Investigation on Log-linear Interpolation of Multi-domain Neural Network Language Model

Zoltán Tüske, Kazuki Irie, Ralf Schlüter, Hermann Ney

24. March 2016

Human Language Technology and Pattern Recognition Group
Prof. Dr.-Ing. H. Ney
Computer Science Department
RWTH Aachen University, Germany

1 Outline

- Introduction
- Multi-domain neural network LM
  - Log-linear interpolation
  - Implementation
- Experiments
- Conclusions

2 Introduction

- State-of-the-art language models (LM) are based on neural networks
  - Better results if (linearly) interpolated with huge count-based LM

- Usually count LMs are trained on different domains, then linear-interpolated
  - Interpolation weights are optimized on target domain validation set
  - Linear interpolation:

$$p(w|h) = \sum_j \lambda_j \cdot p_j(w|h) \qquad \text{with} \qquad \sum_j \lambda_j = 1$$

    - Where: w current word
      h history
      λ_j weight of jth model
  - Optimized using expectation maximization (EM) algorithm
  - Count models are suited to be linearly combined into one single model (with union of n-grams and recomputing back-off weights)

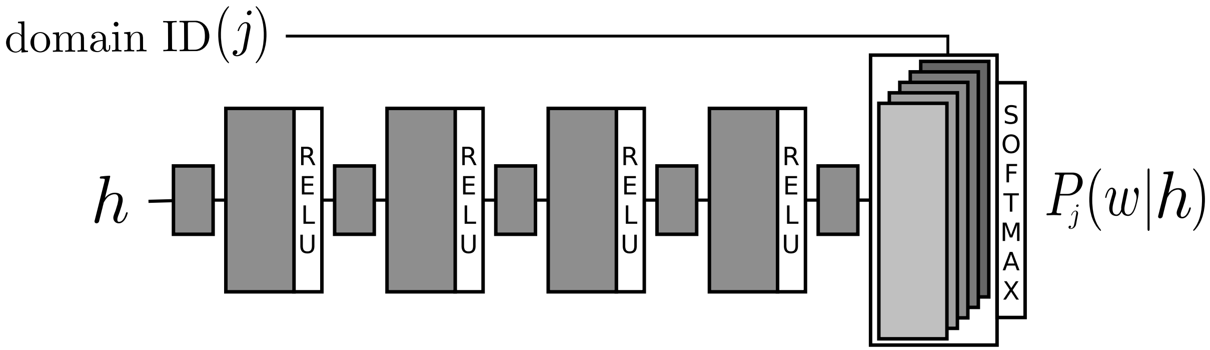
3 Motivation and Goals

- Training multi-domain NNLM
  - Inspired by the great success of multi-task training [Caruana 93]

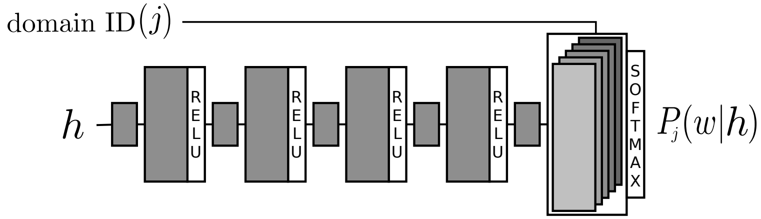
- Similar approach for NNLM as for count models
  - Obtaining single model after interpolation of NNLMs
  - No straightforward method to formulate linear interpolation of NNLMs as a single model
    - Log-linear combination fits better

- Initial investigation using feed-forward NNLM

Joint Model in This Study



- **Multiple posterior estimates**
  - **Active output: selected by the domain of the input vector**
  - **Hidden layers are shared between the domains**
  - **Shared vocabulary, common softmax**
- **Similar as multilingual training in acoustic modeling:**
  - **[Scanzio & Laface+ 08], [Veselý & Karafiát+ 12], [Tüske & Pinto+ 13], [Heigold & Vanhoucke+ 13], [Huang & Li+ 13]**
  - **Outputs are usually not comparable, different tied-triphone targets per language**



- **Special, domain dependent output layer is introduced**
  - **Separate weight matrices and biases allocated for our 11 different domains (j):**

$$A_j = \begin{bmatrix} \vdots \\ a_{wj}^T \\ \vdots \end{bmatrix} \qquad \text{and} \qquad b_j = \begin{bmatrix} \vdots \\ b_{wj} \\ \vdots \end{bmatrix}$$

- **Three types of BN layers:**
  - *Input BN*: **projection layer shared along the LM history (time-delay NN)**
  - *Between-hidden-layer BN*: **low-rank factorization of the hidden layer outputs**
  - *Output BN*: **no word-classes, direct estimation of 150k word posteriors**

- **Last layer of a neural network is a log-linear model with zeroth- and first-order features:**

$$p_j(w|h) = \frac{exp(a_{wj}^T \cdot y + b_{wj})}{\sum_{w'} exp(a_{w'j}^T \cdot y + b_{w'j})}$$

  - $y = y(h)$: **last BN output, a non-linear feature function of h shared between domains**

4 Log-Linear Interpolation of NNLMs

- **Log-linear interpolation [Klakow 98]**

$$p(w|h) = \frac{1}{Z_\lambda} \prod_j p_j(w|h)^{\lambda_j} \qquad \textbf{with} \qquad Z_\lambda = \sum_w \prod_j p_j(w|h)^{\lambda_j}$$

  - **Log-linear interpolation is a convex optimization problem**

- **With the proposed multi-domain NNLM:**

$$\prod_j p_j(w|h)^{\lambda_j} = \frac{\prod_j exp(\lambda_j (a_{wj}^T \cdot y + b_{wj}))}{\prod_j \sum_{w'} exp(\lambda_j (a_{w'j}^T \cdot y + b_{w'j}))}$$

- **Results in:**

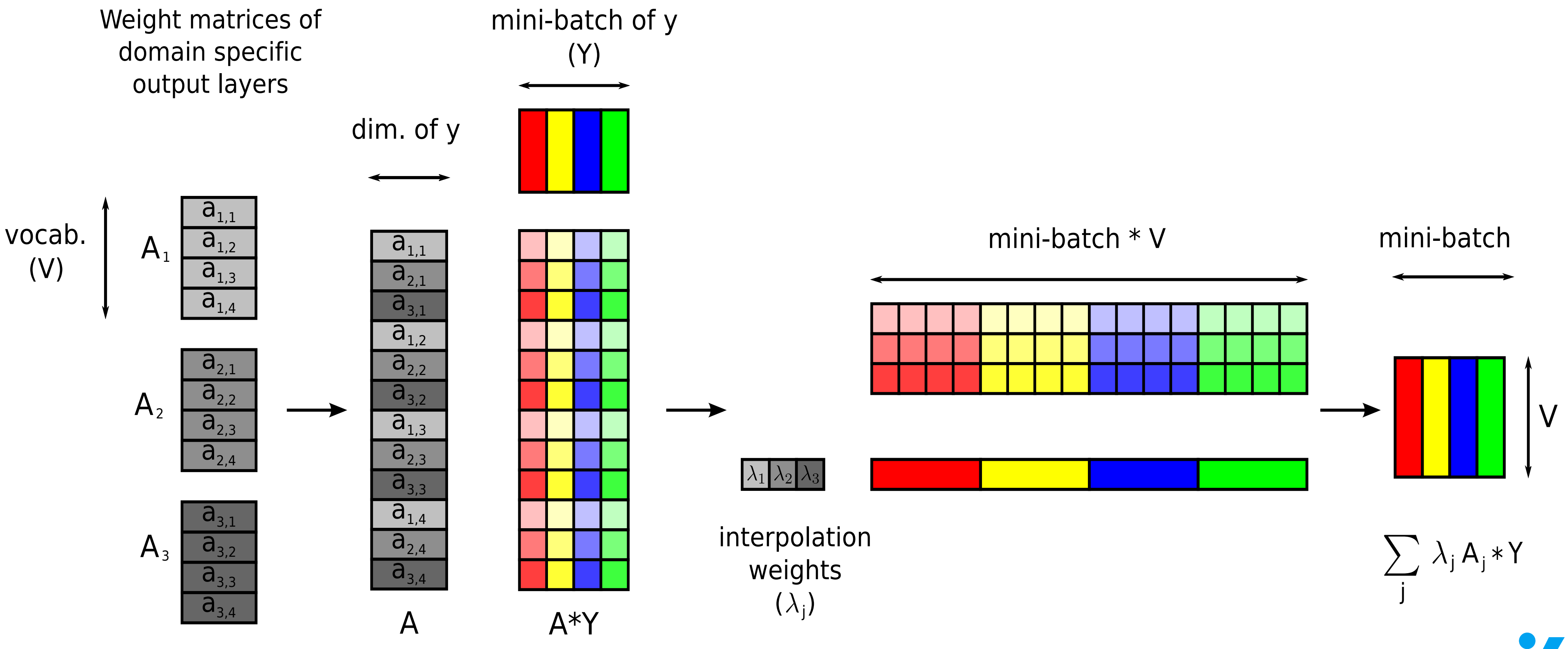
$$p(w|h) = \frac{exp(\tilde{a}_w^T \cdot y + \tilde{b}_w)}{\sum_{w'} exp(\tilde{a}_{w'}^T \cdot y + \tilde{b}_{w'})}$$

where $\tilde{a}_w = \sum_j \lambda_j \cdot a_{wj}$ **and** $\tilde{b}_w = \sum_j \lambda_j \cdot b_{wj}$

- **Single neural network:** *weighted sum* **of the domain dependent linear layers**

5 Implementation

- The interpolation can easily be integrated into NN framework as a linear layer
- The weight matrices (A_j) and biases should be row-wise interleaved
- Mini-batches (column-major format) should be re-interpreted:
  - The interpolation layer performs V times non-overlapping convolution



6 Experimental Setups

- Tests on QUAERO English broadcast news and conversations corpus
- 150K vocabulary
- Dev: 40K, Test: 36K words
- Our data sets for language model training:
  - 3.1B:
    - 11 sub-corpora, used as 11 output targets in multi-domain training
    - Collected from Giga-words, IWSLT, WMT, Quaero, TED
    - Perplexity after linear interpolation of Kneser-Ney smoothed count models: 132.7
  - 50M$\subset$3.1B:
    - Transcription of the acoustic data
    - Blog data, part of the best matching Quaero corpus,
  - 2M$\subset$50M:
    - Only the transcription of the acoustic data
- Acoustic model in the ASR experiments:
  - 12-layer rectified linear unit MLP, speaker independent, after MPE
  - Multilingually initialized MLP, adapted with 250h of English data

7 Experimental Results - (Re-)Optimizing Feed-Forward NNLM

- **Experiments on 50M corpus**
  - **Training time ∼3.5 days on a single GPU, w/o word-classes**
  - **Feeding the best matching 2M subcorpus into NN at the end of the epoch**
- **PPL measured without interpolation with count LM on development set**

- **Optimizing the context**
  - **3 non-BN hidden layers with 1024 nodes**
  - **Projection / between-hidden / before-output BN: 64 / 256 /128 nodes**

N-gram	5	10	20	30
PPL	142.9	126.0	117.4	118.3

Experimental results - (Re-)Optimizing Feed-Forward NNLM

- Effect of discriminative pre-training (DPT) [Seide & Li^{+} 11]
- Optimizing BN, non-BN layer and mini-batch sizes
- 20-gram feed-forward MLP

<table>
<tr><th colspan="2">non-BN</th><th colspan="3">BN size</th><th rowspan="2">DPT</th><th rowspan="2">batch size</th><th rowspan="2">PPL</th></tr>
<tr><th>#</th><th>size</th><th>proj.</th><th>btw.hidden</th><th>output</th></tr>
<tr><td>3</td><td rowspan="5">1024</td><td rowspan="2">64</td><td rowspan="9">256</td><td rowspan="2">128</td><td rowspan="4">-</td><td rowspan="3">64</td><td>117.4</td></tr>
<tr><td>5</td><td>116.2</td></tr>
<tr><td rowspan="4">3</td><td rowspan="7">128</td><td rowspan="7">256</td><td>114.7</td></tr>
<tr><td>128</td><td>117.0</td></tr>
<tr><td rowspan="5">+</td><td rowspan="5">64</td><td>113.7</td></tr>
<tr><td>2048</td><td>112.1</td></tr>
<tr><td rowspan="2">4</td><td>1024</td><td>111.5</td></tr>
<tr><td rowspan="2">2048</td><td>110.5</td></tr>
<tr><td>5</td><td>110.7</td></tr>
</table>

- Our previous best FFNN PPL: 130.9
- Our current best on this 50M corpus: LSTM-RNN, 100.5 [Sundermeyer & Ney^{+} 15]

Experimental results - Effect of More Data and Fine-Tuning

- Training LM on 3.1B words, single GPU $\sim$20 days, w/o word-classes
  - Learning rate adjusted by CV after every $\sim$100M words
- Optional fine-tuning on matched subcorpora: 2M $\subset$ 50M

LM	fine-tuning 50M	fine-tuning 2M	PPL
50M			110.5
		×	109.0
3B			129.0
		×	96.6
	×		101.4
	×	×	96.2

- More (mismatched) data did not help immediately
- But led to a much better MLP initialization before fine-tuning with matched data
- Using multi-domain data led to over 10% rel. imp. compared to the best 50M result
- 50M LSTM-RNN: 100.5

Experimental results - Multi-Domain Training

- Multi-domain training, ∼20 days on single GPU, w/o word-classes

LM	multi domain	log-lin. interp.	fine-tuning 50M	fine-tuning 2M	PPL
50M				×	109.0
3B			×	×	96.2
	×				133.1
	×		×	×	95.7*
	×	×			117.6
	×	×	×	×	94.3

*using the best matching output

- Log-lin. interp.: estimation of 11 parameters led to 10% rel. PPL improvement (133→118)
- Linear interpolation performed better: 114 PPL, but model cannot be merged (and easily fine-tuned)
- Fine-tuning the log-lin. interpolated NNLM led to better results, than taking the best fitting output
- Best: re-training multi-domain output on the BN of the best model followed by interpolation
  - 92.0 PPL

8 Experimental results - ASR Experiments

- Lattice extraction with count model
- Lattice rescoring using `rwthlm` [Sundermeyer & Schlüter+ 14]
  - Traceback lattice approximation
  - Linear-interpolation between NNLM and count LM
- Measuring word error rate
  - After Viterbi (Vi.) or confusion network (CN) decoding of the lattices

Language Model	Dev			Eval		
	PPL	Vi.	CN	PPL	Vi.	CN
KN4	132.7	12.6	12.3	133.4	15.4	15.0
+ 50M FFNN	96.5	11.4	11.1	95.0	14.2	13.8
+ 3B, fine-tune	89.6	10.9	10.7	88.0	13.7	13.4
+ Multi-domain,log-lin,fine-tune	88.5	10.8	9.1	87.0	13.7	13.5
+ 50M LSTM	91.6	10.9	9.0	91.0	13.7	13.5

- Our improved 50M FFNN only slightly behind the LSTM
- Better initialization of FFNN (with the help of mismatched data): significant improvement
- FFNN is 3-4 point PPL better than LSTM (due to more data) but no WER improvement

9 Conclusions

- Re-optimized feed-forward LM: not so far from LSTM
- Multi-domain LM training implementation:
  - Fits naturally to log-linear interpolation
  - Interpolated models can be merged (like count models after lin.interp.)
- With the help of multi-domain data, better optimum can be reached with feed-forward NNLM

- TODOs:
  - Repeating the experiments with LSTM: would mismatched data also lead to better initialization?
  - Log-lin. interpolation: only a few parameters should be estimated
    - Investigation on unsupervised LM adaptation

References

[Caruana 93] R. Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proc. of International Conference on Machine Learning (ICML)*, pp. 41–48, Amherst, MA, USA, June 1993. 4

[Heigold & Vanhoucke+ 13] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, J. Dean. Multilingual acoustic models using distributed deep neural networks. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 8619–8623, 2013. 5

[Huang & Li+ 13] J.-T. Huang, J. Li, D. Yu, L. Deng, Y. Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 7304–7308, 2013. 5

[Klakow 98] D. Klakow. Log-linear interpolation of language models. In *Proc. of International Conference on Spoken Language Processing (ICSLP)*, pp. 1695–1698, Sydney, Australia, Dec. 1998. 7

[Scanzio & Laface+ 08] S. Scanzio, P. Laface, L. Fissore, R. Gemello, F. Mana. On the Use of a Multilingual Neural Network Front-End. In *Proc. Interspeech*, pp. 2711–2714, Brisbane, Australia, Sept. 2008. 5

[Seide & Li+ 11] F. Seide, G. Li, X. Chen, D. Yu. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 24–29, Waikoloa, HI, USA, Dec. 2011. 11

[Sundermeyer & Ney+ 15] M. Sundermeyer, H. Ney, R. Schlüter. From feedforward to recurrent lstm neural networks for language modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 23, No. 3, pp. 517–529, March 2015. 11

[Sundermeyer & Schlüter+ 14] M. Sundermeyer, R. Schlüter, H. Ney. `rwthlm` – The RWTH Aachen University neural network language modeling toolkit. In *Proc. Interspeech*, pp. 2093–2097, Singapore, Sept. 2014. 14

[Tüske & Pinto+ 13] Z. Tüske, J. Pinto, D. Willett, R. Schlüter. Investigation on cross- and multilingual MLP features under matched and mismatched acoustical conditions. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 7349–7353, 2013. 5

[Veselý & Karafiát+ 12] K. Veselý, M. Karafiát, F. Grézl, M. Janda, E. Egorova. The language-independent bottleneck features. In *Proc. of IEEE Workshop on Spoken Language Technology*, pp. 336–341, 2012. 5