Relational Attention Networks via Fully-Connected Conditional Random Fields

Ziyin Liu[‡], Junxiang Chen[†], Paul Pu Liang[♠], Masahito Ueda[‡]
[‡]University of Tokyo, [†]Northeastern University, [♠]Carnegie Mellon University
zliu@cat.phys.s.u-tokyo.ac.jp, jchen@ece.neu.edu,
pliang@cs.cmu.edu, ueda@phys.s.u-tokyo.ac.jp

1 Introduction

This paper proposes a new attention mechanism that aims at building up a new level of interpretability of attention-based neural networks, while at the same time working towards connecting machine learning techniques such as conditional random fields (CRFs) [20], relational deep learning [30, 6] and physics models. The standard attention mechanism [4] involves feeding the encoded representations to a neural network to compute the weights for each encoded state [25, 3, 8, 13, 28]. However, the computation of attention weights is based on a black-box neural network module that cannot be interpreted. In an effort to study why neural networks decide to output such attention, we propose to impose structures via a graphical model to compute the attention weights. Our proposed model is closely related to inference in Conditional Random Fields [20] and aims to generalize the existing work in incorporating structures (inductive biases) into deep models [7, 17]. However, our model is not directly comparable to theirs because their work impose structure on the attentions as a way to incorporate prior knowledge into the task, while ours discovers structures in the task in an end-to-end fashion. In the physics community, our model and algorithm are closely related to spin glass models [1, 2, 31].

2 Method

We consider a problem setting in which the input is a time sequence with arbitrary but bounded length and the output is also a time sequence (output of a single scalar can be thought of as a sequence of length 1). Such tasks include machine translation [4, 25] or natural language style transfer [15]. Traditional approaches use an attention layer defined often using a neural network on top of the encoder states. This standard algorithm for an encoder/decoder network is given in Algorithm 2.

2.1 Fully Connected CRF Attention and Mean-Field Approximation

Let ${h_i}_i^T$ denote the input sequence of length T, d_t output of a decode at time step t. Intuitively, for every output d_t our method defines a series of random variables $\{s_i\}_i^T$ on each of the input, and each of $s_i \in \{0,1\}$ where $s_i = 0$ means that the current input is irrelevant to the output of d_t, and $s_i = 1$ means relevance. The series $\{s_i\}_i^T$ can be seen as a "hard attention" over the input. We can also imagine that the relevance of each input can interact with the relevance of other inputs (for example, in a translation task, the fact that a word the is relevant means that a different the at another position is irrelevant). We model this interation with a CRF, and we can define unary (ϕ_i) and binary potentials (J_{ij}) on s_i, and we want to find the most likely joint state $\arg\max_{s_1,...,s_T} \Pr[s_1, ..., s_T]$ for each d_t. Intuitively, if $J_{ij} > 0$, then s_i and s_j tend to have opposite sign to minimize the energy and so attending on one of these input inhibits the attention on the other; if $J_{ij} < 0$, then s_i and s_j tend to focused on simultaneously. J_{ij} are output from a small neural network. Since an exact inference is intractable, we use mean field approximation to find an approximate solution to the problem. The advantage of mean field approximation is that it is efficient on modern GPUs and is end-to-end trainable. Our algorithm is given in Algorithm 1. This iterative procedure is similar to but more general than the algorithm presented in [18, 32]. In the next section, we derive our update rule and prove the theoretical soundness of our method.

Third workshop on Bayesian Deep Learning (NeurIPS 2018), Montréal, Canada.

Algorithm 1 Mean-field-Attention($\cup_i^T h_i, d_t$). Note that D can be any arbitrarily defined function with necessary smoothness and regularity (e.g., a multilayer perceptron). $\cup$ means concatenation.

```
1: Input: $(\cup_i^T h_i, d_t)$
2: Initialize $\tau \leftarrow 0, k$
3: $\phi_i \leftarrow D_{\text{unary}}(h_i, d_t)$ // computing on-site potential
4: $J_{ij} \leftarrow D_{\text{binary}}(h_i, h_j, d_t)$ // computing interaction potential
5: $\chi_i^{(0)} \leftarrow \phi_i$
6: while $\chi^{(t)}$ not converged AND $\tau < k$ do
7:     $s_i \leftarrow \text{sigmoid}(\chi_i^{(\tau)})$
8:     $\chi_i^{(\tau+1)} \leftarrow \sum_j J_{ij} s_j + \phi_i$
9:     $\tau \leftarrow \tau + 1$
10: Output: sigmoid$(\chi_i^{(\tau-1)})$ // output expected spin at the converged point
```

2.2 Variational Upper Bound of the Mean-field Approximation

Now we show that our algorithm finds a variational upper bound for the optimal solution. In this work we primarily present our work in a theoretical physics framework. The theorem is originally stated in a physical context, and we adapt it for our purpose. In particular, we set the temperature to be 1 and we will present the proofs in the appendix:

Theorem 1. *(Bogoliubov inequality) Let the Hamiltonian of the system be written in terms of a non-interacting Hamiltonian H_0 (i.e. this part contains only unary potentials), and a interacting part H_i (i.e. this contains binary potentials): $H = H_0 + H_i$. Then the free energy of the system F obeys the following inequality:*

$$F \le F_0 \coloneqq \langle H \rangle_0 - S \tag{1}$$

where S is the entropy of the system, and $\langle H \rangle_0$ is the energy of the system given by H_0.

The above theorem guarantees that F_0 is always an upper bound for F. As a result, we can minimize F_0 instead, to obtain an approximate inference procedure. The following theorem gives the optimal update rule if we use a factorizable H_0 as our Hamiltonian:

Theorem 2. *Let $H = \sum_{i,j} J_{ij} s_i s_j + \sum_i \phi_i s_i$, and let $H_0^{(t)} = \sum_i \chi_i^{(t)} s_i$, where $\chi_i^{(0)} = \phi_i$ and $s_i \in \{0, 1\}$. If we set $\chi_i^{(t+1)} = \sum_j J_{ij} \langle s_j \rangle + \phi_i$, and let $F_0^{(t)} \coloneqq \langle H \rangle_{H_0^{(t)}} - S^{(t)}$, where $S^{(t)}$ is the entropy of the system according to Hamiltonian $H_0^{(t)}$. Then*

$$F_0^{(t+1)} \le F_0^{(t)} \tag{2}$$

From the definition of s_i, it is easy to see that $\langle s_i \rangle_{H_0^{(t)}} = 1 \times Pr(s_i = 1) + 0 \times Pr(s_i = 0) = Pr(s_i = 1) = \text{sigmoid}(\chi^{(t)})$, and the above statement is equivalent to the corollary below.

Corollary 2.1. *The $Pr(s_i = 1)$ that is an optimal solution to $F_0^{(t)}$ is $Pr(s_i = 1) \propto \exp[\sum_j J_{ij} \langle s_j \rangle_{H_0^t} + \phi_i]$, and the partition function can be found trivially.*

This establishes the validity of our algorithm; during each iteration the update rule finds a better solution to the full potential system (i.e. H with both unary and binary potentials). Through this procedure, we can unroll our attention mechanism upto step k, and by letting the attentions interact with one another, we are able to plot an interaction plot (section 3.3) that reveals how the network decides to make the current output.

3 Experiments and Dataset

We compare our proposed method, i.e., Mean-Field Attention (MFA), with 5 baseline models: RNN [16], LSTM [14], LSTM with Attention [3], Bidirectional LSTM [24] and VAE Attention [5] on two different tasks: chunk counting task and sorting task. VAE Attention is a model specially designed for increasing output diversity and so is not expected to be strong at benchmark tasks; however, we found it interesting to compare with because our model is specially designed for providing better interpretability to the deep attention models but not for competing on benchmark tasks.

3.1 Chunk Counting Task

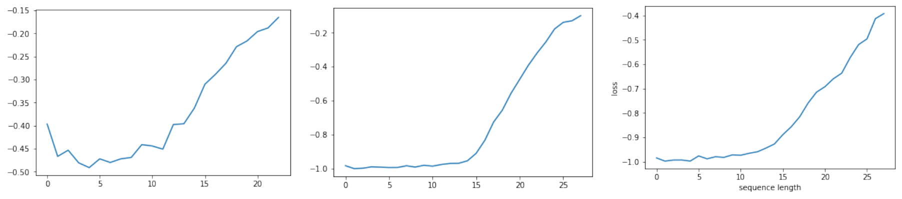
This is a task to demonstrate that MFA is capable of learning a task in which long range relations need not be considered. Given a sequence of 0 and 1, we want to count the number of chunks of ones, for example, if $x = 0010010111$ then we want to output $y = 3$. The training dataset consists of random drawing of proper sequences of length 10 to 30 for about 100000 examples, and testing is done on sequences of length 1 to 100, and so we see whether the model is capable of generalizing to lengths shorter than and longer than the lengths it has observed. This error computed using MSE.

3.2 Sorting Task

The task is to sort a sequence of integers from the smallest to the largest. For example, if $x = [0, 8, 2, 3, 5, 62, 34]$ then we want to output $y = [0, 2, 3, 5, 8, 32, 62]$. This task has become a standard testbed for attention networks [11, 12]; we argue that this task is an essential task to distinguish between attention models and non-attention models because sorting problem has inherent complexity $n \log n$ while a simple RNN structure only has computation complexity n where n is the length of the sequence, and so a network without attention can never learn a proper sorting algorithm. The training data consists of generated integer sequences of length 5 to 15 and testing is performed on sequence of length 1 to 25 in NLL loss. In principle, an RNN with attention should be capable of solving this since its complexity is n^2. From Table 1, our proposed method outperforms other baseline models. While both the standard attention and MFA learned where to focus, it looks like they have learned different strategies of focusing. Our model is also capable of generalizing to sequences of unseen length. Using negative log-likelihood as the error metric, we plot the performance of 3 selected models in Figure 1. A simple LSTM fails to learn perfectly even on the lengths it has observed. When comparing LSTM attention with MFA, we notice that the error of MFA grows slower as the sequence length increases.

Table 1: Performance comparison (lower is better, counting is in MSE, and Sorting is in NLL). For MFA$_k$, the subscript k refers to the maximum number of iterations allowed in the model (see line 6 of algorithm 1).

Model	Counting	Sorting
RNN [16]	720.32	-2.34
LSTM [14]	145.30	-9.13
Bidirectional LSTM [24]	215.91	-8.53
LSTM with Attention [3]	571.64	-18.43
VAE Attention	198.23	-8.44
MFA$_1$	**106.62**	**-20.76**
MFA$_5$	**76.13**	**-25.12**



(a) LSTM (b) LSTM w/ attention (c) MFA

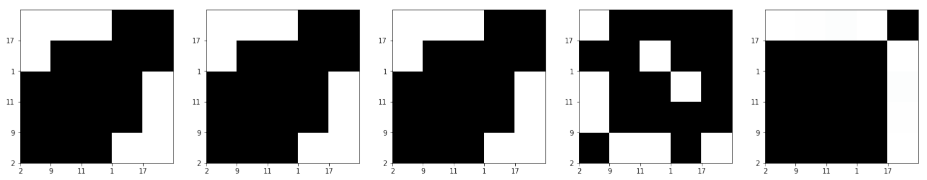
Figure 1: The y axis is negative log-likelihood loss, and the x is the length of testing sequence.

3.3 Interaction Map of Attended Inputs

In this section, we show the interaction relationships between each of the inputs as a heat map, i.e. we plot J_{ij}. This offers new interpretability for deep learning models. The following example shows J_{ij} for a sorting problem. Notice that the output has temporal dimension $t \in 1, .., T$ where $T = 5$, so we expect to have 5 different J_{ij}'s for each of the outputs. Notice that J_{ij} has a graphical structure and we plot a minimum spanning tree on J_{ij} to show the most significant interaction edges.

4 Discussion

This paper proposes a new graphical attention model that adds an additional level of interpretability to deep neural networks. Our method is GPU-efficient and end-to-end trainable. Our model outperforms the baseline models, generalizes to longer unseen sequences and produces an attention comparable or even better than the standard attention models. Future directions involve testing our model thoroughly on larger datasets and using the newly allowed interpretability to shed light on interesting language analysis tasks such as machine translation [4] and multimodal sentiment analysis [21] where the interaction between inputs are complex and dynamic.



(a) t = 1 (b) t = 2 (c) t = 3 (d) t = 4 (e) t = 5

Figure 2: Let white denote connection and black denote non-connection. In this problem, the input is $[2, 9, 11, 1, 17]$, and the network correctly sorts the input: $[1, 2, 9, 11, 17]$. At step 5, we see that all the other inputs have strongest interaction with 17, which is output of the last time step; this suggests that at this step, the network learned to not to consider any interaction between the rest of input. It only needs to identify and output the largest element.

References

[1] E. Agliari, A. Barra, A. Galluzzi, D. Tantari, and F. Tavani. A walk in the statistical mechanical formulation of neural networks. *ArXiv e-prints*, July 2014.

[2] Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Spin-glass models of neural networks. *Phys. Rev. A*, 32:1007–1018, Aug 1985.

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv e-prints*, September 2014.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[5] H. Bahuleyan, L. Mou, O. Vechtomova, and P. Poupart. Variational Attention for Sequence-to-Sequence Models. *ArXiv e-prints*, December 2017.

[6] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks. *ArXiv e-prints*, June 2018.

[7] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018.

[8] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. *CoRR*, abs/1506.07503, 2015.

[9] Jason Eisner. Inside-outside and forward-backward algorithms are just backprop. In *Proceedings of the EMNLP Workshop on Structured Prediction for NLP*, 2016.

[10] Richard P. Feynman. *Statistical Mechanics: A Set of Lectures*. Advanced Book Classics. Boulder, Colo. ; Oxford : Westview Press, 1998.

[11] A. Graves, G. Wayne, and I. Danihelka. Neural Turing Machines. *ArXiv e-prints*, October 2014.

[12] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio G. Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià P. Badia, Karl M. 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*, 2016.

[13] Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *CoRR*, abs/1506.03340, 2015.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[15] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward controlled generation of text. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[16] L. C. Jain and L. R. Medsker. *Recurrent Neural Networks: Design and Applications*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1999.

[17] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. Structured attention networks. *CoRR*, abs/1702.00887, 2017.

[18] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 109–117. Curran Associates, Inc., 2011.

[19] Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. Dynamic evaluation of neural sequence models. *arXiv preprint arXiv:1709.07432*, 2017.

[20] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 2001.

[21] Paul Pu Liang, Ziyin Liu, Amir Zadeh, and Louis-Philippe Morency. Multimodal Language Analysis with Recurrent Multistage Fusion. *ArXiv e-prints*, page arXiv:1808.03920, August 2018.

[22] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002.

[23] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*, 2017.

[24] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November 1997.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[26] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. *The Penn Treebank: An Overview*, pages 5–22. Springer Netherlands, Dordrecht, 2003.

[27] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.

[28] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015.

[29] Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. Breaking the softmax bottleneck: a high-rank rnn language model. *arXiv preprint arXiv:1711.03953*, 2017.

[30] V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D. Reichert, T. Lillicrap, E. Lockhart, M. Shanahan, V. Langston, R. Pascanu, M. Botvinick, O. Vinyals, and P. Battaglia. Relational Deep Reinforcement Learning. *ArXiv e-prints*, June 2018.

[31] F. Zamponi. Mean field theory of spin glasses. *ArXiv e-prints*, August 2010.

[32] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional Random Fields as Recurrent Neural Networks. *ArXiv e-prints*, February 2015.

[33] Konrad Zołna, Devansh Arpit, Dendi Suhubdy, and Yoshua Bengio. Fraternal dropout. *stat*, 1050:16, 2017.

A Proofs

Here the proof is presented in the physics style. For a standard treatment in the probabilitic graphical model literature, see [27], also see [22] for a more physical treatment.

A.1 Theorem 1

The proof for theorem 1 is standard in many statistical mechanics tasks. We point the readers to [10] for a clever and interesting proof in physics.

A.2 Proof for Theorem 2

Let H_0 be the energy of a non-interacting system:

$$H_0 = \sum_{i=1}^{N} \chi_i(s_i) \tag{3}$$

where s_i are defined over the allowable states on the graph $G = (V = \{1, ..., N\}, E = \binom{N}{2})$. In our work, we consider a case where s_i is a Bernoulli variable $\in \{0, 1\}$. The mean field approximation finds the optimal approximation to the full interactive graph using a factorizable energy function.

The full Hamiltonian is a Hamiltonian with pairwise interaction (i.e. fully connected undirected graph):

$$H(s_1, ..., s_N) = \sum_{i,j}^{N,N} J_{ij}(s_i, s_j) + \sum_{i}^{N} \phi_(s_i) \tag{4}$$

Recalling that the free energy $F_0 = H - S = H + \sum P \log P$ is simply the energy of the system with an entropic smoothing term. In this case, we have that the free energy

$$F_0 = \sum_{s_1,...,s_m} H(s_1, s_N) Q_0(s_1, ..., s_N) + \sum_{s_1,...,s_m} Q_0(s_1, ..., s_N) \log Q_0(s_1, ..., s_N) \tag{5}$$

where Q_0 is the probability calculated with respect to the approximate Hamiltonian H_0. Since H_0 contains no binary potential, Q_0 factorizes:

$$Q_0(s_1, ..., s_N) = \frac{1}{Z_0^(N)} \exp[-H_0(s_1, ..., s_N)] = \prod_{i=1}^{N} \frac{1}{Z_0} \exp(-\chi_i(s_i)) = \prod_{i=1}^{N} q_0(s_i) \tag{6}$$

where $Z_0, Z_0^{(N)}$ are the corresponding partition functions, and q_0 are variational probability of state s_i. Notice that, since we take $s_i \in \{0, 1\}$, $q_0 = sigmoid(\chi_i(1))$, now plug into Equation 5 we obtain:

$$F_0 = \sum_{i,j} \sum_{s_i,s_j} J_{ij} q_0(s_i) q_0(s_j) + \sum_{i} \sum_{s_i} \phi_i(s_i) + \sum_{i=1}^{N} \sum_{s_i} q_0(s_i) \log q_0(s_i) \tag{7}$$

Now we want to find a way to update χ_i such that F_0 is minimized after the update. We take the derivative of F_0 with respect to $q_0(s_i)$ and set it to 0 to obtain:

$$q_0(s_i) = \frac{1}{Z_0} \exp[-\chi_i^{(t+1)}(s_i)] \tag{8}$$

where $\chi_i^{(t+1)}(s_i)$ can be identified to be $\sum_j \sum_{s_j} J_{ij}(s_i, s_j) q_0(s_j) + \phi_i(s_i)$. This finishes our derivation of the update rule (along with the corollary). Now since the q_0 found this way is always a local minimum of F_0, we conclude that after each update:

$$F_0^{t+1} \leq F_0 \tag{9}$$

As an additional remark, to arrive at our update rule (line 8 of Algorithm 1), one simply let $J_{ij}(s_i, s_j) \to J_{ij} s_i s_j$, $\phi_i(s_i) \to \phi_i s_i$ and $\chi_i(s_i) \to \chi_i s_i$. Notice that we only used a very basic function for this work, while the theorem above allows for the use of very complicated potential functions such as that being parametrized by a neural network, and this flexibility and generality of the proof allows for a much broader family of models to be chosen and experimented with.

Algorithm 2 `Encoder-Decoder`$(\{\text{word}\}_i)$

```
1: Input: $\{\text{word}_i\}_i$ // The input sequence such as a sequence of words.
2: Initialize $d_0 = \mathbf{0}$, $t = 0$
3: $h_i \leftarrow \texttt{Encoder}(\text{word}_i)$ // For a sentence of length $T$, $i \in \{1, ..., T\}$
4: while $d_t \neq$ END do
5:     $t \leftarrow t + 1$
6:     $a_i^{(t)} \leftarrow \texttt{Mean-field-Attention}(\cup_i^T h_i, d_t)$
7:     $c \leftarrow \sum_i a_i h_i$
8:     $d_t = \texttt{Decoder}(d_{t-1}, c)$
9: Output: $\{d_t\}_t$ // The sequence we want to output
```

B Algorithm for Encoder-Decoder Networks

See Algorithm 2. Notice that the algorithm is completely general and line 6 can be replaced by any other attention mechanism.

C Additional Experiments

C.1 Learned Attention on the chunking task

We then plot the attention of our proposed model and LSTM with attention model for further comparison. The attention of LSTM with attention model seems very imbalanced, suggesting that it has learned an incremental strategy of using the attention mechanism, whereas our proposed model learns to focus on the chunk of ones and it is very balanced.

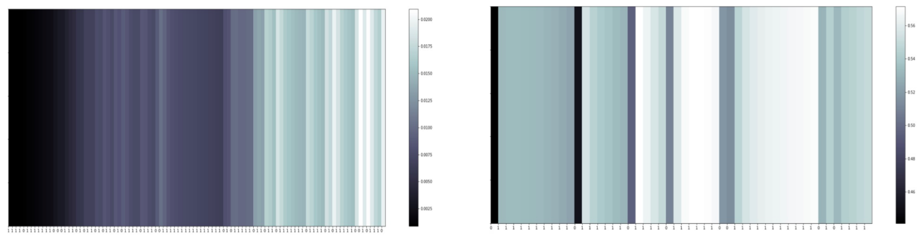

(a) (b)

Figure 3: (a) LSTM with Attention for Chunk Counting. Notice that the traditional attention mechanism outputs an attention that is highly imbalanced and hard to interpret. (b) MF-CRF Attention for Chunk Counting, notice that the attention correctly focuses on the chunk of 1 in a balanced way.

C.2 Learned Attention on the sorting task, and the effect of iteration.

In Figure 4, we show some example learned attentions and show the effect of iteration at refining the attention. We see that the output attention becomes

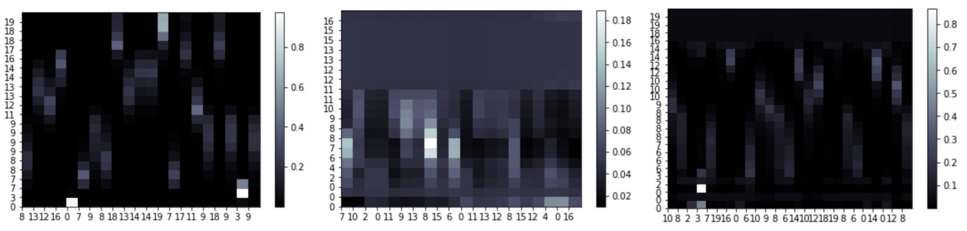

(a) (b) (c)

Figure 4: (a) LSTM with Attention for Sorting. (b) MFA attention for Sorting at iteration 1, we see that it is mostly blurred. (c) MFA attention for Sorting at iteration k, notice that the attention becomes much more accurate and sharpened than at iteration 1.

C.3 Language Modeling Task

Besides the two artificial task, we present a preliminary experiment to study the performance of our model on two real dataset, Penn Treebank [26] and WikiText-2 datasets. PTB is derived from articles of the Wall Street Journal. It contains 929k training tokens and a vocab size limited to 10k words. It is one of the most commonly used benchmarks in language modelling. WikiText-2 is roughly twice the size of PTB, with 2 million training tokens and a vocab size of 33k. Due to the limited time and resource we have for this task, the current result should not be regarded as a actual measure of the performance of the model. In the near future we will finish this part of the experiment. Also notice that at this stage our method is not yet directly comparable to the SOTA results, since our setup is very basic to avoid confounding variables and so does not include most of standard tricks in the NLP community.

Table 2: Language Modelling for Penn Treebank

Methods	Perplexity	Val
[29]	47.69	-
[19]	51.1	-
[23]	52.8	-
[33]	56.8	-
RNN	138.48	144.91
LSTM	127.66	136.82
MFA	109.76	118.21

Table 3: Language Modelling for WikiText-2

Methods	Perplexity	Val
[29]	40.68	-
[19]	44.3	-
[23]	52.0	-
[33]	64.1	-
RNN	164.69	178.01
LSTM	131.32	146.31
MFA	118.31	121.01

D Connection to CRF models

In this section, we briefly link our method to CRFs. In some sense, We are actually combining CRFs with recurrent neural networks dynamically by using a fully connected CRF to model the structural dependencies of the observed data and output the attention vector for the recurrent network. Let T denote the temporal dimension, and in this case it refers to the length of the English sentence, since, at every time step t, we feed in the next word in the sentence an LSTM encoder-decoder. The network outputs an encoded state, h_t for every time step $t \in \{1, ..., T\}$. In a network without attention, the hidden state of the last time step is given to the decoder, which is usually another Recurrent neural network (RNN). The decoder outputs a distribution over the target vocabulary (all possible French words), and usually the word with the highest probability is chosen as the predicted translation. The decoder continues to predict words until it outputs a special symbol that signifies the ending of the translation.

We define the attention vector as a distribution $p(s|\mathbf{h}, q)$ that is modeled as a conditional random field (CRF), where $\mathbf{s}$ represents the position of the feature vector to be attended by our model, $\mathbf{H}$ is a sequence of the hidden states of the encoder, and q is the hidden state of the decoder. In particular, $\mathbf{s}$ is a vector of discrete latent variables $[s_1, ..., s_m]$, and s_i is either 1 or 0 and has conditional probability of being 1 with probability $p(s_i|\mathbf{h}, q)$. In this sense, we model the structure of the hidden states as a graph with m nodes, and the graph is parameterized with cliques whose potentials are defined by θ_C, where C denotes the clique.

The attention distribution is given by:

$$p(\mathbf{s}|\mathbf{H}, q) = \frac{1}{Z}\exp(\sum_{C}\theta_C(\mathbf{s}))$$

where C denotes the cliques in our graphical model, and we use this distribution as the weights to the sequence memories we want to attend on. In this formulation, the context vector is simply the expected value of $\mathbf{x} = s_i h_i$:

$$\mathbf{C_t} = \sum_{i}^{m} \Pr(s_i = 1|\mathbf{H}, q)h_i = \mathbb{E}_{s\sim p(s|\mathbf{H},q)}[\mathbf{x}] \tag{10}$$

Exact solution on some simple structures can be obtained through the forward-backward algorithm [20] and the inside-outside algorithm [9]. While a mean field procedure can be used to improve computation efficiency in the expense of exact inference. We build an undirected graph on the hidden state $\mathbf{H}$ and then we can define states (of the CRF) s_i on each hidden state, and the states are connected to each other through binary and unary potentials.