Learning Sparse Hidden States in Long Short-Term Memory

Niange Yu[1], Cornelius Weber[2], and Xiaolin Hu[1](Corresponding Author)
[1] State Key Laboratory of Intelligent Technology and Systems, Beijing National Research Center for Information Science and Technology, Department of Computer Sicence and Technology, Tsinghua University, Beijing 100084, China
niangeyu@gmail.com, xlhu@tsinghua.edu.cn
[2] Department of Informatics, University of Hamburg, Hamburg, Germany
weber@informatik.uni-hamburg.de

ICANN19
28th International Conference on Artificial Neural Networks

Abstract

Long Short-Term Memory (LSTM) is a powerful recurrent neural network architecture that is successfully used in many sequence modeling applications. Inside an LSTM unit, a vector called “memory cell” is used to memorize the history.

Another important vector, which works along with the memory cell, represents hidden states and is used to make a prediction at a specific step.

Memory cells record the entire history, while the hidden states at a specific time step in general need to attend only to very limited information thereof. Therefore, there exists an imbalance between the huge information carried by a memory cell and the small amount of information requested by the hidden states at a specific step.

We propose to explicitly impose sparsity on the hidden states to adapt them to the required information.

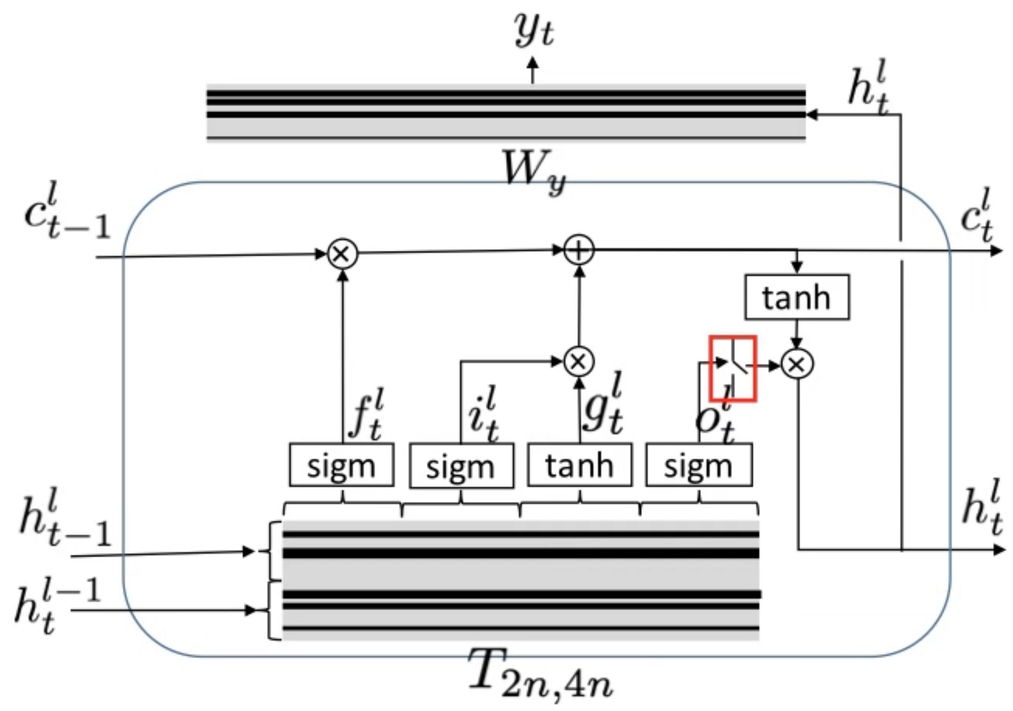

Fig. 1. Sparse LSTM. Given sparse hidden state vectors (h_{t-1}^l, h_t^{l-1}) and h_t^l, only a few rows of the respective transformation matrices $T_{2n,4n}$ and W_y are used.

Method

In order to impose sparsity, we introduce an L1-norm loss term over the output gate at layer l as

$$\mathcal{L}^S(o^l(x,\theta)) = \sum_{1\le b\le B}\sum_{1\le t\le T}|o_t^l(x^{(b)})|.$$

The updating function for w via stochastic gradient decent (SGD) is

$$w \leftarrow w - \left(\eta\frac{\partial\mathcal{L}^{XE}(\theta)}{\partial w} + \sum_{1\le l\le L}\lambda_l\frac{\partial\mathcal{L}^S(o^l(x,\theta))}{\partial w}\right)$$

Experiments

- Language Modeling

The language modeling experiments are conducted on the Penn Treebank dataset. We use the PyTorch implementation of the publicly available baseline model4. The distribution of the output gates o in the baseline model is shown bellow.

Fig. 2. Distributions of the output gate values o_t^l as defined in Eq. (9) for three pre-trained models. $l = 1$ and $l = 2$ stands for the first and second layers respectively. The values are calculated with a batch size of 50 and with 35 unrolled steps $(t = 1, ..., 35)$ on the test set.

The results are shown in Table 1. All the models are trained from scratch in the same setting with the baseline. “Test Perplexity” denotes the perplexity on the test set; lower is better.

Table 1. Learning sparse output gates from scratch.

Method	$\xi, \lambda_1/\lambda_2(10^{-6})$	Test Perplexity	Output Width (1st, 2nd) LSTM	Mult-add* Reduction	Time(ms)/ Speed-Up
baseline	(0.0, 0.0/0.0)	77.88	(1500, 1500)	1.00×	81.7/1×
Ours	(0.1, 0.4/0.4)	77.14	(720, 953)	1.56×	53.3/1.53×
	(0.1, 1.0/1.0)	**76.97**	(436, 797)	1.89×	44.6/1.83×
	(0.1, 1.6/1.6)	77.05	(344, 706)	2.09×	38.2/2.14×
	(0.1, 1.6/4.8)	77.85	**(261, 344)**	**2.75**×	**29.9/2.73**×

*The reduction of multiplication-add operations in matrix multiplications.

Our method can be embedded into ISS easily by adjusting ξ to control the output width. The experimental results are listed bellow

Table 2. ISS with sparse hidden states

Method	Test Perplexity	Memory/Output Width (1st, 2nd) LSTM	Mult-add reduction	Time(ms)/ Speed-Up
baseline*	78.57	(1500, 1500)/(1500, 1500)	1.00×	81.7/1.00×
ISS	78.65	(373, 315)/(373, 315)	7.48×	14.1/5.79×
ISS+Ours ($\xi = 0.10$)	**76.27**	(379, 536)/(229, 445)	5.99×	17.2/4.75×
ISS+Ours ($\xi = 0.20$)	78.37	(194, 542)/(63, 375)	**8.63**×	**13.0/6.28**×

*The baseline is different from the baseline in Tab 1.

- Image Captioning

We experiment our methods with the image captioning model Google NIC as a baseline. The model is trained and evaluated on MSCOCO, which is a widely used benchmark for image captioning. The results are shown bellow.

Table 3. The results of Google NIC with different memory size and ξ on the MSCOCO Karpathy test split. B4, M, R-L and C stand for BLEU4, METEOR, ROUGE-L and CIDEr respectively. Time is in ms.

Model	Time	B4	M	R-L	C
[18]	-	31.3	26.0	54.3	101.3
[1]	-	33.4	26.1	**54.4**	**105.4**
NIC 512	239	**33.5**	25.8	**54.4**	101.7
NIC 1024	503	32.6	26.0	54.1	101.8
NIC 2048	1070	32.3	26.1	53.9	102.1
NIC 2048 ($\xi = 0.3$)	721	33.3	**26.4**	**54.4**	103.0
NIC 2048 ($\xi = 0.4$)	616	32.9	**26.4**	**54.4**	103.3
NIC 2048 ($\xi = 0.5$)	436	32.7	26.0	53.9	102.9

Conclusion

We explore novel sparse LSTM networks in which the activations of the output gates are sparsified. This reduces the amount of information that is passed on for further processing, while not impacting on the memory of the LSTM cells. Experiments were conducted on three tasks including language modeling and image captioning. The proposed method obtains better performance on all tasks at lower computational costs.