Neural Networks for Language Modelling

Chunyang Wu

April 10th, 2014

Outline

Outline

Introduction

- A language model assigns probabilities to word sequences, *e.g.*

  p(Welcome to this presentation on language modeling) $=$?

  p(to on modeling Welcome presentation modeling this) $=$?

- For a good language model, meaningful sentences should be assigned to higher scores than the ambiguous ones.

Outline

Naive solution – *n*-gram

We count the number of observations and get

$$P(w|h) = \frac{C(h, w)}{C(h)}$$

where h is a context history with a fixed length of $n-1$; $C(x)$ is the number of the sequence x contained in the training set.
Thus, *e.g.* bi-gram

$$\begin{aligned} P(w_1, w_2, \ldots, w_L) &= P(w_1, w_2, \ldots, w_{L-1}) P(w_L | w_1, w_2, \ldots, w_{L-1}) \\ &= \cdots \approx P(w_1) P(w_2|w_1) \cdots P(w_L|w_{L-1}) \end{aligned}$$

log version:

$$\log P(w_1, w_2, \ldots, w_L) = \log P(w_1) + \log P(w_2|w_1) + \cdots + \log P(w_L|w_{L-1})$$

Why not *n*-gram?

Long distance dependency, *e.g.*

- `Chunyang felt very sad when he realized he should give a talk in the rcc.`

In bi-gram,
$p(\text{realized}|\text{Chunyang felt very sad when he}) = p(\text{realized}|\text{he})$
$p(\text{realizes}|\text{Chunyang felt very sad when he}) = p(\text{realizes}|\text{he})$

Why not *n*-gram?

Word similarity, *e.g.*

- `Today's presentation is given by Chunyang.`
- `Today's presentation is given by Zoubin.`

$p(\textit{Chunyang}|\ldots) =?, p(\textit{Zoubin}|\ldots) =?$

Outline

Motivation

- Projecting the binary word vector into a low dimension space *e.g.*, $dim = 50, 80, 100$?
- Modeling the probability of word sequences in the new feature space?
- Jointly estimating low-dimension feature space & the sequential probability?

Structure

[Y. Bengio et al., JMLR'03]

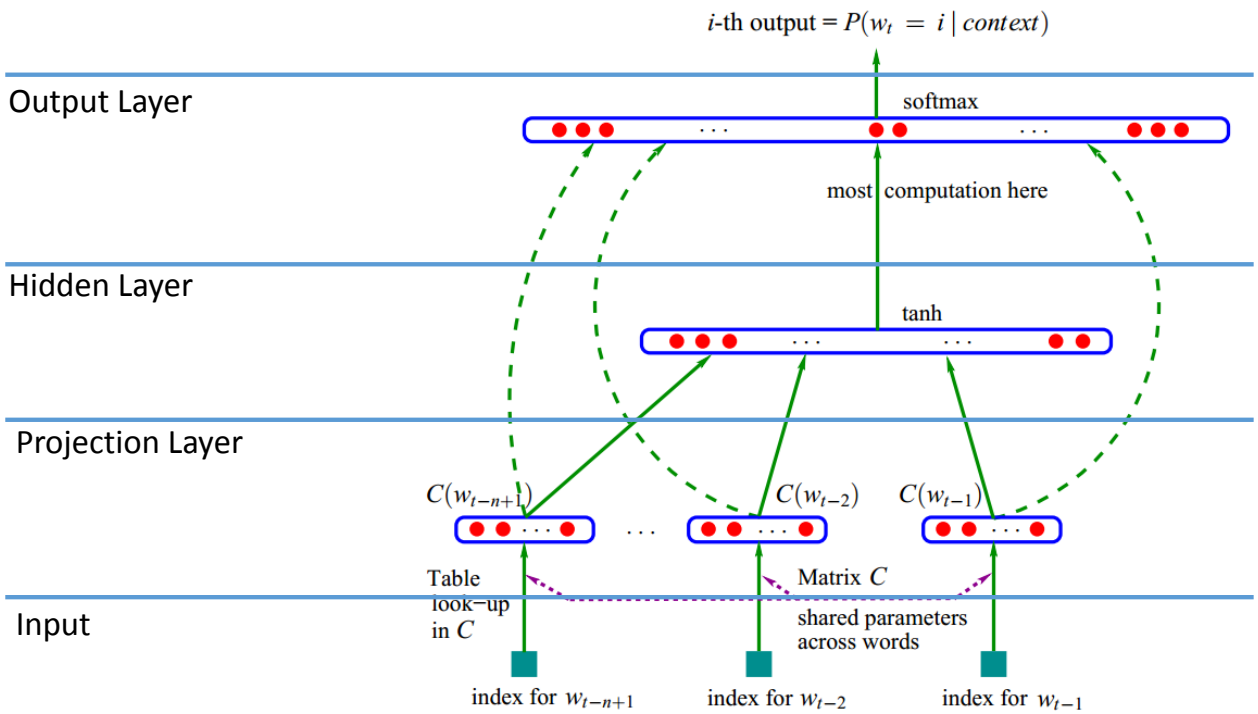


Figure: Feedforward Neural Network LM.

i-th output: $P(w_t = i | w_{t-1}, \ldots, w_{t-n+1})$

Structure – Projection Layer

The output of this layer is

$$\mathbf{g}(\mathbf{w}_{t-n+1}, \ldots, \mathbf{w}_{t-1}) = \begin{pmatrix} \mathbf{C}\mathbf{w}_{t-n+1} \\ \vdots \\ \mathbf{C}\mathbf{w}_{t-1} \end{pmatrix}$$

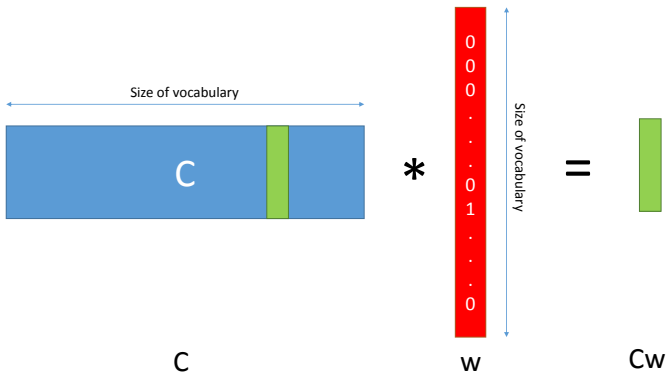


Figure: Projection Layer in Feedforward Neural Network LM.

However, the matrix multiplication is not necessary.

Structure – Hidden & Output Layers

The hidden layer is

$$\mathbf{h}(\ldots) = \tanh(\mathbf{H}^T\mathbf{g}(\ldots) + \mathbf{d})$$

The output layer[1] is

$$P(w_t = i|\mathbf{w}_{t-n+1}, \ldots, \mathbf{w}_{t-1}) = y_i(\ldots) = softmax_i(\mathbf{b} + \mathbf{M}\mathbf{g}(\ldots) + \mathbf{U}\mathbf{h}(\ldots))$$
$$= softmax_i(\mathbf{z}(\ldots)) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

[1] **M** is usually set to be zero.

Limitation of feedforward NNLM

- word similarity ✓
- Long distance dependency $\times$
  Feedforward NNLM is still a language model with a history window of $n - 1$.

Recall

Long distance dependency, *e.g.*

- `Chunyang felt very sad when he realized he should give a talk in the rcc.`

In bi-gram,
$p(\text{realized}|\text{Chunyang felt very sad when he}) = p(\text{realized}|\text{he})$
$p(\text{realizes}|\text{Chunyang felt very sad when he}) = p(\text{realizes}|\text{he})$

RNNLM Structure

[T. Mikolov et al., Interspeech'10]

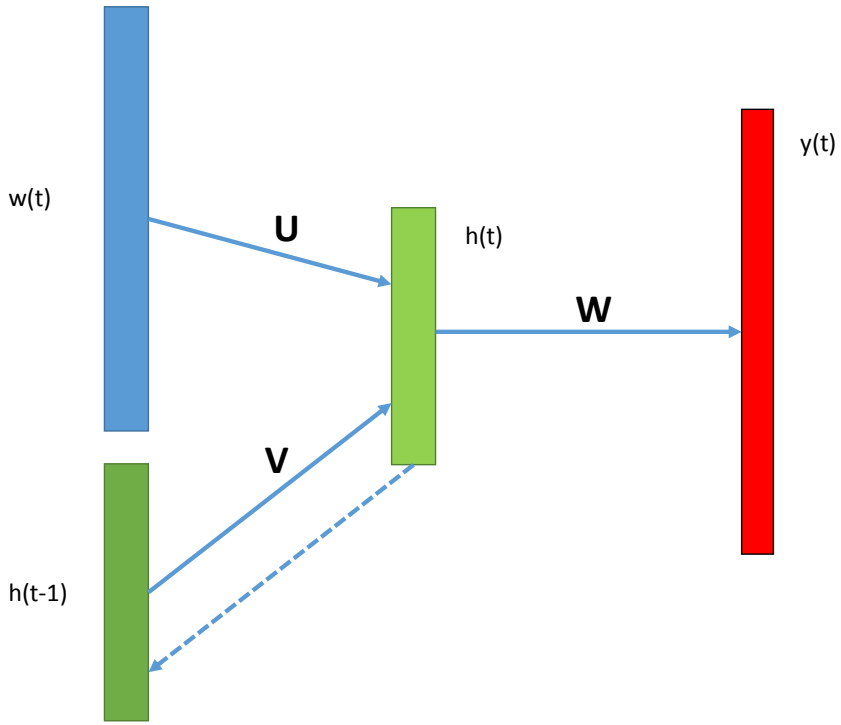


Figure: Recurrent Neural Network LM.

$$y_i(t) = P(w_{t+1} = i | w_t, h_{t-1})$$

RNNLM Structure

$$\mathbf{h}(t) = sigmoid(\mathbf{U}^T\mathbf{w}(\mathbf{t}) + \mathbf{V}^T\mathbf{h}(t-1))$$

$$\mathbf{y}(t) = softmax(\mathbf{W}^T\mathbf{h}(\mathbf{t}))$$

2

[2] biases are absorbed in the matrix.

Training: Back-propagation Through Time

Given an ordered sequence of training instances, the recurrent neural network can be unfolded as that in the figure:

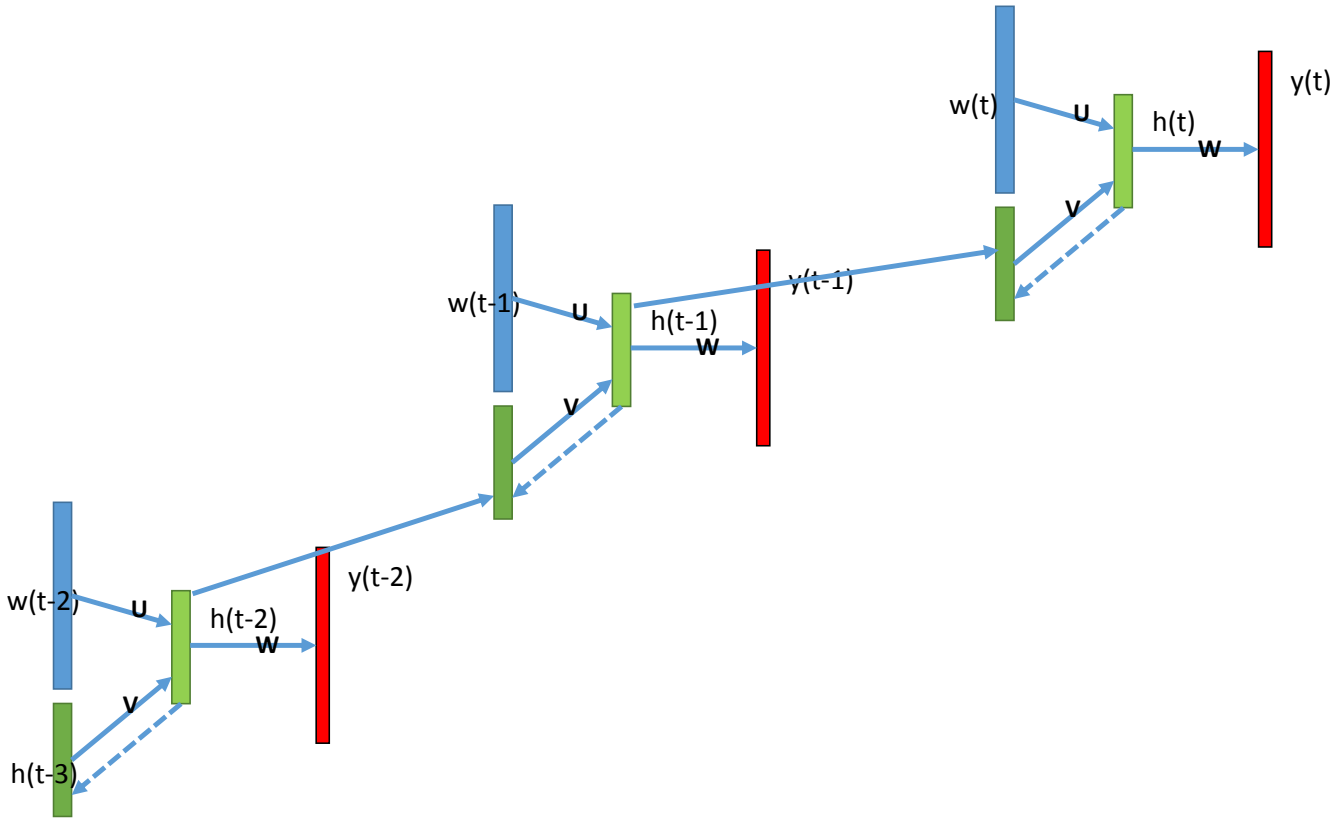


Figure: Unfolding a Recurrent Neural Network.

Training: Back-propagation Through Time

Thus the update strategy is give as

$$\theta = \theta_0 - \frac{\eta}{K} \sum_{i=t_0-K+1}^{t_0} \left. \frac{\partial \mathcal{D}}{\partial \theta} \right|_{t=i}$$

where $\theta = (\mathbf{U}, \mathbf{V}, \mathbf{W})$; η is the learning rate and K is the BPTT step size.

Comparison

The language models are evaluated by per-word perplexity (PP)

$$\exp\left(-\frac{\log \mathcal{L}}{M}\right)$$

where $\mathcal{L}$ is the likelihood of a set of documents measured by a language model and M is the total length of the documents.
Evaluation on Penn Treebank Corpus (T. Mikolov, 2012)

Model	PP
5-gram, KN-smooth (5KN)	141.2
Feedforward neural network, 5 Context (5FFNNLM)	140.2
RNNLM	124.7
5KN + 5FFNNLM	116.7
5KN + RNNLM	105.7

Simple Application

- Used as a feature. *e.g.*,
  - sentence quality estimation, grammar checking, sentence completion.
- Used for re-ranking, *e.g.*,
  - N-best post-processing in machine translation and speech recognition.

Simple Application

[T. Mikolov, 2012]

- *N*-best re-ranking in machine translation in the IWSLT 2005 task

Model	BLEU
Baseline (5-gram)	48.7
+ 300-best RNNLM rerank	51.2

- *N*-best re-ranking in speech recognition in the WSJ task

Model	WER%
Baseline (5-gram)	12.2
+ 100-best RNNLM rerank	10.2

Decoding samples in WSJ speech recolonization task

[T. Mikolov, 2012]

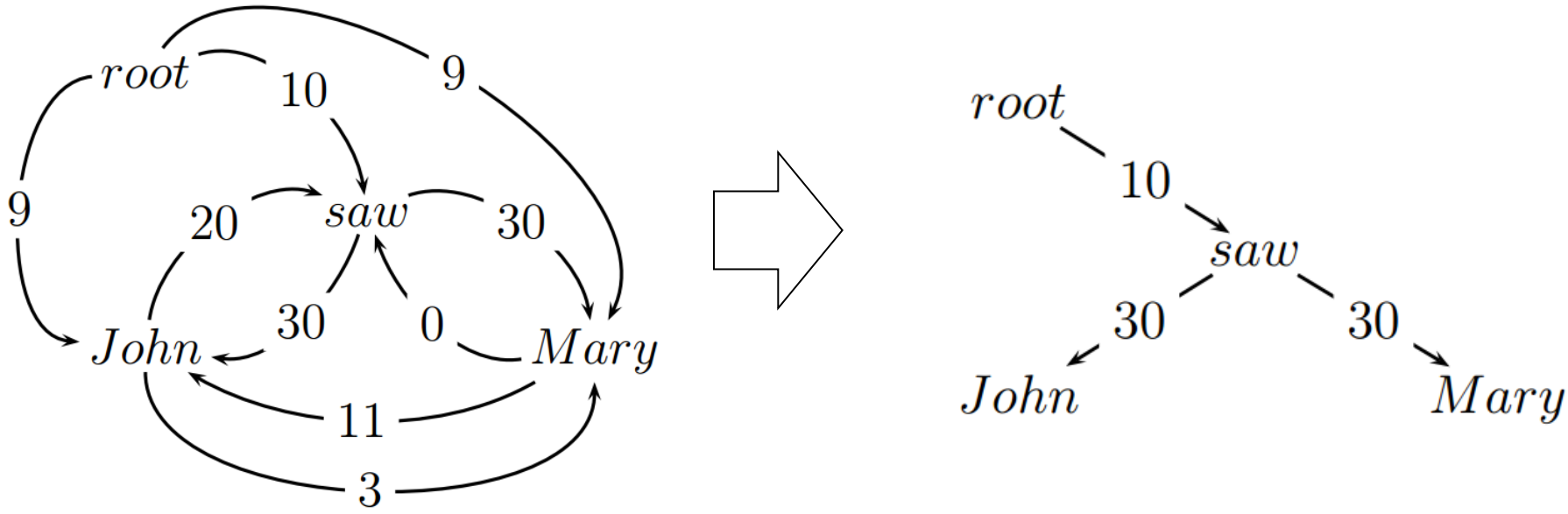
- 5-GRAM+KN: IN TOKYO FOREIGN EXCHANGE TRADING YESTERDAY THE **UNIT** INCREASED AGAINST THE DOLLAR
  +RNNLMrerank: IN TOKYO FOREIGN EXCHANGE TRADING YESTERDAY THE **YEN** INCREASED AGAINST THE DOLLAR
- 5-GRAM+KN: MEANWHILE QUESTIONS REMAIN WITHIN THE E. M. S. **WEATHERED** YESTERDAYS REALIGNMENT WAS ONLY A TEMPORARY SOLUTION
  +RNNLMrerank: MEANWHILE QUESTIONS REMAIN WITHIN THE E. M. S. **WHETHER** YESTERDAYS REALIGNMENT WAS ONLY A TEMPORARY SOLUTION

A Big Issue in Applications of RNNLM

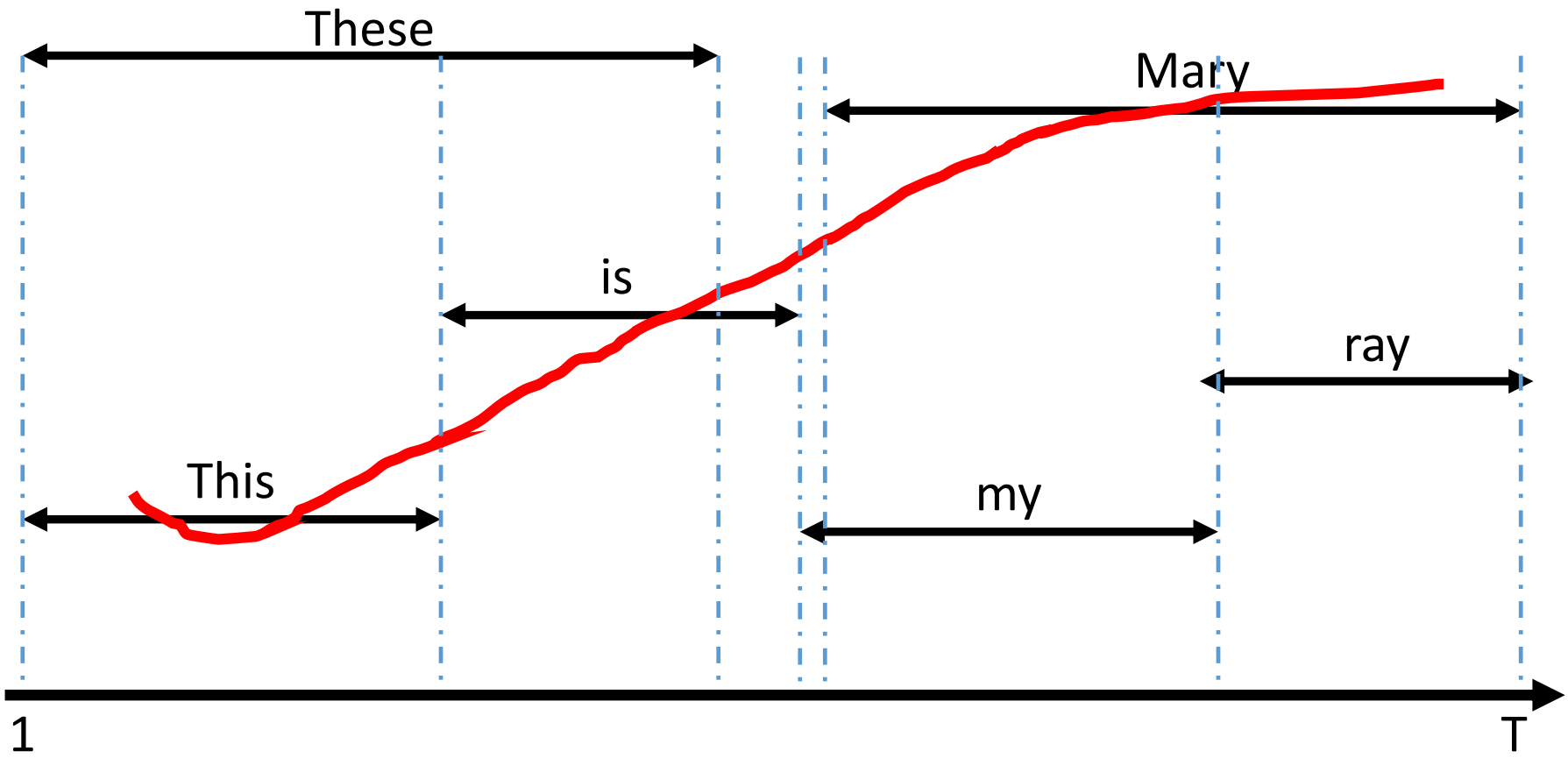
- Decoding is an essential phase in many applications. Variety of dynamic programming algorithms are usually used.

Decoding in Dependency Parsing

Maximum Spanning Tree Algorithm [R. McDonald *et al.*, ACL'06]



Decoding in Speech Recognition



A Skeleton Decoder in Speech Recognition

Suppose the duration of word is omitted in this section: A word is given at each time.
In a typical speech-recognition decoder, the score of recognizing word w at time t (denoted by $s(w,t)$) is given by

$$s(w,t) = A(w,t) + \alpha L(w)$$

where A is the acoustic score given by acoustic model and L is the language-model score. The best score S^* is given by

$$S^* = \max_{w_1,w_2,\ldots,w_M} \sum_t s(w_t,t).$$

In *n*-gram

For uni-gram,

$$S(t) = \max_{w_t} \{S(t-1) + A(t, w_t) + \alpha L(w_t)\}$$

where $S(t)$ represents the best score till time t; w_t is a word candidate. Thus the best hypothesis $w_1 w_2 \ldots w_T$ is given by

$$\arg\max_{w_1 w_2 \ldots w_T} S(T).$$

Time complexity,

$$O(VT)$$

where V is the vocabulary size and T is the length of decoding instance. Complexity of calculating $A(\cdot)$ and $L(\cdot)$ is omitted.

In *n*-gram

For bi-gram,

$$S(t, w_t) = \max_{w_{t-1}} \{S(t-1, w_{t-1}) + A(t, w_t) + \alpha L(w_t|w_{t-1})\}$$

where $S(t, w_t)$ represents the best score till time t with recognizing w_t. Thus the best hypothesis $w_1 w_2 \ldots w_T$ is given by

$$\arg\max_{w_1 w_2 \ldots w_T} S(T, w_T)$$

Time complexity,

$$O(V^2 T).$$

In n-gram

In general, the time complexity of decoding with n-gram is

$$O(V^n T).$$

In RNNLM

For RNNLM, it seems that

$$S(t, \mathcal{H} \cup w_t) = \max_{\mathcal{H}} \{S(t-1, \mathcal{H}) + A(t, w_t) + \alpha L(w_t|\mathcal{H})\}$$

where $\mathcal{H}$ represents the whole decoding sequence. Therefore

$$O(V^T T)?$$

Recall

[T. Mikolov et al., Interspeech'10]

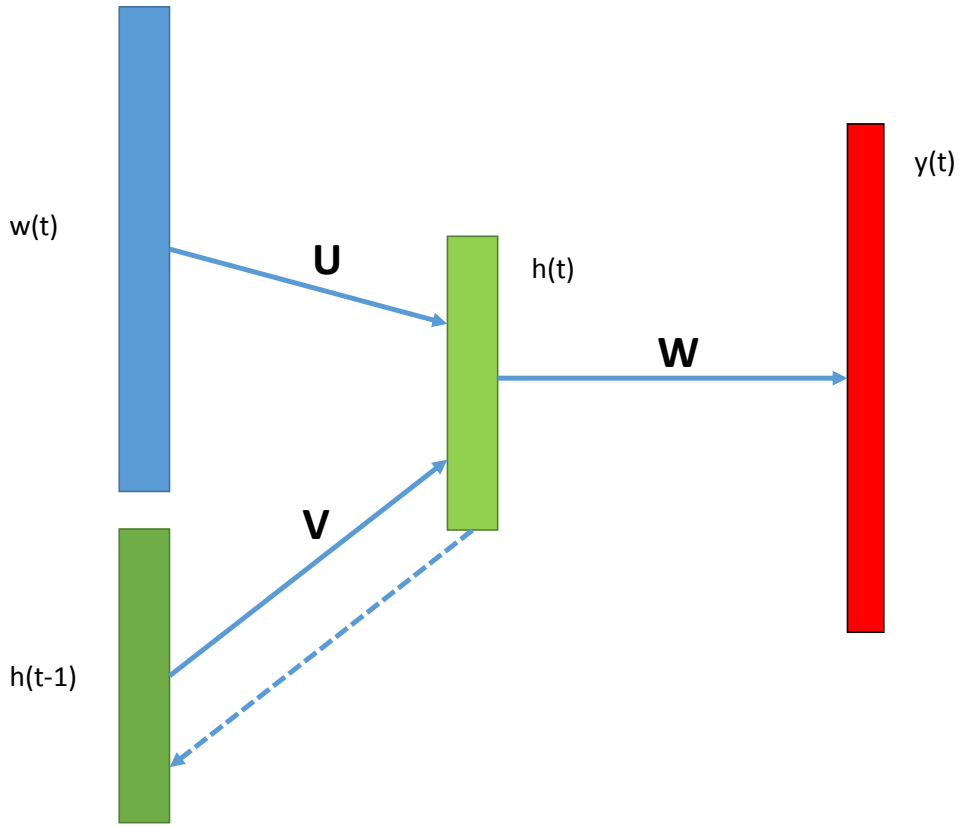


Figure: Recurrent Neural Network LM.

In RNNLM

Potential solution?

$$S(t, \mathbf{w}_t, \mathbf{h}_t) = \max_{\mathbf{w}_{t-1}, \mathbf{h}_{t-1}} \{S(t-1, \mathbf{w}_{t-1}, \mathbf{h}_{t-1}) + A(t, \mathbf{w}_t) + \alpha L(\mathbf{w}_t | \mathbf{w}_{t-1}, \mathbf{h}_{t-1})$$

where

$$\mathbf{h}_t = \mathit{sigmoid}(\mathbf{U}\mathbf{w}_{t-1} + \mathbf{V}\mathbf{h}_{t-1}).$$

Outline

Motivation

- Most of NLP tasks utilize high-dimension atomic symbol features, *e.g.,* a feature template to represent a keyword

$$\texttt{hotel} = (0, 0, 0, \ldots, 0, 0, 0, 1, 0, \ldots, 0, 0, 0)$$

dimension = size of vocabulary

- We really need meaningful distributed representations instead of binary ones, *e.g.,*

$$\texttt{hotel} = (1.02 \quad 2.98 \quad 10.34 \quad 4.27)$$

$$\texttt{motel} = (1.12 \quad 2.71 \quad 10.67 \quad 3.99)$$

Recall: Projection Layer

The output of this layer is

$$\mathbf{g}(\mathbf{w}_{t-n+1}, \ldots, \mathbf{w}_{t-1}) = \begin{pmatrix} \mathbf{C}\mathbf{w}_{t-n+1} \\ \vdots \\ \mathbf{C}\mathbf{w}_{t-1} \end{pmatrix}$$

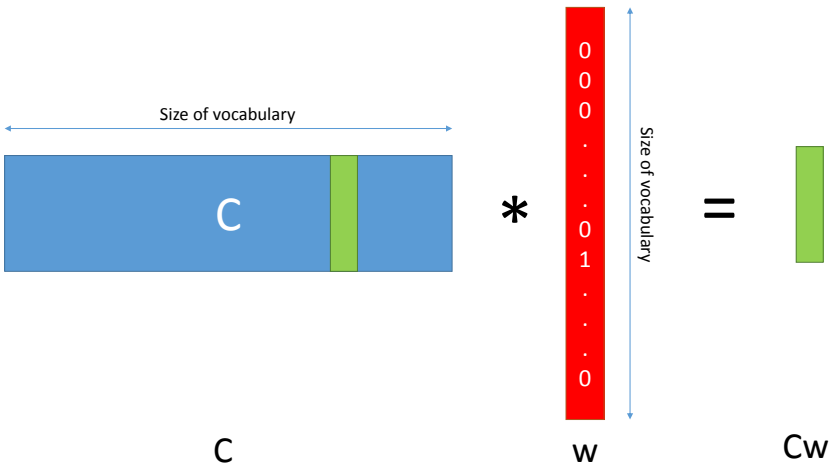


Figure: Projection Layer in Feedforward Neural Network LM.

Why not LSI?

- Bag of words *v.s.* Word sequence
- Comparison of SVD and NNLM word-embedding. I compare the top 3 nearest neighbors of word `good` in both of the models. The distance is measured by euclidean distance. ($dim = 10$)

SVD	Feedfoward NNLM
good	good
unconscious	bad
news	great
resist	strong

Beyond Language model

Inspired by the good properties of word representations given by NNLMs, recent research attempted to trigger meaningful word representation in different tasks.

Multi-task Learning with Neural Network

Multitask Learning [R. Collobert and J. Weston, ICML'08]

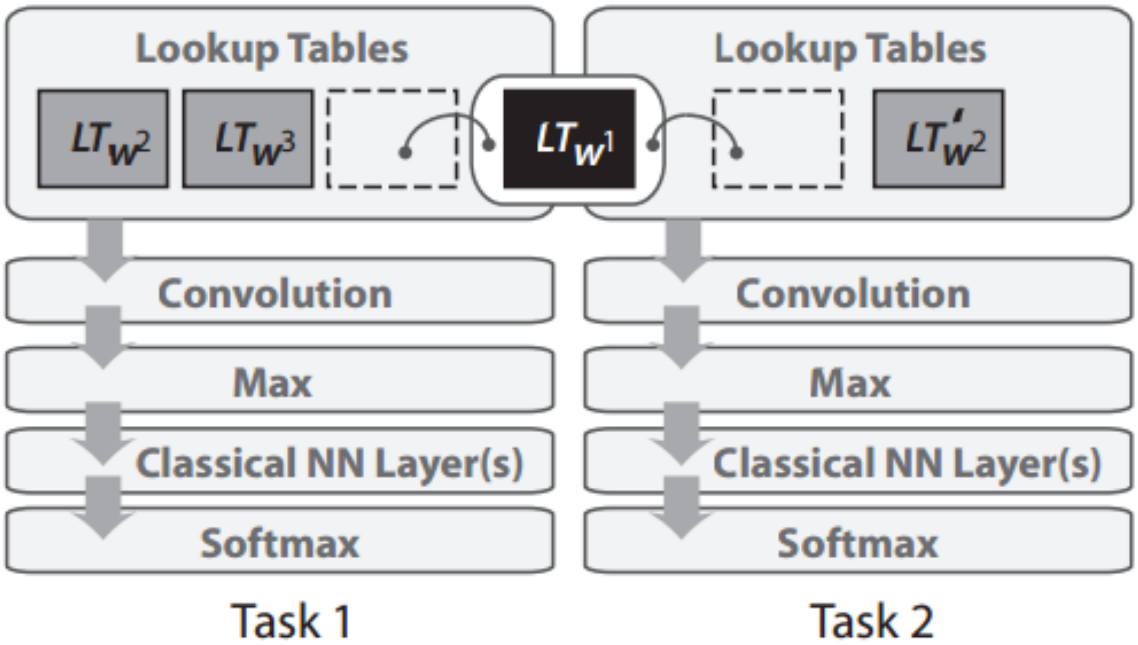


Figure: Multitask Neural Nets.

Word Alignment Modeling

Word Alignment Modeling [N. Yang et al., ACL'13]

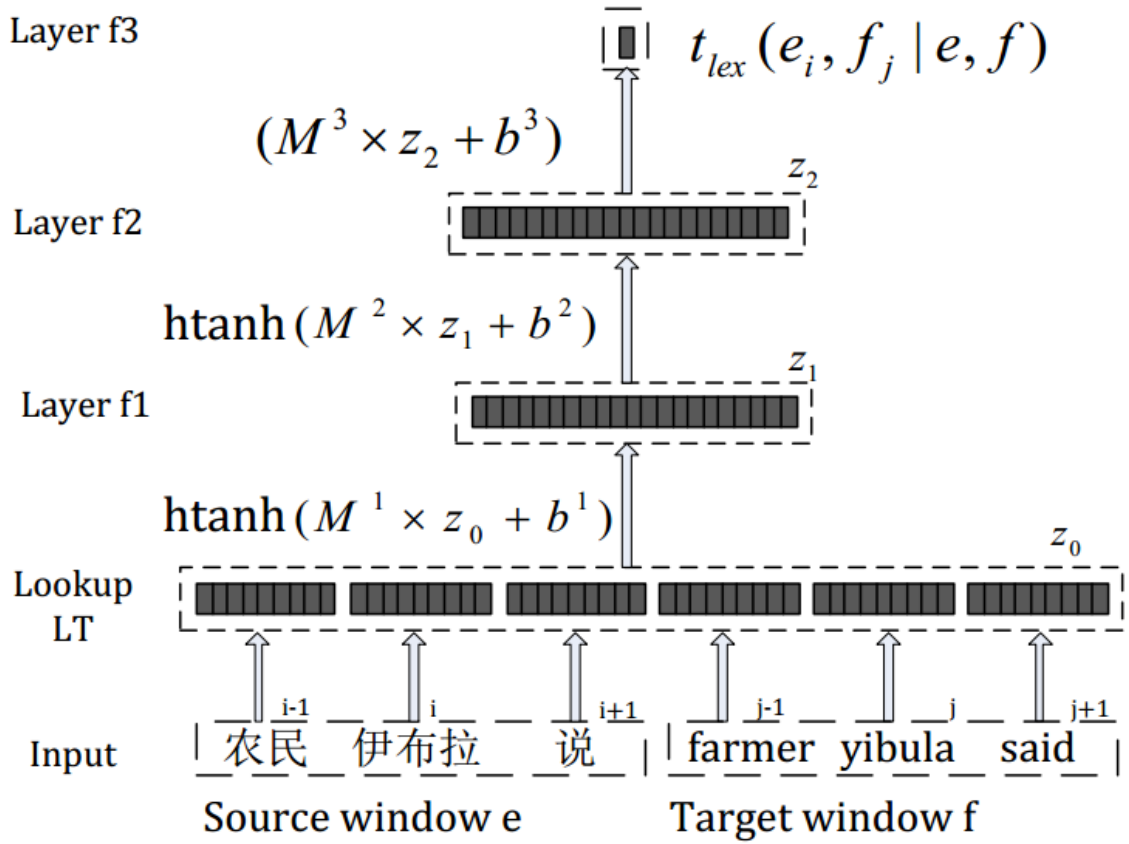


Figure: Word Alignment Modeling with Context Dependent Neural Network.

Outline

Summary

- Two typical types of neural network language models are reviewed.
- Simple applications is so easy and results are good.
- In some tasks like decoding, it potentially requires new algorithms.
- Distributed word representation is helpful.

Q & A