Statistical Language Models Based on RNN

Liu Rong
CSLT of Tsinghua University
2014-10-9

Outline

- Language Model
- Neural Network Based Language Model
  - Standard NNLM
  - RNN
- RNN with LSTM
  - Introduction
  - Toolkit
- References

N-gram-Introduction

- N-Gram
  - A language model is usually formulated as a probability distribution *p*(*s*) over strings *s* that attempts to reflect how frequently a string *s* occurs as a sentence

$$p(s) = p(w_1, w_2, \cdots, w_k) = p(w_1)p(w_2|w_1)\cdots p(w_k|w_1, w_2, \cdots, w_{k-1})$$

  - n-gram

$$p(s) = \prod_{i=1}^{l+1} p(w_i|w_{i-n+1}^{i-1}) \qquad p(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{\sum_{w_i} c(w_{i-n+1}^{i})}$$

  - example(3-gram)

    eg: “This is a example”

    p(This is a example) ≈p(This|<s>)p(is|This <s>)p(a| <s> This is)
    p(example|This is a)p(</s>| is a example)

    => to calculate the p(example|This is a)

    N-Gram: p(example|This is a) = $\frac{C(This\ is\ a\ example)}{C(This\ is\ a)}$

NNLM-Introduction

A Neural Probabilistic Language Model(Bengio et al, NIPS'2000 and JMLR 2003)

- Motivation:
  - LM does not take into account contexts farther than 2 words.
  - LM does not take into account the "similarity" between words.
- Idea:
  - A word w is associated with a distributed feature vector (a real-valued vector in $\mathbb{R}^n$ n is much smaller than size of the vocabulary)
  - Express joint probability function f of words sequence in terms of feature vectors
  - Learn simultaneously the word feature vector and the parameters of f

NNLM-Intoduction

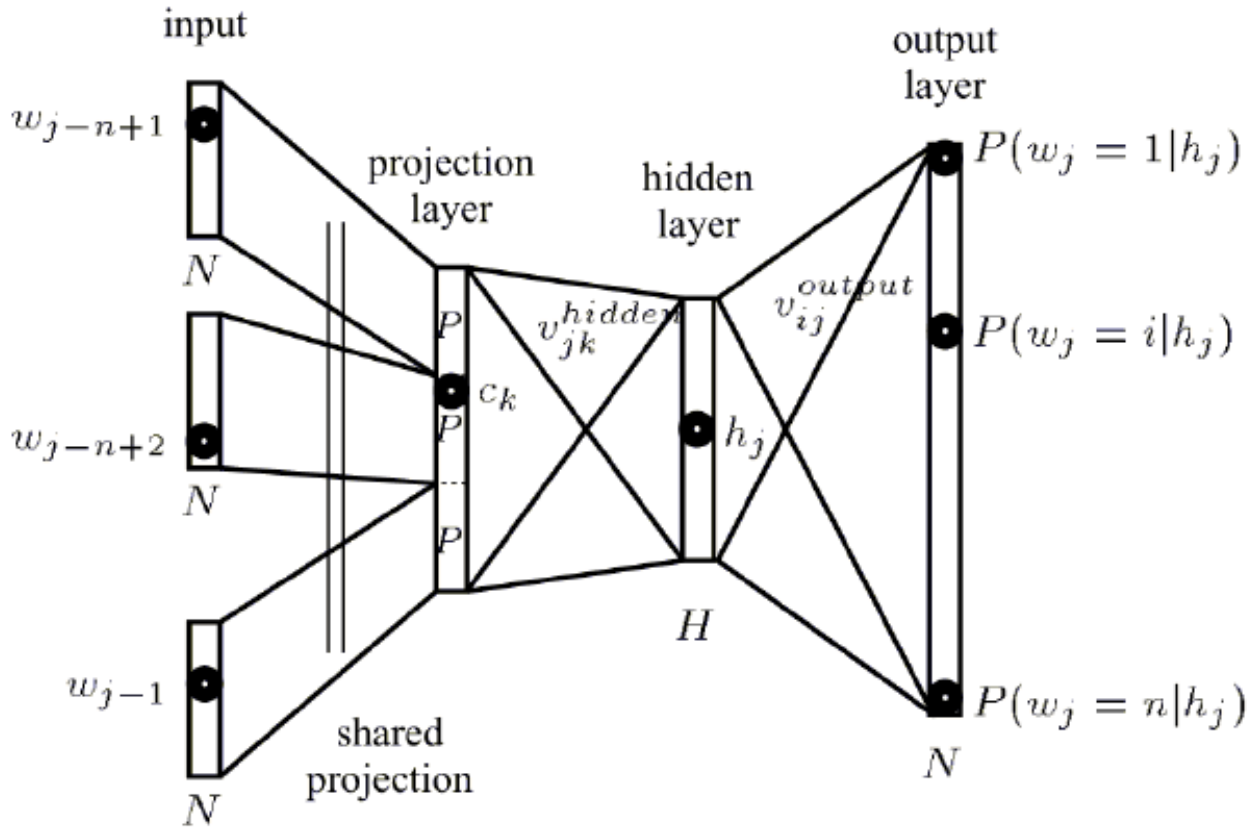


Figure: *Feedforward neural network based LM used by Y. Bengio and H. Schwenk*

[1] Y. Bengio, and R. Ducharme. A neural probabilistic language model. In *Neural Information Processing Systems,* volume 13, pages 932-938. 2001.

NNLM--toolkit

- CSLM Toolkit http://www-lium.univ-lemans.fr/cslm/

  Holger Schwenk; *CSLM - A modular Open-Source Continuous Space Language Modeling Toolkit*, in Interspeech, August 2013.

- NPLM Toolkit http://nlg.isi.edu/software/nplm/

Decoding with large-scale neural language models improves translation. Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang, 2013. In*Proceedings of EMNLP*.

RNNLM-description

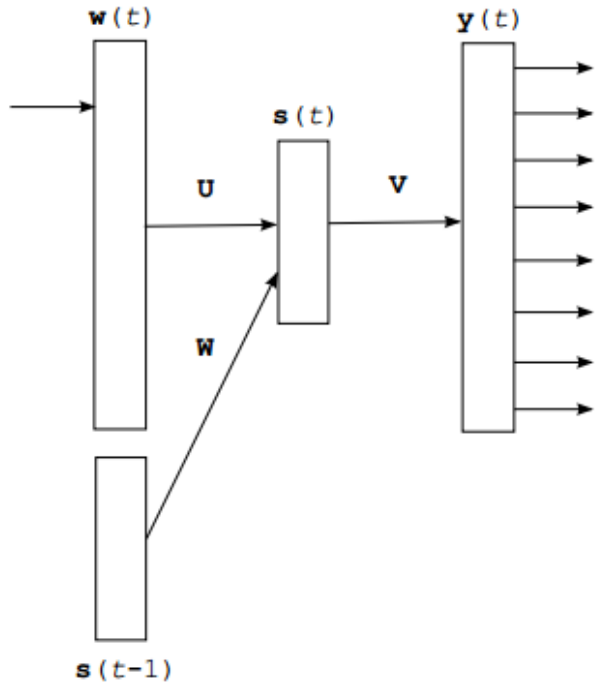
Forward:

$$s(t) = f\big(Uw(t) + Ws(t-1)\big) \quad (1)$$

$$y(t) = g\big(Vs(t)\big) \quad (2)$$

where $f(z) = \frac{1}{1+e^{-z}}, g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}$.

$w(t), y(t)$ *is encoded as* $1 - of - V$ *vector*



BP:

$$e_o(t) = d(t) - y(t) \quad (3)$$

$$V(t+1) = V(t) + s(t)e_o(t)^T\alpha \quad (4)$$

$$U(t+1) = U(t) + w(t)\big[e_o(t)^T Vs(t)\big(1 - s(t)\big)\big]\alpha \quad (5)$$

$$\mathrm{W}(t+1) = W(t) + s(t-1)\big[e_o(t)^T Vs(t)\big(1 - s(t)\big)\big]\alpha \quad (6)$$

RNNLM-Backpropagation Through Time

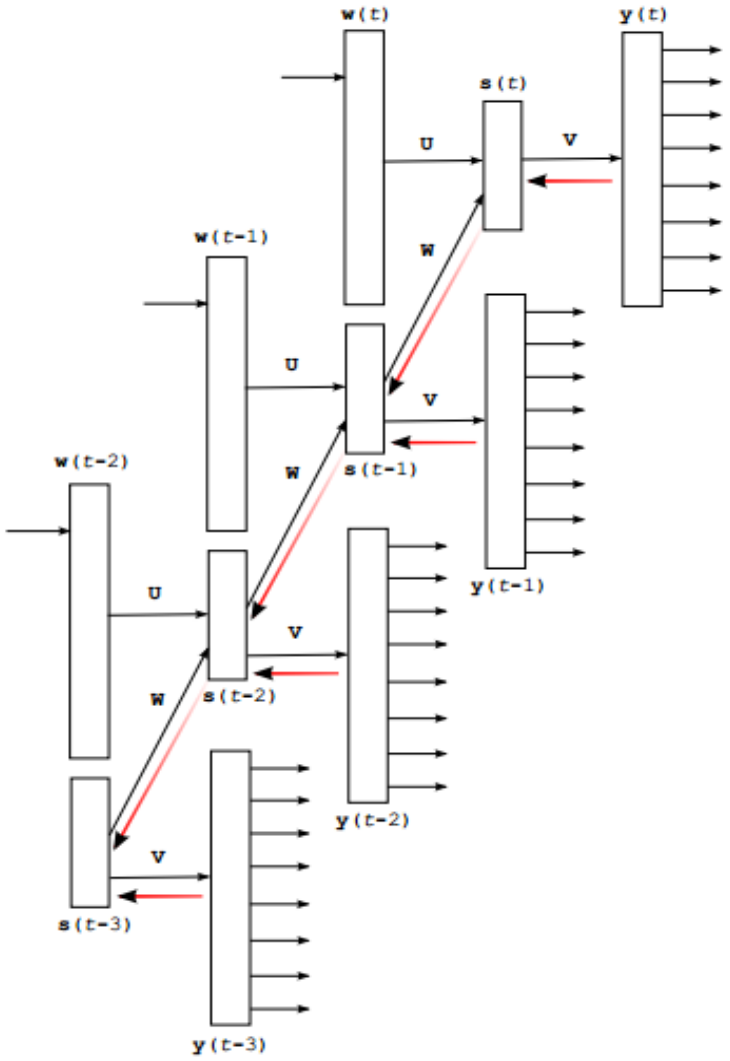
BPTT:

$$W(t+1) = W(t) + \sum_{z=0}^{T} s(t-z-1)\, e_h(t-z)^T \alpha \quad (6)$$

where $e_h(t-z-1) = e_h(t-z)^T W s(t-z-1)\big(1 - s(t-z-1)\big)$

$$e_h(t) = e_o(t)^T V s(t)\big(1 - s(t)\big)$$

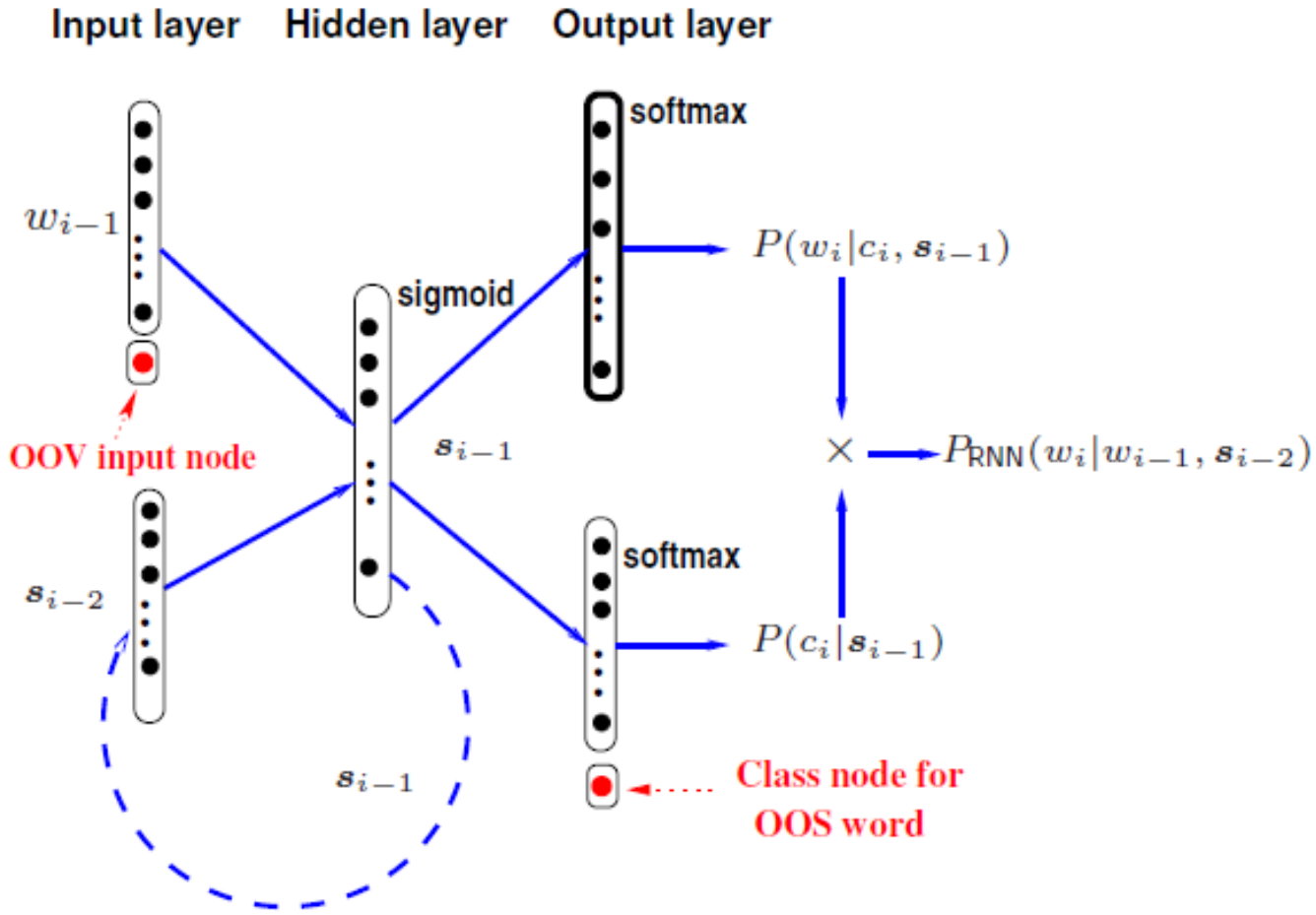
1. The unfolding can be applied for as many time steps as many training examples Were already seen,however the error gradients quickly vanish as they get backpropagated in time,so several steps of unfolding are sufficient.



RNNLM-Classes

1. By using simple classes,can achieve speedups on large data sets more than 100 times

2. Lose a bit of accuracy of the model(usually 5-10% in perplexity)



$$P_{\text{RNN}}(w_i|w_{i-1}, s_{i-2}) = P(w_i|c_i, s_{i-1})P(c_i|s_{i-1}).$$

[] Efficient Lattice Rescoring using Recurrent Neural Network Language Model *X. Liu, Y. Wang, X. Chen..*

RNNLM-Joint Training With Maximum Entropy Model

Hold

RNNLM-LSTM

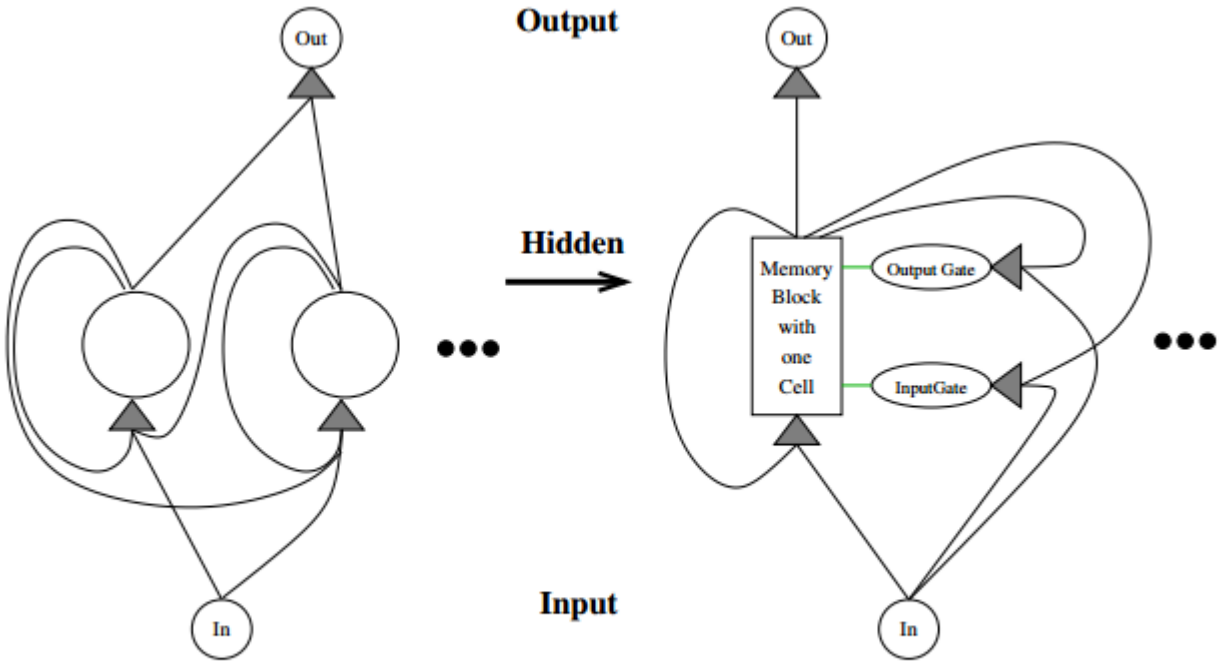


Figure 2.1: Left: RNN with one fully recurrent hidden layer. Right: LSTM network with memory blocks in the hidden layer (only one is shown).

F. A. Gers. Long Short-Term Memory in Recurrent Neural Networks. PhD thesis, Switzerland, 2001. http://felixgers.de/papers/phd.pdf

RNNLM-LSTM

- Traditional LSTM

$$net_{out_j}(t) = \sum_m w_{out_j m}\, y^m(t-1)\ ;\quad y^{out_j}(t) = f_{out_j}(net_{out_j}(t))\ , \tag{2.1}$$

$$net_{in_j}(t) = \sum_m w_{in_j m}\, y^m(t-1)\ ;\quad y^{in_j}(t) = f_{in_j}(net_{in_j}(t))\ . \tag{2.2}$$

$$net_{c_j^v}(t) = \sum_m w_{c_j^v m}\, y^m(t-1)\ ,$$

$$s_{c_j^v}(0) = 0\ ;\quad s_{c_j^v}(t) = s_{c_j^v}(t-1) + y^{in_j}(t)\ g(net_{c_j^v}(t)) \quad \text{for } t > 0\ . \tag{2.6}$$

$$net_k(t) = \sum_m w_{km}\, y^m(t-1)\ ,\quad y^k(t) = f_k(net_k(t))\ , \tag{2.9}$$

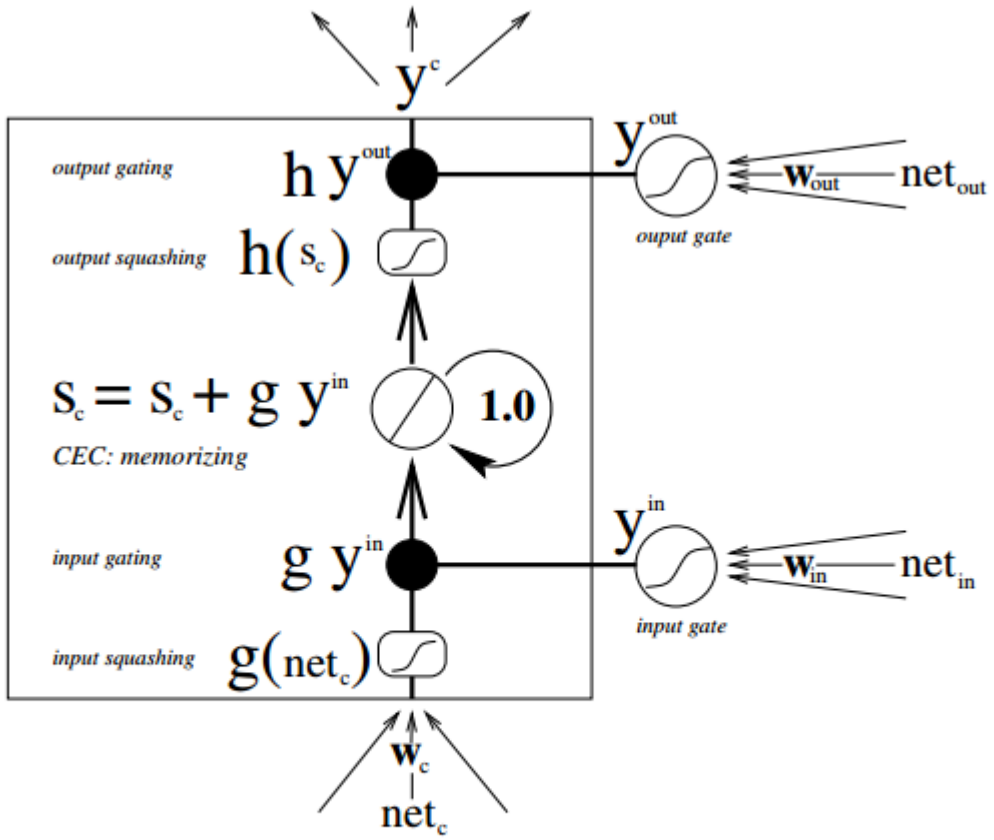


Figure 2.2: The traditional LSTM cell has a linear unit with a recurrent self-connection with weight 1.0 (CEC). Input and output gate regulate read and write access to the cell whose state is denoted s_c. The function g squashes the cell's input; h squashes the cell's output (see text for details).

RNNLM-LSTM

- Traditional LSTM

Hochreiter and Schmidhuber (1997) already solved a wide range of tasks with traditional LSTM: (1) The embedded Reber grammar (a popular regular grammar benchmark); (2) Noise free and noisy sequences with time lags of up to 1000 steps (e.g.; the "2-sequence problem" proposed by Bengio et al., 1994); (3) Continuous-valued tasks the require the storage of values for long time periods and their summation and multiplication (up to a certain precision); (4) Temporal order problems with wildly separated inputs.

RNNLM-LSTM

- LSTM with Forget

$$s_{c_j^v}(t) \;=\; y^{\varphi_j}(t)\; s_{c_j^v}(t-1) + y^{in_j}(t)\;\; g(net_{c_j^v}(t)) \;\;, \qquad (3.2)$$

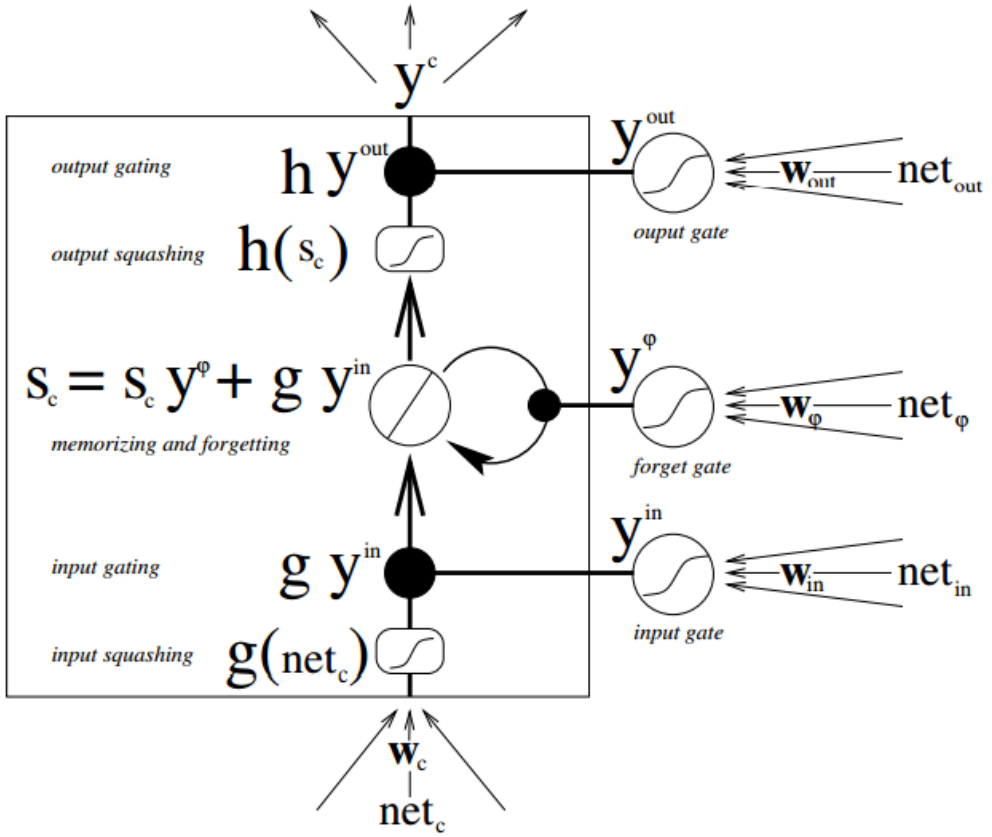
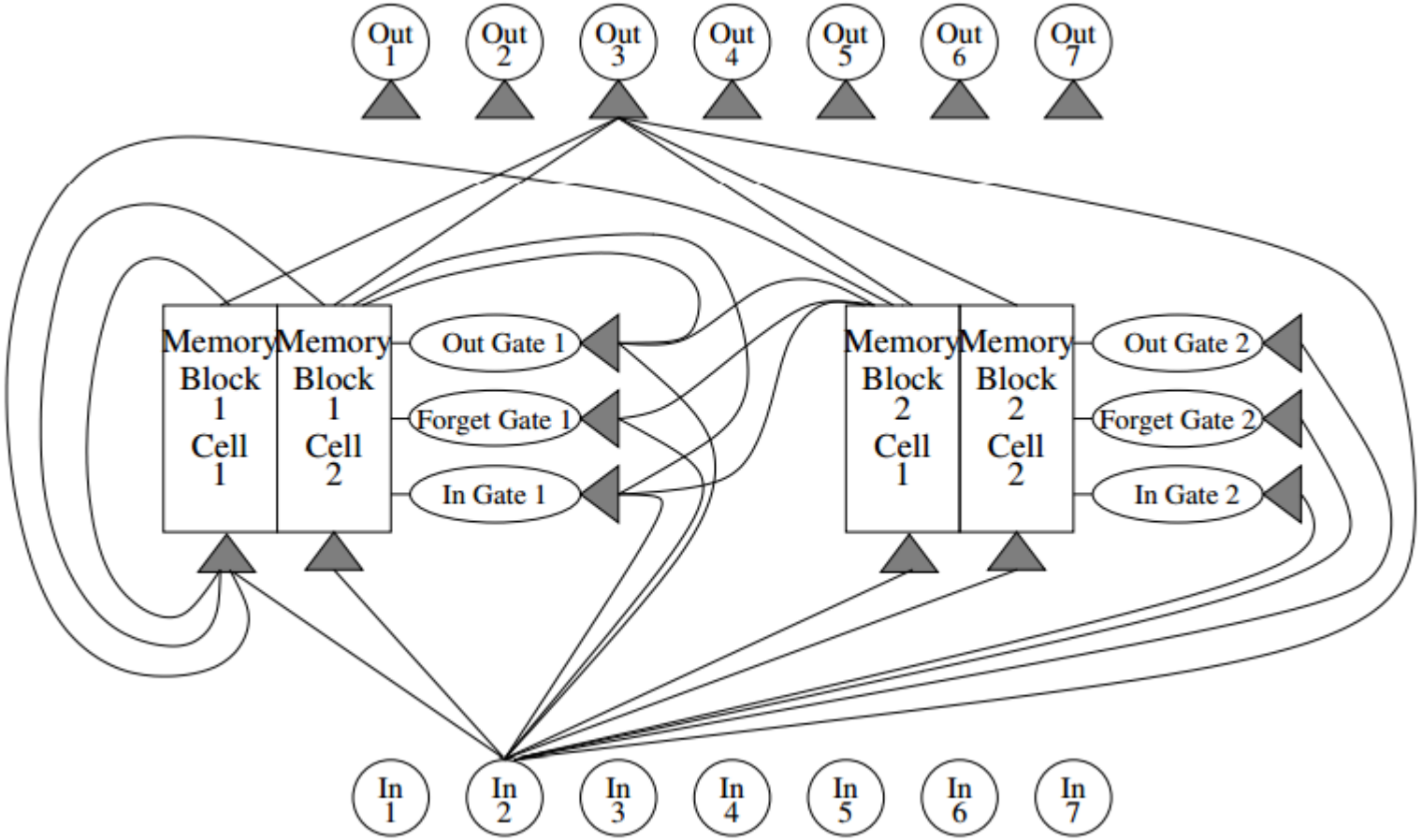


Figure 3.1: Memory block with only one cell for the extended LSTM. A multiplicative forget gate can reset the cell's inner state s_c.

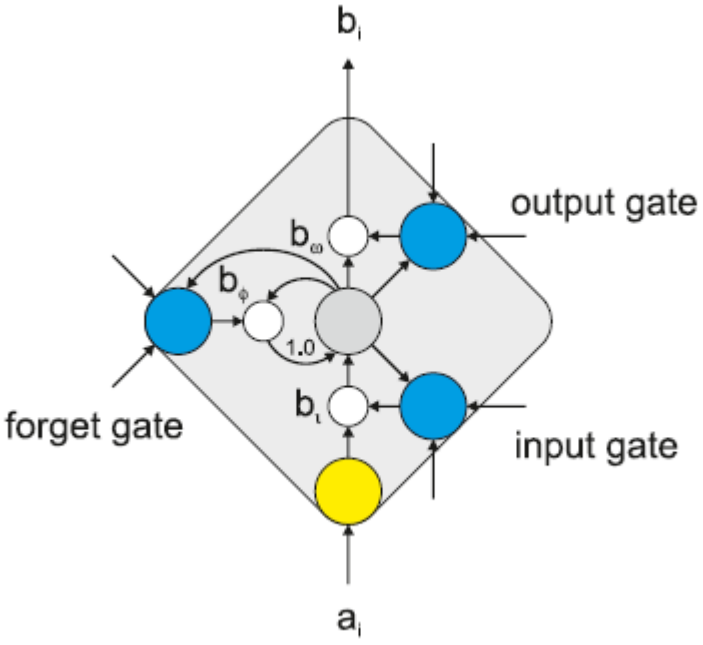
RNNLM-LSTM

- LSTM in Network



RNNLM-LSTM

- LSTM for LM



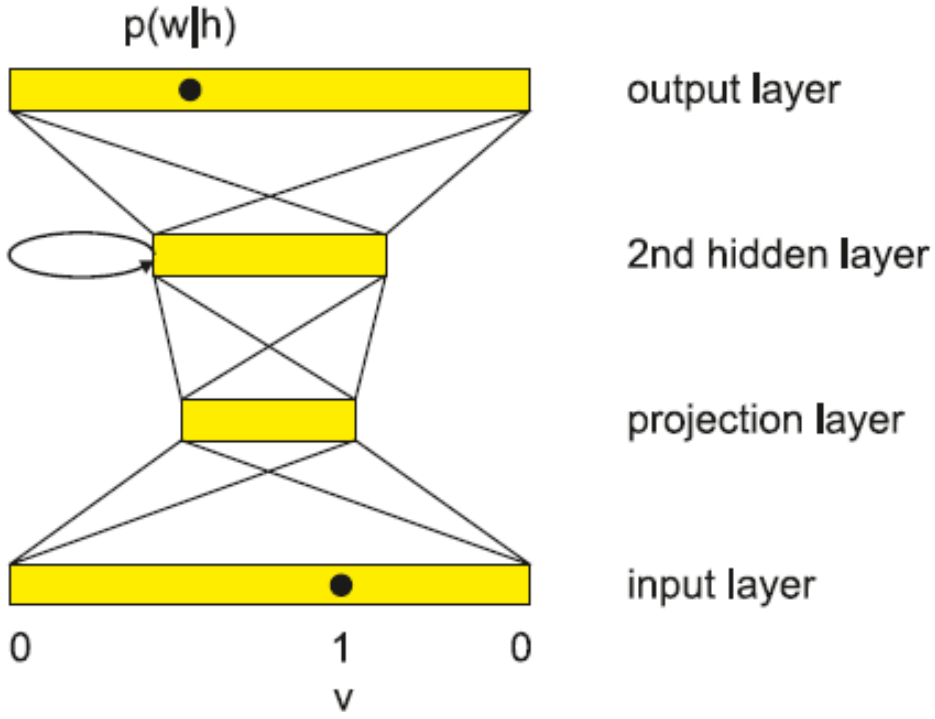


Figure 2: Neural network LM architecture

RNNLM-tool

- LTSM/RNN training, GPU&deep supported http://sourceforge.net/projects/currennt/
- RNNLM: RNN LM toolkit http://www.fit.vutbr.cz/~imikolov/rnnlm/
- RWTHLM: RNN LTSM toolkit http://www-i6.informatik.rwth-aachen.de/web/Software/rwthlm.php
- RNN toolkit from Microsoft http://research.microsoft.com/en-us/projects/rnn/

- Thanks
- Q & A