Multi-Style Generative Reading Comprehension

Kyosuke Nishida[1], **Itsumi Saito**[1], **Kosuke Nishida**[1],
Kazutoshi Shinoda[2]*, **Atsushi Otsuka**[1], **Hisako Asano**[1], **Junji Tomita**[1]
[1]NTT Media Intelligence Laboratory, NTT Corporation [2]The University of Tokyo
kyosuke.nishida@acm.org

Abstract

This study tackles generative reading comprehension (RC), which consists of answering questions based on textual evidence and natural language generation (NLG). We propose a multi-style abstractive summarization model for question answering, called Masque. The proposed model has two key characteristics. First, unlike most studies on RC that have focused on extracting an answer span from the provided passages, our model instead focuses on generating a summary from the question and multiple passages. This serves to cover various answer styles required for real-world applications. Second, whereas previous studies built a specific model for each answer style because of the difficulty of acquiring one general model, our approach learns multi-style answers within a model to improve the NLG capability for all styles involved. This also enables our model to give an answer in the target style. Experiments show that our model achieves state-of-the-art performance on the Q&A task and the Q&A + NLG task of MS MARCO 2.1 and the summary task of NarrativeQA. We observe that the transfer of the style-independent NLG capability to the target style is the key to its success.

1 Introduction

Question answering has been a long-standing research problem. Recently, reading comprehension (RC), a challenge to answer a question given textual evidence provided in a document set, has received much attention. Current mainstream studies have treated RC as a process of extracting an answer span from one passage (Rajpurkar et al., 2016, 2018) or multiple passages (Joshi et al., 2017; Yang et al., 2018), which is usually done by predicting the start and end positions of the answer (Yu et al., 2018; Devlin et al., 2018).

*Work done during an internship at NTT.

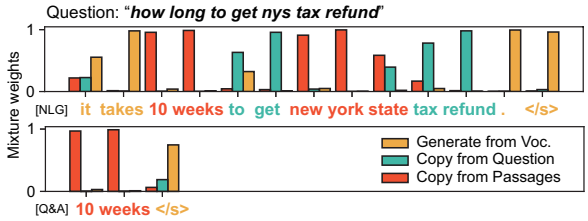

Figure 1: Visualization of how our model generates an answer on MS MARCO. Given an answer style (**top**: NLG, **bottom**: Q&A), the model controls the mixture of three distributions for generating words from a vocabulary and copying words from the question and multiple passages at each decoding step.

The demand for answering questions in natural language is increasing rapidly, and this has led to the development of smart devices such as Alexa. In comparison with answer span extraction, however, the natural language generation (NLG) capability for RC has been less studied. While datasets such as MS MARCO (Bajaj et al., 2018) and NarrativeQA (Kociský et al., 2018) have been proposed for providing abstractive answers, the state-of-the-art methods for these datasets are based on answer span extraction (Wu et al., 2018; Hu et al., 2018). Generative models suffer from a dearth of training data to cover open-domain questions.

Moreover, to satisfy various information needs, intelligent agents should be capable of answering *one* question in *multiple* styles, such as well-formed sentences, which make sense even without the context of the question and passages, and concise phrases. These capabilities complement each other, but previous studies cannot use and control different styles within a model.

In this study, we propose *Masque*, a generative model for multi-passage RC. It achieves state-of-the-art performance on the Q&A task and the Q&A + NLG task of MS MARCO 2.1 and the summary task of NarrativeQA. The main contri-

butions of this study are as follows.

Multi-source abstractive summarization. We introduce the pointer-generator mechanism (See et al., 2017) for generating an abstractive answer from the question and multiple passages, which covers various answer styles. We extend the mechanism to a Transformer (Vaswani et al., 2017) based one that allows words to be generated from a vocabulary and to be copied from the question and passages.

Multi-style learning for style control and transfer. We introduce multi-style learning that enables our model to control answer styles and improves RC for all styles involved. We also extend the pointer-generator to a conditional decoder by introducing an artificial token corresponding to each style, as in (Johnson et al., 2017). For each decoding step, it controls the mixture weights over three distributions with the given style (Figure 1).

2 Problem Formulation

This paper considers the following task:

PROBLEM 1. Given a question with J words $x^q = \{x_1^q, \ldots, x_J^q\}$, a set of K passages, where the k-th passage is composed of L words $x^{p_k} = \{x_1^{p_k}, \ldots, x_L^{p_k}\}$, and an answer style label s, an RC model outputs an answer $y = \{y_1, \ldots, y_T\}$ conditioned on the style.

In short, given a 3-tuple $(x^q, \{x^{p_k}\}, s)$, the system predicts $P(y)$. The training data is a set of 6-tuples: $(x^q, \{x^{p_k}\}, s, y, a, \{r^{p_k}\})$, where a and $\{r^{p_k}\}$ are optional. Here, a is 1 if the question is answerable with the provided passages and 0 otherwise, and r^{p_k} is 1 if the k-th passage is required to formulate the answer and 0 otherwise.

3 Proposed Model

We propose a Multi-style Abstractive Summarization model for QUEstion answering, called *Masque*. Masque directly models the conditional probability $p(y|x^q, \{x^{p_k}\}, s)$. As shown in Figure 2, it consists of the following modules.

1. The **question-passages reader** (§3.1) models interactions between the question and passages.

2. The **passage ranker** (§3.2) finds passages relevant to the question.

3. The **answer possibility classifier** (§3.3) identifies answerable questions.

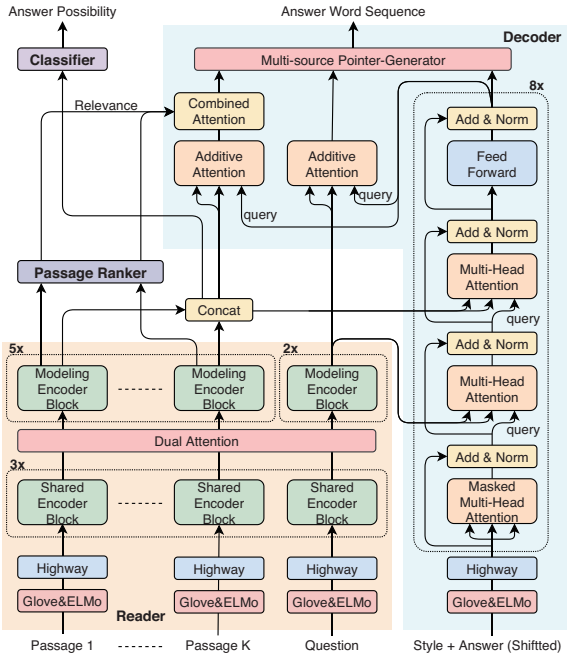

Figure 2: Masque model architecture.

4. The **answer sentence decoder** (§3.4) outputs an answer sentence conditioned on the target style.

Our model is based on multi-source abstractive summarization: the answer that it generates can be viewed as a summary from the question and passages. The model also learns multi-style answers together. With these two characteristics, we aim to acquire the style-independent NLG ability and transfer it to the target style. In addition, to improve natural language understanding in the reader module, our model considers RC, passage ranking, and answer possibility classification together as multi-task learning.

3.1 Question-Passages Reader

The reader module is shared among multiple answer styles and the three task-specific modules.

3.1.1 Word Embedding Layer

Let x^q and x^{p_k} represent one-hot vectors (of size V) for words in the question and the k-th passage. First, this layer projects each of the vectors to a d_{word}-dimensional vector with a pre-trained weight matrix $W^e \in \mathbb{R}^{d_{\text{word}} \times V}$ such as GloVe (Pennington et al., 2014). Next, it uses contextualized word representations via ELMo (Peters et al., 2018), which allows our model to use morphological clues to form robust representations for out-of-vocabulary words unseen in training. Then, the concatenation of the word and con-

textualized vectors is passed to a two-layer highway network (Srivastava et al., 2015) to fuse the two types of embeddings, as in (Seo et al., 2017). The highway network is shared by the question and passages.

3.1.2 Shared Encoder Layer

This layer uses a stack of Transformer blocks, which are shared by the question and passages, on top of the embeddings provided by the word embedding layer. The input of the first block is immediately mapped to a d-dimensional vector by a linear transformation. The outputs of this layer are $E^{p_k} \in \mathbb{R}^{d \times L}$ for each k-th passage, and $E^q \in \mathbb{R}^{d \times J}$ for the question.

Transformer encoder block. The block consists of two sub-layers: a self-attention layer and a position-wise feed-forward network. For the self-attention layer, we adopt the multi-head attention mechanism (Vaswani et al., 2017). Following GPT (Radford et al., 2018), the feed-forward network consists of two linear transformations with a GELU (Hendrycks and Gimpel, 2016) activation function in between. Each sub-layer is placed inside a residual block (He et al., 2016). For an input x and a given sub-layer function f, the output is $\mathrm{LN}(f(x) + x)$, where LN indicates the layer normalization (Ba et al., 2016). To facilitate these residual connections, all sub-layers produce a sequence of d-dimensional vectors. Note that our model does not use any position embeddings in this block because ELMo gives the positional information of the words in each sequence.

3.1.3 Dual Attention Layer

This layer uses a dual attention mechanism to fuse information from the question to the passages as well as from the passages to the question.

It first computes a similarity matrix $U^{p_k} \in \mathbb{R}^{L \times J}$ between the question and the k-th passage, as done in (Seo et al., 2017), where

$$U^{p_k}_{lj} = {w^a}^\top [E^{p_k}_l; E^q_j; E^{p_k}_l \odot E^q_j]$$

indicates the similarity between the l-th word of the k-th passage and the j-th question word. The $w^a \in \mathbb{R}^{3d}$ are learnable parameters. The $\odot$ operator denotes the Hadamard product, and the $[;]$ operator denotes vector concatenation across the rows. Next, the layer obtains the row and column normalized similarity matrices $A^{p_k} = \mathrm{softmax}_j({U^{p_k}}^\top)$ and $B^{p_k} = \mathrm{softmax}_l(U^{p_k})$. It then uses DCN (Xiong et al., 2017) to obtain dual attention representations, $G^{q \to p_k} \in \mathbb{R}^{5d \times L}$ and $G^{p \to q} \in \mathbb{R}^{5d \times J}$:

$$G^{q \to p_k} = [E^{p_k}; \bar{A}^{p_k}; \bar{\bar{A}}^{p_k}; E^{p_k} \odot \bar{A}^{p_k}; E^{p_k} \odot \bar{\bar{A}}^{p_k}]$$
$$G^{p \to q} = [E^q; \bar{B}; \bar{\bar{B}}; E^q \odot \bar{B}; E^q \odot \bar{\bar{B}}].$$

Here, $\bar{A}^{p_k} = E^q A^{p_k}$, $\bar{B}^{p_k} = E^{p_k} B^{p_k}$, $\bar{\bar{A}}^{p_k} = \bar{B}^{p_k} A^{p_k}$, $\bar{\bar{B}}^{p_k} = \bar{A}^{p_k} B^{p_k}$, $\bar{B} = \max_k(\bar{B}^{p_k})$, and $\bar{\bar{B}} = \max_k(\bar{\bar{B}}^{p_k})$.

3.1.4 Modeling Encoder Layer

This layer uses a stack of the Transformer encoder blocks for question representations and obtains $M^q \in \mathbb{R}^{d \times J}$ from $G^{p \to q}$. It also uses another stack for passage representations and obtains $M^{p_k} \in \mathbb{R}^{d \times L}$ from $G^{q \to p_k}$ for each k-th passage. The outputs of this layer, M^q and $\{M^{p_k}\}$, are passed on to the answer sentence decoder; the $\{M^{p_k}\}$ are also passed on to the passage ranker and the answer possibility classifier.

3.2 Passage Ranker

The ranker maps the output of the modeling layer, $\{M^{p_k}\}$, to the relevance score of each passage. It takes the output for the first word, $M^{p_k}_1$, which corresponds to the beginning-of-sentence token, to obtain the aggregate representation of each passage sequence. Given $w^r \in \mathbb{R}^d$ as learnable parameters, it calculates the relevance of each k-th passage to the question as

$$\beta^{p_k} = \mathrm{sigmoid}({w^r}^\top M^{p_k}_1).$$

3.3 Answer Possibility Classifier

The classifier maps the output of the modeling layer to a probability for the answer possibility. It also takes the output for the first word, $M^{p_k}_1$, for all passages and concatenates them. Given $w^c \in \mathbb{R}^{Kd}$ as learnable parameters, it calculates the answer possibility for the question as

$$P(a) = \mathrm{sigmoid}({w^c}^\top [M^{p_1}_1; \ldots; M^{p_K}_1]).$$

3.4 Answer Sentence Decoder

Given the outputs provided by the reader module, the decoder generates a sequence of answer words one element at a time. It is autoregressive (Graves, 2013), consuming the previously generated words as additional input at each decoding step.

3.4.1 Word Embedding Layer

Let y represent one-hot vectors of the words in the answer. This layer has the same components as the word embedding layer of the reader module, except that it uses a unidirectional ELMo to ensure that the predictions for position t depend only on the known outputs at positions previous to t.

Artificial tokens. To be able to use multiple answer styles within a single system, our model introduces an artificial token corresponding to the style at the beginning of the answer (y_1), as done in (Johnson et al., 2017; Takeno et al., 2017). At test time, the user can specify the first token to control the style. This modification does not require any changes to the model architecture. Note that introducing the token at the decoder prevents the reader module from depending on the answer style.

3.4.2 Attentional Decoder Layer

This layer uses a stack of Transformer decoder blocks on top of the embeddings provided by the word embedding layer. The input is immediately mapped to a d-dimensional vector by a linear transformation, and the output is a sequence of d-dimensional vectors: $\{s_1, \ldots, s_T\}$.

Transformer decoder block. In addition to the encoder block, this block consists of the second and third sub-layers after the self-attention block and before the feed-forward network, as shown in Figure 2. As in (Vaswani et al., 2017), the self-attention sub-layer uses a sub-sequent mask to prevent positions from attending to subsequent positions. The second and third sub-layers perform the multi-head attention over M^q and $M^{p_{\text{all}}}$, respectively. The $M^{p_{\text{all}}}$ is the concatenated outputs of the encoder stack for the passages,

$$M^{p_{\text{all}}} = [M^{p_1}, \ldots, M^{p_K}] \in \mathbb{R}^{d \times KL}.$$

Here, the $[,]$ operator denotes vector concatenation across the columns. This attention for the concatenated passages produces attention weights that are comparable between passages.

3.4.3 Multi-source Pointer-Generator

Our extended mechanism allows both words to be generated from a vocabulary and words to be copied from both the question and multiple passages (Figure 3). We expect that the capability of copying words will be shared among answer styles.

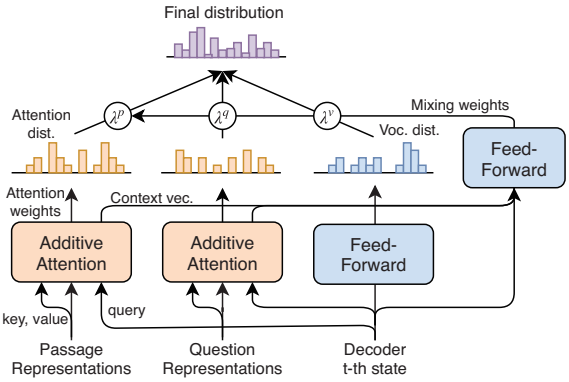

Figure 3: Multi-source pointer-generator mechanism. For each decoding step t, mixture weights $\lambda^v, \lambda^q, \lambda^p$ for the probability of generating words from the vocabulary and copying words from the question and the passages are calculated. The three distributions are weighted and summed to obtain the final distribution.

Extended vocabulary distribution. Let the extended vocabulary, V_{ext}, be the union of the common words (a small subset of the full vocabulary, V, defined by the input-side word embedding matrix) and all words appearing in the input question and passages. P^v then denotes the probability distribution of the t-th answer word, y_t, over the extended vocabulary. It is defined as:

$$P^v(y_t) = \text{softmax}({W^2}^\top (W^1 s_t + b^1)),$$

where the output embedding $W^2 \in \mathbb{R}^{d_{\text{word}} \times V_{\text{ext}}}$ is tied with the corresponding part of the input embedding (Inan et al., 2017), and $W^1 \in \mathbb{R}^{d_{\text{word}} \times d}$ and $b^1 \in \mathbb{R}^{d_{\text{word}}}$ are learnable parameters. $P^v(y_t)$ is zero if y_t is an out-of-vocabulary word for V.

Copy distributions. A recent Transformer-based pointer-generator randomly chooses one of the attention-heads to form a copy distribution; that approach gave no significant improvements in text summarization (Gehrmann et al., 2018).

In contrast, our model uses an additional attention layer for each copy distribution on top of the decoder stack. For the passages, the layer takes s_t as the query and outputs $\alpha_t^p \in \mathbb{R}^{KL}$ as the attention weights and $c_t^p \in \mathbb{R}^d$ as the context vectors:

$$\begin{aligned} e_l^{p_k} &= {w^p}^\top \tanh(W^{pm} M_l^{p_k} + W^{ps} s_t + b^p), \\ \alpha_t^p &= \text{softmax}([e^{p_1}; \ldots; e^{p_K}]), \\ c_t^p &= \textstyle\sum_l \alpha_{tl}^p M_l^{p_{\text{all}}}, \end{aligned} \tag{1}$$

where $w^p, b^p \in \mathbb{R}^d$ and $W^{pm}, W^{ps} \in \mathbb{R}^{d \times d}$ are learnable parameters. For the question, our model

uses another identical layer and obtains $\alpha_t^q \in \mathbb{R}^J$ and $c_t^q \in \mathbb{R}^d$. As a result, P^q and P^p are the copy distributions over the extended vocabulary:

$$P^q(y_t) = \sum_{j:x_j^q=y_t} \alpha_{tj}^q,$$
$$P^p(y_t) = \sum_{l:x_l^{p_{k(l)}}=y_t} \alpha_{tl}^p,$$

where $k(l)$ means the passage index corresponding to the l-th word in the concatenated passages.

Final distribution. The final distribution of y_t is defined as a mixture of the three distributions:

$$P(y_t) = \lambda^v P^v(y_t) + \lambda^q P^q(y_t) + \lambda^p P^p(y_t),$$
$$\lambda^v, \lambda^q, \lambda^p = \mathrm{softmax}(W^m[s_t; c_t^q; c_t^p] + b^m),$$

where $W^m \in \mathbb{R}^{3\times 3d}$ and $b^m \in \mathbb{R}^3$ are learnable parameters.

3.4.4 Combined Attention

In order not to attend words in irrelevant passages, our model introduces a combined attention. While the original technique combined word and sentence level attentions (Hsu et al., 2018), our model combines the word and passage level attentions. The word attention, Eq. 1, is re-defined as

$$\alpha_{tl}^p = \frac{\alpha_{tl}^p \beta^{p_{k(l)}}}{\sum_{l'} \alpha_{tl'}^p \beta^{p_{k(l')}}}.$$

3.5 Loss Function

We define the training loss as the sum of losses via

$$L(\theta) = L_{\mathrm{dec}} + \gamma_{\mathrm{rank}} L_{\mathrm{rank}} + \gamma_{\mathrm{cls}} L_{\mathrm{cls}}$$

where θ is the set of all learnable parameters, and γ_{rank} and γ_{cls} are balancing parameters.

The loss of the decoder, L_{dec}, is the negative log likelihood of the whole target answer sentence averaged over N_{able} answerable examples:

$$L_{\mathrm{dec}} = -\frac{1}{N_{\mathrm{able}}} \sum_{(a,y)\in\mathcal{D}} \frac{a}{T} \sum_t \log P(y_t),$$

where $\mathcal{D}$ is the training dataset. The losses of the passage ranker, L_{rank}, and the answer possibility classifier, L_{cls}, are the binary cross entropy between the true and predicted values averaged over all N examples:

$$L_{\mathrm{rank}} = -\frac{1}{NK} \sum_k \sum_{r^{p_k}\in\mathcal{D}} \begin{pmatrix} r^{p_k} \log \beta^{p_k} + \\ (1 - r^{p_k}) \log(1 - \beta^{p_k}) \end{pmatrix},$$

$$L_{\mathrm{cls}} = -\frac{1}{N} \sum_{a\in\mathcal{D}} \begin{pmatrix} a \log P(a) + \\ (1 - a) \log(1 - P(a)) \end{pmatrix}.$$

Dataset	Subset	Train	Dev.	Eval.
MS MARCO	ALL	808,731	101,093	101,092
	ANS	503,370	55,636	–
	NLG	153,725	12,467	–
NarrativeQA	Summary	32,747	3,461	10,557

Table 1: Numbers of questions used in the experiments.

4 Experiments on MS MARCO 2.1

We evaluated our model on MS MARCO 2.1 (Bajaj et al., 2018). It is the sole dataset providing abstractive answers with multiple styles and serves as a great test bed for building open-domain QA agents with the NLG capability that can be used in smart devices. The details of our setup and output examples are in the supplementary material.

4.1 Setup

Datasets. MS MARCO 2.1 provides two tasks for generative open-domain QA: the **Q&A** task and the Q&A + Natural Language Generation (**NLG**) task. Both tasks consist of questions submitted to Bing by real users, and each question refers to ten passages. The dataset also includes annotations on the relevant passages, which were selected by humans to form the final answers, and on whether there was no answer in the passages.

Answer styles. We associated the two tasks with two answer styles. The NLG task requires a well-formed answer that is an abstractive summary of the question and passages, averaging 16.6 words. The Q&A task also requires an abstractive answer but prefers it to be more concise than in the NLG task, averaging 13.1 words, and many of the answers do not contain the context of the question. For the question "tablespoon in cup", a reference answer in the Q&A task is "16," while that in the NLG task is "There are 16 tablespoons in a cup."

Subsets. In addition to the **ALL** dataset, we prepared two subsets for ablation tests as listed in Table 1. The **ANS** set consisted of answerable questions, and the **NLG** set consisted of the answerable questions and well-formed answers, so that NLG $\subset$ ANS $\subset$ ALL. We note that multi-style learning enables our model to learn from different answer styles of data (i.e., the ANS set), and multi-task learning with the answer possibility classifier enables our model to learn from both answerable and unanswerable data (i.e., the ALL set).

Training and Inference. We trained our model with mini-batches consisting of multi-style an-

Model	NLG		Q&A	
	R-L	B-1	R-L	B-1
BiDAF[a]	16.91	9.30	23.96	10.64
Deep Cascade QA[b]	35.14	37.35	52.01	**54.64**
S-Net+CES2S[c]	45.04	40.62	44.96	46.36
BERT+Multi-PGNet[d]	47.37	45.09	48.14	52.03
Selector+CCG[e]	47.39	45.26	50.63	52.03
VNET[f]	48.37	46.75	51.63	54.37
Masque (NLG; single)	49.19	49.63	48.42	48.68
Masque (NLG; ensemble)	**49.61**	**50.13**	48.92	48.75
Masque (Q&A; single)	25.66	36.62	50.93	42.37
Masque (Q&A; ensemble)	28.53	39.87	**52.20**	43.77
Human Performance	63.21	53.03	53.87	48.50

Table 2: Performance of our and competing models on the MS MARCO V2 leaderboard (4 March 2019). [a]Seo et al. (2017); [b]Yan et al. (2019); [c]Shao (unpublished), a variant of Tan et al. (2018); [d]Li (unpublished), a model using Devlin et al. (2018) and See et al. (2017); [e]Qian (unpublished); [f]Wu et al. (2018). Whether the competing models are ensemble models or not is unreported.

swers that were randomly sampled. We used a greedy decoding algorithm and did not use any beam search or random sampling, because they did not provide any improvements.

Evaluation metrics and baselines. ROUGE-L and BLEU-1 were used to evaluate the models' RC performance, where ROUGE-L is the main metric on the official leaderboard. We used the reported scores of extractive (Seo et al., 2017; Yan et al., 2019; Wu et al., 2018), generative (Tan et al., 2018), and unpublished RC models at the submission time.

In addition, to evaluate the individual contributions of our modules, we used MAP and MRR for the ranker and F_1 for the classifier, where the positive class was the answerable questions.

4.2 Results

Does our model achieve state-of-the-art on the two tasks with different styles? Table 2 shows the performance of our model and competing models on the leaderboard. Our ensemble model of six training runs, where each model was trained with the two answer styles, achieved state-of-the-art performance on both tasks in terms of ROUGE-L. In particular, for the NLG task, our single model outperformed competing models in terms of both ROUGE-L and BLEU-1.

Does multi-style learning improve the NLG performance? Table 3 lists the results of an ablation test for our single model (controlled with the NLG style) on the NLG dev. set[1]. Our model trained with both styles outperformed the model trained with the single NLG style. Multi-style learning enabled our model to improve its NLG performance by also using non-sentence answers.

Model	Train	R-L	B-1
Masque (NLG style; single)	ALL	**69.77**	**65.56**
w/o multi-style learning (§3.4.2)	NLG	68.20	63.95
↪ w/o Transformer (§3.1.2, §3.4.2)	NLG	67.13	62.96
w/o passage ranker (§3.2)	NLG	68.05	63.82
w/o possibility classifier (§3.3)	ANS	69.64	65.41
Masque w/ gold passage ranker	ALL	78.70	78.14

Table 3: Ablation test results on the NLG dev. set. The models were trained with the subset listed in "Train".

Model	Train	MAP	MRR
Bing (initial ranking)	-	34.62	35.00
Masque (single)	ALL	**69.51**	**69.96**
w/o answer decoder (§3.4)	ALL	67.03	67.49
w/o multi-style learning (§3.4.2)	NLG	65.51	65.59
w/o possibility classifier (§3.3)	ANS	69.08	69.54

Table 4: Passage ranking results on the ANS dev. set.

Does the Transformer-based pointer-generator improve the NLG performance? Table 3 shows that our model also outperformed the model that used RNNs and self-attentions instead of Transformer blocks as in MCAN (McCann et al., 2018). Our deep decoder captured the multi-hop interaction among the question, the passages, and the answer better than a single-layer LSTM decoder could.

Does joint learning with the ranker and classifier improve NLG performance? Furthermore, Table 3 shows that our model (jointly trained with the passage ranker and answer possibility classifier) outperformed the model that did not use the ranker and classifier. Joint learning thus had a regularization effect on the question-passages reader.

We also confirmed that the gold passage ranker, which can perfectly predict the relevance of passages, significantly improved the RC performance. Passage ranking will be a key to developing a system that can outperform humans.

Does joint learning improve the passage ranking performance? Table 4 lists the passage ranking performance on the ANS dev. set[2]. The

[1]We confirmed with the organizer that the dev. results were much better than the test results, but there was no problem.

[2]This evaluation requires our ranker to re-rank 10 passages. It is not the same as the Passage Re-ranking task.

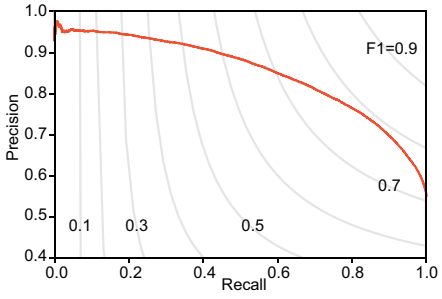

Figure 4: Precision-recall curve for answer possibility classification on the ALL dev. set.

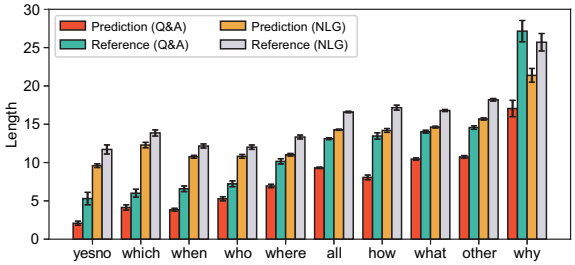

Figure 5: Lengths of answers generated by Masque broken down by the answer style and query type on the NLG dev. set. The error bars indicate standard errors.

ranker shares the question-passages reader with the answer decoder, and this sharing contributed to improvements over the ranker trained without the answer decoder. Also, our ranker outperformed the initial ranking provided by Bing by a significant margin.

Does our model accurately identify answerable questions? Figure 4 shows the precision-recall curve for answer possibility classification on the ALL dev. set. Our model identified the answerable questions well. The maximum F_1 score was 0.7893, where the threshold of answer possibility was 0.4411. This is the first report on answer possibility classification with MS MARCO 2.1.

Does our model control answer lengths with different styles? Figure 5 shows the lengths of the answers generated by our model broken down by the answer style and query type. The generated answers were relatively shorter than the reference answers, especially for the Q&A task, but well controlled with the target style for every query type. The short answers degraded our model's BLEU scores in the Q&A task (Table 2) because of BLEU's brevity penalty (Papineni et al., 2002).

5 Experiments on NarrativeQA

Next, we evaluated our model on NarrativeQA (Kociský et al., 2018). It requires understanding the underlying narrative rather than relying on shallow pattern matching. Our detailed setup and output examples are in the supplementary material.

5.1 Setup

We only describe the settings specific to this experiment.

Datasets. Following previous studies, we used the summary setting for the comparisons with the reported baselines, where each question refers to one summary (averaging 659 words), and there is no unanswerable questions. Our model therefore did not use the passage ranker and answer possibility classifier.

Answer styles. The NarrativeQA dataset does not explicitly provide multiple answer styles. In order to evaluate the effectiveness of multi-style learning, we used the NLG subset of MS MARCO as additional training data. We associated the NarrativeQA and NLG datasets with two answer styles. The answer style of NarrativeQA (**NQA**) is different from that of MS MARCO (**NLG**) in that the answers are short (averaging 4.73 words) and contained frequently pronouns. For instance, for the question "Who is Mark Hunter?", a reference is "He is a high school student in Phoenix."

Evaluation metrics and baselines. BLEU-1 and 4, METEOR, and ROUGE-L were used in accordance with the evaluation in the dataset paper (Kociský et al., 2018). We used the reports of top-performing extractive (Seo et al., 2017; Tay et al., 2018; Hu et al., 2018) and generative (Bauer et al., 2018; Indurthi et al., 2018) models.

5.2 Results

Does our model achieve state-of-the-art performance? Table 5 shows that our single model, trained with two styles and controlled with the NQA style, pushed forward the state-of-the-art by a significant margin. The evaluation scores of the model controlled with the NLG style were low because the two styles are different. Also, our model without multi-style learning (trained with only the NQA style) outperformed the baselines in terms of ROUGE-L. This indicates that our model architec-

Model	B-1	B-4	M	R-L
BiDAF[a]	33.72	15.53	15.38	36.30
DECAPROP[b]	42.00	23.42	23.42	40.07
MHPGM+NOIC[c]	43.63	21.07	19.03	44.16
ConZNet[d]	42.76	22.49	19.24	46.67
RMR+A2D[e]	50.4	26.5	N/A	53.3
Masque (NQA)	**54.11**	**30.43**	**26.13**	**59.87**
w/o multi-style learning	48.70	20.98	21.95	54.74
Masque (NLG)	39.14	18.11	24.62	50.09
Masque (NQA; valid.)[f]	52.78	28.72	25.38	58.94

Table 5: Performance of our and competing models on the NarrativeQA test set. [a]Seo et al. (2017); [b]Tay et al. (2018); [c]Bauer et al. (2018); [d]Indurthi et al. (2018); [e]Hu et al. (2018). [f]Results on the NarrativeQA validation set.

ture itself is powerful for natural language understanding in RC.

6 Related Work and Discussion

Transfer and multi-task learning in RC. Recent breakthroughs in transfer learning demonstrate that pre-trained language models perform well on RC with minimal modifications (Peters et al., 2018; Devlin et al., 2018; Radford et al., 2018, 2019). In addition, our model also uses ELMo (Peters et al., 2018) for contextualized embeddings.

Multi-task learning is a transfer mechanism to improve generalization performance (Caruana, 1997), and it is generally applied by sharing the hidden layers between all tasks, while keeping task-specific layers. Wang et al. (2018) and Nishida et al. (2018) reported that the sharing of the hidden layers between the multi-passage RC and passage ranking tasks was effective. Our results also showed the effectiveness of the sharing of the question-passages reader module among the RC, passage ranking, and answer possibility classification tasks.

In multi-task learning without task-specific layers, Devlin et al. (2018) and Chen et al. (2017) improved RC performance by learning multiple datasets from the same extractive RC setting. McCann et al. (2018) and Yogatama et al. (2019) investigated multi-task and curriculum learning on many different NLP tasks; their results were below task-specific RC models. Our multi-style learning does not use style-specific layers; instead uses a style-conditional decoder.

Generative RC. S-Net (Tan et al., 2018) used an extraction-then-synthesis mechanism for multi-passage RC. The models proposed by McCann et al. (2018), Bauer et al. (2018), and Indurthi et al. (2018) used an RNN-based pointer-generator mechanism for single-passage RC. Although these mechanisms can alleviate the lack of training data, large amounts of data are still required. Our multi-style learning will be a key technique enabling learning from many RC datasets with different styles.

In addition to MS MARCO and NarrativeQA, there are other datasets that provide abstractive answers. DuReader (He et al., 2018), a Chinese multi-document RC dataset, provides longer documents and answers than those of MS MARCO. DuoRC (Saha et al., 2018) and CoQA (Reddy et al., 2018) contain abstractive answers; most of the answers are short phrases.

Controllable text generation. Many studies have been carried out in the framework of style transfer, which is the task of rephrasing a text so that it contains specific styles such as sentiment. Recent studies have used artificial tokens (Sennrich et al., 2016; Johnson et al., 2017), variational auto-encoders (Hu et al., 2017), or adversarial training (Fu et al., 2018; Tsvetkov et al., 2018) to separate the content and style on the encoder side. On the decoder side, conditional language modeling has been used to generate output sentences with the target style. In addition, output length control with conditional language modeling has been well studied (Kikuchi et al., 2016; Takeno et al., 2017; Fan et al., 2018). Our style-controllable RC relies on conditional language modeling in the decoder.

Multi-passage RC. The simplest approach is to concatenate the passages and find the answer from the concatenation, as in (Wang et al., 2017). Earlier pipelined models found a small number of relevant passages with a TF-IDF based ranker and passed them to a neural reader (Chen et al., 2017; Clark and Gardner, 2018), while more recent models have used a neural re-ranker to more accurately select the relevant passages (Wang et al., 2018; Nishida et al., 2018). Also, non-pipelined models (including ours) consider all the provided passages and find the answer by comparing scores between passages (Tan et al., 2018; Wu et al., 2018). The most recent models make a proper trade-off between efficiency and accuracy (Yan et al., 2019; Min et al., 2018).

RC with unanswerable question identification. The previous work of (Levy et al., 2017; Clark and Gardner, 2018) outputted a no-answer score depending on the probability of all answer spans. Hu et al. (2019) proposed an answer verifier to compare an answer with the question. Sun et al. (2018) jointly learned an RC model and an answer verifier. Our model introduces a classifier on top of the question-passages reader, which is not dependent on the generated answer.

Abstractive summarization. Current state-of-the-art models use the pointer-generator mechanism (See et al., 2017). In particular, content selection approaches, which decide what to summarize, have recently been used with abstractive models. Most methods select content at the sentence level (Hsu et al., 2018; Chen and Bansal, 2018) or the word level (Pasunuru and Bansal, 2018; Li et al., 2018; Gehrmann et al., 2018). Our model incorporates content selection at the passage level in the combined attention.

Query-based summarization has rarely been studied because of a lack of datasets. Nema et al. (2017) proposed an attentional encoder-decoder model; however, Saha et al. (2018) reported that it performed worse than BiDAF on DuoRC. Hasselqvist et al. (2017) proposed a pointer-generator based model; however, it does not consider copying words from the question.

7 Conclusion

This study sheds light on multi-style generative RC. Our proposed model, *Masque*, is based on multi-source abstractive summarization and learns multi-style answers together. It achieved state-of-the-art performance on the Q&A task and the Q&A + NLG task of MS MARCO 2.1 and the summary task of NarrativeQA. The key to its success is transferring the style-independent NLG capability to the target style by use of the question-passages reader and the conditional pointer-generator decoder. In particular, the capability of copying words from the question and passages can be shared among the styles, while the capability of controlling the mixture weights for the generative and copy distributions can be acquired for each style. Our future work will involve exploring the potential of our multi-style learning towards natural language understanding.

References

Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *Computing Research Repository (CoRR)*, arXiv:1607.06450. Version 1.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A human generated machine reading comprehension dataset. *Computing Research Repository (CoRR)*, arXiv:1611.09268. Version 3.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 4220–4230.

Richard Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open–domain questions. In *Association for Computational Linguistics (ACL)*, pages 1870–1879.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Association for Computational Linguistics (ACL)*, pages 675–686.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Association for Computational Linguistics (ACL)*, pages 845–855.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository (CoRR)*, arXiv:1810.04805. Version 1.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Workshop on Neural Machine Translation and Generation (NMT@ACL)*, pages 45–54.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 663–670.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 4098–4109.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *Computing Research Repository (CoRR)*, arXiv:1308.0850. Version 5.

Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. 2017. Query-based abstractive summarization using neural networks. *arXiv*, 1712.06100.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a chinese machine reading comprehension dataset from real-world applications. In *Workshop on Machine Reading for Question Answering (MRQA@ACL)*, pages 37–46.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *Computing Research Repository (CoRR)*, arXiv:1606.08415. Version 2.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Association for Computational Linguistics (ACL)*, pages 132–141.

Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, Dongsheng Li, Nan Yang, and Ming Zhou. 2018. Attention-guided answer distillation for machine reading comprehension. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2077–2086.

Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Ming Zhou. 2019. Read + Verify: Machine reading comprehension with unanswerable questions. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning (ICML)*, pages 1587–1596.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2017. Tying word vectors and word classifiers: A loss framework for language modeling. In *International Conference on Learning Representations (ICLR)*.

Sathish Reddy Indurthi, Seunghak Yu, Seohyun Back, and Heriberto Cuayáhuitl. 2018. Cut to the chase: A context zoom-in network for reading comprehension. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 570–575.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistic (TACL)*, 5:339–351.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics (ACL)*, pages 1601–1611.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1328–1338.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Tomás Kociský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistic (TACL)*, 6:317–328.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Computational Natural Language Learning (CoNLL)*, pages 333–342.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. 2018. Guiding generation for abstractive text summarization based on key information guide network. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 55–60.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *Computing Research Repository (CoRR)*, arXiv:1711.05101. Version 1.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *Computing Research Repository (CoRR)*, arXiv:1806.08730. Version 1.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Association for Computational Linguistics (ACL)*, pages 1725–1735.

Preksha Nema, Mitesh M. Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity driven attention model for query-based abstractive summarization. In *Association for Computational Linguistics (ACL)*, pages 1063–1072.

Kyosuke Nishida, Itsumi Saito, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2018. Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension. In *Conference on Information and Knowledge Management (CIKM)*, pages 647–656.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Association for Computational Linguistics (ACL)*, pages 311–318.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 646–653.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2018. CoQA: A conversational question answering challenge. *Computing Research Repository (CoRR)*, arXiv:1808.07042. Version 1.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Association for Computational Linguistics (ACL)*, pages 1683–1693.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Association for Computational Linguistics (ACL)*, pages 1073–1083.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 35–40.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations (ICLR)*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *Computing Research Repository (CoRR)*, arXiv:1505.00387. Version 2.

Fu Sun, Linyang Li, Xipeng Qiu, and Yang Liu. 2018. U-Net: Machine reading comprehension with unanswerable questions. *Computing Research Repository (CoRR)*, arXiv:1810.06638. Version 1.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Shunsuke Takeno, Masaaki Nagata, and Kazuhide Yamamoto. 2017. Controlling target features in neural machine translation via prefix constraints. In *Workshop on Asian Translation (WAT@IJCNLP)*, pages 55–63.

Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. S-Net: From answer extraction to answer synthesis for machine reading comprehension. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 5940–5947.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Densely connected attention propagation for reading comprehension. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4911–4922.

Yulia Tsvetkov, Alan W. Black, Ruslan Salakhutdinov, and Shrimai Prabhumoye. 2018. Style transfer through back-translation. In *Association for Computational Linguistics (ACL)*, pages 866–876.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6000–6010.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced reader-ranker for open-domain question answering. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 5981–5988.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Association for Computational Linguistics (ACL)*, pages 189–198.

Hua Wu, Haifeng Wang, Sujian Li, Wei He, Yizhong Wang, Jing Liu, Kai Liu, and Yajuan Lyu. 2018. Multi-passage machine reading comprehension with cross-passage answer verification. In *Association for Computational Linguistics (ACL)*, pages 1918–1927.

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *International Conference on Learning Representations (ICLR)*.

Ming Yan, Jiangnan Xia, Chen Wu, Bin Bi, Zhongzhou Zhao, Ji Zhang, Luo Si, Rui Wang, Wei Wang, and Haiqing Chen. 2019. A deep cascade model for multi-document reading comprehension. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. 2019. Learning and evaluating general linguistic intelligence. *Computing Research Repository (CoRR)*, arXiv:1901.11373. Version 1.

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 *International Conference on Learning Representations (ICLR)*.

A Supplementary Material

A.1 Experimental Setup for MS MARCO

Model configurations. We trained our model on a machine with eight NVIDIA P100 GPUs. Our best model was jointly trained with the two answer styles in the ALL set for a total of eight epochs with a batch size of 80, where each batch consisted of multi-style answers that were randomly sampled. The training took roughly six days. The hidden size d was 304, and the number of attention heads was 8. The inner state size of the feed-forward networks was 256. The numbers of shared encoding blocks, modeling blocks for a question, modeling blocks for passages, and decoder blocks were 3, 2, 5, and 8, respectively. We used the pre-trained uncased 300-dimensional GloVe (Pennington et al., 2014)[3] and the original 512-dimensional ELMo (Peters et al., 2018)[4]. We used the spaCy tokenizer, and all input words were lowercased except the input for ELMo. The output words were also lowercase. The number of common words in V_{ext} in the extended vocabulary was 5,000. Each passage and each answer were truncated to 100 words for training.

Optimizer. We used Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The weights were initialized using $N(0, 0.02)$, except that the biases of all the linear transformations were initialized with zero vectors. The learning rate was increased linearly from zero to 2.5×10^{-4} in the first 2,000 steps and then annealed to 0 by using a cosine schedule. All parameter gradients were clipped to a maximum norm of 1. An exponential moving average was applied to all trainable variables with a decay rate of 0.9995. The balancing factors for joint learning, λ_{rank} and λ_{cls}, were set to 0.5 and 0.1, respectively.

Regularization. We used a modified version of the L_2 regularization proposed in (Loshchilov and Hutter, 2017), with $w = 0.01$ on all non-bias. We additionally used a dropout (Srivastava et al., 2014) rate of 0.3 for all highway networks and residual and scaled dot-product attention operations in the multi-head attention mechanism. We also used one-sided label smoothing (Szegedy et al., 2016) for the passage relevance and answer possibility labels. We smoothed only the positive labels to 0.9.

Ensemble model. The ensemble model consisted of six training runs with identical architectures and hyperparameters but with different weight initializations. The final answer was decided with a weighted majority, where we used the ROUGE-L score for the dev. set as the weight of each model.

Evaluation settings. We used the official evaluation script. The answers were normalized by making words lowercase.

[3] https://nlp.stanford.edu/projects/glove/

[4] https://allennlp.org/elmo

A.2 Experimental Setup for NarrativeQA

Model configurations. Our best model was jointly trained with the NarrativeQA and MS MARCO NLG datasets for a total of seven epochs with a batch size of 64, where each batch consisted of multi-style answers that were randomly sampled. For efficient multi-style learning, each summary in the NarrativeQA dataset was divided into ten passages (size of 130 words) with sentence-level overlaps such that each sentence in the summary was entirely contained in a passage. Each passage from MS MARCO was also truncated to 130 words. The rest of the configuration was the same as in the MS MARCO experiments.

Evaluation settings. An official evaluation script is not provided, so we used the evaluation script created by Bauer et al. (2018)[5]. The answers were normalized by making words lowercase and removing punctuation marks.

A.3 Output Examples Generated by Masque

Tables 6 and 7 list the generated examples for questions from MS MARCO 2.1 and NarrativeQA, respectively. We can see from the examples that our model could control answer styles appropriately for various question and reasoning types. We did find some important errors: style errors, yes/no classification errors, copy errors with respect to numerical values, grammatical errors, and multi-hop reasoning errors.

[5] https://github.com/yicheng-w/CommonSenseMultiHopQA/

(a) Question: why your body would feel like it is shaking
Relevant Passage: Shaking is a symptom in which a person has tremors (shakiness or small back and forth movements) in part or all of his body. Shaking can be due to cold body temperatures, rising fever (such as with infections), neurological problems, medicine effects, drug abuse, etc. ...Read more.
Reference Answer (Q&A): Shaking can be due to cold body temperatures, rising fever (such as with infections), neurological problems, medicine effects, drug abuse, etc.
Prediction (Q&A): because of cold body temperatures , rising fever , neurological problems , medicine effects , drug abuse . ✓
Reference Answers (NLG): Shaking can be due to cold body temperatures, rising fever, neurological problems, medicine effects and drug abuse. / Body would feel like it is shaking due to cold body temperatures, rising fever, neurological problems, medicine effects, drug abuse.
Prediction (NLG): your body would feel like it is shaking because of cold body temperatures , rising fever , neurological problems , medicine effects , drug abuse . ✓

(b) Question: ___ is the name used to refer to the era of legalized segregation in the united states
Relevant Passage: Jim Crow law, in U.S. history, any of the laws that enforced racial segregation in the South between the end of Reconstruction in 1877 and the beginning of the civil rights movement in the 1950s.
Reference Answer (Q&A): Jim Crow
Prediction (Q&A): jim crow ✓
Reference Answer (NLG): Jim Crow is the name used to refer to the era of legalized segregation in the United States.
Prediction (NLG): jim crow is the name used to refer to the era of legalized segregation in the united states . ✓

(c) Question: average height nba player
Relevant Passage: The average height of an NBA player is around 6 feet 7 inches. The tallest NBA player ever was Gheorghe Muresan, who was 7 feet 7 inches tall. In contrast, the shortest NBA player ever was Tyrone Muggsy Bogues, who was 5 feet 3 inches tall.
Reference Answer (Q&A): Around 6 feet 7 inches
Prediction (Q&A): 6 feet 7 inches ✓
Reference Answers (NLG): The average height of NBA players is around 6 feet, 7 inches. / The height of NBA player is around 6 feet 7 inches.
Prediction (NLG): the average height of an national basketball association player is 6 feet 7 inches . ✓

(d) Question: population of henryetta ok
Relevant Passage: With it's population of 5,927 people, Henryetta represented 14.79% of the county's total. In addition, where the city of Henryetta has a population density of 891 people per square mile [344.2 people/km2], Okmulgee County averages 57.5 people per square mile [22.19 people/km2].
Reference Answer (Q&A): 5927
Prediction (Q&A): the population of henryetta , oklahoma is 5,927 . ✓ (content is OK, but style is NG ✗)
Reference Answer (NLG): The population of Henryetta, Oklahoma is 5,927.
Prediction (NLG): the population of henryetta , oklahoma is 5,927 . ✓

(e) Question: does gameplay programmer need math skill
Relevant Passage: A good computer programmer is more of a problem solver and logical thinker than a math buff. And besides, the industry is peppered with many successful computer programmers who do not really know much about mathematics.
Reference Answer (Q&A): No
Prediction (Q&A): yes ✗
Reference Answers (NLG): No, a gameplay programmer doesn't need math skill. / No, gameplay programmer do not need an math skill.
Prediction (NLG): no , gameplay programmer does not need math skill . ✓

(f) Question: how long does a freezer take to cool down
Relevant Passage: Quick Answer. It takes anywhere from three to 24 hours for a refrigerator to reach safe temperatures for storing food, depending on the size and type of unit. When the refrigerator compartment reaches 40 degrees Fahrenheit and the freezer reaches 5 degrees Fahrenheit, it is safe to transfer food items. Keep Learning.
Reference Answer (Q&A): 24 hours
Prediction (Q&A): 4 to 5 hours ✗
Reference Answers (NLG): A freezer takes 24 hours to cool down. / A freezer take to cool down is 24 hours.
Prediction (NLG): a freezer takes 4 to 12 hours to cool down . ✗

Table 6: Output examples generated by Masque from MS MARCO. The model was trained with the Q&A and NLG styles. The relevant passage is one that an annotator selected to compose the reference answer. The model could control answer styles appropriately for (a) natural language, (b) cloze-style, and (c) keywords questions. (d) The answer style was incorrect. (e) The answers were not consistent between the styles. (f) Copying from numerical words worked poorly. There were some grammatical errors in the generative answers, which are underlined.

(a) Question: Where does Mark broadcast his radio station?
Summary: Mark Hunter (Slater), a high school student in a sleepy suburb of Phoenix, Arizona, starts an FM pirate radio station that broadcasts from the basement of his parents' house. Mark is a loner, an outsider, whose only outlet for his teenage angst and aggression is his unauthorized radio station. His pirate station's theme song is "Everybody Knows" by Leonard Cohen and there are glimpses of cassettes by such alternative musicians as The Jesus and Mary Chain, Camper Van Beethoven, Primal Scream, Soundgarden, Ice-T, Bad Brains, Concrete Blonde, Henry Rollins, and The Pixies. By day, Mark is seen as a loner, hardly talking to anyone around him; by night, he expresses his outsider views about what is wrong with American society. When he speaks his mind about what is going on at his school and in the community, more and more of his fellow students tune in to hear his show. (...)
Reference Answers: In his parent's basement. / His parents' basement.
Prediction (NQA): the basement of his parents ' house ✓
Prediction (NLG): mark broadcast his radio station in the basement of his parents ' house . ✓

(b) Question: Fletch is a reporter for what newspaper?
Summary: Los Angeles Times reporter Irwin "Fletch" Fletcher (Chase) is writing an article exposing drug trafficking on the beaches of Los Angeles. Posing as an addict during his investigation, he is approached by Boyd Aviation executive vice president Alan Stanwyk (Matheson) who mistakenly assumes Fletch is a junkie. Stanwyk claims to have bone cancer, with only months left to live, and wishes to avoid the pain and suffering. Stanwyk offers $50,000 for Fletch to come to his mansion in a few days time, kill him, and then escape to Rio de Janeiro, staging the murder to look like a burglary. Fletch, while not completely convinced on the truth of Stanwyk's story, reluctantly agrees to the plan. Along with his colleague Larry (Davis), he begins investigating Stanwyk instead of completing his drug trafficking expos, much to the disapproval of his overbearing editor Frank Walker (Libertini). Disguised as a doctor, Fletch accesses Stanwyk's file at the hospital and learns Stanwyk lied about having cancer. (...)
Reference Answers: Los Angeles Times / Los Angeles
Prediction (NQA): los angeles times ✓
Prediction (NLG): fletch is a reporter for los angeles times . ✓

(c) Question: How long approximately was the voyage from London to Thailand supposed to take?
Summary: (...) The story is set twenty-two years earlier, when Marlow was 20. With two years of experience, most recently as third mate aboard a crack clipper, Marlow receives a billet as second mate on the barque Judea. The skipper is Captain John Beard, a man of about 60. This is Beard's first command. The Judea is an old boat, belonging to a man "Wilmer, Wilcox or something similar", suffering from age and disuse in Shadewell basin. The 400-ton ship is commissioned to take 600 tons of coal from England to Thailand. The trip should take approximately 150 days. The ship leaves London loaded with sand ballast and heads north to the Senn river to pick up the cargo of coal. On her way, the Judea suffers from her ballast shifting aside and the crew go below to put things right again. The trip takes 16 days because of inclement weather, and the battered ship must use a tug boat to get into port. The Judea waits a month on the Tyne to be loaded with coal. The night before she ships out she is hit by a steamer, the Miranda or the Melissa. The damage takes another three weeks to repair. Three months after leaving London, the Judea ships off for Bangkok. The Judea travels through the North Sea and Britain. 300 miles west of the Lizard a winter storm, 'the famous winter gale of twenty-two years ago', hits. (...)
Reference Answers: Approximately 150 days / 150 days
Prediction (NQA): 150 days ✓
Prediction (NLG): the voyage from london to thailand was supposed to take 150 days . ✓

(d) Question: Why does Jamie start avoiding Landon?
Summmary: (...) During these functions, Landon notices Jamie Sullivan, a girl he has known since kindergarten and who has attended many of the same classes as him, and is also the local minister's daughter. Since he's one of the in-crowd, he has seldom paid any attention to Jamie, who wears modest dresses and owns only one sweater. Jamie is labeled an outsider and a geek. She makes no attempt to wear make-up or otherwise improve her looks or attract attention to herself. Landon has trouble learning his lines for the play. Jamie, who is also in the play, agrees to help him on one condition: Jamie warns Landon not to fall in love with her; he laughs it off and dismisses it as a foolish idea. Landon and Jamie begin practicing together at her house after school. They get to know each other and a spark of affection arises between them. On the opening night of the play, Jamie astounds Landon and the entire audience with her beauty and her voice. Onstage at the peak of the ending to the play, Jamie sings. When Jamie finishes, Landon improvises and kisses her which is not a part of the play. Afterwards, Jamie avoids Landon, and it is not until Landon's friends play a cruel prank on Jamie and he protects her in opposition to his friends that she warms up to him again. Landon asks Jamie on a date soon after, but Jamie says her father doesn't allow her to date. (...)
Reference Answers: Because he kissed her in the play. / He kisses her
Prediction (NQA): he is not a part of the play ✗
Prediction (NLG): he is not a part of the play ✗

Table 7: Output examples generated by Masque from NarrativeQA. The model was trained with the NarrativeQA (NQA) and MS MARCO (NLG) styles. It could control answer styles appropriately for questions that required (a,b) single-sentence reasoning and (c) multi-sentence reasoning. (d) Example of an error in multi-sentence reasoning. There were some grammatical errors in the generative answers, which are underlined.