Aging Memories Generate More Fluent Dialogue Responses with Memory Networks

Omar U. Florez and Erik Mueller
Conversational AI Research
Capital One
omar.florezchoque@capitalone.com

Abstract

The integration of a Knowledge Base (KB) into a neural dialogue agent is one of the key challenges in Conversational AI. Memory networks has proven to be effective to encode KB information into an external memory to thus generate more fluent and informed responses. Unfortunately, such memory becomes full of latent representations during training, so the most common strategy is to overwrite old memory entries randomly.

In this paper, we question this approach and provide experimental evidence showing that conventional memory networks generate many redundant latent vectors resulting in overfitting and the need for larger memories. We introduce *memory dropout* as an automatic technique that encourages diversity in the latent space by 1) Aging redundant memories to increase their probability of being overwritten during training 2) Sampling new memories that summarize the knowledge acquired by redundant memories. This technique allows us to incorporate Knowledge Bases to achieve state-of-the-art dialogue generation in the Stanford Multi-Turn Dialogue dataset. Considering the same architecture, its use provides an improvement of +2.2 BLEU points for the automatic generation of responses and an increase of +8.1% in the recognition of named entities.

1 Introduction

Given the large amount of dialogue data recorded in human-human or human-chatbot interactions, there is a great need for dialogue systems that infer automatic responses grounded to personal knowledge bases. This approach has the advantage of integrating semantic information that is fundamental to achieve dialogue understanding. We want to leverage the contextual information present in a KB (e.g., a calendar of events) to answer queries like *What time is my dentist appointment?*. This task is challenging because existing neural dialogue agents often assume that the dialogue history carries the information needed to provide an answer but struggle to interface with the structured data stored in a KB. This issue breaks end-to-end differentiability when training the model and limits the kind of contextual conversations that people desire.

Memory networks (Miller et al. (2016)) has proven to be effective to encode KB information into an external memory to generate more fluent and informed responses. However, there is no much work in regularizing the latent representations stored in the external memory. Unlike the conventional *dropout* technique used to regularize deep neural networks (Srivastava et al. (2014)), we propose *memory dropout* to attain the same goal (i.e., reduction of overfitting) but with different functionality and designed for memory networks (Weston et al. (2015)). Given the long-term nature of memory networks, we do not ***immediately*** remove redundant memories with some probability as in the original dropout algorithm. Instead, we assign them the current maximum age increasing their probability of being overwritten by more recent latent representations in future training steps. Thus, in contrast to Srivastava et al. (2014), our *memory dropout* is a ***delayed*** regularization mechanism.

The main contributions of our work are the following:

- We introduce a new regularization method called *memory dropout* designed for dealing with overfitting in Memory Augmented Neural Networks. To our best knowledge, ours is the first work on regularizing memory networks.
- We build a neural dialogue agent that uses *memory dropout* to incorporate KB into an external memory for automatic response generation. Our results show that this technique can generate more fluent and accurate responses: an improvement of +2.2 BLUE points and +8.1% Entity F1 score versus not using it in the Stanford Multi-Turn Dialogue dataset.

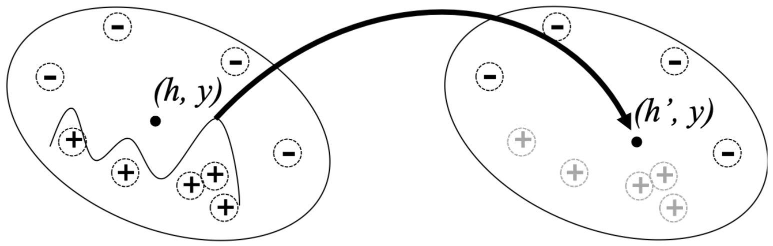


Figure 1: Learning the (h, y) pair transitions the neighborhood of h (represented as an ellipse) to a new state in which a memory h' is drawn as the distribution of positive memories. Small circles represent the uncertainty of using a particular memory to model h'. In the new memory configuration, we age positive keys (now faded in grey) making them more likely of being overwritten by other training examples.

2 Memory Dropout

This section describes the *memory dropout* neural model whose goal is to increase the diversity of latent representations stored in an external memory. For example, consider a neural encoder which receives an observation (x, y) and generates the latent representation h in a hidden layer. As illustrated in Figure 1, we want to incorporate a normalized h (i.e., $\|h\| = 1$) in the long-term memory M. The most similar memory entries to h form a neighborhood in which entries can be positive (+) or negative (-) depending on whether they share the same class label y or not.

An external memory M augments the capacity of a neural encoder by preserving long-term latent representations. Figure 2 illustrates a memory network which consists of arrays K and V to store *keys* (latent representations) and *values* (class labels), respectively, as introduced in Kaiser et al. (2017). To support our technique, we extend this definition with arrays A and S to store the *age* and the *variance* of each key, respectively. The final form of the memory module is as follows:

$$M = (K, V, A, S).$$

Our main goal is to learn a mathematical space in which the margin between positive and negative memories is maximum while retaining the minimum number of positive keys. Core to this idea is the definition of a differentiable *Gaussian Mixture Model* parameterized by both the location and covariance matrix of each positive memory. Sampling from this distribution returns a new positive embedding h' that characterizes its neighbors. So, given the embedding vector h, the collection of P positive keys $K^+ = \{k_1^+, ..., k_P^+\}$ is a subpopulation of the memory keys that can be represented as a linear superposition of P *Gaussian components* aimed at providing a rich class of a density model in the form

$$p(k^+) = \sum_{p=1}^{P} \pi_p \mathcal{N}(k^+ | \mu_p, \Sigma_p) \quad (1)$$

where each Gaussian is centered at a positive key $\mu_p = k_p^+$ with covariance matrix $\Sigma_p = diag(s_p^+)$. Note that without the variances stored in the array S, the uncertainty of each key will be the same and extreme embedding vectors will dominate the likelihood probability. We shall view $\pi = \{\pi_1, ..., \pi_P\}$ in Equation 1 as a vector of probabilities that quantifies the mixing coefficients of the Gaussian components and whose values correspond to the similarity between h and the positive keys K^+

$$\pi = Softmax(h \cdot K^+) \quad \text{such that} \quad 0 \le \pi_p \le 1 \text{ and } \sum_{p=1}^{P} \pi_p = 1. \tag{2}$$

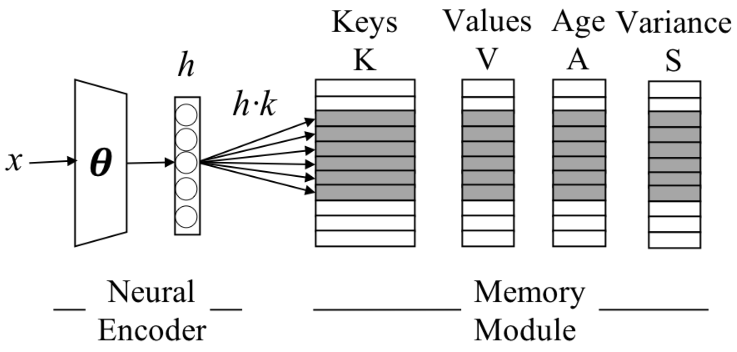


Figure 2: A memory network consists of a neural encoder and an external memory that augments its capacity by preserving longer distinct versions of h during training. Some memory entries (in gray) are positive candidates to correctly answer to the embedding h.

The conditional distribution of a new key k' given a particular Gaussian is

$$p(k'|\pi_{p=i}) = \mathcal{N}(k'|\mu_i, \Sigma_i),$$

so sampling an index i over the P mixture components under the distribution π generates the new key k' as a random variable from $p(k'|\pi_{p=i})$. As being sampled from the mixture model, k' is representative of the subpopulation of positive memories. Then, we incorporate to the external memory the information encoded by the latent vector h to the new key k', reset its corresponding age, and compute its variance to account for the observed uncertainty in approximating h with k'.

$$K[i] = \frac{k' + h}{||k' + h||} \quad V[i] = y \quad A[i] = 0 \quad S[i] = (h - k')^2 \quad A[i^+] := max\{A\}$$

Finally, the *aging mechanism* penalizes redundant keys indexed by i^+ as follows, $A[i^+] = max\{A\}$.

3 Using the Memory Dropout

We study the *memory dropout* neural model in a realistic scenario: As the external memory of a dialogue system that infers automatic responses grounded to a Knowledge Base (KB). Our goal is to leverage the contextual information present in a KB (e.g., a calendar of events) to answer queries like *What time is my dentist appointment?*. This task is challenging because existing neural dialogue agents often assume that the dialogue history carries the information needed to provide an answer but struggle to interface with the structured data stored in a KB. This assumption prevents to have an end-to-end differentiable model to maintain the kind of flexible conversations that people desire. Figure 3 illustrates our proposed architecture formed by a Sequence-to-Sequence model to represent the dialogue history and a Memory Augmented Neural Network (MANN) to encode the KB.

To encode the KB, the addressable memory entries of a Memory Network allows us to generalize with fewer latent representations of the KB, even if they were present once during training (Kaiser et al. (2017)). Inspired by Miller et al. (2016), we store the KB of a given dialogue by decomposing it from its tabular format to a collection of triplets that expresses the following relationship

(subject, relation, object).

For example, an entry in the KB representing a dentist appointment

(**event**=*dentist*, **date**=*the 19th*, **time**=*5pm*, **party**=*Mike*)

would be normalized into 12 different triplets:

[*(dentist, date, the 19th)*, *(dentist, time, 5pm)*, *(dentist, party, Mike)*, *(Mike, time, 5pm)*, *(dentist, date, the 19th)*, *(dentist, time, 5pm)*, *(dentist, party, Mike)*, *(the 19th, event, dentist)*, *(the 19th, party, Mike)*, *(the 19th, time, 5pm)*, *(5pm, event, dentist)*, *(5pm, date, the 19th)*, *(5pm, party, Mike)*, *(Mike, event, dentist)*, *(Mike, date, the 19th)*, *(Mike, time, 5pm)*].

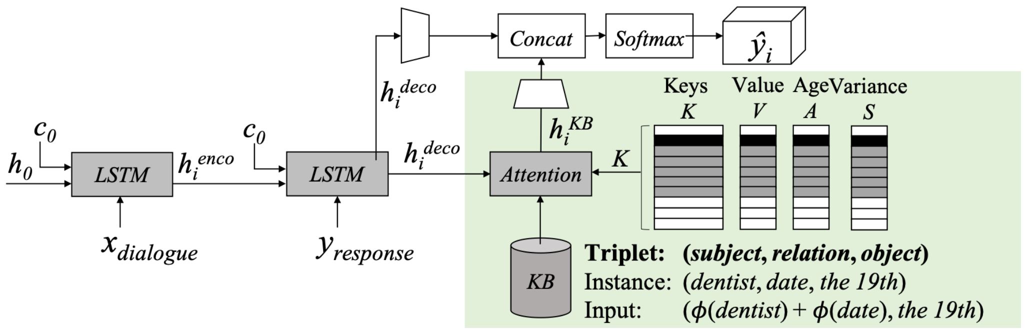


Figure 3: Architecture of the neural dialogue model that incorporates a KB. Note the i^{th} decoding step of the word $\hat{y}_i$ given the attention over the external memory which encodes KB triplets in its keys and that uses *memory dropout* for model regularization.

Each triplet feeds the MANN with the following key-value format

$$\langle \left\| \phi^{emb}(\textbf{\textit{subject}}) + \phi^{emb}(\textbf{\textit{relation}}) \right\| , \textbf{\textit{object}} \rangle$$

where ϕ^{emb} is a trainable embedding function that maps input tokens to a fixed-dimensional vector.

To encode the dialogue, we use an encoder-decoder network architecture (Sutskever et al. (2014)) with an LSTM encoder that outputs a context-sensitive hidden representation h^{enco} based on the dialogue history and an LSTM decoder that generates the next response $\hat{y}$. At every timestep of decoding, the decoder predicts the i^{th} token of the response $\hat{y}$ by computing its corresponding hidden state h_i^{deco} applying the recurrence

$$h_i^{deco} = LSTM(\hat{y}_{i-1}, h_{i-1}^{deco}).$$

Instead of directly using the decoder output h^{deco} to predict over the dialogue vocabulary as in Miller et al. (2016), we combine h^{deco} with the result of querying the memory module. We compute this vector, h_i^{KB}, as the additive attention score between h^{deco} and each *key*. This operation results into the unnormalized probabilities of including its corresponding *value* in the prediction. More formally,

$$\begin{aligned} h_i^{KB} &= W_2 \, tanh(W_1 \, [h_i^{deco}; K]) \\ o_i &= [W_{vocab\_dlg} \, h_i^{deco}; W_{vocab\_KB} \, h_i^{KB}] \\ \hat{y} &= Softmax(o_i) \\ \hat{y}_i &= T[argmax_j \hat{y}[j]] \end{aligned}$$

where W_1, W_2, W_{vocab_dlg}, and W_{vocab_KB} are trainable parameters. Here, o_i represents the concatenation of an extended vocabulary T formed by the logits over the dialogue tokens and the logits over the distinct values stored in the value array V. A Softmax induces a probability distribution over the extended vocabulary and we argmax to obtain the index of the predicted token. Naturally, the objective function is to minimize the cross entropy between the actual and generated responses:

$$J(\theta) = -\sum_{j=1}^{N} \sum_{k=1}^{L_j} \sum_{i=1}^{m} y_i^{j,k} \, log \, p(\hat{y}_i^{j,k})$$

where N is the number of dialogues, L_j is the number of turns in the j^{th} dialogue, m is the length of the generated response, and $y_i^{j,k}$ is the one-hot encoding of the i^{th} word in the actual response.

MODEL	BLEU	ENT.F1	NAVIG.ENT.F1	WEATHER ENT.F1	SCHEDUL.ENT.F1
SEQ2SEQ+ATT.	10.2	30.0%	17.9%	42.4%	30.0%
KVRN	13.2	48.0%	41.3%	47.0 %	**62.9%**
MANN	11.2	50.3%	42.6%	46.8%	51.0%
MANN+MD	**13.4**	**58.4%**	**49.3%**	**53.7%**	60.6%

Table 1: Averaged and per-domain BLEU and Entity F1 scores for the SMTD dataset.

4 Experiments

We evaluate our proposed method in the *Stanford Multi-Turn Dialogue* (SMTD) dataset (Eric et al. (2017)). This dataset consists of 3,031 dialogues in the domain of an in-car assistant, which provides automatic responses grounded to a personalized KB, known only to the in-car assistant. The entries of the KB may contain information for satisfying a query formulated by the driver. There are three types of KBs: a *schedule of events*, the weekly *weather forecast*, and information for point-of-interest *navigation*.

4.1 Baselines

We compare our approach **Memory Augmented Neural Network with Memory Dropout** (MANN+MD) with the following baseline models:

- **Seq2Seq+Attention** (Bahdanau et al. (2015)): An encoder-decoder architecture that maps between sequences with minimal assumptions on the sequence structure and attends to parts of the input to predict the target word. This approach only incorporates information from the dialogue history.
- **Key-Value Retrieval Network+Attention** (Eric et al. (2017)) (KVRN): A memory network with not update memory operations, it also computes attention over the keys of each entry in the KB, and
- **Memory Augmented Neural Network** (MANN): Our proposed model with no *memory dropout* mechanism.

4.2 Training Settings

For all the experiments, we use a word embedding of size 256. The encoders and decoders are bidirectional LSTMs of 3-layers with state size of 256 for each direction. For the memory network models, the number of memory entries is 1,000. We train all models with Adam optimizer (Kingma & Ba (2014)) with a learning rate of 0.001 and initialized all weights from a uniform distribution in $[-0.01, 0.01]$. We also applied dropout (Srivastava et al. (2014)) with keep probability of 95.0% for input and output of recurrent neural networks. Finally, we split the dataset into partitions with ratios 0.8, 0.1, and 0.1 for generating the training, validation, and testing datasets, respectively.

4.3 Results on System Response Generation

Evaluating dialogue systems is challenging because a trained model can generate free-form responses. As in Eric et al. (2017), we also employ two metrics to quantify the performance of a model grounded to a knowledge base.

- **BLEU** (Papineni et al. (2002)) is a metric of fluency that looks at n-gram precisions for $n = 1, 2, 3, 4$ comparing between exact matches of words between responses provided by a model and human annotators.
- **Entity F1** measures the correct retrieval of entities expected to be present in a generated response.

We found that *memory dropout* improves both dialogue fluency and accuracy in recognizing entities compared to other memory networks that did not use *memory dropout*. Table 1 reports averaged and per-domain results on this dataset.

Not attending to the KB seems to have an adverse result in generating automatic responses. For example, the Seq2Seq+Attention model shows the lowest Entity F1 scores (30.0%), indicating its inability to infer responses while being agnostic to the KB. The memory network MANN attends to the KB to predict a response and achieves a BLEU score of 11.2 and an Entity F1 score of 50.3%. With *memory dropout*, the MANN+MD increases both BLEU and Entity F1 scores to 13.4 and 58.4%, respectively. Note that both MANN and MANN+MD share the same architecture and hyperparameters and only differ in the use of *memory dropout*.

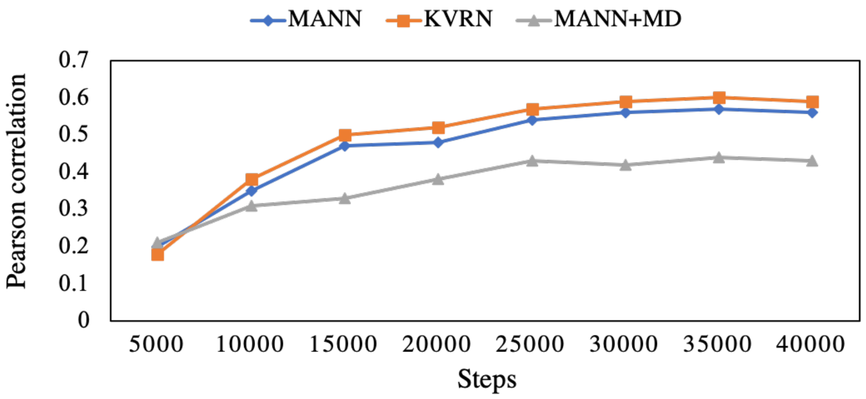


Figure 4: Average Pearson correlation values between pairs of keys at different training steps. The higher, the more correlation, and the maximum value is 1.0. As training progresses, redundant keys may be allocated to the external memory.

On the other hand, KVRN also attends to the KB and it is the best performing neural network that does not use *memory dropout*. This method achieves a BLEU score of 13.2 and Entity F1 score of 48.0%. Our approach outperforms KVRN by $+10.4\%$ in the Entity F1 score and provides slight positive advantage for the BLEU score $+0.2$ setting a new SOTA for this dataset. Interestingly, KVRN only outperforms the other models in the Scheduling Entity F1 domain with 62.9%. This can be because half of these dialogues are commands to execute a task and thus do not require of a KB (e.g., *Set a meeting reminder at 3pm*).

The explicit penalization of redundant keys during training could explain the gains obtained by MANN+MD. In order to test this hypothesis, we study now the correlation of keys in Section 4.4.

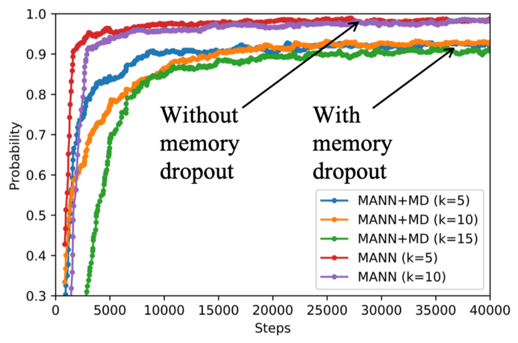
4.4 Results on Correlated Memories

We found that keys in memory tend to become redundant as training progresses. To observe this effect, for each training step we compute the Pearson correlation between each pair of keys and average these values. Figure 4 compare the degree of linear correlations in the memory networks studied in this paper. Comparing the results, we can see that initially all models show low correlations as keys are randomly initialized. In following steps, both MANN and KVRN show increasing correlation values indicating that more redundant keys are stored in memory over time. In contrast, MANN+MD shows low correlation values which do not increase at the same rate than the other methods and reach stable values around step 25,000. We conclude that using *memory dropout* explicitly encourages the overwriting of redundant keys which lead to diverse representations in the latent space.

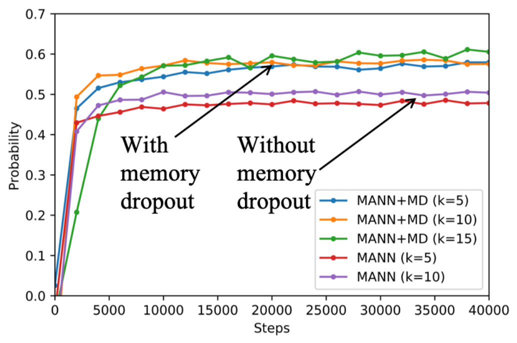
4.5 Results on Overfitting Reduction

In order to test the advantage of using *memory dropout* for overfitting reduction, 1) We compare the Entity F1 scores for the MANN and MANN+MD models during training considering different neighborhood sizes, and 2) We disable traditional *dropout* for the inputs and outputs of the encoder and decoder in an attempt to isolate the contribution of *memory dropout*.

Figure 5 shows two groups of behavior in each plot. During training, no using *memory dropout* (MANN) leads to obtain higher Entity F1 scores, a reasonable outcome considering that not regularizer is present and the external memory increases memory capacity. During testing, MANN shows lower



(a) Training Entity F1



(b) Testing Entity F1

Figure 5: Entity F1 scores considering the use of *memory dropout* in the MANN model.

Entity F1 scores. This is a sign of a model that overfits to its training dataset and have problems to generalize in the testing dataset. On the other hand, using *memory dropout* (MANN+MD) provides a more conservative performance during training but better Entity F1 scores during testing resulting into an average improvement of 10%. Testing with different neighborhood sizes (from 5 to 15 elements) shows in general the same behavior: two groups of well-defined Entity F1 scores grouped by whether they use the memory dropout technique or not.

4.6 Results on the Effect of Memory Size

To test the usefulness of large memories when encoding a KB with *memory dropout*, we compare the models that use an external memory and consider different memory sizes. The task is to compute the Entity F1 score during the automatic response generation.

Figure 6 compares the performance of memory networks. MANN outperforms the KVRN method across different memory sizes. Both methods show higher scores as we increase the memory size until generating overfitting at a particular memory size which translates into lower accuracy values. Adding memory dropout (MANN+MD) increases the Entity F1 scores showing the most accurate results across all memory sizes.

We found that a side-effect of using an external memory is the need for larger memories to accommodate the large number of redundant activations generated during training. As seen in the experiments of Section 4.4 (memory correlations) and Section 4.6 (memory size), using *memory dropout* leads to the storage of diverse keys and therefore we can use smaller memories to obtain higher levels of accuracy in this dataset.

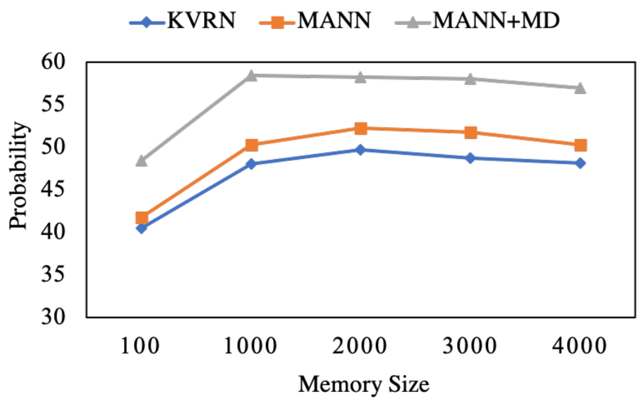


Figure 6: Entity F1 score of the testing dataset considering different memory sizes.

5 Related Work

Deep Neural Networks are models to solve classification tasks that involve non-linearly separable classes. Memory networks consider an external differentiable memory in which a neural encoder manages the hidden states of the model using attention to address similar content in memory (Bahdanau et al. (2014)). Some similar approaches have also studied the problem of few-shot learning as a way to remember infrequent patters, a problem that we also observed when training with small knowledge bases. Some representative examples include (Kaiser et al. (2017)) which also uses an external memory to extend the capacity of the entire architecture. Also, Neural Turing Machines (NTM) (**?**) are differentiable architectures allowing efficient training with gradient descend and showing important properties for associative recall for learning different sequential patterns. In this paper, we extend the key-value architecture introduced in (Kaiser et al. (2017)) because of its simplicity and also it has shown to be effective in learning small datasets in text and visual domains.

Deep models have also been used to train in dialogue agents. Often they model the dialogue state considering belief tracking and generation components (Rojas-Barahona et al. (2017)). More recent architectures consider the use of a knowledge base and an external memory to encode its content (Eric et al. (2017)). Although the key-value architecture of this system allows for incorporating domain-specific knowledge with no need of dialogue state trackers, it overfits to the training dataset impacting the accuracy and fluent generation of responses. Our model contrasts with this work on designing a a memory augmented model that deals directly with overfitting and require smaller memory size, as shown in our experiments.

The regularization of neural networks is also an important problem which have proven to be effective to control overfitting and generate sparse activations during training. Recently, Wang & Niepert (2019) proposes the regularization of state transitions in a recurrent neural network, but the notion of memory is still internal and individual memories cannot be addressed. The popular dropout technique works in hidden layers as a form of model averaging (Srivastava et al. (2014)). In contrast to Srivastava et al. (2014), our *memory dropout* is a delayed (age-sensitive) regularization mechanism that works at the level of memory entries and not individual activations. To the best of our knowledge, our is the first work that addresses the regularization of memory networks and prove its effectivity in a challenging task such as automatic dialogue response.

6 Conclusions

Memory Dropout is a technique for improving memory augmented neural networks by breaking co-adapting memories built during backpropagation. While conventional dropout works at the level of individual activations, our *memory dropout* deals with latent representations of the input. These arrays of activations are stored into an external memory module which resembles areas of the human brain that are content-addressable and sensitive to semantic information (Wixted et al. (2018)). Central to this technique is the idea that age and uncertainty play important roles to regularize the addressable keys of an external memory module that is persistent across training examples. By doing this, we obtain higher BLEU and Entity F1 scores when training a task-oriented dialogue agent that decodes an answer considering the entries of KB stored in the memory module.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*. Association for Computational Linguistics, 2017.

Lukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. 2017. URL `https://openreview.net/pdf?id=SJTQLdqlg`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. CoRR, 2014.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1409, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1147. URL `https://www.aclweb.org/anthology/D16-1147`.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `https://www.aclweb.org/anthology/P02-1040`.

Lina Maria Rojas-Barahona, Milica Gasic, Nikola Mrksic, Pei-Hao Su, Stefan Ultes, Tsung-Hsien Wen, Steve J. Young, and David Vandyke. A network-based end-to-end trainable task-oriented dialogue system. Association for Computational Linguistics, 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1): 1929–1958, January 2014. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=2627435.2670313`.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 3104–3112. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf`.

Cheng Wang and Mathias Niepert. State-regularized recurrent neural networks. 2019. URL `https://arxiv.org/pdf/1901.08817.pdf`.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1410.3916`.

John T. Wixted, Stephen D. Goldinger, Larry R. Squire, Joel R. Kuhn, Megan H. Papesh, Kris A. Smith, David M. Treiman, and Peter N. Steinmetz. Coding of episodic memory in the human hippocampus. *Proceedings of the National Academy of Sciences*, 2018. doi: 10.1073/pnas.1716443115.