LEARNING ASSOCIATIVE INFERENCE USING FAST WEIGHT MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Humans possess the ability to quickly associate recent stimuli in order to deduce the correct response in a novel context. Here we facilitate a neural network model to learn state representations of facts that can be composed together to perform such associative inference. To this end, we augment the LSTM model with an associative memory, dubbed *Fast Weight Memory* (FWM). Through differentiable operations at every step of a given input sequence, the LSTM updates and maintains compositional associations stored in the rapidly changing FWM weights. Our model is trained end-to-end by gradient descent and yields excellent performance on compositional language reasoning problems, meta-reinforcement-learning for POMDPs, and small-scale word-level language modelling.[1]

1 INTRODUCTION

Humans continually adapt in order to understand new situations in changing environments. One important adaptive ability is *associative inference* for composing features extracted from distinct experiences and relating them to each other (Schlichting & Preston, 2015; Gershman et al., 2015). Suppose Alice has shared with you pictures of her toddler. Later, at the office party, you see a man carrying the depicted toddler. Since the toddler yields a shared feature in two different contexts, it may be plausible to infer that the man is Alice's partner, without ever seeing him and Alice together. The ability to rapidly associate and bind together novel stimuli can help to derive knowledge systematically, in addition to the knowledge gained directly from observation.

Virtually all modern cognitive architectures applied to challenging artificial intelligence problems are based on deep artificial neural networks (NNs). Despite their empirical successes and theoretical generality, NNs tend to struggle to generalise in situations similar to the given example (Lake et al., 2017; Phillips, 1995; Lake & Baroni, 2017). This weakness becomes even more severe if the training and test data exhibit systematic differences (Atzmon et al., 2016; Agrawal et al., 2017). For example, during training, the man's representation might never be associated with the toddler's, but during testing this association might be necessary in order to predict the correct answer. In problems where humans excel, this sort of inference is likely ubiquitous since the data is often combinatorially complex such that observations used during training will likely cover just a small fraction of all possible combinations. Such a lack of productivity and systematicity is a long-standing critique of NNs as a substrate of a cognitive architecture (Fodor & Pylyshyn, 1988; Hadley, 1994; McLaughlin, 2009).

The (hidden) state of a neural model is a learned representation of the task-relevant information extracted from the input. In order to generalise to never-seen-before compositions of stimuli, the function which produces the state representation has to be capable of the systematic construction of *all possible* states. This requires a general and differentiable method such as the Tensor Product Representation (TPR; Smolensky (1990)). TPRs provide a general and differentiable method for embedding symbolic structures in vector spaces. A TPR state representation is constructed via the tensor product (i.e. the generalised outer-product) of learned component representations. Under certain constraints, such a mechanism guarantees a unique representation for every possible combination of components (Smolensky, 1990; 2012).

In this work, we augment a recurrent NN (RNN) with an additional TPR-like memory representation. To facilitate the learning of multi-step associative inference, the TPR memory can be queried

[1] Source code and data used in this paper is available at *REDACTED*

multiple times which allows the model to chain together various independent associations. In contrast to previous work, we apply our memory-augmented RNN to much longer sequences which requires the model to update deprecated associations. Furthermore, we demonstrate the generality of our method by applying it to a meta reinforcement learning and small scale language modelling.

In the next section, we cover related memory-augmented NNs. Section 3 describes the FWM in detail. Section 4 demonstrates the generality of our method through experiments in supervised, self-supervised, and meta-reinforcement-learning contexts. The supervised-learning experiments in subsection 4.1 consist of a more challenging version of the bAbI dataset dubbed concatenated-bAbI or catbAbI. The meta reinforcement learning experiment in section 4.2 demonstrates the FWM's ability to learn to explore a partially observable environment through its ability to perform associative inference. Finally, the self-supervised experiments in subsection 4.3 demonstrate that the FWM can compete with the state-of-the-art word-level language models on small benchmark datasets.

2 Related Work

RNNs such as the Long Short-Term Memory (LSTM; Hochreiter & Schmidhuber (1997); Gers et al. (2000)) are in theory capable of implementing any algorithm (Siegelmann & Sontag, 1991). However, linear growth of the hidden state of a fully connected RNN leads to quadratic growth in the number of trainable weights. Early work addressed this issue through the use of additional memory (Das et al., 1992; Mozer & Das, 1993) and differentiable fast weights (Schmidhuber, 1992; 1993). Recently, memory-augmented NNs have solved algorithmic toy problems (Graves et al., 2014; 2016) as well as reasoning and inference problems in synthetic and natural language (Weston et al., 2015b; Xiong et al., 2016).

Inspired by the random-access memory of computer architectures, a common approach is to incorporate a soft and differentiable lookup table into the NN model. Such slot-based memory matrices have shown to be difficult to train (Munkhdalai & Yu, 2017b) and require sophisticated mechanisms for the allocation and deallocation of memory (Csordas & Schmidhuber, 2019). The Transformer-XL (TXL; Dai et al. (2019)), a recent autoregressive variant of the Transformer (Vaswani et al., 2017), can be understood as a slot-based memory-augmented RNN where every new state is pushed into an immutable queue of finite size. Although it is recurrent, the layers of a transformer architecture are strictly forced to use inputs from a lower layer which limits its generality. Nevertheless, a sufficiently deep and well regularised TXL model has achieved state-of-the-art performance in large scale language modelling tasks.

An biologically more plausible alternative of increasing the memory capacity of NNs are fast-changing weights, i.e. statful weights that can adapt as a function of its input. Non-differentiable fast weights or "dynamic links" have been published since 1981 (von der Malsburg, 1981; Feldman, 1982; Hinton & Plaut, 1987). Subsequent work showed that a regular network can be trained by gradient descent to control the fast weights of a separate network (Schmidhuber, 1992) or of itself (Schmidhuber, 1993) in an end-to-end differentiable fashion. Recently, fast weights have made a comeback and achieved good results in small toy problems where regular NNs fall short (Ba et al., 2016a; Schlag & Schmidhuber, 2017; Munkhdalai & Yu, 2017a; Pritzel et al., 2017; Ha et al., 2017; Zhang & Zhou, 2017; Miconi et al., 2018; 2019; Schlag & Schmidhuber, 2018; Munkhdalai et al., 2019; Bartunov et al., 2020).

Most memory-augmented NNs are based a content-based or key-based lookup mechanisms. An alternative to the storage of patterns in a lookup table is the idea that patterns are reconstructed through the implicit iterative minimisation of an energy function, such as in the classical Hopfield network (Willshaw et al., 1969; Hopfield, 1982; Kanerva, 1988) or the modern Hopfield network (Demircigil et al., 2017; Ramsauer et al., 2020). This is is often described as an *auto-associative* type of memory as it reconstructs a previously stored pattern that mostly resembles the current pattern. A much less studied variation is the *hetero-associative* memory (e.g. Kosko (1988)) where the retrieved pattern is is different from the input which is more relevant in our case. An increased difficulty of our work, is to train an LSTM to construct, maintain, and *edit* its associative memory. The ability to edit Hopfield networks is not very well studied as it is not a common use case. For this reason, we use a simple linear hetero-associative memory as it is most closely related to the theory of TPRs and because the output is produced in a single forward computation.

Our work directly builds on two examples of differentiable fast weight memories: the TPR-RNN by Schlag & Schmidhuber (2018) and the Metalearned Neural Memory (MNM) by Munkhdalai et al. (2019). The TPR-RNN is a sentence-level model for reasoning on text. It achieves excellent results on the regular bAbI tasks but performs poorly in the word-level or on other problems (Le et al., 2020). In contrast, the MNM is a word-level model which augments the LSTM with a fully-connected multi-layer feed-forward network as its memory and trains it using a meta-learning objective. Both, MNM and TPR-RNN were developed on the regular bAbI tasks which limits the problem to short sequences (on average X words long). In our experiments we train on an infinite sequence of bAbI stories where we observe MNM to eventually fail to converge or result in NaNs whereas our FWM maintains good performance.

3 Proposed Method

Our FWM is a fast-changing, multi-linear map which is controlled by a slowly-changing, non-linear LSTM. The slow weights of the LSTM are regular NN weights which are updated during training by gradient descent. In contrast, the fast weights of the FWM are updated by the LSTM at every step of the input sequence through a Hebb-like differentiable mechanism. This allows the FWM function to change rapidly even during testing—hence the name *fast* weights. Along with updating the fast weights, the LSTM also generates a FWM query which is used to retrieve information that was previously stored. The retrieved information then becomes a part of the models output.

3.1 The Fast Weight Memory

Given a sequence of tokens $\mathbf{x} = (x_1, ..., x_T)$ from a vocabulary $\mathbb{V}$, the task of language modeling is to train a model which maximizes the joint probability $p(\mathbf{x})$ which is often factorized autoregressively as $p(\mathbf{x}) = p(x_1)\prod_{t=1}^{T-1} p(x_{t+1}|\mathbf{x}_{1:t})$.[2] In this work, we train an RNN model to encode the input sequence $\mathbf{x}_{1:t}$ into $\boldsymbol{h}_t$, the hidden state of the LSTM, and $\boldsymbol{F}_t$, the fast weight tensor of the FWM, to maximize the probability of the next token x_{t+1}.

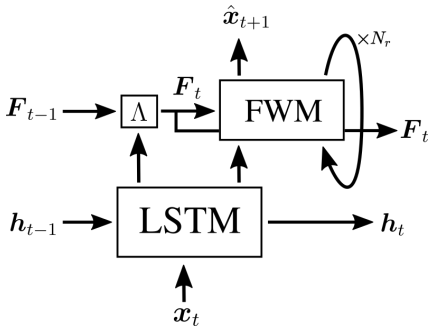

Figure 1: A simplified illustration of the our proposed method where Λ refers to the write mechanism described in section 3.1.2. $\boldsymbol{F}_t$ are the recurrent weights of the FWM which have been generated by the LSTM. The LSTM is a regular slow RNN.

At step t of the input sequence, the input token x_t is embedded in a d_E-dimensional vector space using a lookup table $\boldsymbol{e}_t = \text{embedding}(x_t)$. An LSTM with d_{LSTM} hidden units encodes the sequence of embedded tokens into a fixed size vector representation $\boldsymbol{h}_t = \text{LSTM}(\boldsymbol{e}_t, \boldsymbol{h}_{t-1})$. The probability distribution over the next token $\hat{\boldsymbol{x}}_{t+1} = \text{softmax}(\boldsymbol{W}^{(s)}(\boldsymbol{h}_t + \text{FWM}(\boldsymbol{h}_t, \boldsymbol{F}_t))$ where $\boldsymbol{F}_t \in \mathbb{R}^{d_{\text{FWM}} \times d_{\text{FWM}}^2}$ are the fast weights of the FWM at step t and $\boldsymbol{W}^{(s)} \in \mathbb{R}^{|\mathbb{V}| \times d_{\text{LSTM}}}$. Note that the fast weight matrix $\boldsymbol{F}_t$ is a reshaped third-order tensor $\mathsf{F}_t \in \mathbb{R}^{d_{\text{FWM}} \times d_{\text{FWM}} \times d_{\text{FWM}}}$. This allows us to describe third-order tensor operations using matix multiplications. We'll now describe in detail the FWM function and how its fast weights are updated.

3.1.1 Reading

For each step of the input sequence, the model queries the memory multiple times and uses the previous output as part of the next input. This allows the model to chain associations together into several "inference steps".

[2] We use the notation $\mathbf{x}_{1:t}$ to refer to the sequence $(x_1, x_2, ..., x_t)$.

$$\boldsymbol{n}_t^{(0)} = \phi(\boldsymbol{W}_q \boldsymbol{h}_t) \tag{1}$$

$$\boldsymbol{e}_t^{(i)} = \phi(\boldsymbol{W}_e^{(i)} \boldsymbol{h}_t), 1 \leq i \leq N_r \tag{2}$$

$$\boldsymbol{n}_t^{(i)} = \mathrm{LN}(\boldsymbol{F}_t(\boldsymbol{n}_t^{(i-1)} \otimes \boldsymbol{e}_t^{(i)})), 1 \leq i \leq N_r \tag{3}$$

$$\mathrm{FWM}(\boldsymbol{h}_t, \boldsymbol{F}_t) = \boldsymbol{W}_o \boldsymbol{n}_t^{(N_r)}. \tag{4}$$

Here LN refers to layernorm without the learned element-wise affine map (Ba et al., 2016b), ϕ is the hyperbolic tangent function, and the matrices $\boldsymbol{W}_n, \boldsymbol{W}_e^{(i)}, 0 \leq i \leq N_r \in \mathbb{R}^{d_{\text{FWM}} \times d_{\text{LSTM}}}$ and $\boldsymbol{W}_o \in \mathbb{R}^{d_{\text{LSTM}} \times d_{\text{FWM}}}$ are regular slow weights trained by gradient descent which allows us to decouple the dimensionality of the LSTM from the dimensionality of the FWM. In eq. 3, $\boldsymbol{F}_t$ is the multi-linear map which we query using the LSTM-generated "input" $\boldsymbol{e}$ and the previous retrieval $\boldsymbol{n}$ (except for the first query).

3.1.2 Writing

The FWM is updated at every step t using the write mechanism described in this section. To this end, we extract from the hidden state $\boldsymbol{h}_t$: the write strength β (a scalar bounded by 0 and 1 using the sigmoid function σ), the two key vectors $\boldsymbol{k}_1, \boldsymbol{k}_2$, and the associated value $\boldsymbol{v}$.

$$[\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{v}] = \phi(\boldsymbol{W}_{\text{write}} \boldsymbol{h}_t) \tag{5}$$

$$\beta = \sigma(\boldsymbol{W}_\beta \boldsymbol{h}_t) \tag{6}$$

Writing to memory is a variant of the perceptron rule (Rosenblatt, 1958) and follows a few simple steps. First the currently associated value $\boldsymbol{v}_{\text{old}}$ is retrieved. Second, we remove the old association from the map by subtracting $\boldsymbol{k}_1 \otimes \boldsymbol{k}_2 \otimes \boldsymbol{v}_{\text{old}}$. Third, we add a term such that the end result is $\beta \boldsymbol{v} + (1 - \beta) \boldsymbol{v}_{\text{old}}$. All three steps are achieved using the following update rule:

$$\boldsymbol{F}_t' = \boldsymbol{F}_{t-1} + \beta(\boldsymbol{k}_1 \otimes \boldsymbol{k}_2 \otimes (\boldsymbol{v} - \boldsymbol{v}_{\text{old}})) \tag{7}$$

To further improve stability and keep the fast weights from growing endlessly, we scale down the weights whenever their $||\boldsymbol{F}_t'||_2 > 1$. For this reason, we add a final element-wise scaling.

$$\boldsymbol{F}_t = \frac{\boldsymbol{F}_t'}{\max(1, ||\boldsymbol{F}_t'||_2)} \tag{8}$$

4 Experiments

4.1 Concatenated-bAbI

The bAbI tasks is a popular toy dataset to benchmark neural networks with memory augmentations and reasoning capabilities (Weston et al., 2015a). It consists of a set of short stories with questions embedded in the text. The stories were generated by simulating multiple entities in a virtual environment and cover different contexts in which entities change their state on their own or through an interaction. Each story-sample belongs to one of 20 different tasks that the authors of the dataset considered important for intelligent dialogue agents. The tasks contain questions which require reasoning capabilities like deduction, coreference, or counting. All tasks require some level of symbolic reasoning, and the first neural and non-neural baselines demonstrated poor generalisation performance on test data (Weston et al., 2015a).

We aim to improve the bAbI benchmark as a means of developing intelligent dialogue agents. To this end, we propose concatenated-bAbI (catbAbI): an infinite sequence of bAbI stories. catbAbI is generated from the bAbI dataset. The preprocessing for catbAbI addresses several issues: it removes the supporting facts, leaves the questions embedded in the story, inserts the correct answer after the question mark, and tokenises the full sample into a single sequence of words. As such, catbAbI is designed to be trained in an autoregressive way and analogous to closed-book question answering. During training, a random sample/story from any task is drawn without replacement and concatenated to the ongoing story.

catbAbI models can be trained in two different ways: language modelling mode (LM-mode) or question-answering mode (QA-mode). In LM-mode, the catbAbI models are trained like autoregressive word-level language models. In QA-mode, the catbAbI models are only trained to predict the tokens that are answers to questions—making it more similar to regular bAbI. QA-mode is simply implemented by masking out losses on non-answer predictions.

In both training modes, the model performance is solely measured by its accuracy and perplexity when answering the questions. Performance on non-answers is irrelevant on catbAbI because the tokens are either very predictive or inherently unpredictable, and there is nothing appealing to be learned. Despite measuring performance only for answers, we argue that LM-mode is interesting for three reasons. First, LM-mode removes the bias of knowing which words would benefit from a symbolic inference mechanism. Second, LM-mode trains the model on a sequence with tokens which are inherently unpredictable. Such tokens could also appear in natural language and might harm the model's ability to learn a useful representation of the story. Indeed, in the next section, we will give evidence for such a generalisation gap. Third, the LM-mode setting allows us to compare our method with state-of-the-art language models.

4.1.1 RESULTS

We compare our FWM model with MNM, the previous state-of-the-art on word-level bAbI when trained on all tasks. We also include two strong autoregressive word-level language models as baselines: a regularized LSTM (Merity et al., 2018; Melis et al., 2017) and regularized Transformer-XL (TXL; Dai et al. (2019)). Unfortunately, our attempts at incorportating the fast weight memory of Ba et al. (2016a) failed as were unable to find any hyper-parameter setting that would result in a performance over 15%. For this reason we excluded those results. All four models have different properties which makes a direct comparison difficult. We decided to truncate backpropagation through time (tBPTT) to 200 tokens for all models and limited the amount of GPU memory to at most ~16GB for practical reasons. For every model, we performed a hyper-parameter grid search over the first 3k steps of which a small selection were trained for 30k steps. In general we observe that the bigger models tend to perform better and overfitting is just a minor issue. The best hyper-parameters were used to train a model for 60k steps of which the final validation performance is given in table 1. Further details about the experiment and the dataset were added to the appendix section C.

Table 1: Summary of our catbAbI results. Accuracy and perplexity are measured only w.r.t. answer tokens in the catbAbI test data. Hyperparameters are chosen based on a grid-search with tBPTT fixed to 200 steps and a physical memory cap of ~16GB. Each experiment is run three times of which we report mean and std. Detailed hyper-parameters can be found in the appendix section C.2. MNM was unstable and all runs significantly dropped in accuracy at some point or diverged.

Mode		LSTM	TXL	MNM	FWM
LM-mode	accuracy	74.89% $\pm$ 1.31	86.8% $\pm$ 1.05	73.59% $\pm$ 3.38	84.02% $\pm$ 1.24
	perplexity	1.96 $\pm$ 0.0	1.23 $\pm$ 0.03	1.99 $\pm$ 0.0	1.95 $\pm$ 0.0
QA-mode	accuracy	76.46% $\pm$ 2.19	86.9% $\pm$ 4.05	73.9% $\pm$ 2.05	91.45% $\pm$ 3.38
	perplexity	1.82 $\pm$ 0.1	1.25 $\pm$ 0.14	2.00 $\pm$ 0.08	1.33 $\pm$ 0.09
trainable parameters		8M	24.2M	1.1M	694k
activations per step		4096	2.1M[3]	30.5k	33.3k

In contrast to baselines on bAbI, we find on catbAbI that a regularized, 4-layer deep, and residual LSTM, and a 7-layer deep TXL with memory, achieve strong performance on catbAbI. The increased complexity of catbAbI over regular bAbI allowed us to run much bigger baselines models which clearly improved their performance. In fact, both outperformed MNM, which in QA-mode suffered a TODO% drop in accuracy compared to reported results on regular bAbI. Our FWM achieves strong accuracy on catbAbI while being by far the smallest in parameter count. It significantly outperforms the other RNN models, which suggests the importance of our fast weight memory mechanism. In figure 4.1.1, we demonstrate a simple visualisation which shows how the model has learned to associate facts from different places in the text.

We chose to include the TXL model in our comparison due to its autoregressive nature and strong performance. However, it is essential to point out that the TXLs horizon is larger than the average

bAbI story. Hence, due to the shortness of the stories, catbAbI becomes more of an open-book problem for the TXL model since it has the capability of looking up representations of its previous input. This fundamentally limits the TXL model as it can only condition its prediction on information that is no longer than its attention window to past states. The RNN models (which are general function approximators), for better or for worse, are instead forced to learn to carry the necessary information through time.

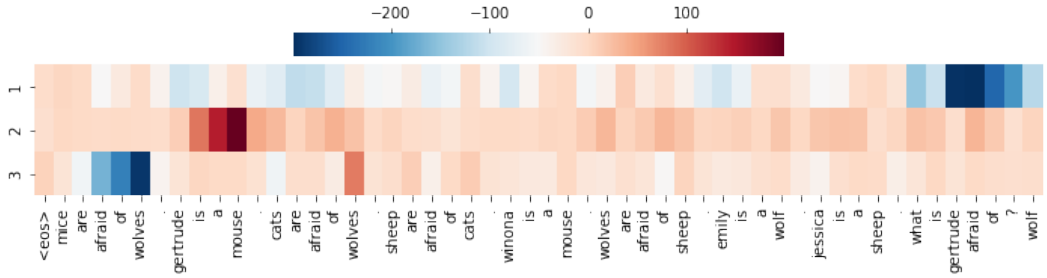

Figure 2: A visualisation of a section of the catbAbI train data. The answer "wolf" is correctly infered upon the input of the "?" token. During that step, the FWM is queried $N_r = 3$ times. The matrix shows the dot product between the query outer product of $\boldsymbol{n}$ and $\boldsymbol{e}$ of each step and the write pattern outer product of $\boldsymbol{k}_1$ and $\boldsymbol{k}_2$ of the previous 50 steps. We can observe that the first step is mostly matching with itself which is likely having no effect. On the second step, the read pattern matches the write pattern of the "gertrude is a mouse" section. Finally, on the last step, we can observe that the query pattern matches the "mice are afraid of wolves" section which likely completes the deduction: Gertrude is a mouse. Mice are afraid of wolves. Therefore, Gertrude is afraid of wolves.

4.2 META-REINFORCEMENT LEARNING

Meta reinforcement learning (Meta-RL) applies meta-learning (Schmidhuber, 1987; Hochreiter et al., 2001; Finn et al., 2017) to the field of reinforcement learning (Schmidhuber, 1994). An agent is trained on multiple environments (or tasks) and receives environmental feedback as part of its input. To maximise its total reward in an environment, the agent has to leverage the feedback signals and adapt. A successful agent is capable of maximising its reward in novel environments that it has not been exposed to during training. Recent work achieved noteable progress in this domain (Santoro et al., 2016; Mishra et al., 2018; Kirsch et al., 2020).

We experiment with tasks drawn randomly from a large set of partially observable Markov decision processes (POMDPs). In this set, every environment consists of precisely five states and three actions. Globally, every environment can be viewed as a sparse directed graph where nodes are locations, and the directed edges are one-way modes of transportation—similar to a metro transit map of a city (Graves et al., 2016). In order to sample a new environment, we sample the adjacency matrix of the graph such that actions are deterministic, and every location is reachable from any other location (see figure 3). We sample graphs such that there are no actions that lead to the same location, and such that not every action is always a valid way of transitioning. We added the exact algorithm to generate graphs, as well as further details, to the appendix section E.

The agent's goal is to reach a reward location. Upon arrival, the agent receives a reward, followed by a random reset of the agent's and reward's location. Whenever the agent takes an action that does not lead to a new location, it receives a penalty. At every step, the agent receives as an input: its current location, the reward location, its last action, and the reward received so far.

We optimise our agent with the Advantage Actor-Critic (A2C) algorithm, a non-asynchronous version of the A3C method (Mnih et al., 2016). We run our experiment for 30 steps and compare our FWM to an LSTM baseline.

Both methods are trained on the same training set of 600 graphs and tested on 600 novel graphs. In our experiments, the LSTM-based agent requires more episodes, a bigger network, and eventually overfits on the training graphs. The FWM-based agent, however, trains faster and generalises to the novel test graphs.

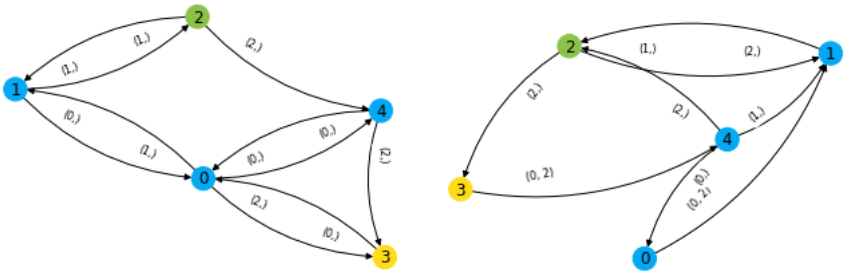

Figure 3: Two randomly generated environments with the agent's location coloured in green and the reward location coloured in yellow. Edge labels indicate the action (here 0, 1, or 2) to be taken in order to transition. The graph and the locations of the agent and reward are set randomly at the beginning of the experiment. If the agent reaches the reward location or did not reach it after six steps, both are randomly reset.

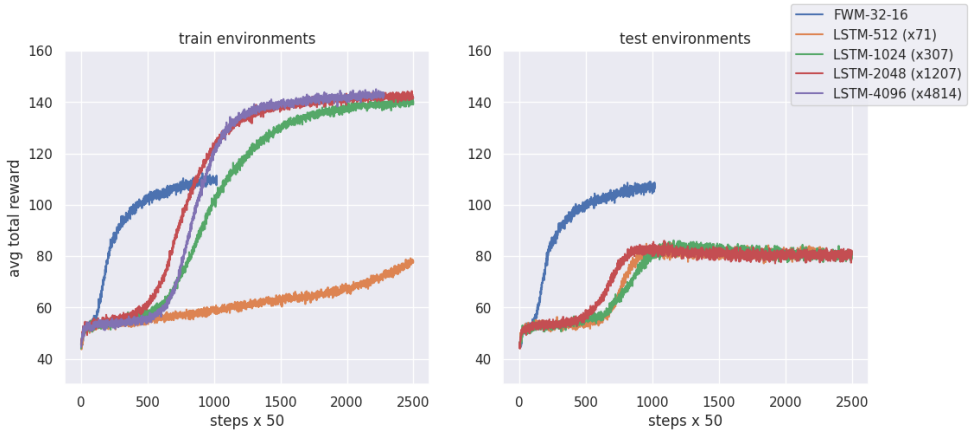

Figure 4: Average total reward of the agents when trained on 600 random graphs (left graph) and tested on 600 different graphs (right graph). The FWM agent (blue) has a slow LSTM with 32 hidden units and a fast weight memory of size 16×16^2. We compare to LSTM agents with different sized hidden states. The largest LSTM has 4096 hidden units (red) which roughly matches the number of temporal variables of the FWM. The FWM has 14k trainable weights which is by far the lowest. The largest LSTM has 67.4M weights which is roughly 4814 times more than the FWM. The relative factor of each LSTM is added to the legend. All LSTMs take longer to train and eventually overfit on the training data. Due to the overfitting, the LSTM does not have to explore, which results in a higher total reward on training environments but a lower total reward on test environments.

We argue that the bAbI stories and the episodes on the graphs are similar in the following three ways. First, in both problems, the network has to construct a useful and *context-specific* representation from its ongoing input. Second, as part of its input, the network repeatedly receives an objective (the reward location versus the question) which requires the exploitation of the context-specific information. Third, the neural network has to come up with a sequence of actions in the environment for RL and a sequence of reasoning steps in catbAbI to optimise its training signal (high reward versus low uncertainty).

4.3 Language Modelling

Comparing FWM to autoregressive language models on catbAbI begs the question: how does FWM perform on popular word-level language modelling datasets such as Penn Treebank (PTB; Mikolov et al. (2010))? It is unclear to which extend a symbolic inference mechanism is beneficial for language modelling. PTB contains virtually no questions and is constructed from news articles which are designed to be easily parsed by the reader. However, not many memory augmented NNs have

been able to bridge from small and toy reasoning tasks to general language models—and those which did, underperformed (Paperno et al., 2016; Sukhbaatar et al., 2015).

For simplicity we use the regularized 3-layer AWD-LSTM (Merity et al., 2018) as the slow RNN in our FWM model without any further hyper-parameter search. Judging from the training the ASGD training didn't help which indicates that with some tuning the final perplexity can be pushed down further. The experimental results in table 2 demonstrate a relative improvement over the AWD-LSTM baselines, which suggest the benefit of our FWM. However, in contrast to catbAbI, all three models achieve very similar results which might indicate that PTB does not benefit as strongly from an associative reasoning capacity. We added the experimental details to the appendix section D.

Since the publication of AWD-LSTM (Merity et al., 2018), various extensions (some of which are orthogonal to our memory augmentation) have been proposed (Krause et al., 2018; Merity et al., 2018; Yang et al., 2018). In this work we are not primarily interested in beating the state-of-the-art in language modelling and leave it for future work to explore the possible synergies between these methods.

Table 2: Average perplexity on the test data of Penn Treebank (PTB). All PTB models have roughly 24M parameters. The AWD-Transformer-XL is the Transformer-XL architecture as reported by with Dai et al. (2019) with the intended and necessary AWD-style regularisation.

Model	PTB	
	Validation	Test
LSTM	60.0	57.3
Transformer-XL (ours)	59.85	55.27
LSTM-FWM (ours)	61.39	59.37

5 Discussion

An order-three memory tensor is a computationally demanding structure for constructing compositional state representations. With vector components in $\mathbb{R}^n$, the tensor product computation has a space and time complexity of $O(n^3)$. For practical reasons, this forces the FWM to remain small, relative to the slow NN, which limits the number of associations that can be maintained at once. Previous work has proposed approximations of such memory tensors in a variance-optimal way (Schlag et al., 2019). However, further improvements might not come from scaling up but from more general symbolic manipulations. Loosely, our fast weight memory can be thought of as the "working memory" of the model—analogous to the working memory in the human brain (Spalding et al., 2018). Recent work has proposed a cognitive model of the human brain which is based on such tensors (Tresp & Ma, 2017).

We address the capacity of the FWM and the necessity of the tensor product from a linear hetero-associative memory perspective in section A of the appendix.

6 Conclusion

Our new FWM is a general fast weights architecture capable of learning from synthetic data to answer questions that require various symbolic reasoning skills. To improve generality, we overcome issues of the popular bAbI dataset by introducing a harder and more general version dubbed catbAbI. We report strong performance on catbAbI and compare with improved baselines based on state-of-the-art language models. In a self-supervised setting, we apply the FWM to word-level language modelling on PTB where it is competitive with the AWD-LSTM baseline. Finally, we report results in a challenging meta-reinforcement learning environment where the agent learns to learn the structure of a randomized graph from its observations and actions and to infer the graph-specific action sequence that maximizes the total reward.

REFERENCES

Aishwarya Agrawal, Aniruddha Kembhavi, Dhruv Batra, and Devi Parikh. C-vqa: A compositional split of the visual question answering (vqa) v1.0 dataset. *ArXiv*, abs/1704.08243, 2017.

Yuval Atzmon, Jonathan Berant, Vahid Kezami, Amir Globerson, and Gal Chechik. Learning to generalize to new compositions in image understanding. *arXiv preprint arXiv:1608.07639*, 2016.

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *Advances In Neural Information Processing Systems*, pp. 4331–4339, 2016a.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016b.

Sergey Bartunov, Jack Rae, Simon Osindero, and Timothy Lillicrap. Meta-learning deep energy-based memory models. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SyljQyBFDH`.

Robert Csordas and Juergen Schmidhuber. Improving differentiable neural computers through memory masking, de-allocation, and link distribution sharpness control. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HyGEM3C9KQ`.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

S. Das, C.L. Giles, and G.Z. Sun. Learning context-free grammars: Capabilities and limitations of a neural network with an external stack memory. In *Proceedings of the The Fourteenth Annual Conference of the Cognitive Science Society, Bloomington*, 1992.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HyzdRiR9Y7`.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

Jerome A Feldman. Dynamic connections in neural networks. *Biological cybernetics*, 46(1):27–39, 1982.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

Samuel J Gershman, Kenneth A Norman, and Yael Niv. Discovering latent causes in reinforcement learning. *Current Opinion in Behavioral Sciences*, 5:43–50, 2015.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

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

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *ICLR 2017*, 2017.

Robert F Hadley. Systematicity in connectionist language learning. *Mind & Language*, 9(3):247–272, 1994.

Geoffrey E Hinton and David C Plaut. Using fast weights to deblur old memories. In *Proceedings of the ninth annual conference of the Cognitive Science Society*, pp. 177–186, 1987.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. Based on TR FKI-207-95, TUM (1995).

S. Hochreiter, A. S. Younger, and P. R. Conwell. Learning to learn using gradient descent. In *Lecture Notes on Comp. Sci. 2130, Proc. Intl. Conf. on Artificial Neural Networks (ICANN-2001)*, pp. 87–94. Springer: Berlin, Heidelberg, 2001.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Daniel D. Johnson. Learning graphical state transitions. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HJ0NvFzxl.

Pentti Kanerva. *Sparse distributed memory*. MIT press, 1988.

Louis Kirsch, Sjoerd van Steenkiste, and Juergen Schmidhuber. Improving generalization in meta reinforcement learning using learned objectives. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1evHerYPr.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3): 455–500, 2009.

Bart Kosko. Bidirectional associative memories. *IEEE Transactions on Systems, man, and Cybernetics*, 18(1):49–60, 1988.

Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. Dynamic evaluation of neural sequence models. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2766–2775, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/krause18a.html.

Joseph B Kruskal. Rank, decomposition, and uniqueness for 3-way and n-way arrays. *Multiway data analysis*, pp. 7–18, 1989.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1378–1387, New York, New York, USA, 20–22 Jun 2016. PMLR. URL http://proceedings.mlr.press/v48/kumar16.html.

Brenden M. Lake and Marco Baroni. Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks. *CoRR*, abs/1711.00350, 2017. URL http://arxiv.org/abs/1711.00350.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

Hung Le, Truyen Tran, and Svetha Venkatesh. Self-attentive associative memory. *arXiv preprint arXiv:2002.03519*, 2020.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.05493.

Brian P McLaughlin. Systematicity redux. *Synthese*, 170(2):251–274, 2009.

Gábor Melis, Chris Dyer, and Phil Blunsom. On the state of the art of evaluation in neural language models. *CoRR*, abs/1707.05589, 2017. URL http://arxiv.org/abs/1707.05589.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyyGPP0TZ.

Thomas Miconi, Kenneth Stanley, and Jeff Clune. Differentiable plasticity: training plastic neural networks with backpropagation. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3559–3568, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/miconi18a.html.

Thomas Miconi, Aditya Rawal, Jeff Clune, and Kenneth O. Stanley. Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1lrAiA5Ym.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1DmUzWAW.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Michael C Mozer and Sreerupa Das. A connectionist symbol manipulator that discovers the structure of context-free languages. In *Advances in neural information processing systems*, pp. 863–870, 1993.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2554–2563. JMLR. org, 2017a.

Tsendsuren Munkhdalai and Hong Yu. Neural semantic encoders. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 1, pp. 397. NIH Public Access, 2017b.

Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. Metalearned neural memory. In *Advances in Neural Information Processing Systems*, pp. 13310–13321, 2019.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://www.aclweb.org/anthology/P16-1144.

Steven Andrew Phillips. *Connectionism and the problem of systematicity*. PhD thesis, University of Queensland, 1995.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adria Puigdomenech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2827–2836. JMLR. org, 2017.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

Imanol Schlag and Jürgen Schmidhuber. Gated fast weights for on-the-fly neural program generation. In *NIPS Metalearning Workshop*, 2017.

Imanol Schlag and Jürgen Schmidhuber. Learning to reason with third order tensor products. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 9981–9993, 2018.

Imanol Schlag, Paul Smolensky, Roland Fernandez, Nebojsa Jojic, Jürgen Schmidhuber, and Jianfeng Gao. Enhancing the transformer with explicit relational encoding for math problem solving. *arXiv preprint arXiv:1910.06611*, 2019.

Margaret L Schlichting and Alison R Preston. Memory integration: neural mechanisms and implications for behavior. *Current opinion in behavioral sciences*, 1:1–8, 2015.

J. Schmidhuber. Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook. Diploma thesis, Inst. f. Inf., Tech. Univ. Munich, 1987. http://www.idsia.ch/˜juergen/diploma.html.

J. Schmidhuber. Learning to control fast-weight memories: An alternative to recurrent nets. *Neural Computation*, 4(1):131–139, 1992.

J. Schmidhuber. On decreasing the ratio between learning complexity and number of time-varying variables in fully recurrent nets. In *Proceedings of the International Conference on Artificial Neural Networks, Amsterdam*, pp. 460–463. Springer, 1993.

J. Schmidhuber. On learning how to learn learning strategies. Technical Report FKI-198-94, Fakultät für Informatik, Technische Universität München, 1994. Revised 1995.

H. T. Siegelmann and E. D. Sontag. Turing computability with neural nets. *Applied Mathematics Letters*, 4(6):77–80, 1991.

P. Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artif. Intell.*, 46(1-2):159–216, November 1990. ISSN 0004-3702. doi: 10.1016/0004-3702(90)90007-M. URL http://dx.doi.org/10.1016/0004-3702(90)90007-M.

Paul Smolensky. Symbolic functions from neural computation. *Phil. Trans. R. Soc. A*, 370(1971): 3543–3569, 2012.

Kelsey N Spalding, Margaret L Schlichting, Dagmar Zeithamova, Alison R Preston, Daniel Tranel, Melissa C Duff, and David E Warren. Ventromedial prefrontal cortex is necessary for normal associative inference and memory integration. *Journal of Neuroscience*, 38(15):3767–3775, 2018.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pp. 2440–2448, 2015.

Volker Tresp and Yunpu Ma. The tensor memory hypothesis. *ArXiv*, abs/1708.02918, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Christoph von der Malsburg. The correlation theory of brain function (internal report 81-2). *Goettingen: Department of Neurobiology, Max Planck Intitute for Biophysical Chemistry*, 1981.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698, 2015a. URL http://arxiv.org/abs/1502.05698.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *International Conference on Learning Representations*, 2015b. URL https://arxiv.org/abs/1410.3916.

David J Willshaw, O Peter Buneman, and Hugh Christopher Longuet-Higgins. Non-holographic associative memory. *Nature*, 222(5197):960–962, 1969.

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning*, pp. 2397–2406, 2016.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. Breaking the softmax bottleneck: A high-rank RNN language model. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HkwZSG-CZ.

Wei Zhang and Bowen Zhou. Learning to update auto-associative memory in recurrent neural networks for improving sequence memorization. *arXiv preprint arXiv:1709.06493*, 2017.