A Memory Model for Question Answering from Streaming Data Supported by Rehearsal and Anticipation of Coreference Information

Vladimir Araujo[1,2], **Alvaro Soto**[2], **Marie-Francine Moens**[1]
[1]KU Leuven, [2]Pontificia Universidad Católica de Chile
vgaraujo@uc.cl, asoto@ing.puc.cl, sien.moens@kuleuven.be

Abstract

Existing question answering methods often assume that the input content (e.g., documents or videos) is always accessible to solve the task. Alternatively, memory networks were introduced to mimic the human process of incremental comprehension and compression of the information in a fixed-capacity memory. However, these models only learn how to maintain memory by backpropagating errors in the answers through the entire network. Instead, it has been suggested that humans have effective mechanisms to boost their memorization capacities, such as rehearsal and anticipation. Drawing inspiration from these, we propose a memory model that performs rehearsal and anticipation while processing inputs to memorize important information for solving question answering tasks from streaming data. The proposed mechanisms are applied self-supervised during training through masked modeling tasks focused on coreference information. We validate our model on a short-sequence (bAbI) dataset as well as large-sequence textual (NarrativeQA) and video (ActivityNet-QA) question answering datasets, where it achieves substantial improvements over previous memory network approaches. Furthermore, our ablation study confirms the proposed mechanisms' importance for memory models.

1 Introduction

The question answering (QA) task is one of the most important and challenging tasks in natural language processing (NLP). A significant advance has been seen in this subject thanks to models based on deep learning (Hermann et al., 2015; Seo et al., 2017; Chen et al., 2017; Devlin et al., 2019). However, these models assume that the whole input (e.g., sentences, paragraphs, etc.) can always be accessed while answering the question. This is a reasonable and practical approach if the input sequence is short but becomes less effective and efficient as the input length grows. These models

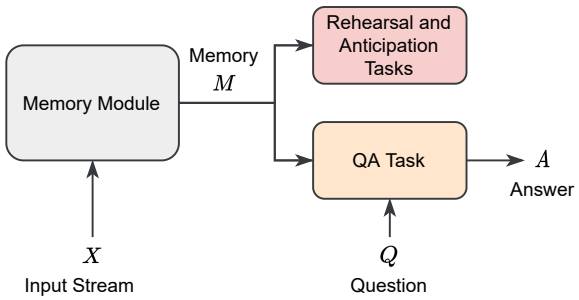


Figure 1: A rehearsal and anticipatory memory model (RAM) for question answering. The model is fed with a stream of data X, incrementally processed to fill a memory M. Such memory is used to obtain question-related Q clues to provide an answer A. Pretext self-supervised tasks improve memorization by continually rehearsing and anticipating coreference information.

are also cognitively implausible, lacking incremental human language processing (Tanenhaus et al., 1995; Keller, 2010) (or visual (Goldstone, 1998)), which could enable the models for online tasks where data input is a stream, such as discourse, dialog or video processing.

Memory-augmented neural networks (MANNs) (Graves et al., 2014) offer a solution to this problem as they introduce mechanisms to compress and remember the input contents. MANNs have shown effectiveness in various tasks (Graves et al., 2016; Liu et al., 2019; Le et al., 2020), including question answering (Miller et al., 2016; Xiong et al., 2016; Han et al., 2019; Zhang et al., 2021). In the latter, given a data sequence, the models process the input incrementally, capturing relevant information in the memory to answer a given query. This is akin to the daily behavior of human beings. For instance, our mental state is updated when we encounter a new text segment while reading (Traxler et al., 1997). Later, we can use the memorized information to solve a specific task (Moscovitch et al., 2016).

Despite the success of MANNs and their similarity to human memory processes, they still have one

Findings of the Association for Computational Linguistics: ACL 2023, pages 13124–13138

significant limitation. Most of these models rely on learning a single task to maintain their memory; for example, a MANN for QA must learn what to store in memory while learning to answer questions. Instead, some studies have suggested that human beings are endowed with specific mechanisms for memorization. On the one hand, rehearsal (Waugh and Norman, 1965; Craik and Watkins, 1973) is a mechanism humans use to memorize information more effectively by repeating information over and over to be remembered. On the other hand, anticipation (Hawkins and Blakeslee, 2004; Wittmann et al., 2007; Cole et al., 2015) suggests that our memory could lead to predictions of upcoming material. Furthermore, both processes are potentially guided by coreference information (Jaffe et al., 2020), an integral part of discourse comprehension.

In this work, we propose a **R**ehearsal and **A**nticipatory **M**emory model (RAM) consisting of a memory module that uses a fusion module to integrate incoming information from a data stream to solve the QA task. This model is supported by two pretext tasks respectively inspired by human rehearsal and anticipation processes to enhance memory representation. These tasks are based on masked modeling (Devlin et al., 2019) of coreference information, allowing the model to anticipate a coreferent and link it to the past through memory and rehearsal. We validate our proposed model with datasets of short synthetic text sequences, long realistic text sequences, and video question answering. Results show that our model significantly outperforms previous well-known memory approaches due to the enhanced memory representation achieved by using the pretext tasks.

2 Proposed Method

In this section, we introduce our model, which processes inputs and builds memory representations incrementally from a data stream, used later to provide an answer to a question (Figure 1). RAM leverages attention and gating mechanisms along with masked modeling-based self-supervision to create a simple but effective method to improve memory representation and memorization. We first describe the problem formulation of QA with MANNs. We then present our basic memory architecture for encoding input and decoding an answer. Finally, we introduce our novel self-supervised mechanisms inspired by rehearsal and anticipation guided by coreference information.

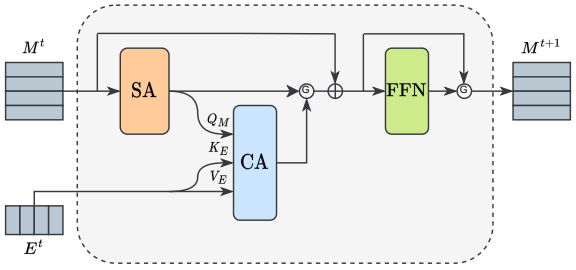


Figure 2: Our memory module consists of a self-attention (SA) and cross-attention (CA) layer that fuses the information from input E^t to memory M^t at step t. It also uses a gating mechanism to allow it to forget.

2.1 Problem Formulation

Let X be an input stream and Q a question related to the content of X. Standard QA approaches consist of a model $G(X, Q)$ that is trained to predict an answer A. In an incremental processing setup (Han et al., 2019), the model does not have access to the whole input, so a fixed-size memory M is used to compress the input stream X one step at a time. Then a model $G(M, Q)$ learns to infer the answer A for any relevant Q.

2.2 Memory Model Overview

Here we present our basic model that incrementally processes the inputs and integrates the relevant information into memory to then answer a question.

Input Encoding: The input to our model is a sequence of segments (sequence of sentences or images in our experiments). A segment $X^t = \{x_n^t\}_{n=1}^N$ at step t with N tokens is encoded by projecting each token into the latent space by using a linear function F. To inject information about the position of the tokens in the segment, we add learnable position embeddings PE. In this way, we obtain the final token embeddings of the segment $E^t = \{e_n^t\}_{n=1}^N$ with dimension d_{model}. We use this procedure to encode the question as well.

$$E^t = F(X^t) + PE \tag{1}$$

Note that this form of representation allows the model to granularly capture and memorize the relevant elements of each segment, which differs from previous approaches that rely on a sequence of sum-pooled representations (Limbacher and Legenstein, 2020; Le et al., 2020).

Memory Module: Our model is augmented with a parametric memory $M = \{m_k\}_{k=1}^K$ with K slots

intended to compress the information from the input stream effectively. For this we implement a module that borrows the information fusion idea from (Dou et al., 2022) and extends it to integrate information into the memory incrementally.

As shown in Figure 2, at step t, our module receives two inputs: a segment representation E^t and a memory M^t to be updated. First, a self-attention layer (SA) is used over the memory to allow the slots to interact with each other. Then, a cross-attention (CA) operation is performed to merge the tokens' information into memory (Eq. 2). Note that the query matrix Q_M (with dimension d_k) is computed from the memory and the key K_E and value V_E matrices from the segment. We employ a residual connection around the two layers.

As a result, we obtain an intermedial memory state $\hat{M}^t$ (Eq. 3) with K slots, which has been updated with information from step t. Intuitively each slot can, for example, describe an object or an entity in the input or composition of them.

$$\mathrm{CA}(M, E) = softmax\left(\frac{Q_M K_M^\top}{\sqrt{d_k}}\right) V_E \quad (2)$$

$$\hat{M}^t = \mathrm{CA}(\mathrm{SA}(M^t), E^t) \quad (3)$$

To allow the model to "forget," an essential ability for memory networks, we employ a gating mechanism proposed by Hutchins et al. (2022). First, the memory candidates are computed (Eq. 4). Unlike the standard LSTM gating, the candidate only is computed using the intermediate memory state $\hat{M}^t$ as it indirectly depends on M^t. Later, the input (Eq. 5) and forget gates (Eq. 6) which are used to obtain the new memory state (Eq. 7) are calculated. Note that the matrices W and bias vectors b are trainable.

$$z_t^k = tanh(W_z \hat{m}_t^k + b_z) \quad (4)$$

$$i_t^k = \sigma(W_i \hat{m}_t^k + b_i - 1) \quad (5)$$

$$f_t^k = \sigma(W_f \hat{m}_t^k + b_f + 1) \quad (6)$$

$$m_{t+1}^k = m_t^k \odot f_t^k + z_t^k \odot i_t^k \quad (7)$$

Finally, the new memory state M^{t+1} is projected with a feedforward layer (FFN). We replace the standard residual connection with the gating mechanism explained above. Note that the parameters are not tied, so this gating mechanism is not attached to the previous one.

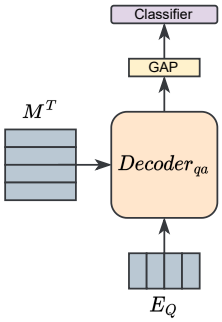


Figure 3: The output decoder takes the question Q and the last memory state M^T as inputs to obtain an answer A using a classifier.

Output Decoding: After the model has processed the entire input stream and obtained our memory fed M^T, we use a decoder to obtain the answer (Figure 3). We use a standard transformer decoder (Vaswani et al., 2017), which allows contextualizing the question token embeddings and aggregating question-relevant clues from memory (Eq. 8). Then, we perform a global average pooling (GAP) of the resulting representations H_Q and use a classifier W_{clf} to predict the answer (Eq. 9) by optimizing a loss $\mathcal{L}_{qa}$. To allow H_Q to capture enough information to answer the question, we perform multi-hop reasoning by iteratively updating H_Q states with the same decoder. This is equivalent to having a multilayer decoder with tied weights.

$$H_Q = Decoder_{qa}(E_Q, M^T) \quad (8)$$

$$A = W_{clf}(H_Q) \quad (9)$$

2.3 Rehearsal and Anticipation Mechanisms

Rehearsal and anticipation are important processes that occur in our brain to refresh memory and prevent forgetting. On the one hand, rehearsal consists of continually bringing to our mind information already experienced to strengthen those memories (Waugh and Norman, 1965; Craik and Watkins, 1973). On the other hand, anticipation is like imagining the future. It has been found that anticipation can fire memory-forming regions of the brain — even before an event has occurred (Mackiewicz et al., 2006; Wittmann et al., 2007). Furthermore, rehearsal and anticipation were found to be related, might be using the same machinery (Cole et al., 2015), and guided by coreference information (Jaffe et al., 2020).

Motivated by these findings, we propose implementing rehearsal as the reconstruction of the past

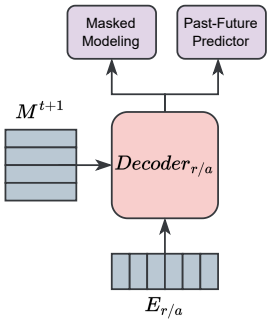


Figure 4: Our rehearsal and anticipation (r/a) decoder receives a masked input $E_{r/a}$ and the more recent memory updated M^{t+1} to predict the masked tokens and whether the segment belongs to the past or future.

(Zhang et al., 2021) and anticipation as the prediction of the future (Oord et al., 2018; Araujo et al., 2021). To use the same machinery, we pose these processes as masked modeling tasks that predict past and future coreference-related tokens.

Rehearsal: This process is performed at step t by randomly selecting a previous ($< t$) segment to mask some of its tokens. Using a standard transformer decoder (Figure 4), we compute an input segment representation E_r to obtain a contextual representation H_r. This process aggregates relevant clues through cross-attention from the updated memory M^{t+1} to the representations (Eq. 10).

Anticipation: This process is performed similarly to rehearsal (Eq. 10), but the next $(t+1)$ segment embedding E_a is used instead. We use the same decoder to compute the representation H_a.

$$H_{r/a} = Decoder_{r/a}(E_{r/a}, M^t) \quad (10)$$

where r and a denote rehearsal and anticipation, respectively. The final hidden vectors (H_r or H_a) corresponding to masked tokens are fed into a classifier to predict the actual token.

What to rehearse and anticipate: Our model has to store relevant information to predict future elements and, at the same time, avoid forgetting crucial information to predict the elements of previous segments. For this to happen, it is important to mask significant tokens. Unlike the standard masked modeling, we do not mask tokens randomly but instead focus on masking coreference-related tokens (for instance, person, object, or event).

The idea stems from the suggestion that coreference information may be helpful in retrieving probable antecedents from memory and in informing expectations about future words in language comprehension (Jaffe et al., 2020). Intuitively, by focusing on coreference relationships, our model can anticipate a coreferent and link it to the past through memory and rehearsal, also connecting the related pieces of discourse information.

Separating the tasks: Since the task is the same for both mechanisms, the model would learn to predict the mask regardless of whether it is past or future information. Therefore, we include an additional signal to teach the model to distinguish where the segment comes from. We use the [CLS] token to create a binarized prediction task to classify whether the segment belongs to the past or the future. As a result, we obtain three loss functions that we additively combine to obtain our final self-supervised modeling loss $\mathcal{L}_{ssm}$ (Eq. 11). Our loss function is used in conjunction with the QA loss $\mathcal{L}_{qa}$ to optimize the whole model (Eq. 12).

$$\mathcal{L}_{ssm} = \mathcal{L}_{anticipation} + \mathcal{L}_{rehearsal} + \mathcal{L}_{binary} \quad (11)$$

$$\mathcal{L} = \mathcal{L}_{qa} + \mathcal{L}_{ssm} \quad (12)$$

Novelty of our proposed method: Although our model builds on the RM setup and model (Zhang et al., 2021) incorporating anticipation along with rehearsal, we introduce important novelties.

RM trains an additional QA model for each downstream task (*history sampler*), using the complete input text to determine the important items to answer a question and then use them to rehearse. A key novelty of our method is that the memory uses coreference-related tokens that can be obtained by general-purpose part-of-speech (POS) taggers, making it task agnostic as we do not need to train additional models. By rehearsing and anticipating this information, we induce the model to softly learn coreference relationships, which is important for discourse comprehension.

In addition, RM updates its memory using single-head cross-attention and a disaggregated GRU network. Instead, our model implements self-attention, cross-attention, and gating in the same memory module, providing better integration of the incoming information with the one in the memory. Finally, our model is light and efficient due to the above features. Our memory module is parallelizable as it does not use a sequential GRU model and does not need to train additional QA models to decide what to rehearse or anticipate.

3 Experiments

3.1 Experimental Settings

Implementation Details: We use PyTorch to implement our model. We set the head number in all attention layers to 8, our decoders have 3 weight-sharing layers and K=20 memory slots. We tie the input embedding and output embedding as it has been demonstrated that it helps to improve language models (Press and Wolf, 2017) and memory networks (Liu et al., 2017). Also, this avoids having multiple output embeddings for the masked and QA modeling, reducing the model size.

We initialize the weights and bias of the gating mechanism using a truncated normal distribution with a mean of 0 and a standard deviation of 0.1, and add a constant of -1 and +1 to the input (Eq. 5) and forget gates (Eq. 6). We use Adam optimizer with a learning rate of 4e-4 to train our model for 300 epochs with a batch size of 128. RAM for short sequence QA has $d_{model} = 128$, and for long sequence QA has $d_{model} = 256$, resulting in 1.2M and 3M parameters, respectively.

For the anticipation and rehearsal masked modeling, we use a POS tagging model to compute coreference-related tokens (nouns, pronouns, and verbs) for all the datasets. We rely on the FLAIR library (Akbik et al., 2019), which provides fast and accurate POS tagging models. We mask coreference-related tokens up to a maximum of 40% of segment tokens, which have been shown to be beneficial (Wettig et al., 2023).

Baselines: We compare our model with very well-known and recent memory networks that incrementally process the input data: DNC (Graves et al., 2016), NUTM (Le et al., 2019), H-Mem (Limbacher and Legenstein, 2020), DAM (Park et al., 2021), STM (Le et al., 2020), CT (Rae et al., 2020), and RM (Zhang et al., 2021). We borrowed the setup of Zhang et al. (2021), in which baselines and memory size were adapted for a fair comparison with our model. Regarding the specific hyperparameters, the NUTM core number was set to 4, the STM query number was set to 8, and the DAM memory block number was set to 2. The CT model uses a 3-layer transformer and a compression ratio of 5. Besides, all the models were set to K=20.

We also include direct methods that solve the task by accessing the entire input. We specify them in the section of each task explored.

Method	Mean Error	Best Error
DNC†	16.70 ± 7.60	3.8
NUTM†	5.60 ± 1.90	3.3
H-Mem	8.93 ± 0.73	7.65
DAM†	1.53 ± 1.33	0.16
STM†	0.39 ± 0.18	0.15
CT†	0.81 ± 0.26	0.34
RM†	0.33 ± 0.15	0.12
RAM	**0.25 ± 0.16**	**0.10**

Table 1: Test error rates (in %) on the 20 bAbI QA tasks for models using 10k training examples. We report mean ± std. and best error over 10 runs. † is reported from Zhang et al. (2021).

3.2 Short Sequence QA

For short sequence QA, we use the bAbI dataset (Weston et al., 2015), a synthetic benchmark widely used to evaluate memory networks. This dataset consists of 20 short-sequence reasoning tasks (less than 100 words) that have to be solved with a common architecture. We use the percent error rate as the metric, which would be the complement of accuracy. Table 1 shows the results. Our model achieves the best result compared to all baselines over 10 runs. Also, RAM has a low variance, being comparable with the most competitive models.

3.3 Long Sequence QA

For long sequence QA, we use NarrativeQA (Kočiský et al., 2018), a dataset with long input contents. It contains about 1,5000 stories and corresponding summaries (more than 600 words). In addition, it includes around 47,000 questions. We adopt the multiple-choice form proposed by the authors. Given a question associated with a summary, the idea is to retrieve the correct answer from a pool of answer candidates drawn from the associated questions. We use mean reciprocal rank (MRR) (Voorhees and Tice, 2000) as the metric to measure how far down the ranking the first relevant answer is. We include two direct models: AS Reader (Kadlec et al., 2016) and E2E-MN (Sukhbaatar et al., 2015) as additional baselines.

Table 2 shows the results for the validation and test sets. Our model achieves 8.46% and 7.38% percentage differences of improvement with respect to the memory baseline RM for validation and test set, respectively. Note that our model also outperforms direct QA methods, being 9.49% and 7.73%, the percentage difference for validation and test set.

Method	Setting	Val	Test
AS Reader†	Direct	26.9	25.9
E2E-MN†	Direct	29.1	28.6
DNC†	Memory	25.8	25.2
NUTM†	Memory	27.7	27.2
HMem	Memory	26.2	25.5
DAM†	Memory	28.1	27.5
STM†	Memory	27.2	26.7
CT†	Memory	28.7	28.3
RM†	Memory	29.4	28.7
RAM	Memory	**32.0**	**30.9**

Table 2: Mean reciprocal rank (in %) on Narrative QA for all the models. † is reported from Zhang et al. (2021).

Method	Setting	Test
E-MN†	Direct	27.9
E-SA†	Direct	31.8
HCRN†	Direct	37.6
DNC†	Memory	30.3
NUTM†	Memory	33.1
HMem	Memory	31.9
DAM†	Memory	32.4
STM†	Memory	33.7
CT†	Memory	35.4
RM†	Memory	36.3
RAM	Memory	**37.4**

Table 3: Accuracy (in %) on ActivityNet-QA for all the models. † is reported from Zhang et al. (2021).

These results demonstrate that incremental memory processing constitutes an effective and efficient approach over direct methods when well-designed.

3.4 Video QA

For video QA, we use ActivityNet-QA (Yu et al., 2019a), which consists of 58,000 QA pairs about 5,800 complex web videos, derived from the popular ActivityNet dataset. We used the same procedure as for the bAbI task to predict the answer. As evaluation metric for the test set, we use accuracy.

As the input stream is video, we adapt our network to make it work with the visual modality. Following the original setup of ActivityNet-QA (Yu et al., 2019a), we use fixed sampling to obtain 20 frames as the sequence that our model will process incrementally. To encode the images and decode the masked tokens, we closely follow the framework proposed by Xie et al. (2022) for masked image modeling. The images of the stream are linearly projected to obtain a sequence of patch embeddings with a 32×32 pixel resolution, to which position embeddings are added.

For the rehearsal and anticipation tasks, we use a learnable mask token vector to replace each masked patch. To decode the masked patches, we use a linear layer to yield patch logits to compute the L1 loss considering only the masked patches. The inputs are images, so we no longer have content words to mask. As an alternative, we propose using the objects in the scene as tokens to mask. Therefore, we use the object detector YOLO (Jocher et al., 2022) to compute the patches of the objects in a frame. We use all the object categories but constrain the masking up to 40% of tokens.

Table 3 shows the test set results. Our model achieves a percentage difference improvement of 2.98% with respect to the model RM and 5.49% with respect to the model CT. Note that our model performs almost comparable to the best direct baseline, HCRN, having only a 0.5% percentage difference of improvement.

4 Further Experimentation

4.1 Ablation Study

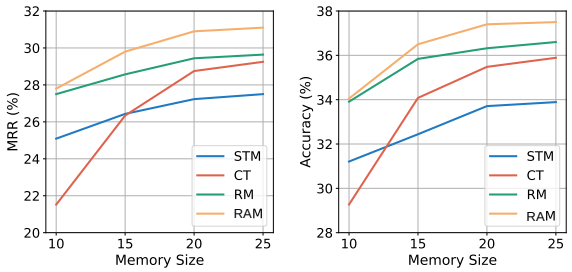
The ablation study allows us to understand which components or configurations impact our model's performance most. We are interested in analyzing the impact of the proposed pretext tasks and masking strategy.

Table 4 shows the four ablation results we performed compared with the default model. First, we found that randomly masking tokens in the masked modeling task slightly affects the results for both bAbI and NarrativeQA. However, for ActivityNet-QA, the impact is negligible, suggesting that random masking is good enough to capture relevant visual information, as shown in (Xie et al., 2022).

Also, we found that when we remove the pretext tasks entirely (w/o $\mathcal{L}_{ssm}$), the performance drops dramatically, resulting in being competitive with the no-self-supervised memory STM. By removing only the anticipation task (w/o $\mathcal{L}_{anticipation}$) or only the rehearsal task (w/o $\mathcal{L}_{rehearsal}$), the performance approaches the RM model. This means that the backbone of our model is more robust, possibly due to the gating mechanism or the fusion module. Note that removing the anticipation task impacts performance more than removing the rehearsal task, but both mechanisms help to boost

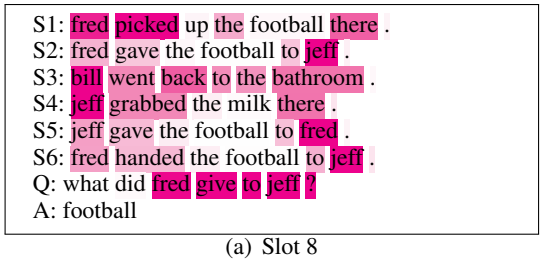
Method	bAbI (Error Rate)	NarrativeQA (MRR)	ActivityNet-QA (Acc)
RAM	0.25 ± 0.16	30.9	37.4
RAM (random masking)	0.28 ± 0.19	30.6	37.4
RAM w/o $\mathcal{L}_{anticipation}$	0.31 ± 0.17	28.9	35.9
RAM w/o $\mathcal{L}_{rehearsal}$	0.34 ± 0.17	29.7	36.4
RAM w/o $\mathcal{L}_{ssm}$	0.37 ± 0.18	28.0	35.0

Table 4: Ablation results for all the datasets. All results are obtained on test sets.

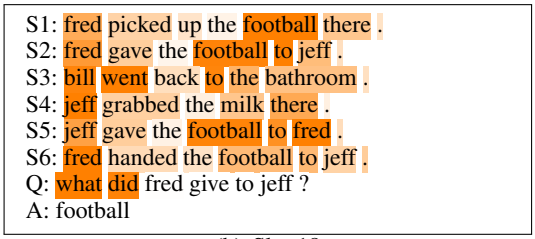


(a) NarrativeQA (b) ActivityNet-QA

Figure 5: Performance of models across different memory sizes for (a) NarrativeQA and (b) ActivityNet-QA.



(a) Slot 8



(b) Slot 18

Figure 6: Attention visualization for a random example of bAbi task 5. Each subfigure shows the intensity of attention from a memory slot to each word of the sentence during reading and from each word of the question to the memory slot. Strongly highlighted tokens mean higher attention weights. The most relevant slots to answer the question are shown: Slot (a) 8 and (b) 18.

performance. This supports that rehearsal is a crucial mechanism for memory networks (Park et al., 2021; Zhang et al., 2021) and confirms that anticipation is a helpful signal to improve memorization.

4.2 Effect of Memory Size

Figure 5 shows the results of our model and some of the more robust memory-based baselines for NarrativeQA and ActivityNet-QA. We evaluate all the models using 10 to 25 memory slots. We do not include the results of the bAbI task because we found that the performance does not vary substantially when changing the memory size of the models.

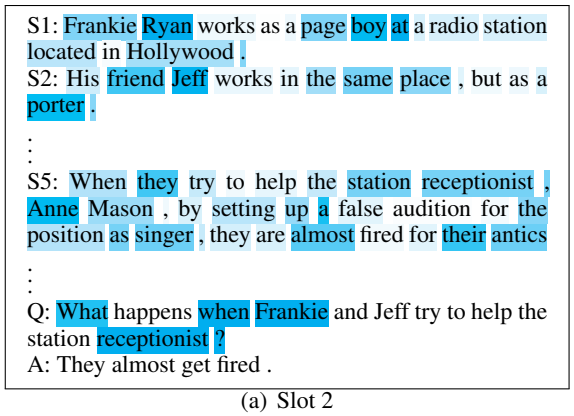
We found that the performance for all models gradually increases with growing memory size. As shown, the CT model is the most affected when the size is reduced, followed by the STM model. The behavior of STM, RM, and RAM is slightly similar but with a difference in performance. We found that our model with low memory is almost equivalent to the performance of RM, but as memory capacity increases, the performance gap also increases. We noticed that memory slots higher than 25 do not show substantial improvements, suggesting that 20-25 slots are adequate for these tasks.

4.3 What is the Attention Capturing?

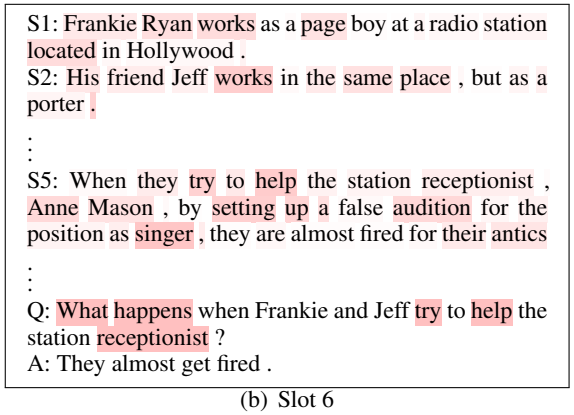
This section aims to reveal what kind of interactions are taking place between the inputs and the memory slots by visualizing the attention weights. Our approach uses cross-attention to both integrate information in the memory and also to retrieve query-relevant information from the memory to obtain the answer. Therefore, we analyze the attention of those layers obtained from random examples.

Figure 6 shows an example from bAbI task 5. In this example, the objective is reasoning over argument relations. We found that 20 slots of memory capture several repeated patterns, so we analyze some of them. We notice that slot 8 is specialized to capture relationships between entities (e.g., fred, jeff), as more attention is seen on entity names and action verbs across all the steps. Besides, the attention from the query to slot 8 was related to the entities and the activity they carried out.

Regarding slot 18, we notice that this slot is specialized to capture the relation between entities and objects. We found that some slots became more



(a) Slot 2



(b) Slot 6

Figure 7: Attention visualization for a random example of NarrativeQA. Each subfigure shows the intensity of attention from a memory slot to each word of the sentence during reading and from each word of the question to the memory slot. Strongly highlighted tokens mean higher attention weights. The most relevant slots to answer the question are shown: Slot (a) 2 and (b) 6. Vertical dots refer to sentences not presented in that space.

specialized to capture specific items, such as this one for the "football" object. We can see that all entities have high attention and the word "football", which suggests that this slot is tracking the state of the football through the steps. Interestingly, we found that the question attention over the slots is very high for the word "what", showing that slot 8 contains information about the football.

Figure 7 shows an example from NarrativeQA. In this case, the example consists of several sentences, so we only include some relevant ones to solve the question. The example shows an outcome question inquiring about the specific result of a particular event. We notice that slot 2 mainly tracks people of the story, for instance, "Ryan", "Jeff" and "Anne". It is also possible to see that attention is paid to the pronoun tokens.

Regarding slot 6, we find that the attention is a bit more distributed across the tokens. However, more attention is paid to verbs (for example, "try" and "help"), which suggests that this slot specializes in events and participants. Note that attention to the question also presents the same behavior.

Figure 8 shows an example from ActivityNet-QA. This example presents a counting task across the steps. In this case, the input stream is images. Our visualizations show bright regions when attention is high. We find a behavior similar to text-based models since some slots specialize in entities or the interaction between them or with objects.

We see that slot 7 is paying attention to the objects and the people around, suggesting that this slot captures the interaction between entities and objects across the steps. Note that slot 7 got the most attention for the words "how many", which supports the hypothesis that this slot tracks entity information.

As for slot 12, we find that it is specialized to capture information about the objects (cups, in this case). The focus is mainly on the cups; however, sometimes, in other places in the image. This suggests that attention tracks the interaction between objects in the scene.

These results support our initial hypothesis that slots could capture information about entities, objects, and relations between them. Also, it is possible to see that our model can track the state of those entities or objects across the steps, suggesting it can represent sequentially coreference-related information when updating the memory.

5 Related Work

5.1 Question Answering

In recent years there has been a significant advance in QA. This is thanks to the new datasets and novel deep learning architectures proposed to solve them. One of the most important datasets is SQuAD (Rajpurkar et al., 2016), which proposes span prediction as the task for QA about a given paragraph. SQuAD has been one of the reasons for progress in the field (Liu et al., 2021), as several attention models have been proposed for this task. For example, BiDAF (Seo et al., 2017) proposes a specialized bi-directional attention mechanism.

Recently, more complex QA tasks were proposed, such as long-range QA (Kočiský et al., 2018; Pang et al., 2022) and commonsense QA (Talmor et al., 2019). Also, other modalities were explored, using images (Johnson et al., 2017; Goyal et al., 2017) or videos (Lei et al., 2018; Yu et al., 2019b)

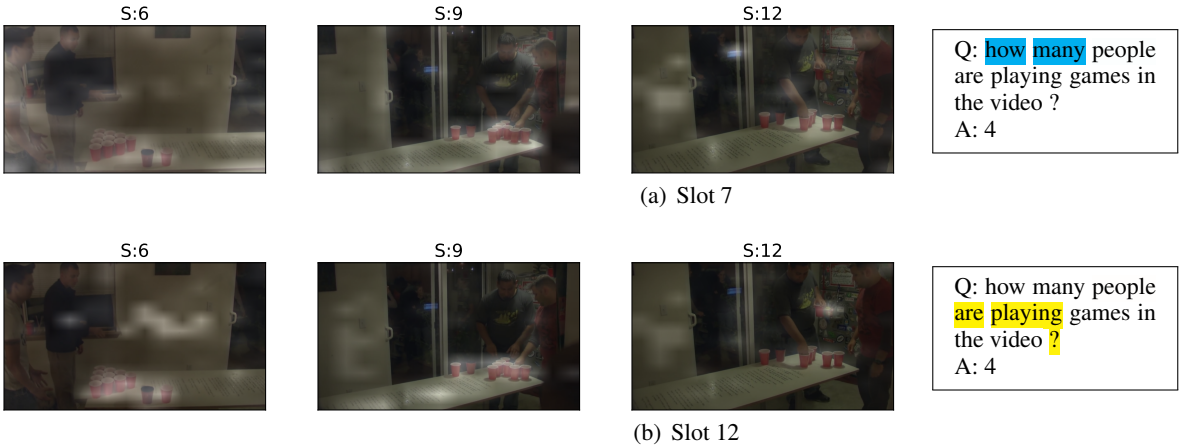


Figure 8: Attention visualization for a random example of ActivityNet-QA. Each subfigure shows the intensity of attention from a memory slot to each patch of the frame during reading and from each word of the question to the memory slot. Brighter regions of images mean higher attention weights. We show the most relevant slots and frames to answer the question: (a) Slot 7 and (b) Slot 12.

as input. For these, large pre-trained models (Devlin et al., 2019) have achieved outstanding results.

The models mentioned above always have access to the whole input. However, this is ineffective when the input is a data stream or a long sequence. For this reason, memory networks have been proposed to perform incremental processing to accumulate knowledge into memory and afterward perform the QA task (Miller et al., 2016; Xiong et al., 2016; Han et al., 2019; Zhang et al., 2021).

5.2 Memory-augmented Neural Networks

MANNs introduce models with external memory to store and access the past contents by differentiable operators. These models attempt to mimic human memory processes, bringing them closer to biological plausibility. NTM (Graves et al., 2014) and DNC (Graves et al., 2016) are well-known memorization and reasoning models which use memory instead of original input to generate inferences.

In this line of research, several papers have proposed new approaches for different NLP tasks, such as language modeling (Santoro et al., 2018), dialogue (Zhang et al., 2019), machine translation (Kaiser et al., 2017), coreference resolution (Liu et al., 2019), question answering (Henaff et al., 2017), and visual question answering (Xiong et al., 2016). These approaches propose new ways of storing and retrieving information (Henaff et al., 2017; Banino et al., 2020), relational or associative mechanisms (Santoro et al., 2018; Limbacher and Legenstein, 2020; Le et al., 2020), and methods to improve memory representation and prevent forgetting (Park et al., 2021; Zhang et al., 2021).

In recent years, some works have been proposed with the latter objective. DAM (Park et al., 2021) introduced a rehearsal task to ensure a current input is well stored by reproducing it using the memory. RM (Zhang et al., 2021) was proposed to tackle the long-term memorization problem, as this model can rehearse a current input and previous ones employing a recollection and familiarity task. Our work extends these approaches and builds on recent findings from neurocognitive science that posit that our memory is guided by rehearsal and anticipation of coreference information, and they might use the same machinery to improve human memorization (Cole et al., 2015). Unlike DAM and RM, our model proposes to perform not only rehearsal but also anticipation using masked modeling and a task to detect whether the information belongs to the future or the past. These mechanisms aim to improve memorization and prevent forgetting.

6 Conclusion

This article presents RAM, a QA memory model supported by two self-supervised pretext tasks to improve memory presentation. It uses masked modeling to mimic the human processes of rehearsal and anticipation of coreference information. Our experiments on short sequence QA, long sequence QA, and video QA show that our model substantially outperforms previous memory models. Our further experimentation demonstrates that the rehearsal and anticipation tasks help to improve memorization and that the model leverages coreference information to support the processes.

Limitations

This work has some limitations regarding the architecture and the data used: Our model assumes that masked modeling could be used as rehearsal and anticipation tasks; however, other approaches could also be effective. We use transformer layers, so our model scalability is tied to the scalability of the transformer model. Also, our approach relies on obtaining additional annotation from pre-trained models for the masking process, so we are limited to the misprediction of those models. We only tested our model with the English language; further exploration of other languages would be valuable to validate the language-independent functionality of the model.

Ethics Statement

Our article presents a novel approach and has not been published in whole or in part elsewhere. The data used to train the model does not imply any violation of privacy. The potential negative social impacts from this work are similar to any other NLP models. Question answering models could potentially be used to create malicious chatbots. This work does not include experimentation with humans or animals.

Acknowledgements

This work was partially funded by European Research Council Advanced Grant 788506, FONDECYT grant 1221425, and National Center for Artificial Intelligence CENIA FB210017, Basal ANID.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Vladimir Araujo, Andrés Villa, Marcelo Mendoza, Marie-Francine Moens, and Alvaro Soto. 2021. Augmenting BERT-style models with predictive coding to improve discourse-level representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3022, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Andrea Banino, Adrià Puigdomènech Badia, Raphael Köster, Martin J. Chadwick, Vinicius Zambaldi, Demis Hassabis, Caswell Barry, Matthew Botvinick, Dharshan Kumaran, and Charles Blundell. 2020. Memo: A deep network for flexible combination of episodic memories. In *International Conference on Learning Representations*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Scott N. Cole, Søren R. Staugaard, and Dorthe Berntsen. 2015. Inducing involuntary and voluntary mental time travel using a laboratory paradigm. *Memory & Cognition*, 44(3):376–389.

Fergus I.M. Craik and Michael J. Watkins. 1973. The role of rehearsal in short-term memory. *Journal of Verbal Learning and Verbal Behavior*, 12(6):599–607.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Aishwarya Kamath, Zhe Gan, Pengchuan Zhang, Jianfeng Wang, Linjie Li, Zicheng Liu, Ce Liu, Yann LeCun, Nanyun Peng, Jianfeng Gao, and Lijuan Wang. 2022. Coarse-to-fine vision-language pre-training with fusion in the backbone. In *NeurIPS*.

Robert L. Goldstone. 1998. Perceptual learning. *Annual Review of Psychology*, 49(1):585–612.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476.

Moonsu Han, Minki Kang, Hyunwoo Jung, and Sung Ju Hwang. 2019. Episodic memory reader: Learning what to remember for question answering from streaming data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4407–4417, Florence, Italy. Association for Computational Linguistics.

Jeff Hawkins and Sandra Blakeslee. 2004. *On intelligence*. Times Books, Westminster, CO.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. In *International Conference on Learning Representations*.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1693–1701, Cambridge, MA, USA. MIT Press.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. 2022. Block-recurrent transformers. In *Advances in Neural Information Processing Systems*.

Evan Jaffe, Cory Shain, and William Schuler. 2020. Coreference information guides human expectations during natural reading. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4587–4599, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. 2022. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 908–918, Berlin, Germany. Association for Computational Linguistics.

Lukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. 2017. Learning to remember rare events. In *International Conference on Learning Representations*.

Frank Keller. 2010. Cognitively plausible models of human language processing. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 60–67, Uppsala, Sweden. Association for Computational Linguistics.

Tomáš Kočiský, 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 Linguistics*, 6:317–328.

Hung Le, Truyen Tran, and Svetha Venkatesh. 2019. Learning to remember more with less memorization. In *International Conference on Learning Representations*.

Hung Le, Truyen Tran, and Svetha Venkatesh. 2020. Self-attentive associative memory. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5682–5691. PMLR.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. 2018. TVQA: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, Brussels, Belgium. Association for Computational Linguistics.

Thomas Limbacher and Robert Legenstein. 2020. H-mem: Harnessing synaptic plasticity with hebbian memory networks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Fei Liu, Trevor Cohn, and Timothy Baldwin. 2017. Improving end-to-end memory networks with unified weight tying. In *Proceedings of the Australasian Language Technology Association Workshop 2017*, pages 16–24, Brisbane, Australia.

Fei Liu, Luke Zettlemoyer, and Jacob Eisenstein. 2019. The referential reader: A recurrent entity network for anaphora resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5925, Florence, Italy. Association for Computational Linguistics.

Nelson F. Liu, Tony Lee, Robin Jia, and Percy Liang. 2021. Do question answering modeling improvements hold across benchmarks?

Kristen L. Mackiewicz, Issidoros Sarinopoulos, Krystal L. Cleven, and Jack B. Nitschke. 2006. The effect of anticipation and the specificity of sex differences for amygdala and hippocampus function in emotional memory. *Proceedings of the National Academy of Sciences*, 103(38):14200–14205.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language*

Processing, pages 1400–1409, Austin, Texas. Association for Computational Linguistics.

Morris Moscovitch, Roberto Cabeza, Gordon Winocur, and Lynn Nadel. 2016. Episodic memory and beyond: The hippocampus and neocortex in transformation. *Annual Review of Psychology*, 67(1):105–134.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel Bowman. 2022. QuALITY: Question answering with long input texts, yes! In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States. Association for Computational Linguistics.

Taewon Park, Inchul Choi, and Minho Lee. 2021. Distributed associative memory network with memory refreshing loss. *Neural Networks*, 144:33–48.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Adam Santoro, Ryan Faulkner, David Raposo, Jack Rae, Mike Chrzanowski, Théophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy Lillicrap. 2018. Relational recurrent neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7310–7321, Red Hook, NY, USA. Curran Associates Inc.

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

Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael K. Tanenhaus, Michael J. Spivey-Knowlton, Kathleen M. Eberhard, and Julie C. Sedivy. 1995. Integration of visual and linguistic information in spoken language comprehension. *Science*, 268(5217):1632–1634.

Matthew J. Traxler, Michael D. Bybee, and Martin J. Pickering. 1997. Influence of connectives on language comprehension: Eye tracking evidence for incremental interpretation. *The Quarterly Journal of Experimental Psychology Section A*, 50(3):481–497.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Nancy C. Waugh and Donald A. Norman. 1965. Primary memory. *Psychological Review*, 72(2):89–104.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2023. Should you mask 15% in masked language modeling? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2985–3000, Dubrovnik, Croatia. Association for Computational Linguistics.

Bianca C. Wittmann, Nico Bunzeck, Raymond J. Dolan, and Emrah Düzel. 2007. Anticipation of novelty recruits reward system and hippocampus while promoting recollection. *NeuroImage*, 38(1):194–202.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. 2022. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9653–9663.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of the*

33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, page 2397–2406. JMLR.org.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019a. ActivityNet-QA: A dataset for understanding complex web videos via question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):9127–9134.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019b. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, pages 9127–9134.

Zheng Zhang, Minlie Huang, Zhongzhou Zhao, Feng Ji, Haiqing Chen, and Xiaoyan Zhu. 2019. Memory-augmented dialogue management for task-oriented dialogue systems. *ACM Trans. Inf. Syst.*, 37(3).

Zhu Zhang, Chang Zhou, Jianxin Ma, Zhijie Lin, Jingren Zhou, Hongxia Yang, and Zhou Zhao. 2021. Learning to rehearse in long sequence memorization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12663–12673. PMLR.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Left blank.

☑ A2. Did you discuss any potential risks of your work?
Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Left blank.

☑ B1. Did you cite the creators of artifacts you used?
Left blank.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☒ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not reported for all the experiments.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.