Attention-based ASR utilizing Byte-Pair Encoding

Albert Zeyer, Ralf Schlüter

Human Language Technology and Pattern Recognition — Lehrstuhl Informatik 6
Fakultät für Mathematik, Informatik und Naturwissenschaften
RWTH Aachen University

End-to-end

Current state-of-the-art: **hybrid HMM/ANN** approach

- usually based on initial Gaussian mixture HMM training
- operates on phone level using pronunciation lexica and full word forms
- end-to-end scheme possible:
  - lattice-free MMI [Povey$^+$ 2016]
  - GMM-free incl. phonetic decision tree [Gosztolya$^+$ 2016]

What is **"end-to-end"**?

- lower end: input features, e.g. MFCCs
- upper end: ouput labels, e.g. characters, words, subwords; here: byte-pair encoding (BPE)
- aim: homogeneous modeling, training and decoding
- (no pronunciation lexicon, no phone model and clustering)

Models:

- HMM, CTC, ASG, LF-MMI
- encoder-decoder with attention (here)
- inverted HMM / segmental RNN, recurrent transducer, recurrent neural aligner

Except for HMM, discriminative model: $p(y_1^N | x_1^T) = \prod_{i=1}^{N} p(y_i | [y_1^{i-1}], x_1^T)$

Model: Encoder-decoder with attention

Encoder

- high-level feature representation/transformation
- deep bi-directional LSTM network
- max-pooling in time: optional sub-sampling following each LSTM layer
- input feature vector sequence x_1^T,
- encoder output:

$$h_1^{T'} = \mathsf{LSTM}_{\#\mathrm{enc}} \circ \cdots \circ \mathsf{max\text{-}pool}_1 \circ \mathsf{LSTM}_1(x_1^T),$$

  - $T' = \mathrm{red} \cdot \mathrm{T}$ with time reduction factor red,
  - $\#\mathrm{enc}$: number of encoder layers, $\#\mathrm{enc} \geq 2$.

Model: Encoder-decoder with attention

Decoder

- attention energies for encoder time-step t and decoder step (output label position) i:

$$e_{i,t} = v^\top \tanh(W[s_i, h_t, \beta_{i,t}]),$$

  with trainable vector v and trainable matrix W, current decoder state s_i, and encoder state h_t.
  Note: no dependence on symbol y_i to be hypothesized in position i!
- attention weight feedback: influence of attention used in earlier decoder steps

$$\beta_{i,t} = \sigma(v_\beta^\top h_t) \cdot \sum_{k=1}^{i-1} \alpha_{k,t}, \text{ with trainable vector } v_\beta.$$

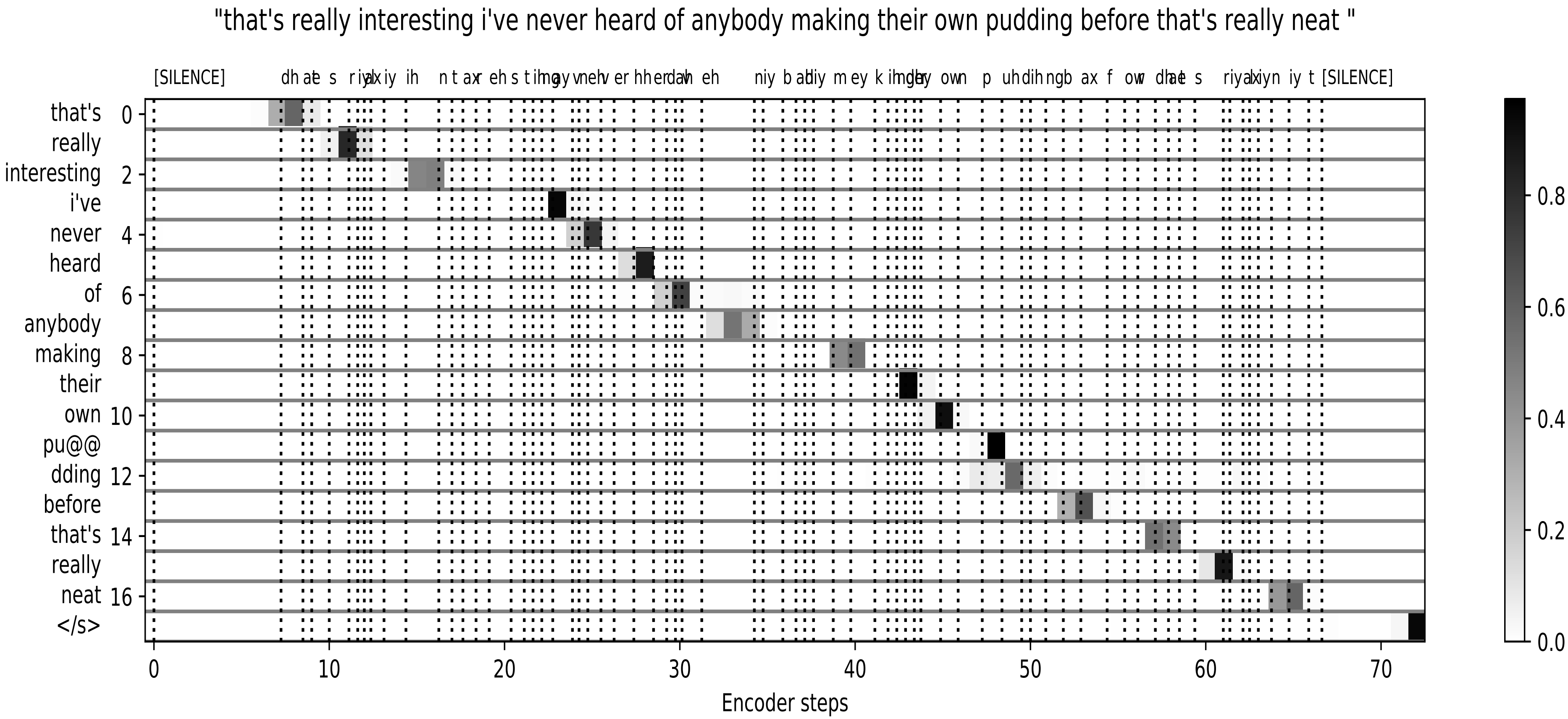
- attention weights: $\alpha_i = \mathrm{softmax}_t(e_i)$, normalized over time
- attention context vector: input to decoder

$$c_i = \sum_{t=1}^{T'} \alpha_{i,t} h_t$$

- decoder state: $s_i = \mathrm{LSTMCell}(s_{i-1}, y_{i-1}, c_{i-1})$
- decoder output prediction probability:

$$p(y_i|y_1^{i-1}, x_1^T) = \mathrm{softmax}(\mathrm{linear} \circ \mathrm{maxout} \circ \mathrm{linear}(s_i, y_{i-1}, c_i))$$

Attention Example



- grayscale reflects attention weights for each time frame

Time reduction and pretraining

- input sequences are much longer than output sequences (e.g. 30 times)
- continuous input: downscaling / time reduction can work
- time reduction factors:
  - low, e.g. 8 or less: hard to get to convergence
    (did not converge at all for us in most cases, or was very bad)
  - high, e.g. 16 or 32: converges fast and nicely

- deep hybrid LSTM models can benefit from layer-wise pretraining:
  start with 1 or 2 layers, add more and more layers
- same for the deep LSTM encoder
  - first we showed that it works for machine translation
  - also works for speech recognition
- pretraining in speech recognition with time reduction scheduling:
  - start with high time reduction (32), and then reduce (to 8)
  - get better performance with lower time reduction
- pretraining with other scheduling variants:
  - label smoothing (initially disabled)
  - dropout (initially disabled)

Pretraining

- Pretraining study for machine translation (WMT 2017 German→English task):

encoder num. layers	BLEU [%] no pretrain	BLEU [%] with pretrain
2	29.3	-
3	29.9	-
4	29.1	30.3
5	-	30.3
6	-	30.6
7	-	**30.9**

- Time reduction for ASR (Switchboard), directly starting with:
  - time reduction factor 8, 2 layers, or
  - time reduction factor 32, 6 layers:

  did not work
- might work with more careful tuning, not needed with pretraining

Optimal time reduction factor

- always with pretraining, starting with time reduction factor 32
- Switchboard 300h, Hub5'00 (SWB+CH) results:

factor	WER [%]
4	(out of memory)
8	**20.4**
16	21.0
32	21.9

- (factors in between not straight forward with our max-pooling time reduction)

Byte-Pair Encoding

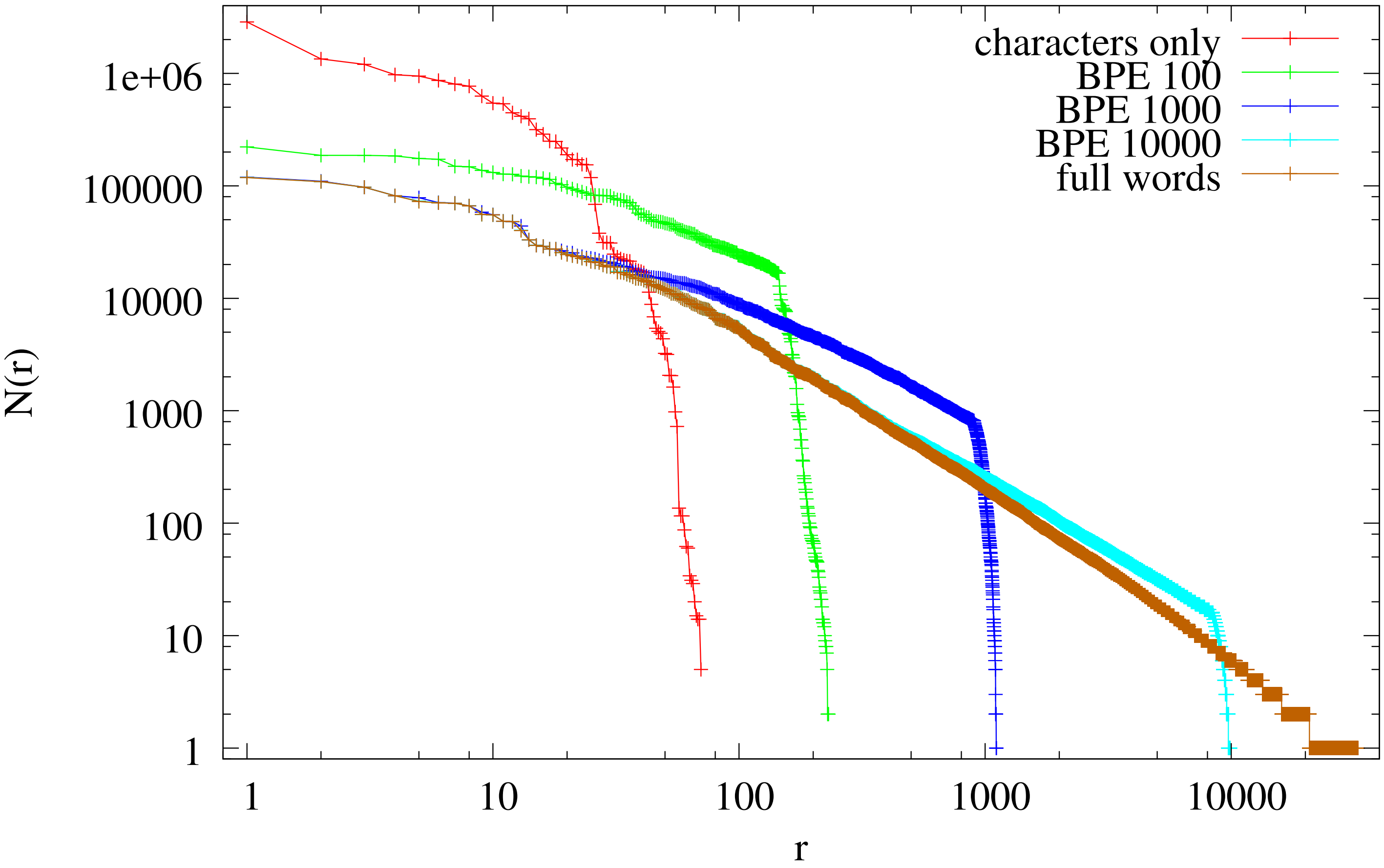
Goals: enable open vocabulary and avoid pronunciation lexicon

- character vocabulary:
  - small label set
  - enables open vocabulary
  - pronunciation ambiguous/context dependent
- word vocabulary:
  - large label set
  - fixed/limited vocabulary
  - pronunications well-defined

Byte-pair encoding:

- starts from character vocabulary and corresponding segmentation
- iteratively merges most frequent label bigrams ("byte-pairs")
- always keeps byte-pair constituents in vocabulary
- word internal processing only, word internal boundary symbol '@', attached to left character
- leads to intermediate size of label set

Byte-Pair Encoding: Beyond Zipf's Law [Montemurro 2001]



- rank r vs. frequency $N(r)$ in training corpus

Switchboard 300h

- 6 layer encoder attention model, 1 layer decoder
- [1]added noise from external data. [2]added the lexicon, i.e. also additional data.

model	LM	label unit		WER[%] Hub5'00 SWB	WER[%] Hub5'00 CH	WER[%] Hub5'01
JHU, LF MMI, 2016	4-gram	CDp		9.6	19.3	
hybrid (this work)	LSTM	CDp		**8.3**	**17.3**	12.9
Edinburgh, attention, 2016	3-gram	words		25.8	46.0	
Toyota, attention, 2017	none	chars		23.1	40.8	
Stanford, CTC, 2015	RNN	chars		21.4	40.2	
Baidu DeepSpeech, CTC[1], 2014	RNN	chars		20.0	31.8	
Microsoft, CTC[2], 2017	word RNN	chars		14.0	**25.3**	
attention (this work)	none	BPE	10K	13.5	27.1	19.9
attention (this work)	none	BPE	1K	13.1	26.1	19.7
attention (this work)	LSTM	BPE	1K	**11.8**	25.7	18.1

LibriSpeech 1000h

- 6 layer encoder attention model, 1 layer decoder

model	LM	label unit	WER[%]			
			dev		test	
			clean	other	clean	other
JHU, hybrid, FFNN, 2015	4-gram	CDp	4.90	12.98	5.51	13.97
JHU, LF MMI, LSTM, 2016					4.28	
Baidu DeepSpeech2, CTC, 2015	4-gram	chars			5.33	13.25
Facebook, ASG (CTC), 2017					4.80	14.50
Salesforce, CTC, PL, 2017			5.10	14.26	5.42	14.70
attention (this work)	none	BPE	4.87	14.37	4.87	15.39
	4-gram		4.79	14.31	4.82	15.30
	LSTM		**3.54**	**11.52**	**3.82**	**12.76**

Beam Search Error Analysis

- on LibriSpeech, without language model
- reference-related search errors:
  percentage of segments with recognition score worse than reference score:

beam size	search errors [%] (WER [%])			
	dev		test	
	clean	other	clean	other
4	1.52 (4.87)	1.68 (14.53)	1.07 (4.87)	1.70 (15.49)
8	0.96 (4.88)	0.98 (14.40)	0.76 (4.87)	1.02 (15.39)
12	0.81 (4.87)	0.59 (14.37)	0.61 (4.86)	0.71 (15.39)
16	0.70 (4.87)	0.52 (14.36)	0.50 (4.86)	0.58 (15.37)
32	0.26 (4.87)	0.14 (14.34)	0.19 (4.86)	0.20 (15.34)

Conclusion

- encoder-decoder-attention model for large-vocabulary speech recognition
- target labels: BPE subword units
- pretraining for encoder with time reduction scheduling
- joint beam search with a separately trained LSTM LM
- current experimental results:
  - 300h-Switchboard:
    competitive results compared to other end-to-end models
    WERs are still higher than the conventional hybrid systems
  - 1000h-LibriSpeech:
    near to state-of-the-art

- open issues:
  - robustness of training process?
  - how to accomodate lower amounts of training data?
  - need for further model structuring
    $\rightarrow$ as encoder is similar to networks used in hybrid approach: more robust attention model?

Thank you for your attention

Questions?