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. In addition to extract answers, previous works usually predict an additional "no-answer" probability to detect unanswerable cases. However, they fail to validate the answerability of the question by verifying the legitimacy of the predicted answer. To address this problem, we propose a novel read-then-verify system, which not only utilizes a neural reader to extract candidate answers and produce no-answer probabilities, but also leverages an answer verifier to decide whether the predicted answer is entailed by the input snippets. Moreover, we introduce two auxiliary losses to help the reader better handle answer extraction and no-answer detection, and investigate three different architectures for the answer verifier. Our experiments on the SQuAD 2.0 dataset show that our system achieves a score of 74.2 F1 on the test set, outperforming all previous approaches at the time of submission (Aug. 23th, 2018).

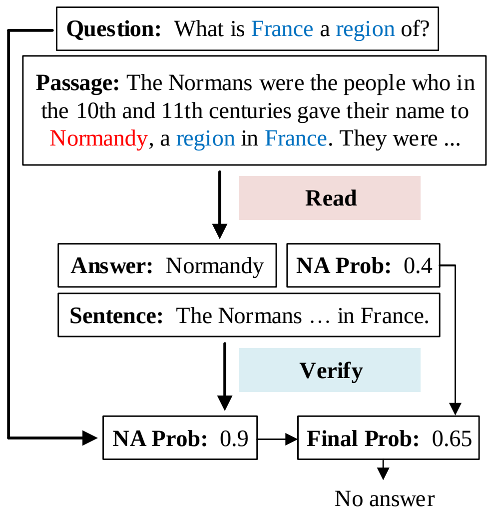

Figure 1: An overview of our approach. The reader first extracts a candidate answer and produce a no-answer probability (NA Prob). The answer verifier then checks whether the extracted answer is legitimate or not. Finally, the system aggregates previous results and outputs the final prediction.

1 Introduction

The ability to comprehend text and answer questions is crucial for natural language processing. Due to the creation of various large-scale datasets (Hermann et al., 2015; 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 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 cases. To deal with unanswerable cases, systems must learn to identify a wide range of linguistic 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, so that they can produce a no-answer probability and output a candidate answer simultaneously. However, they have not considered to further validate the answerability of the question by verifying the legitimacy of the predicted answer.

[*] Preprint. Work in progress.

Here, "answerability" denotes whether the question is answerable or not, and "legitimacy" means if the extracted text is entailed by the passage and the question. Human, on the contrary, tends to first find a plausible answer given a question, and then checks if there exists any contradictory semantics.

In this paper, we propose a read-then-verify system that aims to 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 deciding whether the extracted candidate is legitimate or not. 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 the downstream answer verification always requires a candidate answer, the reader must be able to extract plausible answers for all questions. However, previous approaches 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 regardless of the answerability of the question. In order to not conflict with no-answer detection, we leverage a *multi-head pointer network* 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 of span scores.

Second, in addition to the standard reading phase, we introduce an additional answer verification phase. 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 determine that no answer exists since the question asks for an impossible condition. 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 it. The first one is a sequential model that considers two sentences as a long sequence, while the second one attempts to model inferences 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. Moreover, the best system achieves a score of 74.2 F1 on the test set, outperforming all previous approaches at the time of submission (Aug. 23th, 2018).

2 Background

Existing reading comprehension models focus on answering questions where a correct answer is guaranteed to exist. However, they are not able to identify unanswerable questions but tend to return an unreliable text span. Consequently, we first give a brief introduction on the unanswerable reading comprehension task, and then investigate current solutions.

2.1 Task Description

Given a context passage and a question, the machine needs to not only find answers to answerable questions but also detect unanswerable cases. The passage and the question are described as sequences of word tokens, denoted as $P = \{x_i^p\}_{i=1}^{l_p}$ and $Q = \{x_j^q\}_{j=1}^{l_q}$ respectively, where l_p is passage length and l_q is question length. Our goal is to predict an answer A, which is constrained as a segment of text in the passage: $A = \{x_i^p\}_{i=l_a}^{l_b}$, or return an empty string if there is no answer, where l_a and l_b indicate the answer boundary.

2.2 No-Answer Reader

To predict an answer span, current approaches first embed and encode both of passage and question into two series of fix-sized vectors. Then they leverage various attention mechanisms, such as bi-attention (Seo et al., 2017) or reattention (Hu et al., 2018a), to build interdependent representations for passage and question, which are denoted as $U = \{u_i\}_{i=1}^{l_p}$ and $V = \{v_j\}_{j=1}^{l_q}$ respectively. Finally, they summarize the question representation into a dense vector t, and utilize the pointer network (Vinyals et al., 2015) to produce two scores over passage words that indicate the answer boundary (Wang et al., 2017; Yu et al., 2018):

$$o_j = w_v^{\mathrm{T}} v_j \ , \ t = \sum_{j=1}^{l_q} \frac{e^{o_j}}{\sum_{k=1}^{l_q} e^{o_k}} v_j$$
$$\alpha, \beta = \text{pointer\_network}(U, t)$$

where α and β are the span scores for answer start and end bounds.

In order to additionally detect if the question is unanswerable, previous approaches (Levy et al., 2017; Clark and Gardner, 2018) attempt to predict a special no-answer score z in addition to the distribution over answer spans. Concretely, a shared softmax function can be applied to normalize both of no-answer score and span scores, yielding a joint no-answer objective defined as:

$$\mathcal{L}_{joint} = -\log \left(\frac{(1-\delta)e^z + \delta e^{\alpha_a \beta_b}}{e^z + \sum_{i=1}^{l_p} \sum_{j=1}^{l_p} e^{\alpha_i \beta_j}} \right)$$

where a and b are the ground-truth start and end positions, and δ is 1 if the question is answerable and 0 otherwise. At test time, a question is detected as being unanswerable once the normalized no-answer score exceeds some threshold.

3 Approach

In this section we describe our read-then-verify system for machine reading comprehension with 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 check the legitimacy of the predicted answer. We enhance the reader with two novel auxiliary losses, and investigate three different architectures for the answer verifier.

3.1 Reader with Auxiliary Losses

Although previous no-answer readers are capable of jointly learning answer extraction and no-answer detection, there exists two problems for each individual task. For the answer extraction, previous readers are not trained to find candidate answers for unanswerable questions. In our system, however, the reader is required to extract a plausible answer that is fed to the downstream answer verification for all questions. As for no-answer detection, a confliction could be triggered due to the shared normalization between span scores and no-answer score: an over-confident span probability would cause an unconfident no-answer probability, and vice versa. Therefore, inaccurate confidences on answer extraction, which has been observed by Clark et al. (2018), could cause imprecise predictions on no-answer detection. To address the above issues, we propose two auxiliary losses to optimize and enhance each task independently without interfering with each other.

Independent Span Loss: This loss is designed to concentrate on answer extraction. 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 unanswerable cases as positive examples, and consider plausible answers as ground-truth span labels. In order to not conflict with no-answer detection, we propose to use a *multi-head pointer network* to additionally produce another pair of span scores $\tilde{\alpha}$ and $\tilde{\beta}$:

$$\tilde{o}_j = \tilde{w}_v^{\mathrm{T}} v_j \ , \ \tilde{t} = \sum_{j=1}^{l_q} \frac{e^{\tilde{o}_j}}{\sum_{k=1}^{l_q} e^{\tilde{o}_k}} v_j$$
$$\tilde{\alpha}, \tilde{\beta} = \text{pointer\_network}(U, \tilde{t})$$

where multiple heads share the same network architecture but with different parameters.

Then, we define an independent span loss as:

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

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 span scores.

Independent No-Answer Loss: Despite a multi-head pointer network being used to prevent the confliction problem, no-answer detection can still be weakened since no-answer score z is normalized with span scores. Therefore, we consider exclusively encouraging the prediction on no-answer detection. This is achieved by introducing an independent no-answer 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

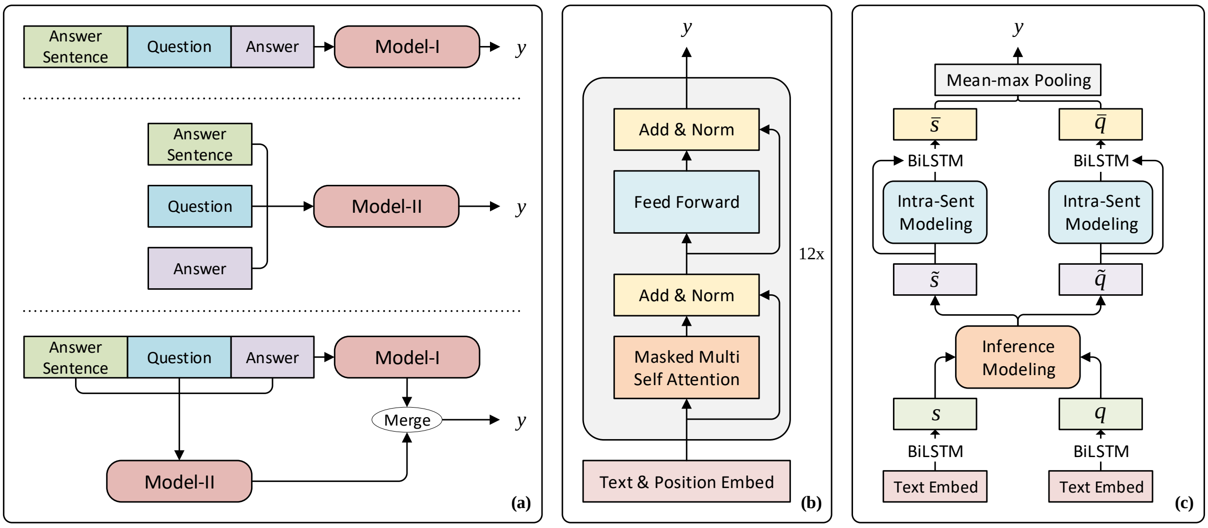

Figure 2: Overview of our answer verification architectures. (a) Input structures for running three different models. (b) Finetuned Transformer model proposed by Radford et al. (2018). (c) Our proposed interactive model.

z without considering the shared normalization of 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.

3.2 Answer Verifier

After the no-answer reader extracts a candidate answer, an answer verifier is used to check the answer's legitimacy by recognizing local textual 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 encodes two sentences codependently, and (3) a hybrid model that takes both of the two approaches into account.

3.2.1 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., 2018a), 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 $X = \{x_i\}_{i=1}^{l_m}$, 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 $h_n^{l_m}$ is then fed into a linear projection layer followed by a softmax function to output the no-answer probability y:

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

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

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

3.2.2 Model-II: Interactive Architecture

Since answer verification requires to model inferences 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 is the length of answer sentence, and $[\cdot;\cdot]$ denotes concatenation.

Inference Modeling: An 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:

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

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. (2018a) 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 an 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.

3.2.3 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 joint 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 integration of two different architectures. In practice we use a simple concatenation to merge the two sources of information.

4 Experimental Setup

4.1 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.

4.2 Training and Inference

Our no-answer reader is trained on passages, while the answer verifier is trained on oracle sentences.

Model	Dev		Test	
	EM	F1	EM	F1
BNA[1]	59.8	62.6	59.2	62.1
DocQA[2]	61.9	64.8	59.3	62.3
DocQA + ELMo	65.1	67.6	63.4	66.3
VS[3]−Net†	-	-	68.4	71.3
SAN[3]	-	-	68.6	71.4
FusionNet++(ensemble)[4]	-	-	70.3	72.6
RMR + ELMo + Verifier	**72.3**	**74.8**	**71.7**	**74.2**
Human	86.3	89.0	86.9	89.5

Table 1: Comparison of different approaches on the SQuAD 2.0 test set, extracted on Aug 23, 2018: Levy et al. (2017)[1], Clark et al. (2018)[2], Liu et al. (2018b)[3] and Huang et al. (2018)[4]. † indicates unpublished works.

	HasAns		All		NoAns
	EM	F1	EM	F1	ACC
RMR	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
RMR + ELMo	79.4	86.8	71.4	73.7	77.0
- indep-I	78.9	86.5	71.2	73.5	76.7
- indep-II	79.5	86.6	69.4	71.4	75.1
- both	78.7	86.2	70.0	71.9	75.3

Table 2: Comparison of readers with different auxiliary losses.

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

Table 3: Comparison of different architectures for the answer verifier.

Model-I follows a procedure 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.

4.3 Implementation

We use the Reinforced Mnemonic Reader (RMR) (Hu et al., 2018a), 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. 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 words, and truncate sentences for not exceeding 150 words.

[1]We compute the mean of passage-level probability and sentence-level probability as the joint probability.

5 Evaluation

5.1 Main Results

We first submit our approach on the hidden test set of SQuAD 2.0 for evaluation, which is shown in Table 1. As we can see, our system achieves an EM score of 71.7 and a F1 score of 74.2, outperforming all previous approaches.

5.2 Ablation Study

Next, we do an ablation study on the SQuAD 2.0 development set to show the effects of our

Configuration	All EM	All F1	NoAns ACC
RMR	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
RMR + 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 4: Comparison of readers with different answer verifiers on the SQuAD 2.0 dev set.

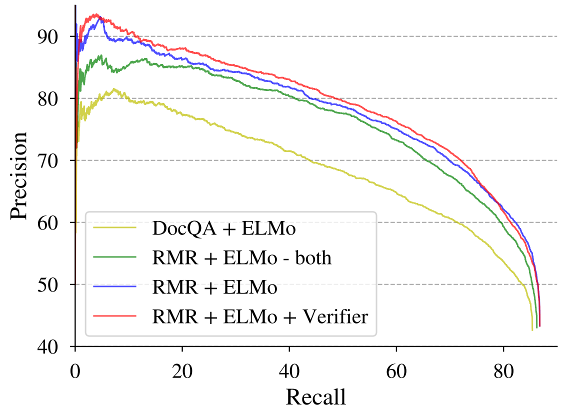

Figure 3: Precision-Recall curves of F1 score.

proposed methods for each individual component. Table 2 first shows the ablation results of different auxiliary losses on the reader. Removing the independent span loss (indep-I) results in a performance drop for all answerable questions (HasAns), indicating that this loss helps the model in better identifying the answer boundary. Ablating independent no-answer loss (indep-II), on the other hand, causes little influence on HasAns, but leads to a severe decline on no-answer accuracy (NoAns ACC). This suggests that a confliction between answer extraction and no-answer detection indeed happens. Finally, deleting both of two losses causes a degradation of more than 1.5 points on the overall performance in terms of F1, with or without ELMo embeddings.

Table 3 details the results of various architectures for the answer verifier. Model-III outperforms all of other competitors, achieving a no-answer accuracy of 76.2. This illustrates that the combination of two different architectures can bring in further improvement. Adding ELMo embeddings, however, does not boost the performance. We hythosize that the bytepair encoding (Sennrich et al., 2016) from Model-I and the word/character embeddings from Model-II have provided enough representation capacities.

After doing separate ablations on each component, we then compare the performance of the whole system, as shown in Table 4. We notice that the combination of base reader with any answer verifier can always result in considerable performance gains, and combining the reader with Model-III obtains the best result. We conjecture that the improvement comes from the significant increasement on no-answer accuracy: the metric raises by 4 points when using RMR as the base reader. The similar observation can be found when ELMo embeddings are used, demonstrating that the improvements are consistent and stable.

Finally, we plot the precision-recall curves of F1 score on the development set in Figure 3. We observe that RMR + ELMo + Verifier achieves the best precision when the recall is less than 80. After the recall exceeds 80, the precision of RMR + ELMo becomes slightly better. Ablating two auxiliary losses, however, leads to an overall degradation on the curve, but it still outperforms the baseline by a large margin.

5.3 Error Analysis

To perform error analysis, we first categorize all examples on the development set into 5 classes:

- Case1: the question is answerable, the no-answer probability is less than the threshold, and the answer is correct.
- Case2: the question is unanswerable, and the no-answer probability is larger than the threshold.
- Case3: almost the same as case1, except that the predicted answer is wrong.
- Case4: the question is unanswerable, but the no-answer probability is less than the threshold.
- Case5: the question is answerable, but the no-answer probability is larger than the threshold.

Configuration	Case1 ✓	Case2 ✓	Case3 ✗	Case4 ✗	Case5 ✗
RMR - both	27.8	37.3	6.5	12.7	15.7
RMR	27	**39.9**	**5.9**	**10.2**	17
RMR + Verifier	**30.3**	38.2	8.4	11.8	**11.3**
RMR + ELMo - both	31.5	38.3	5.6	11.8	12.8
RMR + ELMo	31.2	**40.2**	**5.5**	**9.9**	13.2
RMR + ELMo + Verifier	**32.5**	39.8	6.5	10.3	**10.9**

Table 5: Percentage of five categories. Correct predictions are denoted with ✓, while wrong cases are marked with ✗.

Phenomenon	Percentage	
	All	Error
Negation	9	0
Antonym	20	8
Entity Swap	21	24
Mutual Exclusion	15	16
Impossible Condition	4	14
Other Neutral	24	32
Answerable	7	6

Table 6: Linguistic phenomena exhibited by all negative examples (data from (Rajpurkar et al., 2018)) and sampled error cases of RMR + ELMo + Verifier.

We then show the percentage of each category in the Table 5. As we can see, the base reader trained with auxiliary losses is notably better at case2 and case4 compared to the baseline, implying that our proposed losses help the model mainly improve upon unanswerable cases. After adding the answer verifier, we observe that although the system's performance on unanswerable cases slightly decreases, the results on case1 and case5 have been improved. This demonstrates that the answer verifier does well on detecting answerable question rather than unanswerable one. However, there are still more than 20% of examples that are misclassified even with our best system. Therefore we argue that the main performance bottleneck lies in no-answer detection instead of answer extraction.

Next, in order to understand the challenges our approach faces, we manually investigate 50 incorrectly predicted unanswerable examples (based on F1) that are randomly sampled from the development set. Following the types of negative examples defined by Pranav et al. (2018), we categorize the sampled examples and show them in Table 6. As we can see, our system is good at recognize negation and antonym. The frequency of negation decreases from 9 to 0 and only 4 antonym examples are predicted wrongly. We argue that this is because the two cases are relatively easier to identify. Both of negation and antonym only require to detect one single word in the question, such as "never" or "not" for negation and "increase" to "decrease" for antonym. However, "impossible condition" and "other neutral" cases acount for 44% of the error set, suggesting that our system performs poorly on these cases.

6 Related Work

Reading Comprehension Datasets. Various large-scale machine reading comprehension (MRC) datasets, such as cloze-style test (Hermann et al., 2015), extractive benchmark (Rajpurkar et al., 2016; Joshi et al., 2017) and abstractive benchmark (Nguyen et al., 2016; Kočiskỳ et al., 2018), have been proposed. However, these datasets still guarantee that the given context must contain an answer. Recently, Some works construct negative examples by retrieving passages for existing questions based on Lucene and TF-IDF (Tan et al., 2018; Clark and Gardner, 2018), or using crowdworkers to craft unanswerable questions (Rajpurkar et al., 2018). Compared to automatically generated negative examples, human-annotated examples are more difficult to detect since they are are relevant to the passage and a plausible answer is contained. Therefore, we choose to work on the SQuAD 2.0 dataset in this paper.

Neural Networks for Reading Comprehension. Neural reading models typically leverage various attention mechanisms (Seo et al., 2017; Hu et al., 2018a) to build interdependent representations of passage and question, and sequentially predict the answer boundary (Wang et al., 2017; Yu et al., 2018; Hu et al., 2018b). However, these approaches are not designed to handle no-answer cases. To address this problem, previous works (Levy et al., 2017; Clark and Gardner, 2018) normalize an additional no-answer score with scores of 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 (RTE), or known as natural language inference (NLI), requires systems to understand entailment, contradiction or semantic neutrality between two sentences (Dagan et al., 2010; Marelli et al., 2014; Bowman et al., 2015). This task is strongly related to no-answer detection, where the machine needs to understand if the passage and the question entails the answer. To recognize entailment, various branches of works have been proposed, including encoding-based approach (Bowman et al., 2016), interaction-based approach (Parikh et al., 2016) and sequence-based approach (Radford et al., 2018). In this paper we investigate the last two branches and further propose a hybrid architecture that combines both of them properly.

Answer Validation. Early answer validation task aims at ranking multiple candidate answers to return a most reliable one (Magnini et al., 2002). Later, the answer validation exercise has been proposed to decide whether an answer is correct or not according to a given supporting text and a question (Rodrigo et al., 2008), but the dataset is too small for neural network-based approaches. Recently, Tan et al. (2018) propose an answer validation based method to validate the answer for reading comprehension tasks, by comparing the question with the passage. Our answer verifier, on the contrary, denoises the passage by comparing question with answer sentence, so as to focus on finding local entailment that supports the answer.

7 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 answer verifier to validate the legitimacy of the predicted answer. 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 all previous approaches at the time of submission (Aug. 23th, 2018). Looking forward, we plan to design new network structures for the answer verification task that requires to handle questions with more complicated inferences.

Acknowledgments

We would like to thank Pranav Rajpurkar and Robin Jia for their helps with SQuAD 2.0 submissions.

References

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

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*.

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

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(1):105–105.

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*.

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. 2018a. Reinforced mnemonic reader for machine reading comprehension. In *Proceedings of IJCAI*.

Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, DongshengLi, Nan Yang, and Ming Zhou. 2018b. Attention-guided answer distillation for machine reading comprehension. In *Proceedings of EMNLP*.

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

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 ACL*, 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. 2018a. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018b. Stochastic answer networks for machine reading comprehension. In *Proceedings of ACL*.

Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. 2002. Is it the right answer?: exploiting web redundancy for answer validation. In *Proceedings of ACL*.

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.

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*.

Álvaro Rodrigo, Anselmo Peñas, and Felisa Verdejo. 2008. Overview of the answer validation exercise 2008. In *Workshop of CLEF*, pages 296–313. Springer.

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

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. *JMLR*, pages 1929–1958.

Chuanqi Tan, Furu Wei, Qingyu Zhou, Nan Yang, Weifeng Lv, and Ming Zhou. 2018. I know there is no answer: Modeling answer validation for machine reading comprehension. In *Proceedings of NLPCC*, pages 85–97. Springer.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of NIPS*.

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*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *Proceedings of ICLR*.