On Using Monolingual Corpora in Neural Machine Translation
&
Towards better decoding and integration of language models in sequence to sequence models

Rebekka Hubert

Department of Computational Linguistics (ICL)
University of Heidelberg

1. Monolingual Corpora in NMT
2. On language models in end-to-end systems
3. Conclusions

Motivation

- NMT systems suffer from low resources and domain restriction
- monolingual corpora exhibit linguistic structure
- integrate language model trained on monolingual corpora into a NMT model suffering from low resources or domain restriction

Model - based on [Bahdanau et al., 2014]

- encoder-decoder framework

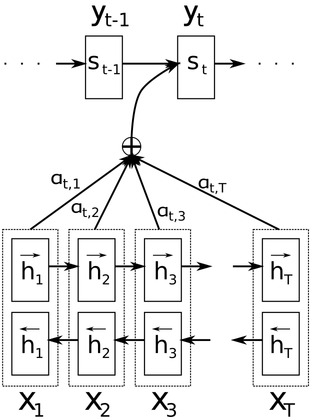

Figure 1: overview of the model [Bahdanau et al., 2014]

Encoder

- bidirectional RNN
- sequence of annotation vectors is concatenation of pairs of hidden states

$$h_j^T = [\overleftarrow{h}_j^T; \overrightarrow{h}_j^T]$$

- h_j encodes information about j-word w.r.t. all of the surrounding words in the sentence

Decoder

- emulates searching through a source sentence during decoding
- at each timestep t
  - compute $s_t = f_r(s_{t-1}, y_{t-1}, c_t)$
  - computes context vector c_t (*expected annotation*)

$$c_t = \sum_{j=1}^{T_x} \alpha_{tj} h_j$$

with

$$\alpha_{tj} = \frac{exp(e_{tj})}{\sum_{k=1}^{T_x} exp(e_{tk})}; e_{tj} = a(s_{t-1}, h_j, y_{t-1})$$

  - a: soft-alignment model (feedforward neural network)
  - α: attention

Decoder

- compute probability of target word y_t with deep output layer

$$p(y_t \mid y_{<t}) \propto exp(y_t^T(W_o f_o(s_t^{TM}, y_{t-1}, c_t) + b_o))$$

  - y_t: k-dimensional, one-hot encoded vector indicating one of the words in target vocabulary
    - for large target vocabularies: importance sampling technique [Jean et al., 2014]
  - $W_o \in \mathbb{R}^{K_y \times l}$: weight matrix
  - f_o: neural network with two-way maxout non-linearity
  - b_o: bias

Sampling technique

- partition training corpus and define small subset $V\prime$ of unique target words for each partition
- at each update only the vectors associated with the sampled words in $V\prime$ and the correct word are updated
- equivalent to

$$p(y_t \mid y_{<t}, x) = \frac{exp\{y_t^T \phi(y_{t-1}, s_t, c_t) + b_t\}}{\sum_{y_k \in V\prime} exp\{y_k^T \phi(y_{t-1}, s_t, c_t) + b_k\}}$$

- *approximates* exact output probability

Maxout Networks

- characterized by the Maxout activation function
- *Generalization* of ReLu: $h_i(x) = max_{j \in [1,k]} z_{ij}$
- $z_{ij} = x^T W ... ij + b_{ij}$; $W \in \mathbb{R}^{d \times m \times k}$
- piecewise linear approximation to an arbitrary convex function
- locally linear almost anywhere
- no sparse representation can be produced
- many learned parameters - best used in combination with dropout

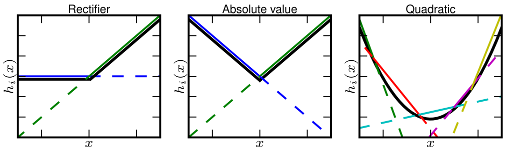

Figure 2: maxout activation function implementing/approximating different functions [Goodfellow et al., 2013]

Training

- Train NMT model with bilingual corpus of pairs $(x^{(n)}, y^{(n)})$

$$max_{\theta} \frac{1}{N} = \sum_{n=1}^{N} log \ p_{\theta}(y^{(n)} \mid x^{(n)})$$

- θ: set of trainable parameters
- Model setup
  - RNNLM as language model: set $c_t = 0$
  - NMT as described above
  - pre-train RNNLM and NMT separately

Integrating the language model - Fusions

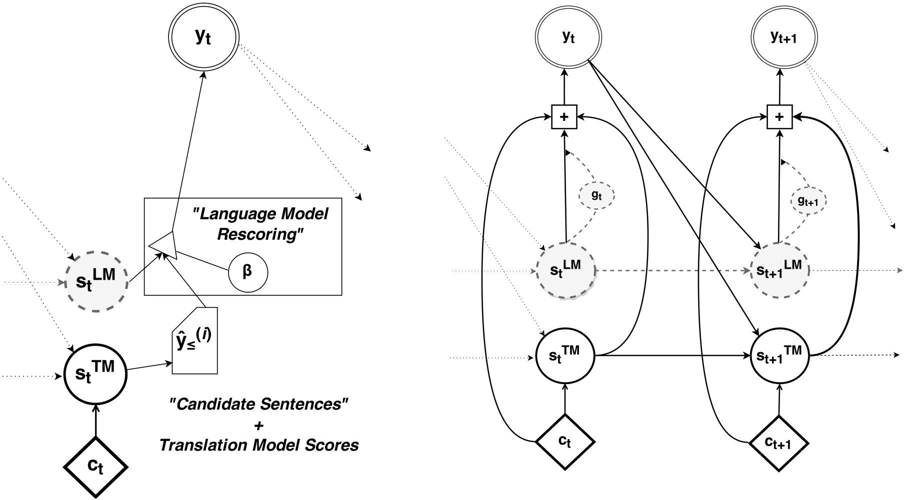

Figure 3: Shallow and Deep Fusion [Gülçehre et al., 2015]

Shallow Fusion

- at each timestep t
  - NMT: compute score of possible next words for all hypotheses $\{y^{(i)}_{<t-1}\}$
  - sort new hypotheses (old hypotheses with new word added)
  - select top K hypotheses as candidates $\{\hat{y}^{(i)}_{\leq t}\}_{i=1,\ldots,K}$
  - rescore hypotheses with weighted sum of words (add score of new word)

$$log\ \ p(y_t = k) = log\ \ p_{TM}(y_t = k) + \beta log\ \ p_{LM}(y_t = k)$$

  - β: hyper-parameter

Deep Fusion

- integrate RNNLM and decoder of NMT by concatenating their hidden states
- fine-tune model to use both hidden states by tuning only output parameters
- new input of deep output layer

$$p(y_t \mid y_{<t}, x) \propto exp(y_t(W_o f_o(s_t^{TM}, s_t^{LM}, y_{t-1}, c_t) + b_o))$$

- keeps learned monolingual structure of LM intact

Deep Fusion - Balancing LM and TM

- use controller mechanism to dynamically weight LM and TM models

$$g_t = \sigma(v_g^T s_t^{LM} + b_g)$$

- multiply controller output with hidden state of LM to control magnitude of LM signal
- decoder only regards LM when deemed necessary

Corpora

- parallel datasets

	Chinese	English
# of Sentences	436K	
# of Unique Words	21K	150K
# of Total Words	8.4M	5.9M
Avg. Length	19.3	13.5

(a) Zh-En

	Turkish	English
# of Sentences	160K	
# of Unique Words	96K⋆	95K
# of Total Words	11.4M⋆	8.1M
Avg. Length	31.6	22.6

(b) Tr-En

	Czech	English
# of Sentences	12.1M	
# of Unique Words	1.5M	911K
# of Total Words	151M	172M
Avg. Length	12.5	14.2

(c) Cs-En

	German	English
# of Sentences	4.1M	
# of Unique Words	1.16M†	742K
# of Total Words	11.4M†	8.1M
Avg. Length	24.2	25.1

(d) De-En

Table 1: datasets used [Gülçehre et al., 2015]

  - Zh-En: training on SMS/CHAT, test on conversational speech
  - all others: close domains
- monolingual dataset: English gigaword corpus

Results - Translation Tasks

	Development Set		Test Set			
	dev2010	**tst2010**	**tst2011**	**tst2012**	**tst2013**	**Test 2014**
Previous Best (Single)	15.33	17.14	18.77	18.62	18.88	-
Previous Best (Combination)	-	17.34	18.83	18.93	18.70	-
NMT	14.50	18.01	18.40	18.77	19.86	18.64
NMT+LM (Shallow)	14.44	17.99	18.48	18.80	19.87	18.66
NMT+LM (Deep)	**15.69**	**19.34**	**20.17**	**20.23**	**21.34**	**20.56**

(a) Tr-En for each set

	SMS/CHAT		CTS	
	Dev	Test	Dev	Test
PB	15.5	14.73	21.94	21.68
+ CSLM	16.02	15.25	23.05	22.79
HPB	15.33	14.71	21.45	21.43
+ CSLM	15.93	15.8	22.61	22.17
NMT	17.32	17.36	23.4	**23.59**
Shallow	16.59	16.42	22.7	22.83
Deep	**17.58**	**17.64**	**23.78**	23.5

(b) Zh-En, PB: phrase-based, HPB: hierarchical phrase based

	De-En		Cs-En	
	Dev	Test	Dev	Test
NMT Baseline	25.51	23.61	21.47	21.89
Shallow Fusion	25.53	23.69	21.95	22.18
Deep Fusion	**25.88**	**24.00**	**22.49**	**22.36**

(c) De-En and Cs-En translation tasks

Table 2: translation results on parallel corpora [Gülçehre et al., 2015]

Results - RNNLM perplexity on development set

	Zh-En	Tr-En	De-En	Cs-En
Perplexity	223.68	163.73	78.20	78.20
Average g	0.23	0.12	0.28	0.31
Std Dev g	0.0009	0.02	0.003	0.008

Table 3: Perplexity of RNNLM on dev. set and g [Gülçehre et al., 2015]

Conclusions

- Fusion
  - generally improves performance, regardless of the size of the parallel corpora
  - Deep Fusion achieves bigger improvements than Shallow Fusion
  - Deep Fusion makes a model incorporate the LM information as needed
- domain similarity
  - divergence in domains generally hurts model performance
  - LM is most useful when domains of parallel and monolingual corpora are close
- controller mechanism
  - makes model more robust to domain mismatch
  - most active on De-En/Cs-En as LM was able to model the target sentences best here

Motivation - [Chorowski and Jaitly, 2016]

- DNN Automatic Speech Recognition (ASR) systems suffer from
  - overconfidence
    - peaked probability distribution prevents model from finding sensible alternatives
    - harms learning deep layers as gradient approximates 0 on correct characters: $[y_i = c] - p(y_i \mid y_{<i}, x)$
  - integrating LM to combat this results in incomplete transcripts
- idea: better integrate language models

Model - based on [Chan et al., 2015]

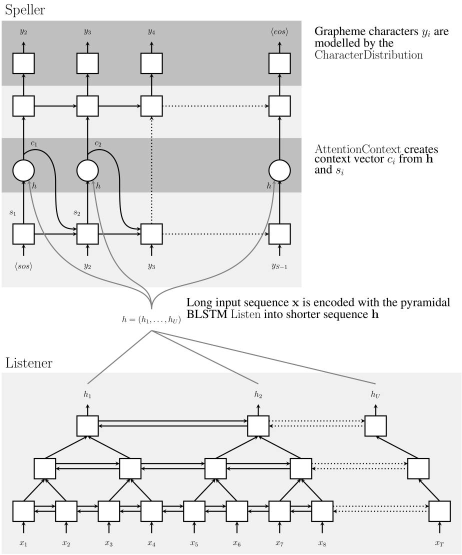

Figure 4: Listen, Attend and Spell model [Chan et al., 2015]

Model

- encodes into a space-delimited sequence of characters directly using the encoder - decoder structure
- Listener: pyramid BLSTM (pBLSTM)

$$h_t^j = pBLSTM(h_{t-1}^{j-1}, [h_{2t}^{j-1}, h_{2i+1}^{j-1}])$$

- Speller
  - attention-based LSTM transducer

$$\begin{aligned} c_i &= AttentionContext(s_t, h) \\ s_t &= RNN(s_{t-1}, y_{t-1}, c_{t-1}) \\ P(y_t \mid x, y_{<t}) &= CharacterDistribution(s_t, c_t) \end{aligned}$$

  - RNN: 2-layer LSTM
  - CharacterDistribution: MLP with softmax
  - beam search to find character sequence y^*

Model - AttentionContext for decoder timestep t

$$c_t = \sum_u \alpha_{t,u} h_u$$

$$\alpha_{t,u} = \frac{exp(e_{t,u})}{\sum_u exp(e_t, u)}$$

$$e_{t,u} = w^T tanh(Ws_{t-1} + Vh_j + Uf_{t,j} + b)$$

$$f_t = F * \alpha_{t-1}$$

- α very sharp probability distribution (attention)
- $e_{t,u}$: location aware scoring mechanism [Chorowski et al., 2015]
  - employs convolutional filters over the previous attention weights
  - extract k $f_{t,j} \in \mathbb{R}^k$ vectors with $F \in \mathbb{R}^{k \times r}$ for every position j of α_{t-1}

Training

- minimize cross-entropy between ground-truth characters and model predictions
- criterion

$$loss(y, x) = -\sum_i \sum_c T(y_i, c) log(p)(y_t \mid y_{<t}, x)$$

- $T(y_i, c)$: target-label function, implemented differently for each model

Language Model Integration - based on [Gülçehre et al., 2015] Shallow Fusion

- extends beam search with a language modelling term

$$y^* = argmin_y - log(p(y \mid x)) - \lambda log(p_{LM}(g)) - \gamma coverage$$

- λ, γ: tunable parameters
- coverage: term promoting longer transcript to avoid incomplete transcripts
- overconfidence drastically influences $-log(p(y \mid x))$ if deviation from network's best guess
- using a non-heuristic measurements does not change results, yet is far more difficult to compute

Experimental Setup

- dataset
  - Wall Street Journal dataset
  - extracted and augmented 80-dimensional mel-scale filterbanks
  - vocabulary consists of lower case letters, space, apostrophe, noise marker, SOS, EOS
- LM
  - extended-vocabulary trigram LM constructed with Kaldi [Povey et al., 2011] and spelling lexicon [Miao et al., 2015]
- baseline
  - Listener: 4 BLSTM, 3 time-pooling layers
  - Speller: LSTM, character embeddings, attention MLP
  - beam size: 10
- final model: baseline + LM

Improvement tests - Overconfidence

- problem
  - good prediction of speller results in little training signal through the attention mechanism to the listener
  - next predicted character may have low accuracy
  - beam search's usefulness is greatly reduced
- Solution 1: Temperature Parameter T

$$p(y_i) = \frac{exp(l_i/T)}{\sum_j exp(l_j/T)}$$

  - increase T: more uniform distribution; more deletion errors; better beam search results
  - constrain emission of EOS

Improvement tests - Overconfidence: Label Smoothing

- variants of label smoothing
  - unigram label smoothing
  - neighborhood smoothing
- model is regularized
- higher entropy of network predictions

Improvement tests - Overconfidence

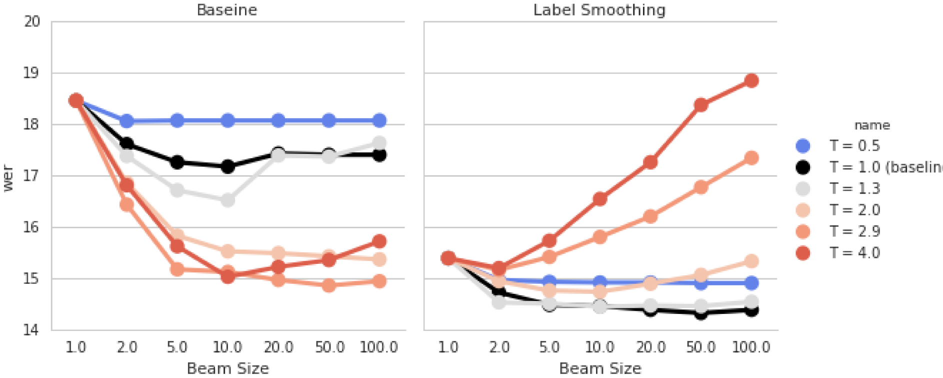

Figure 5: influence of temperature parameter on softmax [Chorowski and Jaitly, 2016]

- problem: using LM + beam search results in incomplete transcripts
- solution: extend beam search criterion with coverage term to promote long transcripts

$$coverage = \sum_j [\sum_i \alpha_{ij} > \tau]$$

- prevents looping over the utterance

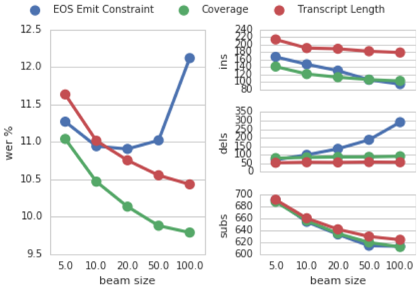

Figure 6: impact of techniques to promote long sequences [Chorowski and Jaitly, 2016]

Results

Model	Parameters	dev93	eval92
CTC [3]	26.5M	-	27.3
seq2seq [12]	5.7M	-	18.6
seq2seq [24]	5.9M	-	12.9
seq2seq [22]	-	-	10.5
Baseline	6.6M	17.9	14.2
Unigram LS	6.6M	**13.7**	**10.6**
Temporal LS	6.6M	14.1	10.7

(a) baseline configuration and results

Model	dev93	eval92
seq2seq [12]	-	9.3
CTC [3]	-	8.2
CTC [6]	-	7.3
Baseline + Cov	12.6	8.9
Unigram LS + Cov.	9.9	7.0
Temporal LS + Cov.	**9.7**	**6.7**

(b) trigram language model

Table 4: results on WSJ [Chorowski and Jaitly, 2016]

- competitive with other non-HMM techniques at the time

Conclusion

- integrating LM into sequence-to-sequence models is competitive with other approaches, if done successfully
- integrating a LM into a NN has proven to be difficult
- LM is especially useful in cases the NN itself is uncertain
- ablation study to see the actual effect of the LM on the results is advisable

Critique and Discussion

References I

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014).
Neural machine translation by jointly learning to align and translate.
cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation.

[Chan et al., 2015] Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015).
Listen, attend and spell.
CoRR, abs/1508.01211.

[Chorowski et al., 2015] Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. (2015).
Attention-based models for speech recognition.
CoRR, abs/1506.07503.

References II

[Chorowski and Jaitly, 2016] Chorowski, J. and Jaitly, N. (2016).
Towards better decoding and language model integration in sequence to sequence models.
CoRR, abs/1612.02695.

[Goodfellow et al., 2013] Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013).
Maxout networks.

[Gülçehre et al., 2015] Gülçehre, Ç., Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H., Bougares, F., Schwenk, H., and Bengio, Y. (2015).
On using monolingual corpora in neural machine translation.
CoRR, abs/1503.03535.

[Jean et al., 2014] Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2014).
On using very large target vocabulary for neural machine translation.
CoRR, abs/1412.2007.

References III

[Miao et al., 2015] Miao, Y., Gowayyed, M., and Metze, F. (2015).

EESEN: end-to-end speech recognition using deep RNN models and wfst-based decoding.

CoRR, abs/1507.08240.

[Povey et al., 2011] Povey, D., Ghoshal, A., Boulianne, G., Goel, N., Hannemann, M., Qian, Y., Schwarz, P., and Stemmer, G. (2011).

The kaldi speech recognition toolkit.

In *In IEEE 2011 workshop*.