}$ are the augmented ground-truth answer boundaries. The final span probability is obtained using a simple mean pooling over the two pairs of softmax-normalized scores.

Independent No-Answer Loss: One problem of the previous independent loss is that it may conflict with the joint no-answer loss. For unanswerable questions, now the model is required to find an "answer" as well as detect that no answer exists. To alleviate this problem, we consider exclusively encouraging the model's confidence on no-answer detection. This is achieved by introducing another independent loss as:

$$\mathcal{L}_{indep-II} = -(1-\delta)\log\sigma(z) - \delta\log(1-\sigma(z))$$

where σ is the sigmoid activation function. Through this loss, we expect the model to produce a more confident prediction on no-answer score z without considering the shared normalization from answer extraction.

Finally, we combine the above losses as follows:

$$\mathcal{L} = \mathcal{L}_{joint} + \gamma\mathcal{L}_{indep-I} + \lambda\mathcal{L}_{indep-II}$$

where γ and λ are two hyper-parameters that control the weight of two auxiliary losses.

2.2 Answer Verifiers

After the no-answer reader extracts a candidate answer, an answer verifier is used to further recognize the local, fine-grained entailment between the answer sentence and the question. We explore three different architectures for this task (shown in Figure 2): (1) a sequential model that considers the inputs as a long sequence, (2) an interactive model that learns the interaction between the answer sentence and the question, and (3) a hybrid model that takes both of the two approaches into account.

2.2.1 Model-I: Sequential Architecture

In Model-I, we convert the answer sentence and the question along with the extracted answer into an ordered input sequence. Then we adapt the recently proposed Finetuned Transformer model (Radford et al., 2018) to perform the answer verification. The model is a multi-layer Transformer decoder (Liu et al., 2018), which is first trained with a language modeling objective on a large unlabeled text corpus and then finetuned on the specific target task.

Specifically, given an answer sentence s, a question q and an extracted answer a, we concatenate the two sentences with the answer while adding a delimiter token in between to get $[s; q; \$; a]$. The sequence, which can also be denoted as a series of tokens $X = [x_1, ..., x_m]$, is then encoded by a multi-head self-attention operation followed by position-wise feed-forward layers as follows:

$$h_0 = W_e[X] + W_p$$
$$h_i = \text{transformer\_block}(h_{i-1}), \forall i \in [1, n]$$

where X denotes the sequence's indexes in the vocab, W_e is the token embedding matrix, W_p is the position embedding matrix, and n is the number of layers.

The last token's activation h_n^m is then fed into a linear projection layer followed by a softmax function to output the no-answer probability y:

$$p(y|X) = \text{softmax}(h_n^m W_y)$$

A standard cross-entropy objective is used to minimize the negative log-likelihood:

$$\mathcal{L}(\theta) = \sum_{(X,y)} \log p(y|X)$$

2.2.2 Model-II: Interactive Architecture

Since the answer verification task requires the model to recognize local entailment between two sentences, therefore we also consider an interaction-based approach that has the following layers:

Encoding: We embed words using the GloVe embedding (Pennington et al., 2014), and also embed characters of each word with trainable vectors. We run a bi-directional long short-term memory network (BiLSTM) (Hochreiter and Schmidhuber, 1997) to encode the characters and concatenate two last hidden states to get character-level embeddings. In addition, we use a binary feature to indicate if a word is part of the answer. All embeddings along with the feature are then concatenated and encoded by a weight-shared BiLSTM, yielding two series of contextual embeddings:

$$s_i = \text{BiLSTM}([\text{word}_i^s; \text{char}_i^s; \text{fea}_i^s]), \forall i \in [1, l_s]$$
$$q_j = \text{BiLSTM}([\text{word}_j^q; \text{char}_j^q; \text{fea}_j^q]), \forall j \in [1, l_q]$$

where l_s and l_q are the lengths of answer sentence and question respectively, and $[\cdot; \cdot]$ denotes concatenation.

Inference Modeling: A inference modeling layer is used to capture the interactions between two sentences and produce two inference-aware sentence representations. We first compute the dot products of all tuples $< s_i, q_j >$ as attention weights, and then normalize these weights so as to obtain attended vectors as follows:

$$a_{ij} = s_i^\intercal q_j, \forall i \in [1, l_s], \forall j \in [1, l_q]$$
$$b_i = \sum_{j=1}^{l_q} \frac{e^{a_{ij}}}{\sum_{k=1}^{l_q} e^{a_{ik}}} q_j \ , \ c_j = \sum_{i=1}^{l_s} \frac{e^{a_{ij}}}{\sum_{k=1}^{l_s} e^{a_{kj}}} s_i$$

We separately fuse local inference information between aligned pairs $\{(s_i, b_i)\}_{i=1}^{l_s}$ and $\{(q_j, c_j)\}_{j=1}^{l_q}$ using a weight-shared function F:

$$\tilde{s}_i = F(s_i, b_i) \ , \ \tilde{q}_j = F(q_j, c_j)$$

A heuristic fusion function $o = F(x, y)$ proposed by Hu et al. (2018) is used as:

$$r = \text{gelu}\left(W_r[x; y; x \circ y; x - y]\right)$$
$$g = \sigma\left(W_g[x; y; x \circ y; x - y]\right)$$
$$o = g \circ r + (1 - g) \circ x$$

where gelu is the Gaussian Error Linear Unit (Hendrycks and Gimpel, 2016), $\circ$ is element-wise multiplication, and the bias term is omitted.

Intra-Sentence Modeling: Next we apply a intra-sentence modeling layer to capture self correlations inside each sentence. The input is first passed through another BiLSTM layer for encoding. We then use the same attention mechanism described above, only now between the sentence and itself, and we set $a_{ij} = -inf$ if $i = j$ to ensure that the word is not aligned with itself. Another fusion function is used to produce $\hat{s}_i$ and $\hat{q}_j$ respectively.

Prediction: Before the final prediction, we apply a concatenated residual connection and model the sentences with BiLSTM as:

$$\bar{s}_i = \text{BiLSTM}([\tilde{s}_i; \hat{s}_i]) \ , \ \bar{q}_j = \text{BiLSTM}([\tilde{q}_j; \hat{q}_j])$$

A mean-max pooling operation is then applied to summarize the representation of two sentences. All summarized vectors are then concatenated and fed into a feed-forward classifier that consists of a projection sub-layer with gelu activation and a softmax output sub-layer. As before, we optimize the negative log-likelihood objective function.

2.2.3 Model-III: Hybrid Architecture

To explore how the features extracted by model-I and model-II can be integrated to yield better representation capacities, we investigate the combination of the above two models, namely Model-III. We merge the output vectors of two models into a single combined representation. An unified feed-forward classifier is then applied to output the no-answer probability. Such design allows us to test whether the performance can benefit from the combination of two different architectures. In practice we use a simple concatenation to merge the two sources of information.

3 Experimental Setup

3.1 Dataset

We evaluate our approach on the SQuAD 2.0 dataset (Rajpurkar et al., 2018). SQuAD 2.0 is a new machine reading comprehension benchmark that aims to test the models whether they have truely understood the questions by knowing what they don't know. It combines answerable questions from the previous SQuAD 1.1 dataset (Rajpurkar et al., 2016) with 53,775 unanswerable questions about the same passages. Crowdsourcing workers craft these questions with a plausible answer in mind, and make sure that they are relevant to the corresponding passages.

3.2 Training and Inference

Our no-answer reader is trained on the entire passages, while the answer verifier is trained on oracle sentences. Model-I is trained with a combination of unsupervised pre-training and supervised fine-tuning. That is, the model is first optimized with a language modeling objective on a large unlabeled text corpus to initialize its parameters. Then it adapts the parameters to the answer verification task with our supervised objective. For model-II, we directly train it with the supervised loss. Model-III, however, consists of two different architectures that require different training procedures. Therefore, we initialize Model-III with the pre-trained parameters from both of model-I and model-II, and then fine-tune the whole model until convergence.

At test time, the reader first produces a candidate answer as well as a passage-level no-answer probability. The answer verifier then validates the extracted answer and outputs a sentence-level probability. Following the official evaluation setting, a question is detected to be unanswerable once the joint no-answer probability exceeds some threshold [1]. We tune this threshold to maximize F1 score on the development set, and report both of EM (Exact Match) and F1 metrics. We also evaluate the performance on no-answer detection with an accuracy metric, where its threshold is set as 0.5 by default.

Model	EM	F1
BNA[1]	59.8	62.6
DocQA[2]	61.9	64.8
DocQA + ELMo	65.1	67.6
R.M-Reader	66.9	69.1
R.M-Reader + Verifier	**68.5**	**71.5**
R.M-Reader + ELMo	71.4	73.7
R.M-Reader + ELMo + Verifier	**72.3**	**74.8**
Human	86.3	89.0

Table 1: Comparison of different approaches on the SQuAD 2.0 dev set: Levy et al. (2017)[1] and Clark et al. (2018)[2].

3.3 Implementation

We use the Reinforced Mnemonic Reader (R.M-Reader) (Hu et al., 2018), one of the state-of-the-art reading comprehension models on the SQuAD 1.1 dataset, as our base reader. The reader is configurated with its default setting, and trained with the no-answer objective with our auxiliary losses. ELMo embeddings (Peters et al., 2018) are exclusively listed in our experimental configuration. The hyper-parameter γ is set as 0.3, and λ is 1. As for answer verifiers, we use the original configuration from Radford et al. (2018) for model-I. As for model-II, the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0008 is used, the hidden size is set as 300, and a dropout (Srivastava et al., 2014) of 0.3 is applied for preventing overfitting. The batch size is 48 for the reader, 64 for model-II and 32 for model-I and model-III. The GloVe 100D embeddings (Pennington et al., 2014) are used for the reader, and 300D embeddings for model-II and model-III. We truncate passages so that the length is less than 300, and truncate sentences for not exceeding 150.

[1]We compute the mean of passage-level and sentence-level no-answer probabilities as the joint probability.

Table 2: Ablation study of each individual component on the SQuAD 2.0 dev set.

(a) Comparison of reader with different auxiliary losses.

Reader	HasAns		All		NoAns
	EM	F1	EM	F1	ACC
R.M-Reader	**72.6**	**81.6**	**66.9**	**69.1**	**73.1**
- indep-I	71.3	80.4	66.0	68.6	72.8
- indep-II	72.4	81.4	64.0	66.1	69.8
- both	71.9	80.9	65.2	67.5	71.4

(b) Comparison of different architectures for answer verifier.

Answer Verifier	NoAns ACC
Model-I	74.5
Model-II	74.6
Model-II + ELMo	75.3
Model-III	**76.2**
Model-III + ELMo	76.1

4 Evaluation

4.1 Main Results

We first evaluate our approach on the development set of SQuAD 2.0, which is shown in Table 1. As we can see, the base reader augmented with our auxiliary losses already outperforms previous state-of-the-art results: R.M-Reader achieves 69.1 F1 while R.M-Reader + ELMo reaches a F1 of 73.7. The whole system further boosts the performance to 71.5 and 74.8 F1 respectively.

4.2 Ablation Study

Next, we do an ablation study on the SQuAD 2.0 development set to show the effects of our proposed methods for each individual component. Table 2(a) first shows the ablation results of different auxiliary losses on the reader. Removing the independent span loss (indep-I) results in a performance drop on both of EM and F1 for all answerable questions (HasAns), indicating that this loss helps the model in better identifying candidate answers' boundaries. Ablating independent no-answer loss (indep-II), on the other hand, causes little influence on the performance of HasAns, but leads to a severe decline of more than 3 points on no-answer accuracy. This suggests that a confliction indeed happens when only adding independent span loss. Finally, deleting both of two losses causes a drop of more than 1.5 points on the overall performance in terms of EM and F1.

Table 2(b) details the results of various architectures for the answer verifier. Model-III outperforms all of the other competitors, achieving a no-answer accuracy of 76.2. This illustrates that the combination of two different architectures can bring in richer set of features. Adding ELMo embeddings, however, does not further improve the performance. We hythosize that the bytepair encoding (Sennrich et al., 2016) from Model-I and the word/character embeddings from Model-II provide enough representation capacities for this verification task.

Configuration	All		NoAns
	EM	F1	ACC
R.M-Reader	66.9	69.1	73.1
+ Model-I	68.3	71.1	76.2
+ Model-II	68.1	70.8	75.6
+ Model-II + ELMo	68.2	70.9	75.9
+ Model-III	**68.5**	**71.5**	**77.1**
+ Model-III + ELMo	68.5	71.2	76.5
R.M-Reader + ELMo	71.4	73.7	77.0
+ Model-I	71.8	74.4	77.3
+ Model-II	71.8	74.2	78.1
+ Model-II + ELMo	72.0	74.3	78.2
+ Model-III	**72.3**	**74.8**	**78.6**
+ Model-III + ELMo	71.8	74.3	78.3

Table 3: Comparison of readers with different answer verifiers on the SQuAD 2.0 dev set.

After doing separate ablations on each component, we then compare the performance of the whole system, as shown in Table 3. We notice that the combination of base reader with any answer verifier can always result in considerable performance gains: R.M-Reader + Model-III achieves the best result, outperforming the base reader by more than 2 points in terms of F1. We conjecture that the improvement comes from the significant increasement on no-answer accuracy: the metric raises by 4 points with our best answer verifier. The similar observation can be found when ELMo embeddings are used in the reader, indicating that the improvement is consistent and stable.

5 Related Work

Reading Comprehension Datasets. Recent years have witnessed rapid progress in the task of machine reading comprehension. This task has evolved from early cloze-style test (Hermann et al., 2015; Hill et al., 2016), in which the goal is to predict the missing word in a passage, to answer extraction test (Rajpurkar et al., 2016; Trischler et al., 2016), which aims to choose a text span in the passage as the answer. Later, several datasets have extended from reading a single passage to comprehending multiple passages, (Nguyen et al., 2016), a document (Joshi et al., 2017) or even an entire book (Kočiskỳ et al., 2018). However, all of these datasets still guarantee that the given context must contain an answer. In this work we choose to focus on the SQuAD 2.0 dataset, which mixes answerable questions from SQuAD 1.1 with a large-scale, high-quality set of unanswerable questions.

Neural Networks for Reading Comprehension. Neural reading models typically leverage various attention mechanisms such as bidirectional attention (Seo et al., 2017), self attention (Wang et al., 2017) and reattention (Hu et al., 2018) to build codependent representations of the passage and the question. Researchers further incorporate these neural readers with a ranking model for open-domain question answering (Chen et al., 2017a). However, all of these models are not designed to handle no-answer cases: they tend to make over-confident guesses on unanswerable questions and return a span relying on type-matching heuristics. To address this problem, previous works (Levy et al., 2017; Clark and Gardner, 2018) attempted to learn an additional no-answer probability in addition to a distribution over answer spans. Our no-answer reader extends existing approaches by introducing two auxiliary losses that independently optimize and enhance answer extraction as well as no-answer detection.

Recognizing Textual Entailment. Recognizing textual entailment requires systems to understand entailment, contradiction or semantic neutrality between two sentences (Marelli et al., 2014; Bowman et al., 2015). This task is strongly related to the no-answer detection, where models are requested to understand whether a candidate answer is entailed by the passage and the question. To recognize entailment, various works have been proposed. The first branch of works is the sentence encoding-based approach (Mou et al., 2015; Bowman et al., 2016), in which two sentences are encoded separately to form two independent representations for final classification. Another branch is the interaction-based approach (Parikh et al., 2016; Chen et al., 2017b) that aims to compare aligned text pairs between two sentences and then aggregate the results. Unlike previous works, the recent proposed sequence-based approach (Radford et al., 2018) attempts to encode sentences and capture interactions simultaneously. In this paper we focus on the last two methods and further propose a hybrid architecture that combines both of them properly.

6 Conclusion

We proposed a read-then-verify system that is able to abstain from answering when a question has no answer given the passage. We first introduce two auxiliary losses to help the reader concentrate on answer extraction and no-answer detection respectively, and then utilize an additional answer verifier to further validate if the candidate answer is entailed by input snippets. We have explored three different architectures for the answer verifier and find the hybrid architecture yields the best performance. Our system has achieved state-of-the-art results on the SQuAD 2.0 dataset, outperforming the previous best published model by more than 7 points. Looking forward, we expect to design new network structures for the answer verification task that requires to handle questions with more complicated inferences.

References

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

Samuel R Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. *arXiv preprint arXiv:1603.06021*.

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

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced lstm for natural language inference.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of ACL*.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv preprint arXiv:1606.08415*.

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

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

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

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In *Proceedings of IJCAI*.

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

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

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gáabor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association of Computational Linguistics*, 6:317–328.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *LREC*, pages 216–223.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv preprint arXiv:1512.08422*.

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*.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

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

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word prepresentations. In *Proceedings of NAACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of ACL*.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units.

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

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, pages 1929–1958.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of ACL*.