Retrospective Reader for Machine Reading Comprehension

Zhuosheng Zhang[1,2,3], **Junjie Yang**[2,3,4], **Hai Zhao**[1,2,3,*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction
and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, Shanghai, China
[4]SJTU-ParisTech Elite Institute of Technology, Shanghai Jiao Tong University, Shanghai, China
zhangzs@sjtu.edu.cn, jj-yang@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Machine reading comprehension (MRC) is an AI challenge that requires machines to determine the correct answers to questions based on a given passage. MRC systems must not only answer questions when necessary but also tactfully abstain from answering when no answer is available according to the given passage. When unanswerable questions are involved in the MRC task, an essential verification module called verifier is especially required in addition to the encoder, though the latest practice on MRC modeling still mostly benefits from adopting well pre-trained language models as the encoder block by only focusing on the "reading". This paper devotes itself to exploring better verifier design for the MRC task with unanswerable questions. Inspired by how humans solve reading comprehension questions, we proposed a retrospective reader (Retro-Reader) that integrates two stages of reading and verification strategies: 1) sketchy reading that briefly investigates the overall interactions of passage and question, and yields an initial judgment; 2) intensive reading that verifies the answer and gives the final prediction. The proposed reader is evaluated on two benchmark MRC challenge datasets SQuAD2.0 and NewsQA, achieving new state-of-the-art results. Significance tests show that our model is significantly better than strong baselines.

1 Introduction

Be certain of what you know and be aware what you don't. That is wisdom.

Confucius (551 BC - 479 BC)

Machine reading comprehension (MRC) aims to teach machines to answer questions after comprehending given passages (**???**), which is a fundamental and long-standing goal of natural language understanding (NLU) (**?**). It has significant application scenarios such as question answering and dialog systems (**??????**). The early MRC systems (**?????**) were designed on a latent hypothesis that all questions can be answered according to the given passage (Figure 1-[a]), which is not always true for real-world cases. The recent progress on the MRC task has required that the model must be capable of distinguishing those unanswerable questions to avoid giving plausible answers (**?**). MRC task with unanswerable questions may be usually decomposed into two subtasks: 1) answerability verification and 2) reading comprehension. To determine unanswerable questions requires a deep understanding of the given text and requires more robust MRC models, making MRC much closer to real-world applications. Table 1 shows an unanswerable example from SQuAD2.0 MRC task (**?**).

Passage: *Computational complexity theory is a branch of the theory of computation in theoretical computer science that focuses on classifying computational problems according to their inherent difficulty, and relating those classes to each other. A computational problem is understood to be a task that is in principle amenable to being solved by a computer, which is equivalent to stating that the problem may be solved by mechanical application of mathematical steps, such as an algorithm.*
Question: *What cannot be solved by mechanical application of mathematical steps*?
Gold Answer: ⟨no answer⟩ **Plausible answer:** *algorithm*

Table 1: An unanswerable MRC example.

So far, a common reading system (reader) which solves MRC problem generally consists of two modules or building steps as shown in Figure 1-[a]: 1) building a robust language model (LM) as encoder; 2) designing ingenious mechanisms as decoder according to MRC task characteristics.

Pre-trained language models (PrLMs) such as BERT (**?**) and XLNet (**?**) have achieved success on various natural language processing tasks, which broadly play the role of a powerful encoder (**???**). However, it is quite time-consuming and resource-demanding to impart massive amounts of general knowledge from external corpora into a

*Corresponding author. This paper was partially supported by National Key Research and Development Program of China (No. 2017YFB0304100), Key Projects of National Natural Science Foundation of China (U1836222 and 61733011), Huawei-SJTU long term AI project, Cutting-edge Machine reading comprehension and language model.

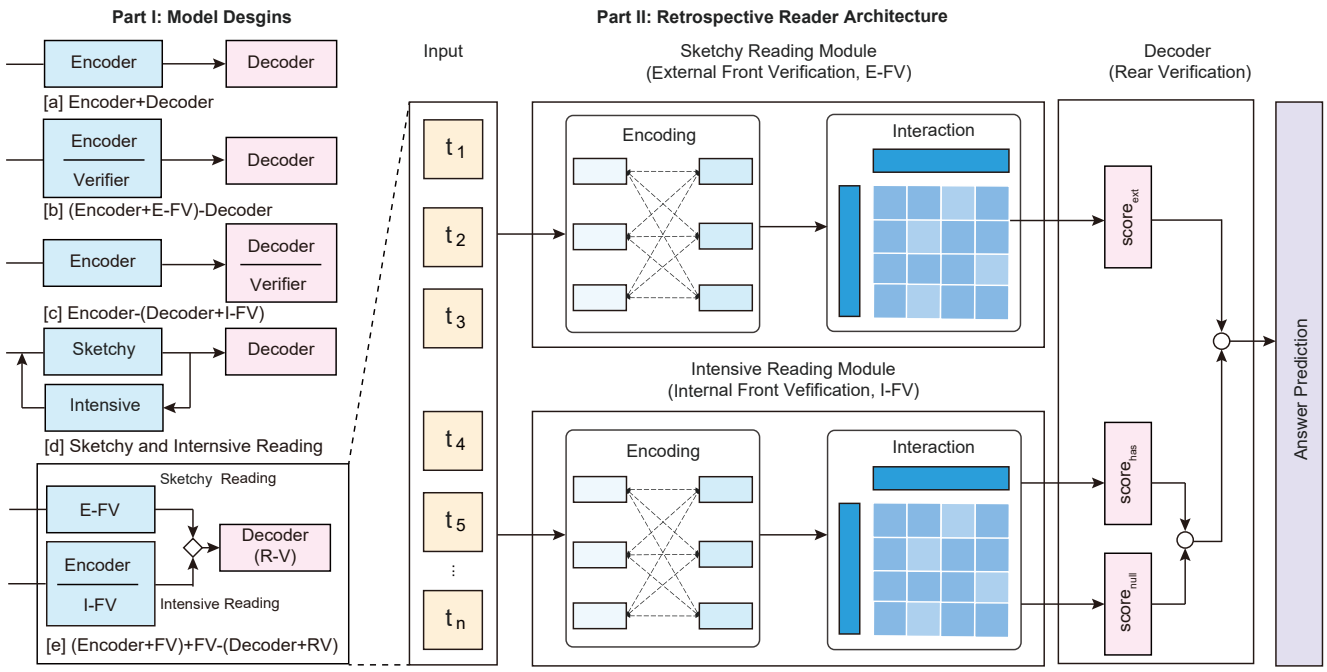

Figure 1: Reader overview. For the left part, models [a-c] summarize the instances in previous work, and model [d] is ours, with the implemented version [e]. In the names of models [a-e], "(·)" represents a module, "+" means the parallel module and "-" is the pipeline. The right part is the detailed architecture of our proposed Retro-Reader.

deep language model via pre-training.

Recently, most MRC readers keep the primary focus on the encoder side, i.e., the deep PrLMs (**???**), as readers may simply and straightforwardly benefit from a strong enough encoder. Meanwhile, little attention is paid to the decoder side[1] of MRC models (**???**), though it has been shown that better decoder or better manner of using encoder still has a significant impact on MRC performance, no matter how strong the encoder it is (**?????**).

For the concerned MRC challenge with unanswerable questions, a reader has to handle two aspects carefully: 1) give the accurate answers for answerable questions; 2) effectively distinguish the unanswerable questions, and then refuse to answer. Such requirements lead to the recent reader's design by introducing an extra verifier module or answer-verification mechanism. Most readers simply stack the verifier along with encoder and decoder parts in a pipeline or concatenation way (Figure 1-[b-c]), which is shown suboptimal for installing a verifier.

As a natural practice of how humans solve complex reading comprehension (**??**), the first step is to read through the full passage along with the question and grasp the general idea; then, people re-read the full text and verify the answer if not so sure. Inspired by such a reading and comprehension pattern, we proposed a retrospective reader (Retro-Reader, Figure 1-[d]) that integrates two stages of reading and verification strategies: 1) sketchy reading that briefly touches the relationship of passage and question, and yields an initial judgment; 2) intensive reading that verifies the answer and gives the final prediction. Our major contributions are three folds:[2]

1. We propose a new retrospective reader design which is capable of effectively performing answer verification instead of simply stacking verifier in existing readers.
2. Experiments show that our reader can yield substantial improvements over strong baselines and achieve new state-of-the-art results on benchmark MRC tasks.
3. For the first time, we apply the significance test for the concerned MRC task and show that our models are significantly better than the baselines.

2 Related Work

The research of machine reading comprehension have attracted great interest with the release of a variety of benchmark datasets (**??????**). The early trend is a variety of attention-based interactions between passage and question, including Attention Sum (**?**), Gated attention (**?**), Self-matching (**?**), Attention over Attention (**?**) and Bi-attention (**?**). Recently, PrLMs dominate the encoder design for MRC and achieve great success. These PrLMs include ELMo (**?**), GPT (**?**), BERT (**?**), XLNet (**?**), RoBERTa (**?**), ALBERT (**?**), and ELECTRA (**?**). They show strong capacity for capturing the contextualized sentence-level language representa-

[1]We define *decoder* here as the task-specific part in an MRC system, such as passage and question interaction and answer verification.

[2]Our source code is available at https://github.com/cooelf/AwesomeMRC.

tions and greatly boost the benchmark performance of current MRC. Following this line, we take PrLMs as our backbone encoder.

In the meantime, the study of the decoder mechanisms has come to a bottleneck due to the already powerful PrLM encoder. Thus this work focuses on the non-encoder part, such as passage and question attention interactions, and especially the answer verification.

To solve the MRC task with unanswerable questions is though important, only a few studies paid attention to this topic with straightforward solutions. Mostly, a treatment is to adopt an extra answer verification layer, the answer span prediction and answer verification are trained jointly with multi-task learning (Figure 1-[c]). Such an implemented verification mechanism can also be as simple as an answerable threshold setting broadly used by powerful enough PrLMs for quickly building readers (**??**). **?** (**?**) appended an empty word token to the context and added a simple classification layer to the reader. **?** (**?**) used two types of auxiliary loss, independent span loss to predict plausible answers and independent no-answer loss the to decide answerability of the question. Further, an extra verifier is adopted to decide whether the predicted answer is entailed by the input snippets (Figure 1-[b]). **?** (**?**) developed an attention-based satisfaction score to compare question embeddings with the candidate answer embeddings (Figure 1-[c]). **?** (**?**) proposed a verifier layer, which is a linear layer applied to context embedding weighted by start and end distribution over the context words representations concatenated to `[CLS]` token representation for BERT (Figure 1-[c]).

Different from these existing studies which stack the verifier module in a simple way or just jointly learn answer location and non-answer losses, our Retro-Reader adopts a two-stage humanoid design (**??**) based on a comprehensive survey over existing answer verification solutions.

3 Our Proposed Model

We focus on the span-based MRC task, which can be described as a triplet $\langle P, Q, A\rangle$, where P is a passage, and Q is a query over P, in which a span is a right answer A. Our system is supposed to not only predict the start and end positions in the passage P and extract span as answer A but also return a null string when the question is unanswerable.

Our retrospective reader is composed of two parallel modules: a *sketchy reading module* and an *intensive reading module* to conduct a two-stage reading process. The intuition behind the design is that the sketchy reading makes a coarse judgment (external front verification) about the answerability, whether the question is answerable; and then the intensive reading jointly predicts the candidate answers and combines its answerability confidence (internal front verification) with the sketchy judgment score to yield the final answer (rear verification). [3]

[3]Intuitively, our model is supposed to be designed as shown in Figure 1-[d]. In the implementation, we find that modeling the entire reading process into two parallel modules is both simple and practicable with basically the same performance, which results in a parallel reading module design at last as shown in Figure 1-[e].

3.1 Sketchy Reading Module

Embedding We concatenate question and passage texts as the input, which is firstly represented as embedding vectors to feed an encoder (i.e., a PrLM). In detail, the input texts are first tokenized to word pieces (subword tokens). Let $T = \{t_1, \ldots, t_n\}$ denote a sequence of subword tokens of length n. For each token, the input embedding is the sum of its token embedding, position embedding, and token-type embedding. Let $X = \{x_1, \ldots, x_n\}$ be the outputs of the encoder, which are embedding features of encoding sentence tokens of length n. The input embeddings are then fed to the interaction layer to obtain the contextual representations.

Interaction Following **?**, the encoded sequence X is processed to a multi-layer Transformer (**?**) for learning contextual representations. For the following part, we use $\mathbf{H} = \{h_1, \ldots, h_n\}$ to denote the last-layer hidden states of the input sequence.

External Front Verification After reading, the sketchy reader will make a preliminary judgment, whether the question is answerable given the context. We implement this reader as an external front verifier (E-FV) to identify unanswerable questions. The pooled first token (the special symbol, `[CLS]`) representation $h_1 \in \mathbf{H}$, as the overall representation of the sequence, is passed to a fully connection layer to get classification logits $\hat{y}_i$ composed of answerable ($logit_{ans}$) and unanswerable ($logit_{na}$) elements. We use cross entropy as training objective:

$$\mathbb{L}^{ans} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right] \quad (1)$$

where $\hat{y}_i \propto \text{SoftMax}(\text{FFN}(h_1))$ denotes the prediction and y_i is the target indicating whethter the question is answerbale or not. N is the number of examples. We calculate the difference as the external verification score: $score_{ext} = logit_{na} - logit_{ans}$, which is used in the later rear verification as effective indication factor.

3.2 Intensive Reading Module

The objective of the intensive reader is to verify the answerability, produce candidate answer spans, and then give the final answer prediction. It employs the same encoding and interaction procedure as the sketchy reader, to obtain the representation **H**. In previous studies (**???**), **H** is directly fed to a linear layer to yield the prediction.

Question-aware Matching Inspired by previous success of explicit attention matching between passage and question (**????**), we are interested in whether the advance still holds based on the strong PrLMs. Here we investigate two alternative question-aware matching mechanisms as an extra layer. Note that this part is only used for ablation in Table 7 as a reference for interested readers. We do not use any extra matching part in our submission on test evaluations (e.g., in Tables 2-3) for the sake of simplicity as our major focus is the verification.

To obtain the representation of each passage and question, we split the last-layer hidden state $\mathbf{H}$ into $\mathbf{H}^Q$ and $\mathbf{H}^P$ as the representations of the question and passage, according to its position information. Both of the sequences are padded to the maximum length in a minibatch. Then, we investigate two potential question-aware matching mechanisms, 1) Transformer-style multi-head *cross attention* (CA) and 2) traditional *matching attention* (MA).

• **Cross Attention** We feed the $\mathbf{H}^Q$ and $\mathbf{H}$ to a revised one-layer multi-head attention layer inspired by **?** (**?**). Since the setting is $\mathbf{Q} = \mathbf{K} = \mathbf{V}$ in multi-head self attention,[4] which are all derived from the input sequence, we replace the input to $\mathbf{Q}$ with $\mathbf{H}$, and both of $\mathbf{K}$ and $\mathbf{V}$ with $\mathbf{H}^Q$ to obtain the question-aware context representation $\mathbf{H}'$.

• **Matching Attention** Another alternative is to feed $\mathbf{H}^Q$ and $\mathbf{H}$ to a traditional matching attention layer (**?**), by taking the question presentation $\mathbf{H}^Q$ as the attention to the representation $\mathbf{H}$:

$$\begin{aligned} \mathbf{M} &= \text{SoftMax}(\mathbf{H}(\mathbf{W}\mathbf{H}^Q + \mathbf{b} \otimes \mathbf{e})^\mathsf{T}), \\ \mathbf{H}' &= \mathbf{M}\mathbf{H}^Q, \end{aligned} \tag{2}$$

where $\mathbf{W}$ and $\mathbf{b}$ are learnable parameters. $\mathbf{e}$ is a all-ones vector and used to repeat the bias vector into the matrix. $\mathbf{M}$ denotes the weights assigned to the different hidden states in the concerned two sequences. $\mathbf{H}'$ is the weighted sum of all the hidden states and it represents how the vectors in $\mathbf{H}$ can be aligned to each hidden state in $\mathbf{H}^Q$. Finally, the representation $\mathbf{H}'$ is used for the later predictions. If we do not use the above matching like the original use in BERT models, then $\mathbf{H}' = \mathbf{H}$ for the following part.

Span Prediction The aim of span-based MRC is to find a span in the passage as answer, thus we employ a linear layer with SoftMax operation and feed $\mathbf{H}'$ as the input to obtain the start and end probabilities, s and e:

$$s, e \propto \text{SoftMax}(\text{FFN}(\mathbf{H}')). \tag{3}$$

The training objective of answer span prediction is defined as cross entropy loss for the start and end predictions,

$$\mathbb{L}^{span} = -\frac{1}{N}\sum_{i}^{N}[\log(p^s_{y^s_i}) + \log(p^e_{y^e_i})] \tag{4}$$

where y^s_i and y^e_i are respectively ground-truth start and end positions of example i. N is the number of examples.

Internal Front Verification We adopted an internal front verifier (I-FV) such that the intensive reader can identify unanswerable questions as well. In general, a verifier's function can be implemented as a cross-entropy loss (I-FV-CE), binary cross-entropy loss (I-FV-BE), or regression-style mean square error loss (I-FV-MSE). The pooled representation $h_1^{'} \in \mathbf{H}'$, is passed to a fully connected layer to get the classification logits or regression score. Let $\hat{y}_i \propto$ *Linear*(h_1) denote the prediction and y_i is the answerability target, the three alternative loss functions are as defined as follows:

(1) We use cross entropy as loss function for the classification verification:

$$\begin{aligned} \bar{y}_{i,k} &= \text{SoftMax}(\text{FFN}(h_1^{'})), \\ \mathbb{L}^{ans} &= -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\left[y_{i,k}\log\bar{y}_{i,k}\right], \end{aligned} \tag{5}$$

where K means the number of classes ($K = 2$ in this work). N is the number of examples.

(2) For binary cross entropy as loss function for the classification verification:

$$\begin{aligned} \bar{y}_i &= \text{Sigmoid}(\text{FFN}(h_1)), \\ \mathbb{L}^{ans} &= -\frac{1}{N}\sum_{i=1}^{N}\left[y_i\log\bar{y}_i + (1-y_i)\log(1-\bar{y}_i)\right]. \end{aligned} \tag{6}$$

(3) For the regression verification, mean square error is adopted as its loss function:

$$\bar{y}_i = \text{FFN}(h_1^{'}), \tag{7}$$

$$\mathbb{L}^{ans} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2. \tag{8}$$

During training, the joint loss function for FV is the weighted sum of the span loss and verification loss:

$$\mathbb{L} = \alpha_1\mathbb{L}^{span} + \alpha_2\mathbb{L}^{ans}, \tag{9}$$

where α_1 and α_2 are weights.

Threshold-based Answerable Verification Following previous studies (**????**), we adopt threshold based answerable verification (TAV), which is a heuristic strategy to decide whether a question is answerable according to the predicted answer start and end logits finally. Given the output start and end probabilities s and e, and the verification probability v, we calculate the has-answer score $score_{has}$ and the no-answer score $score_{null}$:

$$\begin{aligned} score_{has} &= \max(s_k + e_l), 1 < k \le l \le n, \\ score_{null} &= s_1 + e_1, \end{aligned} \tag{10}$$

We obtain a difference score between $score_{null}$ and the $score_{has}$ as the final *no-answer* score: $score_{diff} = score_{null} - score_{has}$. An answerable threshold δ is set and determined according to the development set. The model predicts the answer span that gives the *has-answer* score if the final score is above the threshold δ, and null string otherwise.

TAV is used in all our models as the last step for the answerability decision. We denote it in our baselines with (+TAV) as default in Table 2-3, and omit the notation for simplicity in analysis part to avoid misunderstanding to keep on the specific ablations.

[4] In this work, $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ correspond to the items $Q^{l+1}_m x^l_i$, $K^{l+1}_m x^l_j$, $V^{l+1}_m x^l_j$, respectively.

3.3 Rear Verification

Rear verification (RV) is the combination of predicted probabilities of E-FV and I-FV, which is an aggregated verification for final answer.

$$v = \beta_1 score_{diff} + \beta_2 score_{ext}, \quad (11)$$

where β_1 and β_2 are weights. Our model predicts the answer span if $v > \delta$, and null string otherwise.

4 Experiments

4.1 Setup

We use the available PrLMs as encoder to build baseline MRC models: BERT (**?**), ALBERT (**?**), and ELECTRA (**?**). Our implementations of BERT and ALBERT are based on the public Pytorch implementation from Transformers.[5] ELECTRA is based on the Tensorflow release.[6] We use the pre-trained LM weights in the encoder module of our reader, using all the official hyperparameters.[7] For the fine-tuning in our tasks, we set the initial learning rate in {2e-5, 3e-5} with a warm-up rate of 0.1, and L2 weight decay of 0.01. The batch size is selected in {32, 48}. The maximum number of epochs is set in 2 for all the experiments. Texts are tokenized using wordpieces (**?**), with a maximum length of 512. Hyper-parameters were selected using the dev set. The manual weights are $\alpha_1 = \alpha_2 = \beta_1 = \beta_2 = 0.5$ in this work.

For answer verification, we follow the same setting according to the corresponding literatures (**???**), which simply adopts the answerable threshold method described in §3.2. In the following part, +TAV (for all the baseline modes) denotes the baseline verification for easy reference, which is equivalent to the baseline implementations in public literatures (**???**).

4.2 Benchmark Datasets

Our proposed reader is evaluated in two benchmark MRC challenges.

SQuAD2.0 As a widely used MRC benchmark dataset, SQuAD2.0 (**?**) combines the 100,000 questions in SQuAD1.1 (**?**) with over 50,000 new, unanswerable questions that are written adversarially by crowdworkers to look similar to answerable ones. The training dataset contains $87k$ answerable and $43k$ unanswerable questions.

NewsQA NewsQA (**?**) is a question-answering dataset with 100,000 human-generated question-answer pairs. The questions and answers are based on a set of over 10,000 news articles from CNN supplied by crowdworkers. The paragraphs are about 600 words on average, which tend to be longer than SQuAD2.0. The training dataset has $20k$ unanswerable questions among $97k$ questions.

[5]https://github.com/huggingface/transformers.

[6]https://github.com/google-research/electra.

[7]BERT$_{\texttt{large}}$; ALBERT$_{\texttt{xxlarge}}$; ELECTRA$_{\texttt{large}}$.

Model	Dev		Test	
	EM	F1	EM	F1
Human	-	-	86.8	89.5
BERT (**?**)	-	-	82.1	84.8
NeurQuRI (**?**)	80.0	83.1	81.3	84.3
XLNet (**?**)	86.1	88.8	86.4	89.1
RoBERTa (**?**)	86.5	89.4	86.8	89.8
SG-Net (**?**)	-	-	87.2	90.1
ALBERT (**?**)	87.4	90.2	88.1	90.9
ELECTRA (**?**)	88.0	90.6	88.7	91.4
Our implementation				
ALBERT (+TAV)	87.0	90.2	-	-
Retro-Reader on ALBERT	87.8	90.9	88.1	91.4
ELECTRA (+TAV)	88.0	90.6	-	-
Retro-Reader on ELECTRA	**88.8**	**91.3**	**89.6**	**92.1**

Table 2: The results (%) for SQuAD2.0 dataset. The results are from the official leaderboard. TAV: threshold-based answerable verification (§3.2).

4.3 Evaluation

Metrics Two official metrics are used to evaluate the model performance: Exact Match (EM) and a softer metric F1 score, which measures the average overlap between the prediction and ground truth answer at the token level.

Significance Test With the rapid development of deep MRC models, the dominant models have achieved very high results (e.g., over 90% F1 scores on SQuAD2.0), and further advance has been very marginal. Thus a significance test would be beneficial for measuring the difference in model performance.

For selecting evaluation metrics for the significance test, since answers vary in length, using the F1 score would have a bias when comparing models, i.e., if one model fails on one severe example though works well on the others. Therefore, we use the tougher metric EM as the goodness measure. If the EM is equal to 1, the prediction is regarded as right and vice versa. Then the test is modeled as a binary classification problem to estimate the answer of the model is exactly right (EM=1) or wrong (EM=0) for each question. According to our task setting, we used McNemars test (**?**) to test the statistical significance of our results. This test is designed for paired nominal observations, and it is appropriate for binary classification tasks (**?**).

The p-value is defined as the probability, under the null hypothesis, of obtaining a result equal to or more extreme than what was observed. The smaller the p-value, the higher the significance. A commonly used level of reliability of the result is 95%, written as $p = 0.05$.

4.4 Results

Tables 2-3 compare the leading single models on SQuAD2.0 and NewsQA. Retro-Reader on ALBERT and Retro-Reader on ELECTRA denote our final models (i.e., our submissions

Model	Dev		Test	
	EM	F1	EM	F1
Existing Systems				
BARB (**?**)	36.1	49.6	34.1	48.2
mLSTM (**?**)	34.4	49.6	34.9	50.0
BiDAF (**?**)	-	-	37.1	52.3
R2-BiLSTM (**?**)	-	-	43.7	56.7
AMANDA (**?**)	48.4	63.3	48.4	63.7
DECAPROP (**?**)	52.5	65.7	53.1	66.3
BERT (**?**)	-	-	46.5	56.7
NeurQuRI (**?**)	-	-	48.2	59.5
Our implementation				
ALBERT (+TAV)	57.1	67.5	55.3	65.9
Retro-Reader on ALBERT	**58.5**	**68.6**	**55.9**	**66.8**
ELECTRA (+TAV)	56.3	66.5	54.0	64.5
Retro-Reader on ELECTRA	56.9	67.0	54.7	65.7

Table 3: Results (%) for NewsQA dataset. The results except ours are from **?** (**?**) and **?** (**?**). TAV: threshold based answerable verification (§3.2).

to SQuAD2.0 online evaluation), which are respectively the ALBERT and ELECTRA based retrospective reader composed of both sketchy and intensive reading modules without question-aware matching for simplicity. According to the results, we make the following observations:

1) Our implemented ALBERT and ELECTRA baselines show the similar EM and F1 scores with the original numbers reported in the corresponding papers (**??**), ensuring that the proposed method can be fairly evaluated over the public strong baseline systems.

2) In terms of powerful enough PrLMs like ALBERT and ELECTRA, our Retro-Reader not only significantly outperforms the baselines with p-value < 0.01,[8] but also achieves new state-of-the-art on the SQuAD2.0 challenge.[9]

3) The results on NewsQA further verifies the general effectiveness of our proposed Retro-Reader. Our method shows consistent improvements over the baselines and achieves new state-of-the-art results.

5 Ablations

5.1 Evaluation on Answer Verification

Table 4 presents the results with different answer verification methods. We observe that either of the front verifiers boosts the baselines, and integrating both as rear verification works the best. Note that we show the HasAns and NoAns only for completeness. Since the final predictions are based on the threshold search of answerability scores (§3.2), there exists a tradeoff between the HasAns and NoAns accuracies. We

[8]Besides the McNemar's test, we also used paired t-test for significance test, with consistent findings.

[9]When our models were submitted (*Jan 10th 2020* and *Apr 05, 2020* for ALBERT- and ELECTRA-based models, respectively), our Retro-Reader achieved the first place on the SQuAD2.0 Leaderboard (https://rajpurkar.github.io/SQuAD-explorer/) for both single and ensemble models.

Model	All		HasAns		NoAns	
	EM	F1	EM	F1	EM	F1
BERT	78.8	81.7	74.6	80.3	83.0	83.0
+ E-FV	79.1	82.1	73.4	79.4	84.8	84.8
+ I-FV-CE	78.6	82.0	73.3	79.5	84.5	84.5
+ I-FV-BE	78.8	81.8	72.6	78.7	85.0	85.0
+ I-FV-MSE	78.5	81.7	73.0	78.6	84.8	84.8
+ RV	79.6	82.5	73.7	79.6	85.2	85.2
ALBERT	87.0	90.2	82.6	89.0	91.4	91.4
+ E-FV	87.4	90.6	82.4	88.7	92.4	92.4
+ I-FV-CE	87.2	90.3	81.7	87.9	92.7	92.7
+ I-FV-BE	87.2	90.2	82.2	88.4	92.1	92.1
+ I-FV-MSE	87.3	90.4	82.4	88.5	92.3	92.3
+ RV	87.8	90.9	83.1	89.4	92.4	92.4

Table 4: Results (%) with different answer verification methods on the SQuAD2.0 dev set. *CE*, *BE*, and *MSE* are short for the two classification and one regression loss functions defined in §3.2.

Method	Prec.	Rec.	F1	Acc.
ALBERT	91.70	93.42	92.55	86.14
Retro-Reader on ALBERT	**94.30**	92.38	93.33	87.49
ELECTRA	92.71	92.58	92.64	86.30
Retro-Reader on ELECTRA	93.27	**93.51**	**93.39**	**87.60**

Table 5: Performance on the unanswerable questions from SQuAD2.0 dev set.

Method	EM	F1
ALBERT	87.0	90.2
Two-model Ensemble	87.6	90.6
Retro-Reader	87.8	90.9

Table 6: Comparisons with Equivalent Parameters on the dev set of SQuAD2.0.

Method	SQuAD2.0		NewsQA	
	EM	F1	EM	F1
BERT	78.8	81.7	51.8	62.5
+ CA	78.8	81.7	52.1	62.7
+ MA	78.3	81.4	52.4	62.6
ALBERT	87.0	90.2	57.1	67.5
+ CA	87.3	90.3	56.0	66.3
+ MA	86.8	90.0	55.8	66.1

Table 7: Results (%) with matching interaction methods on the dev sets of SQuAD2.0 and NewsQA.

see that the final RV that combines E-FV and I-FV shows the best performance, which we select as our final implementation for testing.

We further conduct the experiments on our model performance of the 5,945 unanswerable questions from the SQuAD 2.0 dev set. Results in Table 5 show that our method improves the performance on unanswerable questions by a large margin, especially in the primary F1 and accuracy metrics.

5.2 Comparisons with Equivalent Parameters

When using sketchy reading module for external verification, we have two parallel modules that have independent parameters. For comparisons with equivalent parameters, we add an ensemble of two baseline models, to see if the advance is purely from the increase of parameters. Table 6 shows the results. We see that our model can still outperform two ensembled models. Although the two modules share the same design of the Transformer encoder, the training objectives (e.g., loss functions) are quite different, one for answer span prediction, the other for answerable decision. The results indicate that our two-stage reading modules would be more effective for learning diverse aspects (verification and span prediction) for solving MRC tasks with different training objectives. From the two modules, we can easily find the effectiveness of either the span prediction or answer verification, to improve the modules correspondingly. We believe this design would be quite useful for real-world applications.

5.3 Evaluation on Matching Interactions

Table 7 shows the results with different interaction methods described in §3.2. We see that merely adding extra layers could not bring noticeable improvement, which indicates that simply adding more layers and parameters would not substantially benefit the model performance. The results verified the PrLMs' strong ability to capture the relationships between passage and question after processing the paired input by deep self-attention layers. In contrast, answer verification could still give consistent and substantial advance.

Passage:
Southern California consists of a heavily developed urban environment, home to some of the largest urban areas in the state, along with vast areas that have been left undeveloped. It is the third most populated megalopolis in the United States, after the Great Lakes Megalopolis and the Northeastern megalopolis. Much of southern California is famous for its large, spread-out, suburban communities and use of automobiles and highways. The dominant areas are Los Angeles, Orange County, San Diego, and Riverside-San Bernardino, each of which are the centers of their respective metropolitan areas...

Question:
What are the second and third most populated megalopolis after Southern California?

Answer:
Gold: ⟨no answer⟩
ALBERT (+TAV): Great Lakes Megalopolis and the Northeastern megalopolis.
Retro-Reader over ALBERT: ⟨no answer⟩
$score_{has} = 0.03, score_{na} = 1.73, \delta = -0.98$

Table 8: Answer prediction examples from the ALBERT baseline and Retro-Reader.

5.4 Comparison of Predictions

To have an intuitive observation of the predictions of Retro-Reader, we give a prediction example on SQuAD2.0 from baseline and Retro-Reader in Table 8, which shows that our method works better at judging whether the question is answerable on a given passage and gets rid of the plausible answer.

6 Conclusion

As machine reading comprehension tasks with unanswerable questions stress the importance of answer verification in MRC modeling, this paper devotes itself to better verifier-oriented MRC task-specific design and implementation for the first time. Inspired by human reading comprehension experience, we proposed a retrospective reader that integrates both sketchy and intensive reading. With the latest PrLM as encoder backbone and baseline, the proposed reader is evaluated on two benchmark MRC challenge datasets SQuAD2.0 and NewsQA, achieving new state-of-the-art results and outperforming strong baseline models in terms of newly introduced statistical significance, which shows the choice of verification mechanisms has a significant impact for MRC performance and verifier is an indispensable reader component even for powerful enough PrLMs used as the encoder. In the future, we will investigate more decoder-side problem-solving techniques to cooperate with the strong encoders for more advanced MRC.

Vitae maxime ullam harum, corporis amet quod