Neural Structured Turing Machine

Hao Liu [*1] **Xinyi Yang** [*2] **Zenglin Xu** [2]

Abstract

Differential external memory has greatly improved attention based model, such as *neural Turing machines*(NTM), however none of existing attention models and memory augmented attention models has considered real-valued structured attention, which may hinders their representation power. Here we take structured information into consideration and propose a *Neural Structured Turing Machine*(NSTM). Our preliminary experiment on varied length sequence copy task shows the benefit of such structured attention mechanism, leading to a novel perspective for improving differential external memory based mechanism, e.g., NTM.

1. Introduction

Neural processes involving attention have been largely studied in Neuroscience and Computational Neuroscience(Itti et al., 1998). Basically, an attention model is a method that take n arguments and a context and return a vector which is supposed to be the summary of the argument focusing on information linked to the context. More formally, it returns a weighted arithmetic mean of the augment and the weights are chosen according the relevance of each argument of each argument given the context.

Attention mechanism can be commonly divided into soft attention and hard attention. Soft attention is a fully differentiable deterministic mechanism that can be plugged into an existing system, and the gradients are propagated through the attention mechanism at the same time they are propagated through the rest of the network. Contrast to soft attention, hard attention is a stochastic process where it instead of using all the hidden states as an input for the decoding, the system samples a hidden state with a stochastic probabilities. Commonly used approaches to estimate the gradient are Monte Carlo sampling and REINFORCE.

Recent work have advanced attention mechanism to fully differentiable addressable memory. Such as *Memory Networks*(Weston et al., 2014), *Neural Turing Machines*(Graves et al., 2014) and many other models(Kurach et al., 2016; Chandar et al., 2016; Gulcehre et al., 2016), all of which use external memory in which they can read (eventually write). Fully differentiable addressable memory enables model to bind values to specific locations in data structures(Fodor & Pylyshyn, 1988). Differentiable memory is important because many of the algorithms and things that we do every day on a computer are actually very hard for a computer to do because computers work in absolutes. While most neural network models dont actually do that. They work with real numbers, smoother sort of curves, and that makes them actually a great deal easier to train, because it means that what you see and hope that they provide, that you can easily track back how to improve them by tweaking the parameters.

However, currently none of attention models and other differential memory mechanisms considers the structured information between memory location and query content when addressing memory, instead they simply use content to calculate a relevance score, which may hinder their representation power. Here we show how to use a general structured inference mechanism to model complex dependencies between memory locations and query in Neural Turing Machine(NTM), we validate through experiments the power of incorporating real-valued structured attention in NTM.

2. Model

2.1. Energy Based Structured Attention

Let m_{t-1} denotes input, q_t denotes the query, to enable query based content search, input m_{t-1} can be processed with query q_t by $x = m_{t-1}^{\top} W q_t$ where W comes from a neural network(e.g., MLP or parameters), $y = (y_1, y_2, ..., y_n)$ denotes the attention value we want to predict, here $y_i \in [0, 1]$, $E(x, y, w) : R^K \rightarrow R$ is an energy function, defined as $E = \sum_i f_i(y_i, x; w_u) +$

[*]Equal contribution [1]SMILE Lab & Yingcai Honors College, University of Electronic Science and Technology of China [2]SMILE Lab & Big Data Research Center School of Computer Science and Engineering, University of Electronic Science and Technology of China. Correspondence to: Zenglin Xu <zlxu@gmail.com>.

Proceedings of the ICML 17 Workshop on Deep Structured Prediction, Sydney, Australia, PMLR 70, 2017.

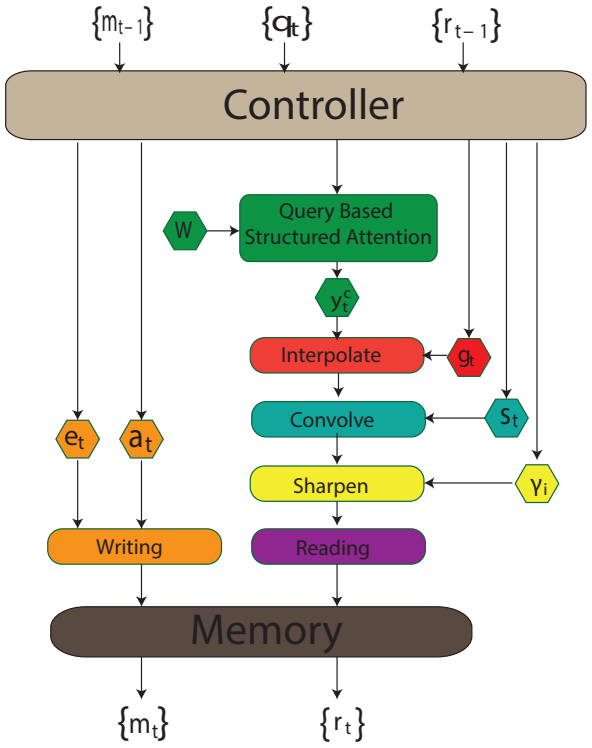

Figure 1. Neural Structured Turing machine

$\sum_\alpha f_\alpha(y_\alpha, x; w_\alpha)$ which can be viewed as unary function plus higher order energy functions. Our aim is obtain $y^\star = \arg min_{y\in\mathcal{Y}} \sum_i f_i(y_i, x; w_u) + \sum_\alpha f_\alpha(y_\alpha, x; w_\alpha)$. Since this kind of model in general is intractable although a variety of methods have been developed in the context of predicting discrete outputs(Zheng et al., 2015b; Chen et al., 2015; Zheng et al., 2015a). A Markov random fields (MRFs) with Gaussian potentials is also real-valued and when the precision matrix is positive semi-definite and satisfies the spectral radius condition (Weiss & Freeman, 2001) then message passing is possible for exact inference. However, MRF with Gaussian potentials may not be powerful enough for extracting complex features from memory and model complex dependencies between query and memory. We turn to proximal method to remedy, for more about proximal method refer to (Parikh et al., 2014). We made the following assumptions: $f_i(y_i, x; w) = g_i(y_i, h_i(x, w))$ and $f_\alpha(y_\alpha, x; w_\alpha) = h_\alpha(x; w) g_\alpha(w_\alpha^T y_\alpha)$, then

$$E = \sum_i g_i(y_i, h_i(x, w)) + \sum_\alpha h_\alpha(x; w) g_\alpha(w_\alpha^T y_\alpha) \quad (1)$$

The above equations can be solved by primal-dual method effectively. Denote the output of t-th iteration of as y_t^c.

2.2. Controller Parameters

Having shown how to address memory using structured attention mechanism, now we turn to other addressing situation, in some cases, we may want to read from specific memory locations instead of reading specific memory values. This is called location-based addressing, and to implement it, we need three more stages. In the second stage, a scalar parameter $g_t \in (0, 1)$ called the interpolation gate, blends the content weight vector y_t^c with the previous time step's weight vector y_{t-1} to produce the gated weighting y_t^g.This allows the system learn when to use (or ignore) content-based addressing, $y_t^g \leftarrow g_t y_t^c + (1 - g_t) y_{t-1}$.

We'd like the controller to be able to shift focus to other rows. Let's suppose that as one of the system's parameters, the range of allowable shifts is specified. For example, a head's attention could shift forward a row (+1), stay still (0), or shift backward a row(-1) given a shift modulo R, so that a shift forward at the bottom row of memory moves the head's attention to the top row, and similarly for a shift backward at the top row. After interpolation, each head emits a normalized shift weighting s_t and the following convolutional shift with shift weight s_t, $\tilde{y}_t(i) \leftarrow \sum_i (y_t^g(j)) s_t(i - j)$.

In order to prevent shift weight s_t from blurring, we define sharpening as following, $y_t(i) \approx \tilde{y}_t(i)^{\gamma_i}$, here g_t, s_t and γ_i are controller parameters.

2.3. Writing

Writing involves two separate steps: erasing, then adding. In order to erase old data, a write head will need a new vector, the length-C erase vector e_t , in addition to our length-R normalized weight vector y_t. The erase vector is used in conjunction with the weight vector to specify which elements in a row should be erased, left unchanged, or something in between. If the weight vector tells us to focus on a row, and the erase vector tells us to erase an element, the element in that row will be erased. Then the write head uses a length-C add vector a_t to complete the writing.

$$m_t^{erased} \leftarrow m_{t-1}(i)[1 - y_t(i) e_t]$$
$$m_t(i) \leftarrow m_t^{erased}(i) + y_t(i) a_t$$

2.4. Optimization and Learning

The general idea of primal dual solvers is to introduce auxiliary variables z to decompose the high order terms. We can then minimize z and y alternately through computing their proximal operator. In particular, we can transform the primal problem in Eq.(1) into the following saddle point problem.

$$E = \min_{y\in\mathcal{Y}} \max_{z\in\mathcal{Z}} \sum_i g_i(y_i, h_i(x, w_\alpha)) - \sum_\alpha h_\alpha(x, w) g_\alpha^\star(z_\alpha) + \sum_\alpha h_\alpha(x, w) \langle w_\alpha^T y_\alpha, z_\alpha \rangle$$

where $g^{\star}_{\alpha}(\cdot)$ is the convex function of $g_{\alpha}(z^{\star})$: $\sup\{\langle z^{\star}, z\rangle - g_{\alpha}(z)|z \in Z\}$. Note that the whole inference process has two stages: first we compute the unaries $h(x, w_u)$ with a forward pass, followed by a MAP inference. This kind of optimization approach was also proposed in(Wang et al., 2016), note that both are special cases of (Domke, 2012).

3. Experiment

The goal is not only to establish that NSTM is able to solve the problems, but also that it is able to do so by learning compact internal programs. The hallmark of such solutions is that they generalise well beyond the range of the training data. For example, we were curious to see if a network that had been trained to copy sequences of length up to 20 could copy a sequence of length 100 with no further training.

We compare NSTM to different work namely NTM with a feedforward controller, NTM with an LSTM controller, and a standard LSTM network. All the tasks were supervised learning problems with binary targets; all networks had logistic sigmoid output layers and were trained with the cross-entropy objective function.

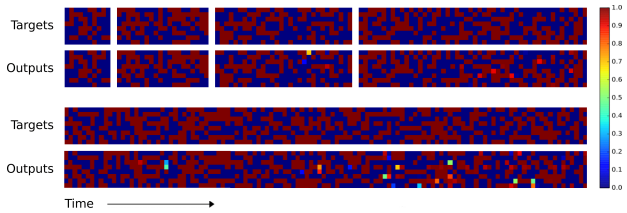

Figure 2. Results of Neural Structured Turing Machine on difference sequences. The four pairs of plots in the top row depict network outputs and corresponding copy targets for test sequences of length 10, 20, 30, and 50, respectively. The plots in the bottom row are for a length 120 sequence. The network was only trained on sequences of up to length 20. Compared to other methods shown in Figure 2 and Figure 1, the first four sequences are reproduced by Neural Structured Turing Machine with high confidence and very few mistakes.

3.1. Copy

To exams whether NSTM can store and recall a long sequence of arbitrary information, we apply NSTM to copy task. This experiment was designed to qualify whether having an structured attention system allows for better performance. The network is presented with an input sequence of random binary vectors followed by a delimiter flag. Storage and access of information over long time periods has always been problematic for RNNs and other dynamic architectures. There are three different models compared:*Neural Turing Machine* with LSTM controller, a basic LSTM and NSTM proposed in this paper. To qualify NSTM's performance on longer sequences which were never seen before, the networks were trained to copy sequences of eight bit random vectors, where the sequence lengths were randomised between 1 and 20. The target sequence was simply a copy of the input sequence (without the delimiter flag). Note that no inputs were presented to the network while it receives the targets, to ensure that it recalls the entire sequence with no intermediate assistance. Since the training sequences had between 1 and 20 random vectors, the LSTM and NTMs were compared using sequences of lengths 10, 20, 30, 50 and 120. Results of NSTM shown in Figure 3, compared to LSTM and NTM shown in Figure 2 and Figure 1, NSTM remembers longer and can recall a longer sequence of arbitrary information, we can draw a conclusion that it is beneficial to enable structured attention.

3.2. Repeat Copy

To compare NTM and NSTM on learning a simple nested function, repeat copy task is proposed, since it requires the model to output the copied sequence a specified number of times and then emit an end maker. The model is fed with random length sequences of random binary vectors, followed by a scalar value indicating the desired number of copies which presents through a separate input channel. The model must be both able to interpret the extra input and keep count of number of copies it has performed so far to be able to emit the end maker at the correct time. The models were trained to reproduce sequences of size eight random binary vectors, where both the sequence length and the number of repetitions were chosen randomly from one to ten. The input representing the repeat number was normalised to have mean zero and variance one. Comparison between Neural Structured Turing Machine with LSTM controller, LSTM, NTM with feedforward/LSTM controller on repeat copy Learning task shown in Figure 3. It shows that compared to NTM and LSTM, Neural Structured Turing Machine allows faster convergence although all of them eventually converge to the optimum.

4. Related work

4.1. Attention Mechanism and Neural Turing Machine

In *Neural Turing Machine*(Graves et al., 2014), the authors consider augment attention model with differential external memory to enable neural networks to bind values to specific locations in data structure by learning how to use this memory. (Kim et al., 2017) proposes to use a linear-chain CRF to model discrete attention in a structured way. However, their model is limited to discrete situation, which means that the dependency this kind of structured attention

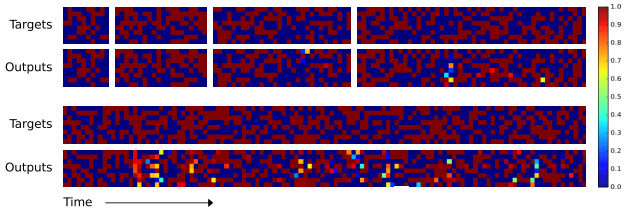

Table 1. Results of Neural Turing Machine on different length sequences in copy task.

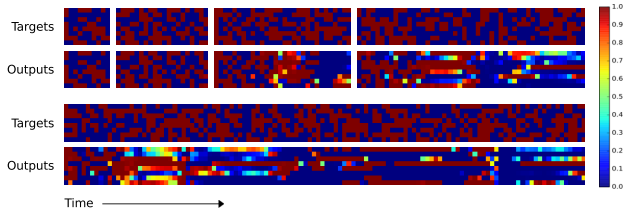

Table 2. Results of Long short-term memory on different length sequences in copy task.

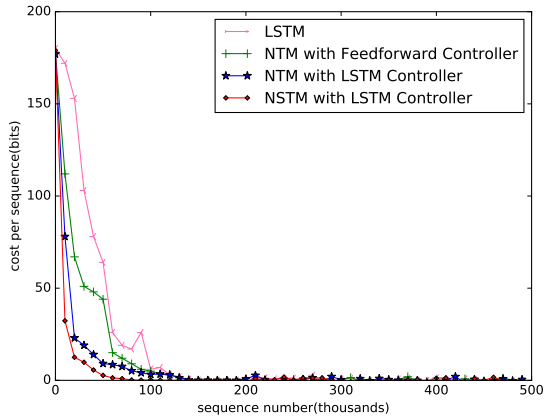

Figure 3. Comparison between Neural Structured Turing Machine with LSTM controller, LSTM, NTM with feedforward/LSTM controller on repeat copy Learning task. The compared baselines *NTM* and LSTM use the same parameter setting as in (Graves et al., 2014).

can model is rather limited. (Mnih et al., 2014) views vision as a sequential decision task, the model sequentially chooses small windows of information on the data, integrating information from all past windows to make its next decision. (Gregor et al., 2015) combines an attention mechanism with a sequential variational auto-encoder and makes both reading and writing sequential tasks. (Xu et al., 2015) use attention-based model to dynamically select needed features based based on the intuitions of attention.

4.2. Neural Structured Models

It is natural to combine MRF, CRF and neural network to enable (deep) neural structured model. (LeCun et al., 2006; Collobert et al., 2011; Chen et al., 2014; Schwing & Urtasun; Chen et al., 2015) parametrize the potentials of a CRF using deep features. Alternatively, non-iterative feed-forward predictors can be constructed using structured models as motivation (Domke, 2013; Li & Zemel, 2014; Zheng et al., 2015b). (Tompson et al., 2014) jointly train convolutional neural networks and a graphical model for pose estimation, (Johnson et al., 2016) also combines graphical models with neural network for behaviour modelling.(Meshi et al., 2010; Taskar et al., 2005) used local-consistency relaxation message-passing algorithms to do MAP inference in a logical MRF but using the dual objectives from fast message-passing algorithms for graphical model inference as the inner dual. (Schwing et al., 2012) used this for latent variable modelling, also using message-passing duals. (Chen et al., 2015) used the inner dual method twice to even more aggressively dualize expensive inferences during latent variable learning . (Chávez, 2015) revisited the inner dual idea to train deep models with structured predictors attached to them.

5. Conclusion and future work

Neural Turing Machine is a very promising way to achieve general intelligence, it shows that accessing to external memory is crucial to learn and generalise. This paper further improve upon this ideas by considering structured real-valued addressing attention mechanism, our preliminary experiments show the benefit of such structured attention in *Neural Turing Machine*. Future work include conducting more experiments on various tasks including important vision and NLP tasks and exploring the recurrence nature of structure attention mechanism.

References

Chandar, A. P. Sarath, Ahn, Sungjin, Larochelle, Hugo, Vincent, Pascal, Tesauro, Gerald, and Bengio, Yoshua. Hierarchical memory networks. *CoRR*, 2016.

Chávez, Hugo. Paired-dual learning for fast training of latent variable hinge-loss mrfs: Appendices. 2015.

Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. *Semantic image segmentation with deep convolutional nets and fully connected crfs*. ICCV, 2014.

Chen, Liang-Chieh, Schwing, Alexander, Yuille, Alan, and Urtasun, Raquel. Learning deep structured models. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1785–1794, 2015.

Collobert, Ronan, Weston, Jason, Bottou, Léon, Karlen, Michael, Kavukcuoglu, Koray, and Kuksa, Pavel. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

Domke, Justin. Generic methods for optimization-based modeling. In *AISTATS*, 2012.

Domke, Justin. Learning graphical model parameters with approximate marginal inference. *PAML*, 2013.

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

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

Gregor, Karol, Danihelka, Ivo, Graves, Alex, Rezende, Danilo Jimenez, and Wierstra, Daan. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Gulcehre, Caglar, Chandar, Sarath, Cho, Kyunghyun, and Bengio, Yoshua. Dynamic neural turing machine with soft and hard addressing schemes. *arXiv preprint arXiv:1607.00036*, 2016.

Itti, Laurent, Koch, Christof, and Niebur, Ernst. A model of saliency-based visual attention for rapid scene analysis. *PAMI*, 20(11):1254–1259, 1998.

Johnson, Matthew J., Duvenaud, David K., Wiltschko, Alex, Adams, Ryan P., and Datta, Sandeep R. Composing graphical models with neural networks for structured representations and fast inference. In *NIPS*, 2016.

Kim, Yoon, Denton, Carl, Hoang, Luong, and Rush, Alexander M. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

Kurach, Karol, Andrychowicz, Marcin, and Sutskever, Ilya. Neural random access machines. *ERCIM News*, 2016, 2016.

LeCun, Yann, Chopra, Sumit, Hadsell, Raia, and Huang, Jie. A tutorial on energy-based learning. 2006.

Li, Yujia and Zemel, Richard S. Mean-field networks. *CoRR*, abs/1410.5884, 2014.

Meshi, Ofer, Sontag, David, Jaakkola, Tommi S., and Globerson, Amir. Learning efficiently with approximate inference via dual losses. In *ICML*, 2010.

Mnih, Volodymyr, Heess, Nicolas, Graves, Alex, et al. Recurrent models of visual attention. In *NIPS*, 2014.

Parikh, Neal, Boyd, Stephen, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

Schwing, Alexander G. and Urtasun, Raquel. Fully connected deep structured networks.

Schwing, Alexander G., Hazan, Tamir, Pollefeys, Marc, and Urtasun, Raquel. Efficient structured prediction with latent variables for general graphical models. In *ICML*, 2012.

Taskar, Ben, Chatalbashev, Vassil, Koller, Daphne, and Guestrin, Carlos. Learning structured prediction models: a large margin approach. In *ICML*, 2005.

Tompson, Jonathan, Jain, Arjun, LeCun, Yann, and Bregler, Christoph. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.

Wang, Shenlong, Fidler, Sanja, and Urtasun, Raquel. Proximal deep structured models. In *NIPS*, 2016.

Weiss, Yair and Freeman, William T. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Neural computation*, 2001.

Weston, Jason, Chopra, Sumit, and Bordes, Antoine. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

Xu, Kelvin, Ba, Jimmy, Kiros, Ryan, Cho, Kyunghyun, Courville, Aaron, Salakhudinov, Ruslan, Zemel, Rich, and Bengio, Yoshua. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pp. 2048–2057, 2015.

Zheng, Shuai, Jayasumana, Sadeep, Romera-Paredes, Bernardino, Vineet, Vibhav, Su, Zhizhong, Du, Dalong, Huang, Chang, and Torr, Philip H. S. Conditional random fields as recurrent neural networks. *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1529–1537, 2015a.

Zheng, Shuai, Jayasumana, Sadeep, Romera-Paredes, Bernardino, Vineet, Vibhav, Su, Zhizhong, Du, Dalong, Huang, Chang, and Torr, Philip HS. Conditional random fields as recurrent neural networks. In *ICCV*, pp. 1529–1537, 2015b.