EXTENSIONS OF RECURRENT NEURAL NETWORK LANGUAGE MODEL

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget,
Jan "Honza" Černocký, Sanjeev Khudanpur

Speech@FIT, Brno University of Technology,
Johns Hopkins University

25. 5. 2011

Overview

- Introduction
- Model description
- Extensions
- Empirical evaluation
- Current work

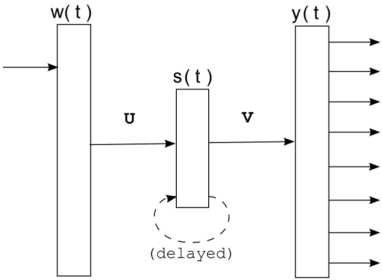
Introduction

- Neural network based LMs outperform standard backoff n-gram models
  - Words are projected into low dimensional space, similar words are automatically clustered together
  - Smoothing is solved implicitly
  - Standard backpropagation algorithm (BP) is used for training
  - In [Mikolov2010], we have shown that recurrent neural network (RNN) architecture is competitive with the standard feedforward architecture

Introduction

- In this presentation, we will show:
  - Importance of ”backpropagation through time” (BPTT) [Rumelhart et al. 1986] training algorithm for RNN language models
  - Simple speed-up technique that reduces computational complexity $10\times$ - $100\times$
  - Results after combining randomly initialized RNN models
  - Comparison of different advanced LM techniques on the same data set
  - Results on large data sets and LVCSR experiments

Model description - recurrent NN

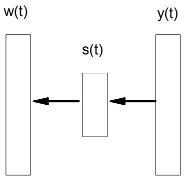


- Input layer $\mathbf{w}$ and output layer $\mathbf{y}$ have the same dimensionality as the vocabulary
- Hidden layer $\mathbf{s}$ is orders of magnitude smaller
- $\mathbf{U}$ is the matrix of weights between input and hidden layer, $\mathbf{V}$ is the matrix of weights between hidden and output layer

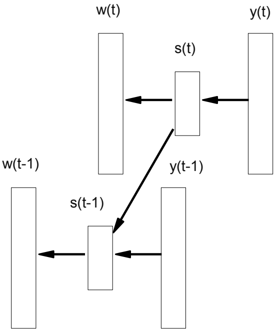
Backpropagation through time

- Training of RNNs by normal backpropagation is not optimal
- Backpropagation through time (BPTT) is efficient algorithm for training recurrent neural networks
- BPTT works by unfolding the recurrent part of the network in time to obtain usual feedforward representation of the network; such deep network is then trained by backpropagation
- For on-line learning, ”truncated BPTT” is used

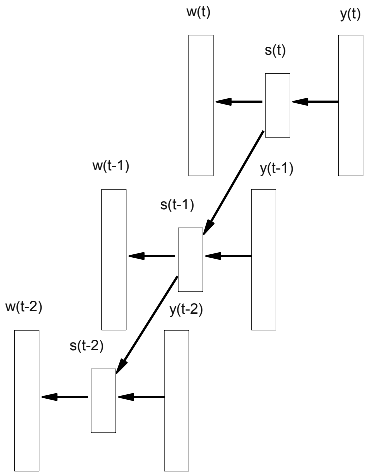
RNN unfolded in time



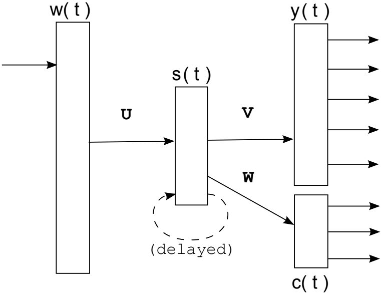
RNN unfolded in time



RNN unfolded in time



Factorization of the output layer



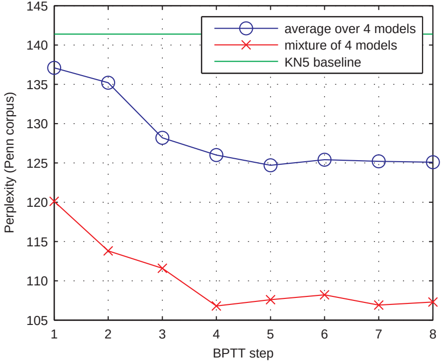
$$P(w_i|history) = P(c_i|\mathbf{s}(t))P(w_i|c_i, \mathbf{s}(t)) \quad (1)$$

- Words are assigned to ”classes” based on their unigram frequency
- First, class layer is evaluated; then, only words belonging to the predicted class are evaluated, instead of the whole output layer $\mathbf{y}$ [Goodman2001]
- Provides speedup in some cases more than $100\times$

Empirical evaluation - Setup description

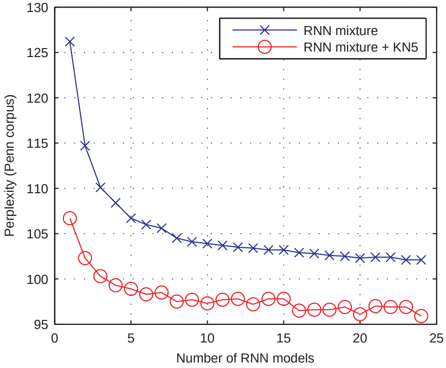
- We have used the Penn Treebank Corpus, with the same vocabulary and data division as other researchers:
  - Sections 0-20: training data, 930K tokens
  - Sections 21-22: validation data, 74K tokens
  - Sections 23-24: test data, 82K tokens
  - Vocabulary size: 10K

Importance of BPTT training



- Importance of BPTT training on Penn Corpus. BPTT=1 corresponds to standard backpropagation.

Combination of randomly initialized RNNs



- By linearly interpolating outputs from randomly initialized RNNs, we obtain better results

Comparison of different language modeling techniques

Model	Perplexity
Kneser-Ney 5-gram	141
Random forest [Xu 2005]	132
Structured LM [Filimonov 2009]	125
Feedforward NN LM	116
Syntactic NN LM [Emami 2004]	110
RNN trained by BP	113
RNN trained by BPTT	106
4x RNN trained by BPTT	98

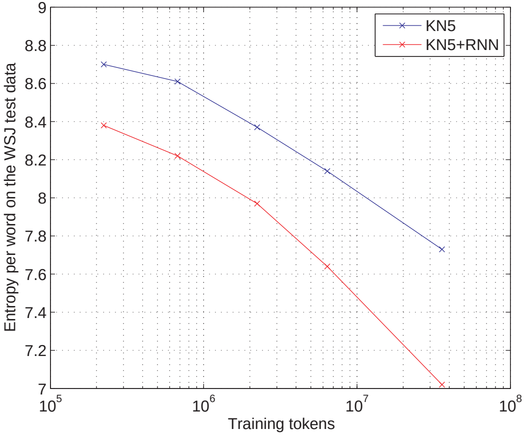
- Comparison of different language modeling techniques on Penn Corpus. Models are interpolated with the baseline 5-gram backoff model.

Speedup with different amount of classes

Classes	RNN	RNN+KN5	Min/epoch	Sec/test
30	134	112	12.8	8.8
100	136	114	9.1	5.6
1000	131	111	16.1	15.7
4000	127	108	44.4	57.8
Full	123	106	154	212

- Values around `sqrt(vocabulary size)` lead to the largest speed-ups

Improvements with increasing amount of data



- The improvement obtained from a single RNN model over the best backoff model **increases** with more data!

Current work

- Dynamic evaluation for model adaptation
- Combination and comparison of RNNs with many other advanced LM techniques
- More than 50% improvement in perplexity on large data set against modified Kneser-Ney smoothed 5-gram

Current work - ASR

- Almost 20% reduction of WER (Wall Street Journal) with simple ASR system, against backoff 5-gram model (WER $17.2\% \rightarrow 14.4\%$)
- Almost 10% reduction of WER (Broadcast News) with state of the art IBM system, against backoff 4-gram model (WER $13.1\% \rightarrow 12.0\%$)

Toolkit

- Our experiments can be repeated using toolkit available at `http://www.fit.vutbr.cz/~imikolov/rnnlm/`

- Thanks for attention!