Promising Accurate Prefix Boosting for Sequence-to-sequence ASR

Karthick Baskar, Lukáš Burget, Shinji Watanabe and Martin Karafiat

Johns Hopkins University

Brno University of Technology
Faculty of Information Technology

14th May 2019

Premilinaries

Premilinaries

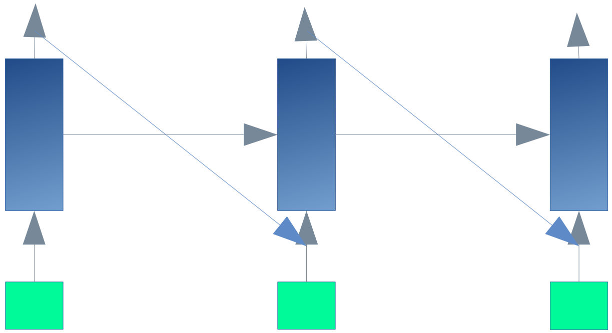
- What is Prefix ?
  - In the context of ASR, prefix refers to a partial sequence

Premilinaries

- What is Prefix ?
  - In the context of ASR, prefix refers to a partial sequence
- Why boost accurate prefix ??
  - Training by boosting correct prefixes (accurate) over wrong prefixes will help model to rectify its own errors

Encoder - Decoder

- Encoder:
  - recurrent layers
  - entire input sequence to fixed-length vector
- Decoder:
  - recurrent layers with final softmax layer
  - predict probability for the next symbol of the output sequence in an auto-regressive fashion
  - learns an implicit language model for the output sequences



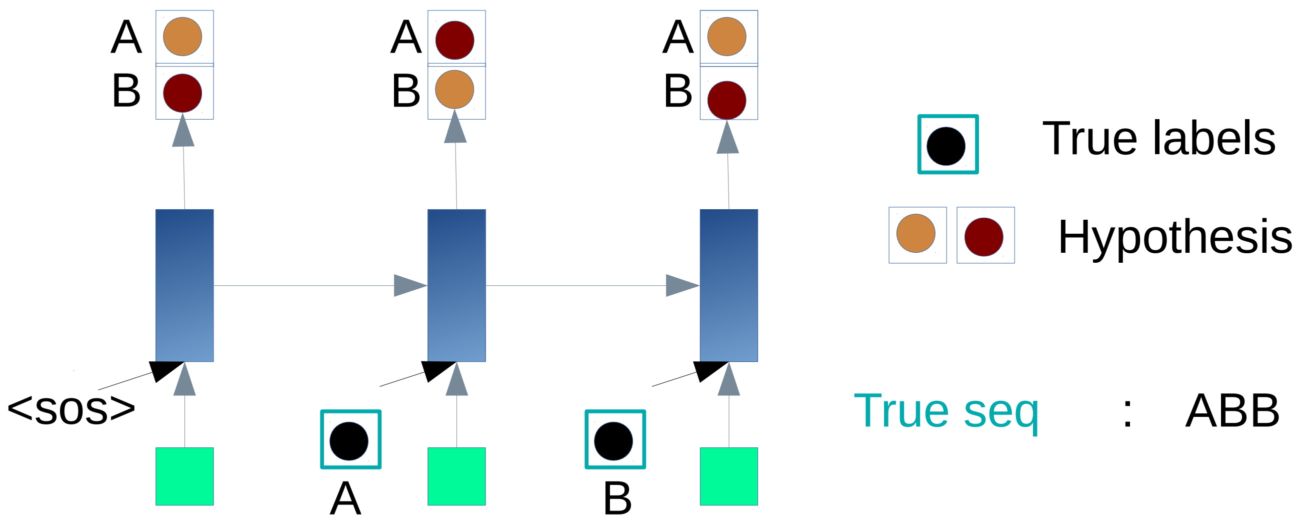
Problem Overview

- **Exposure bias**
  - Training: output character is conditioned on the previous true character
  - Testing: the model needs to rely on its own previous predictions
- **Error criterion mismatch**
  - Training: the objective is the conditional maximum likelihood (cross entropy) for maximizing the probability of the correct sequence
  - Testing: Character error rate (CER) or word error rate (WER)

Training: Minimize cross-entropy loss of each target token y_l^* (character)

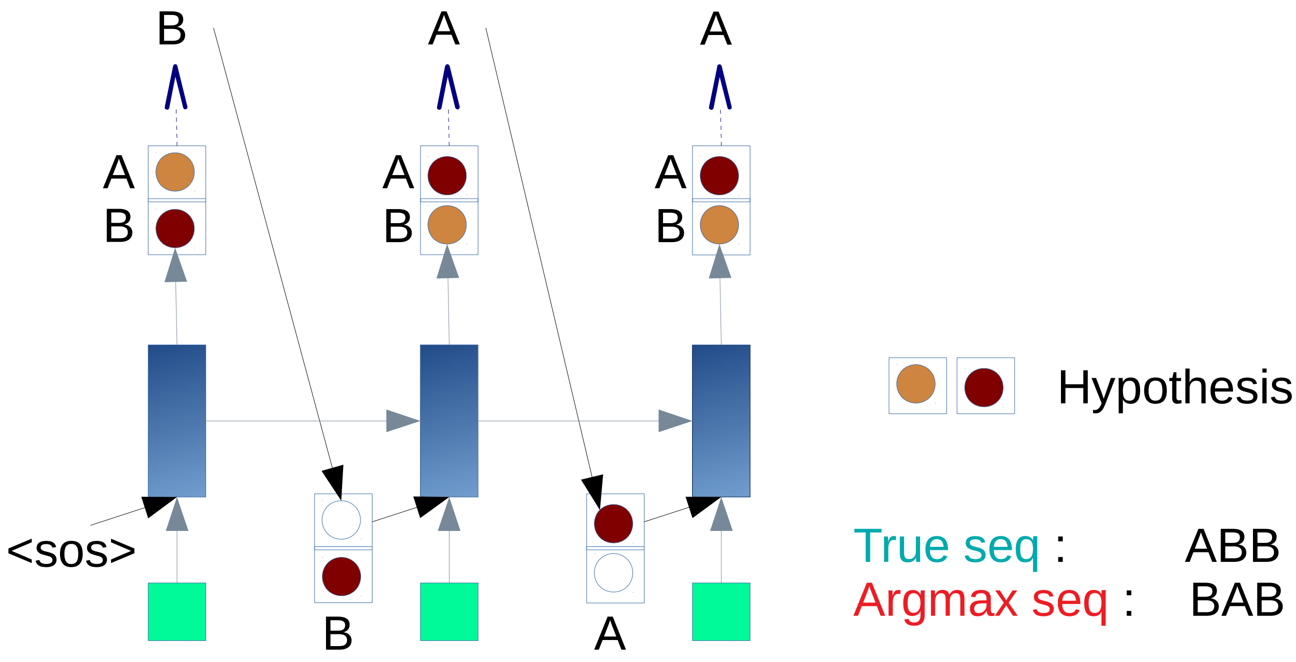
$$\mathrm{logp}(\mathrm{y}^*|X) = \sum_l logp(y_l^*|X)$$

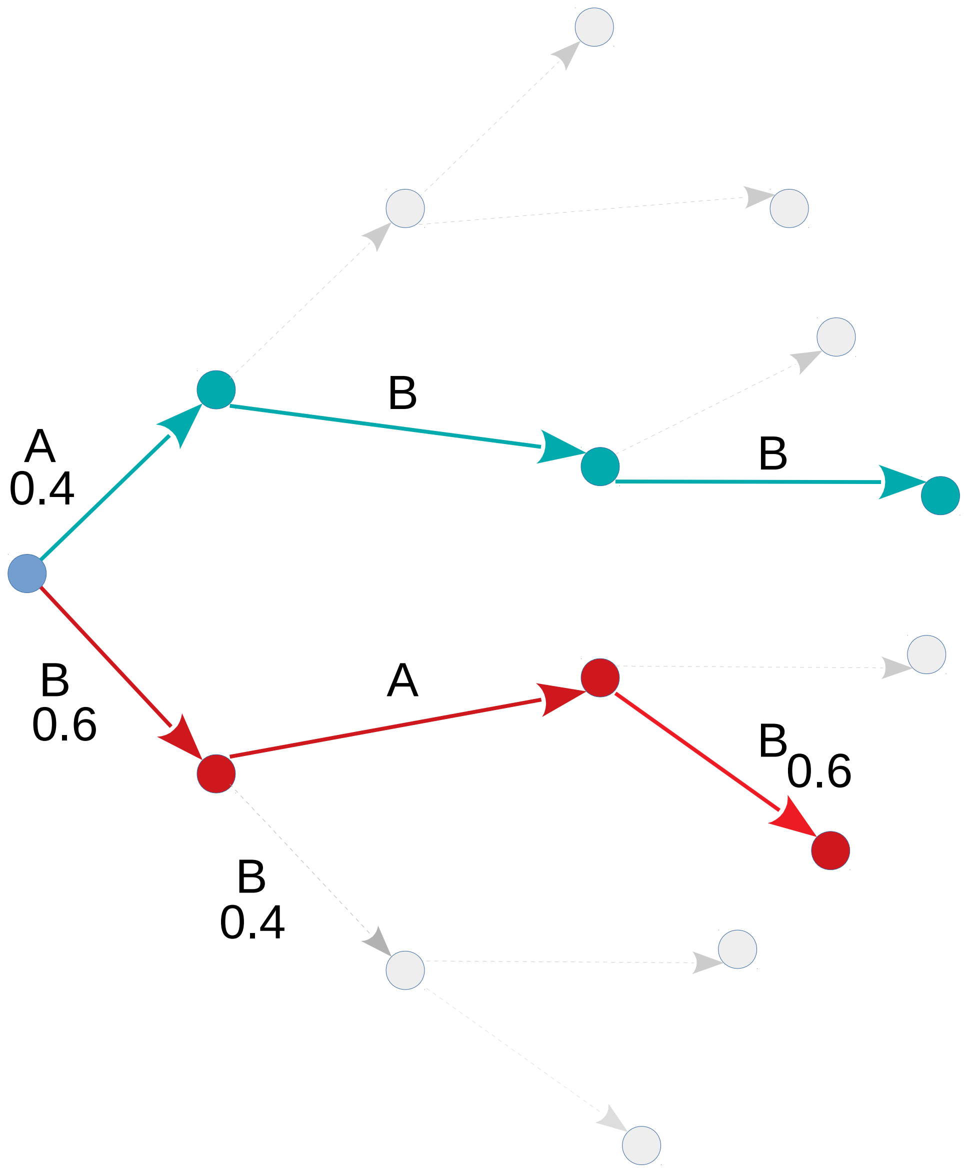
Teacher-forcing: Feed previous token from ground-truth as auxiliary info to predict current token

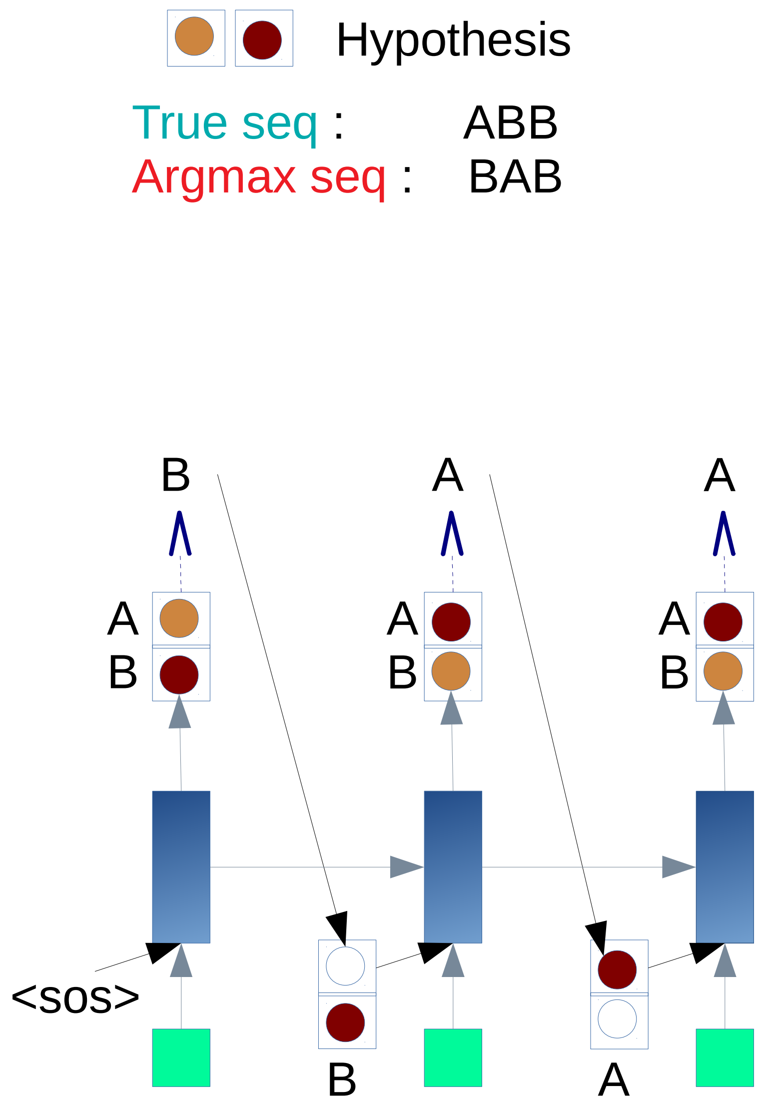


Decoding:

- Previous token from hypothesis is fed to predict current token
- Output sequence is predicted in two ways
  - Greedy (argmax) search
  - Beam search




A
0.4
B
B
B
0.6
A
B
0.6
B
0.4


Hypothesis
True seq : ABB
Argmax seq : BAB
B
A
A
A
B
A
B
A
B
<sos>
B
A

Decrease the training loss for the predicted paths !!

Training is matched to testing

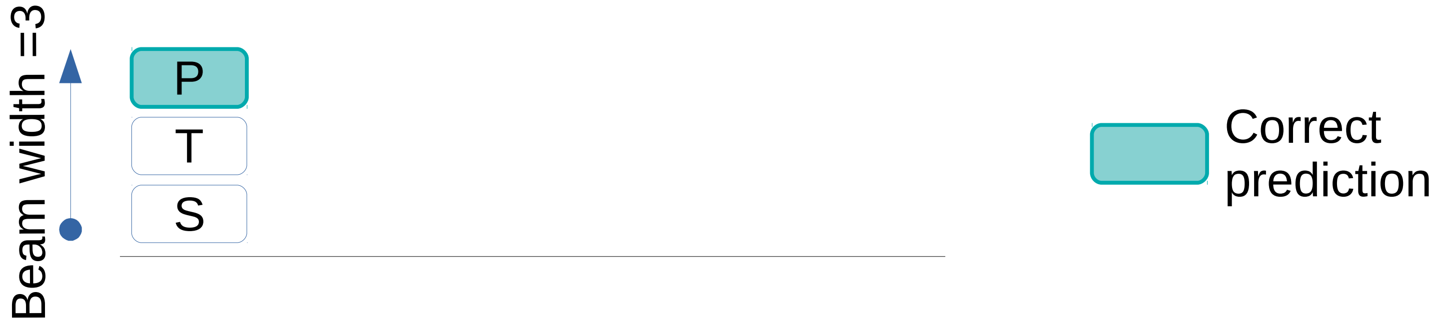
Is there a technique to train only with predictions as previous tokens ??

Decoding:

- Previous token from hypothesis is fed to predict current token
- Output sequence is predicted in two ways
  - Greedy (argmax) search
  - Beam search

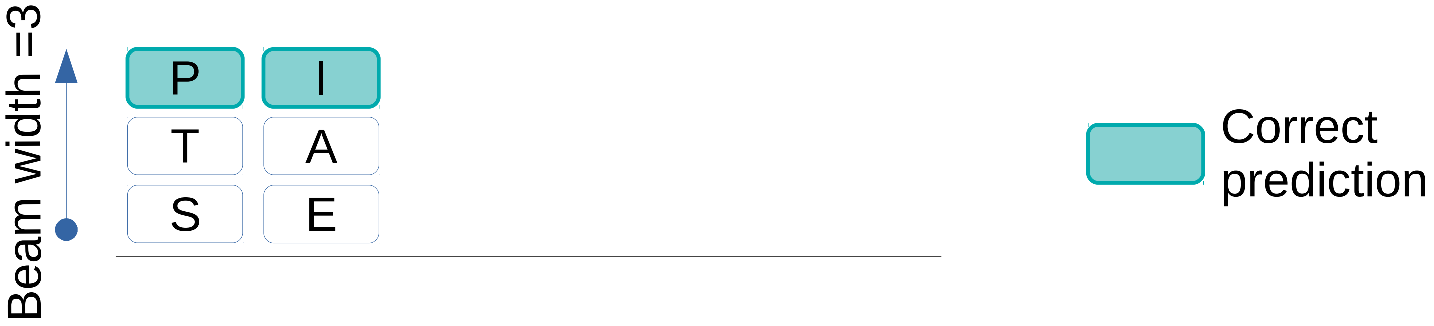
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



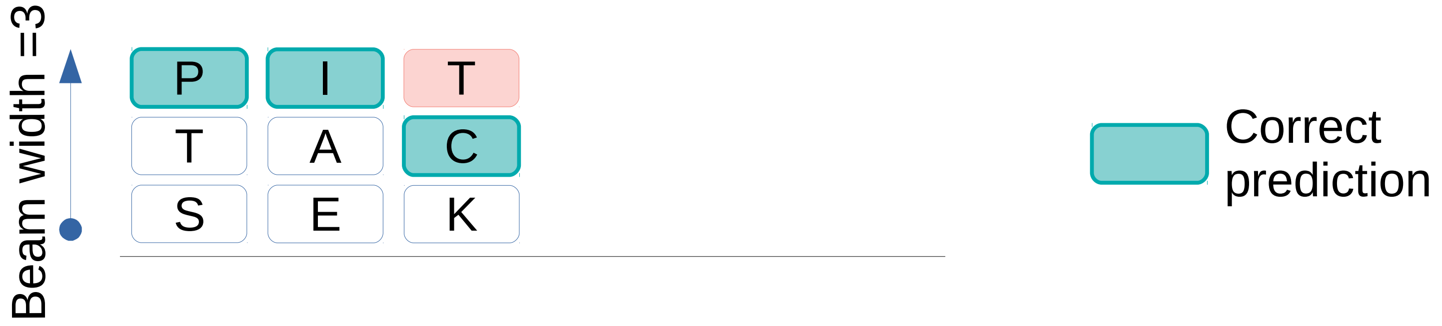
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



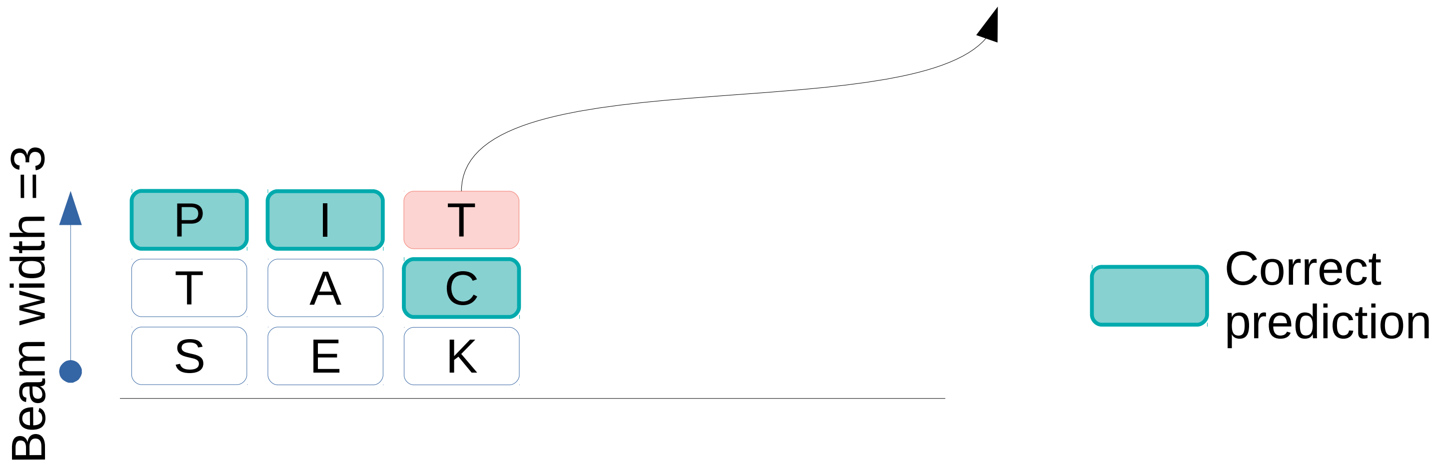
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



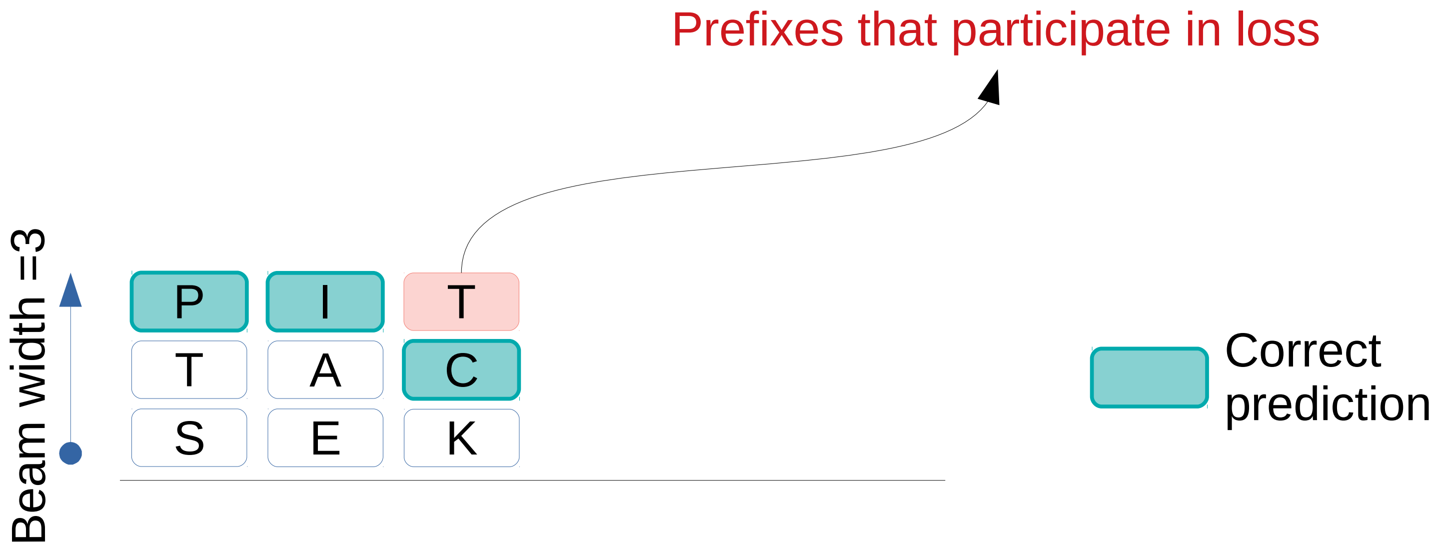
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



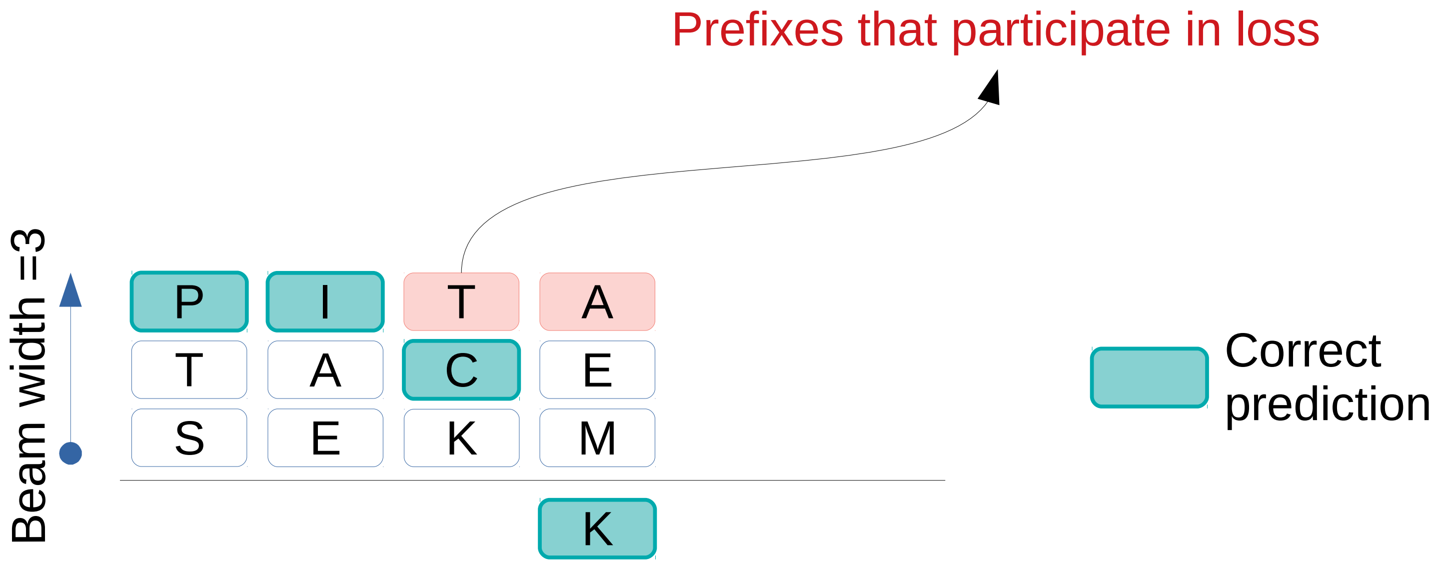
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



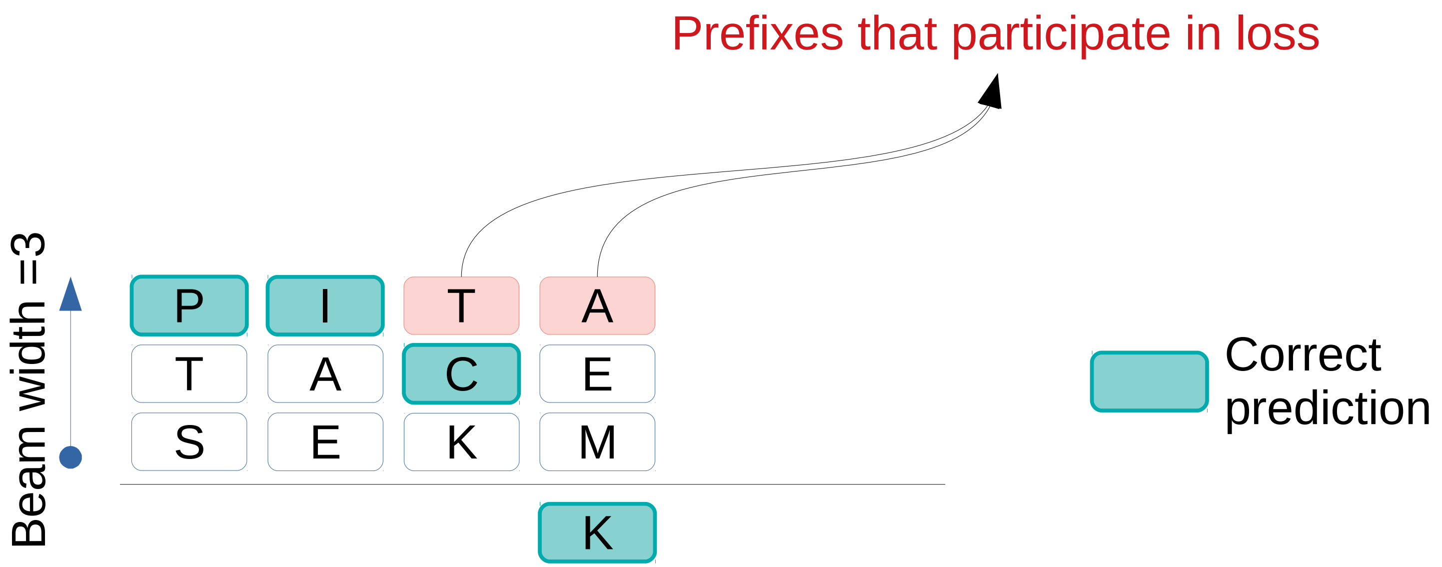
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



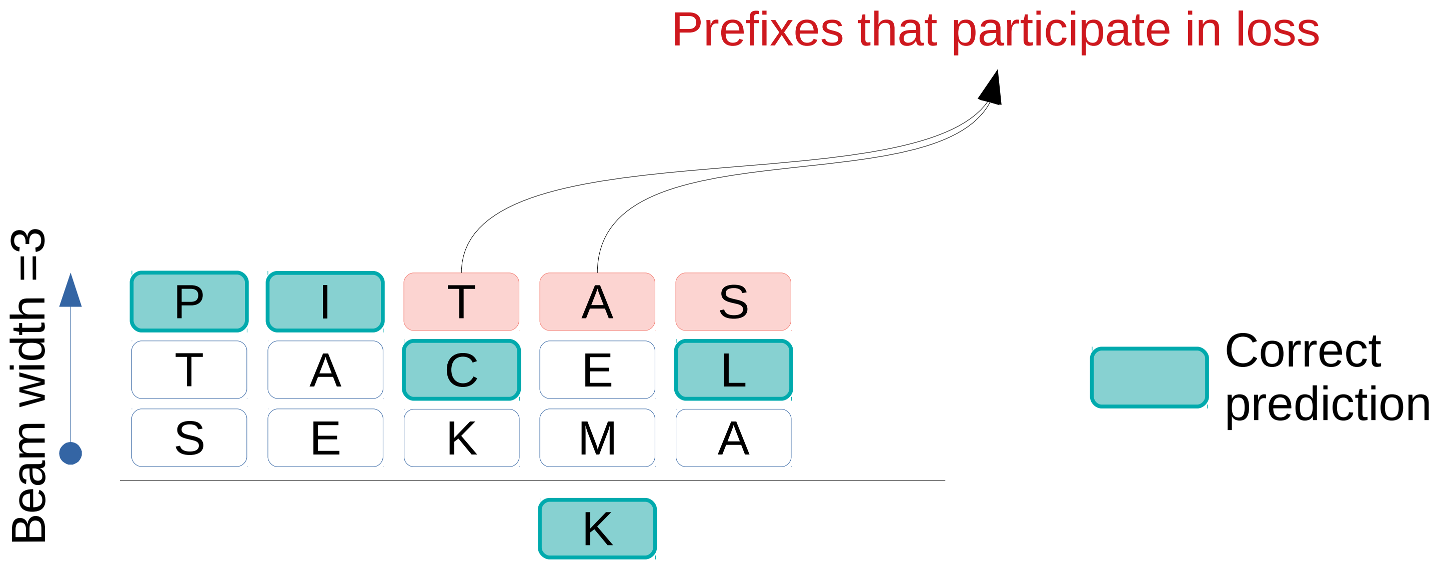
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



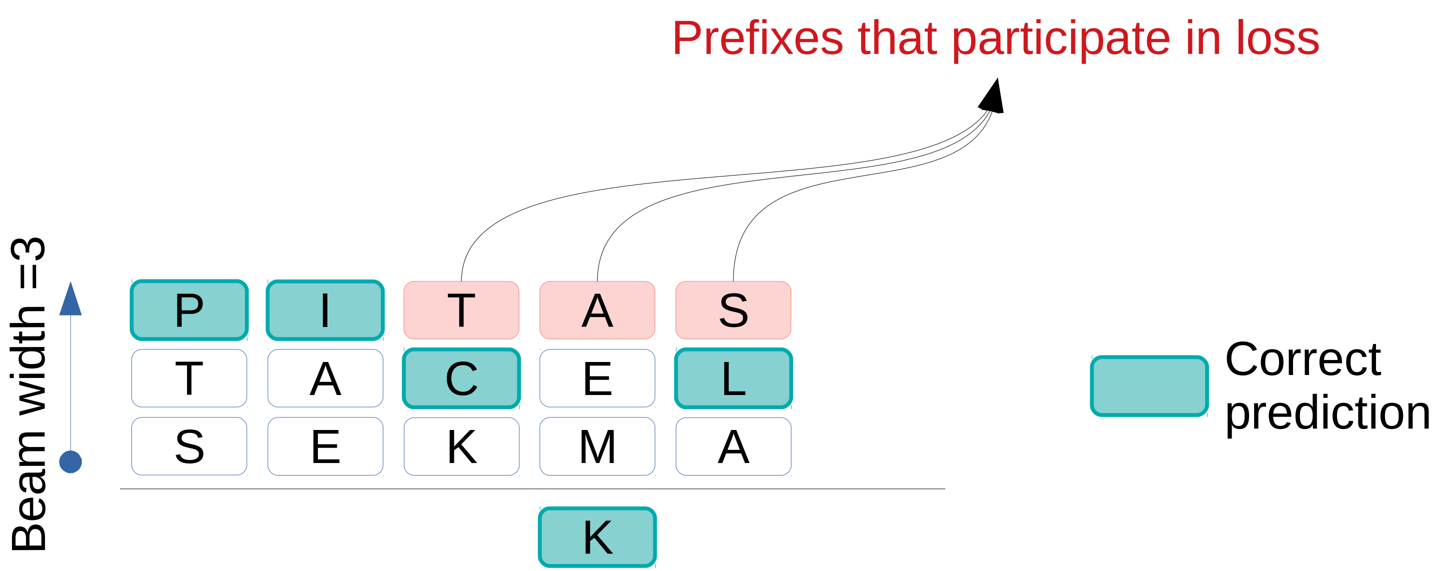
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



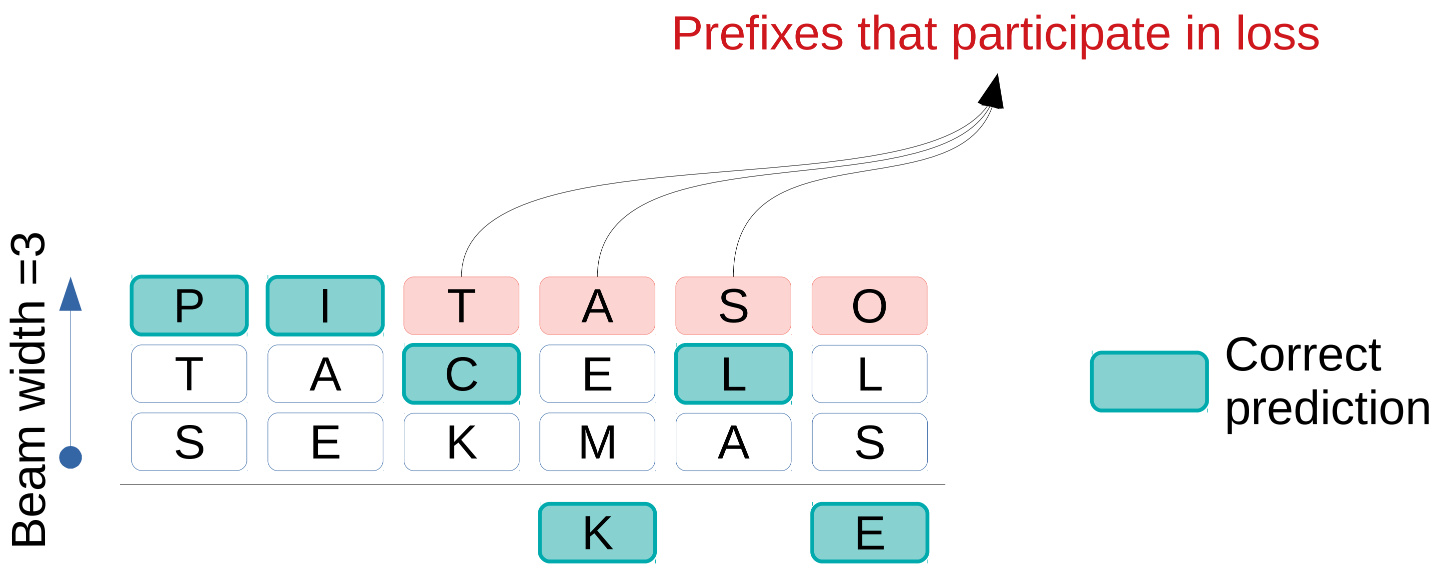
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



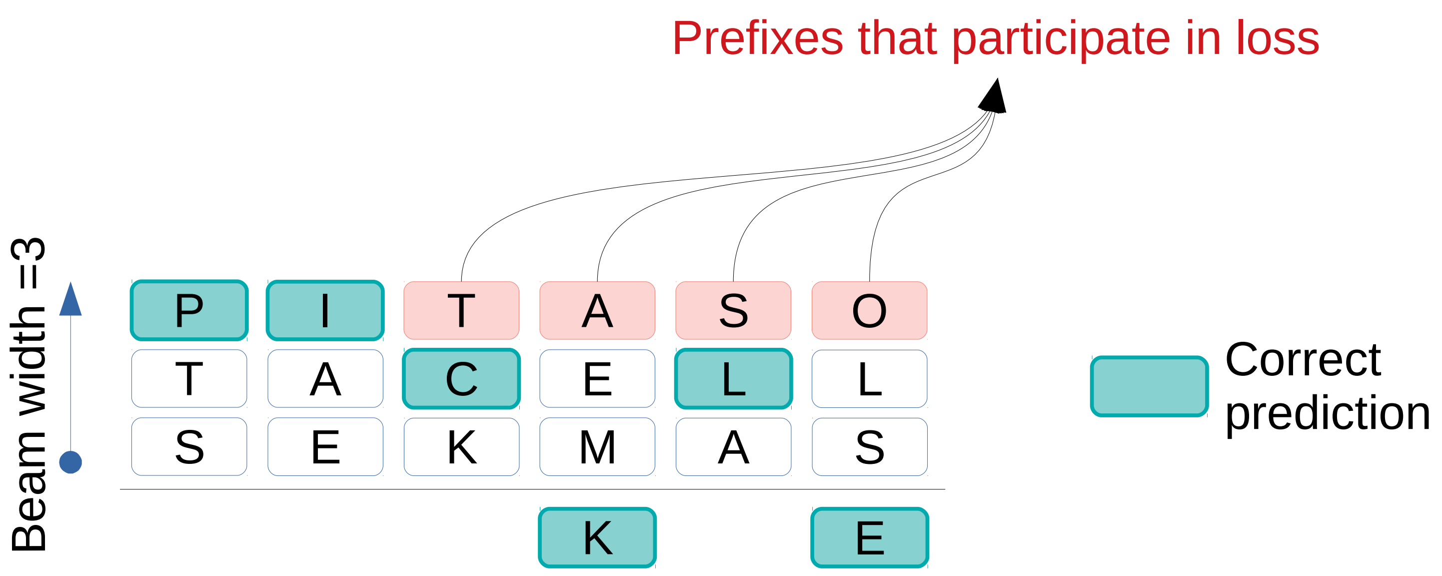
How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



How to match beam-search decoding with training ??

- Need to consider multiple hypothesis generated during beam-search
- Training objective must keep prefix at top of the beam
- Helps to survive pruning by keeping scores higher in the beam



Choose weights

score of true label is better than predicted label by a specific margin

$$\mathcal{L}_{MM} = \sum_{l} - \underbrace{s(y_l^*, X)}_{\text{True label score}} + \max_y \left(\underbrace{s(y, X)}_{\text{predicted label score}} + \underbrace{\alpha \mathrm{Acc}(y_l^*, y)}_{\text{margin}} \right)$$

Maximum margin objective

Choose weights

score of true label is better than predicted label by a specific margin

weight . (true label score) >= (Margin) + weight . (scores of other labels)

$$\mathcal{L}_{MM} = \sum_{l} - s(y_l^*, X) + \max_y \left(s(y, X) + \alpha \mathrm{Acc}(y_l^*, y) \right)$$

True label score

predicted label score

margin

Choose weights

score of true label is better than predicted label by a specific margin

weight . (true label score) >= (Margin) + weight . (scores of other labels)

$$\mathcal{L}_{MM} = \sum_{l} - \underbrace{s(y_l^*, X)}_{\text{True label score}} + \max_y \left(\underbrace{s(y, X)}_{\text{predicted label score}} + \underbrace{\alpha \mathrm{Acc}(y_l^*, y)}_{\text{margin}} \right)$$

Label → Prefix

Maximum margin objective

Choose weights

score of true label is better than predicted label by a specific margin

weight . (true label score) >= (Margin) + weight . (scores of other labels)

$$\mathcal{L}_{MM} = \sum_{l} - s(y_l^*, X) + \max_y \left(s(y, X) + \alpha \mathrm{Acc}(y_l^*, y)\right)$$

True label score

predicted label score

margin

Label → Prefix

Better for training the encoder-decoder because they contain more informative training signals at each step

Maximum margin objective

Choose weights

↓

score of true label is better than predicted label by a specific margin

weight . (true label score) >= (Margin) + weight . (scores of other labels)

$$\mathcal{L}_{MM} = \sum_{l} - \underbrace{s(y_l^*, X)}_{\text{True label score}} + \max\nolimits_y \left(\underbrace{s(y, X)}_{\text{predicted label score}} + \underbrace{\alpha \mathrm{Acc}(y_l^*, y)}_{\text{margin}} \right)$$

Label → Prefix

Better for training the encoder-decoder because they contain more informative training signals at each step

$$\mathcal{L}_{MM} = \sum_{l} - s(y_{1:l}^*, X) + \max\nolimits_y \left(s(y_{1:l}, X) + \alpha \mathrm{Acc}(y_{1:l}^*, y_{1:l}) \right)$$

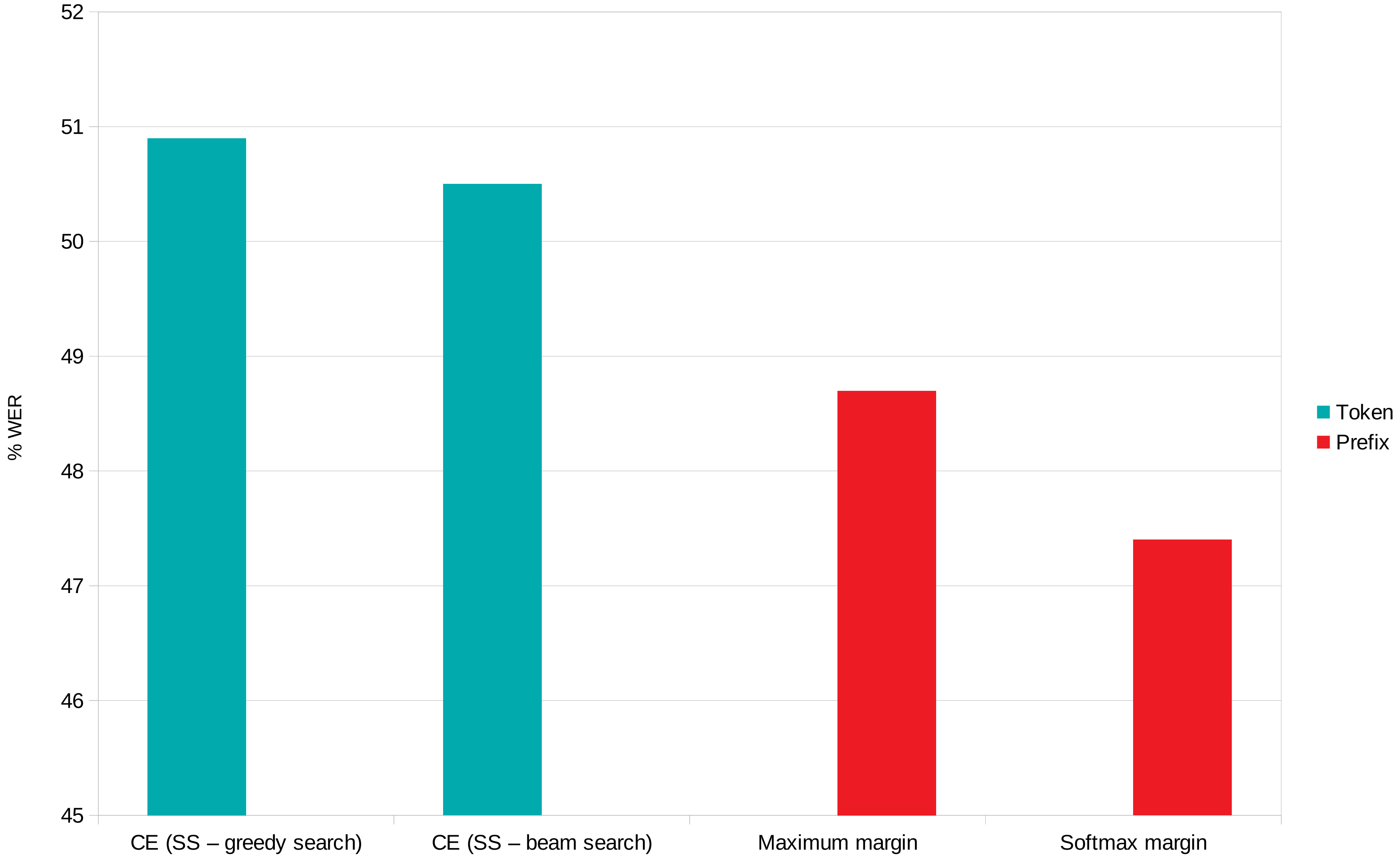
- Hard maximum is replaced by soft maximum "softmax" $(\log \sum \exp)$
- Softmax margin* showed noticeable gains over max margin empirically

$$\mathcal{L}_{SM} = \sum_{l} - s(y^*_{1:l}, X) + \log(\textstyle\sum_y \exp(s(y_{1:l}, X) + \alpha \mathrm{Acc}(y^*_{1:l}, y_{1:l})))$$

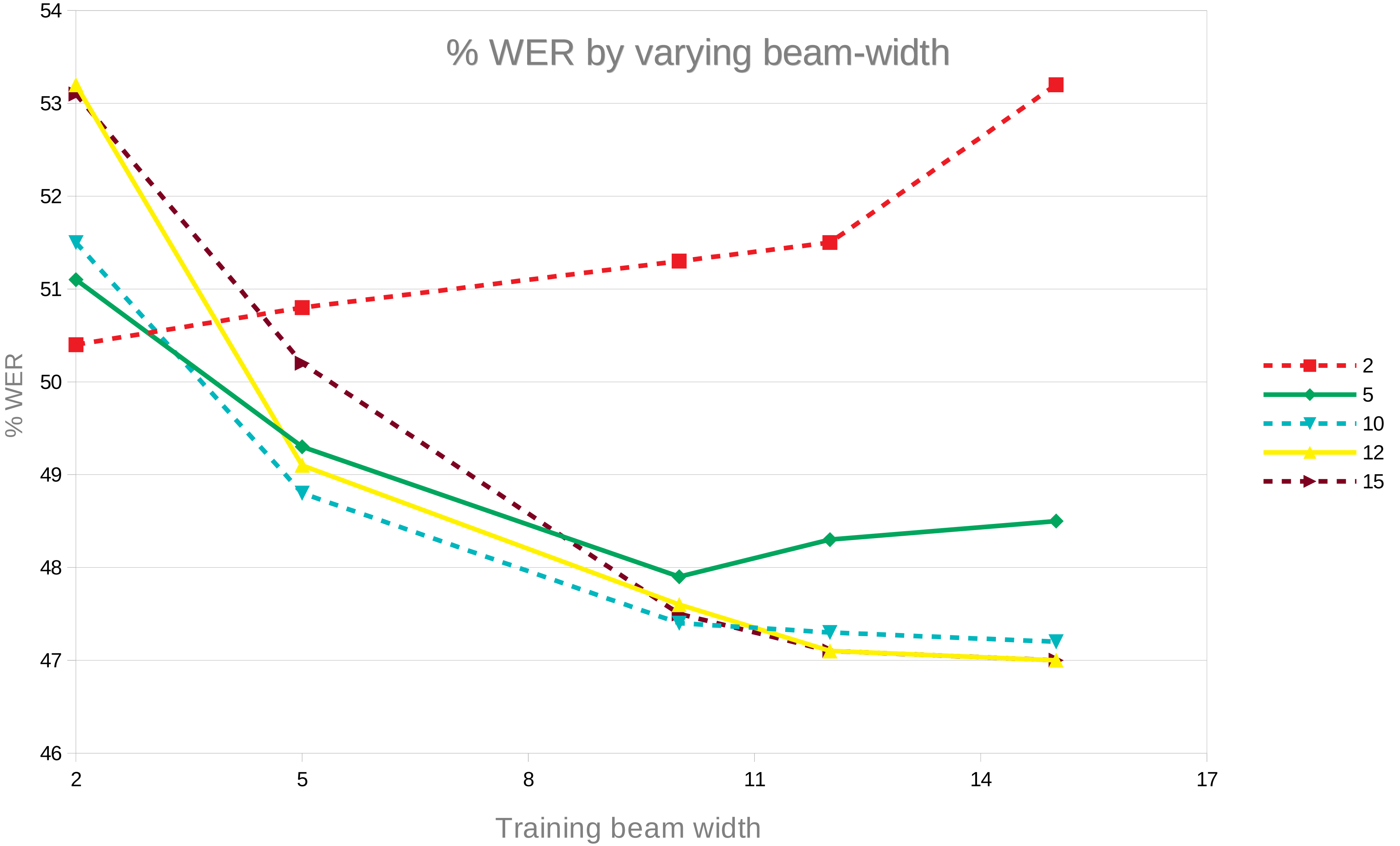
- Generalization of boosted MMI (bMMI) criterion

* K. Gimpel and N. A. Smith, "Softmax-margin training for structured log-linear models," 2010

Promising accurate prefix boosting (PAPB)



% WER on held-out set with PAPB



Comparison with sequence-level objective

- Sequence-level optimization technique: Minimum Bayes Risk Criterion*

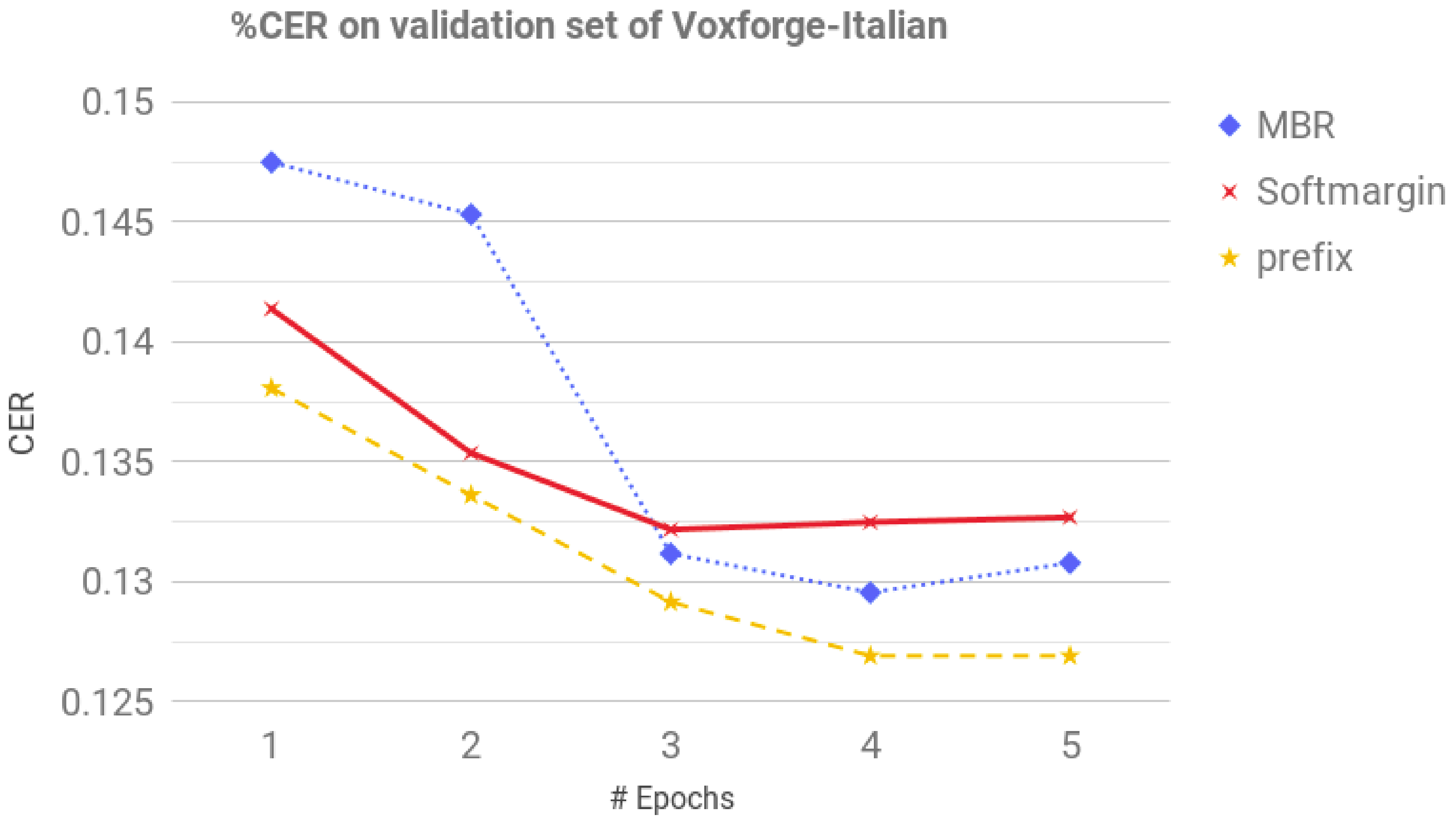
$$\mathcal{L}_{MBR} = E_{p(\boldsymbol{y}|\boldsymbol{X})}\left[\mathrm{Acc}(y^*, y)\right] = \sum_{y \epsilon Y} p(y|X)\mathrm{Acc}(y^*, y)$$

- Obtain sequence predictions from model distribution and backpropagate a sequence-level objective

- Y denotes the N-best sequences selected using beam search

* R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, and A. Kannan, “Minimum word error rate training for attention-based sequence-to-sequence models,” in ICASSP, 2018, pp. 4839–4843, IEEE, 2018

CER on held-out set with PAPB



Impact of pretraining and CE regularization

CE	Pretraining	MBR (%WER)	% Rel. drop	PAPB (%WER)	% Rel. drop
Y	Y	11.5	-	10.8	-
Y	N	Hard to train	-	14.9	27.5
N	Y	13.8	16.7	11.5	6.1

Impact of pretraining and CE regularization

CE	Pretraining	MBR (%WER)	% Rel. drop	PAPB (%WER)	% Rel. drop
Y	Y	11.5	-	10.8	-
Y	N	Hard to train	-	14.9	27.5
N	Y	13.8	16.7	11.5	6.1

- Pretraining is crucial for sequence-level objective such as MBR training

Impact of pretraining and CE regularization

CE	Pretraining	MBR (%WER)	% Rel. drop	PAPB (%WER)	% Rel. drop
Y	Y	11.5	-	10.8	-
Y	N	Hard to train	-	14.9	27.5
N	Y	13.8	16.7	11.5	6.1

- Pretraining is crucial for sequence-level objective such as MBR training
- PAPB did show convergence without pretraining

Impact of pretraining and CE regularization

CE	Pretraining	MBR (%WER)	% Rel. drop	PAPB (%WER)	% Rel. drop
Y	Y	11.5	-	10.8	-
Y	N	Hard to train	-	14.9	27.5
N	Y	13.8	16.7	11.5	6.1

- Pretraining is crucial for sequence-level objective such as MBR training
- PAPB did show convergence without pretraining
- CE regularization provides 6.1 % and 16.7% relative gain for PAPB and MBR

Recognition performance on WSJ corpus

Effect of LM on token level, sequence level and prefix (partial sequence) level training

Model type	No RNNLM		Character RNNLM		Word RNNLM	
	%CER	%WER	%CER	%WER	%CER	%WER
CE	4.6	12.9	2.5	5.8	2.0	4.8
MBR	4.3	11.5	2.5	5.4	2.1	4.3
PAPB	4.0	10.8	2.1	4.5	2.0	**3.8**
Deep-CNN*	-	10.5	-	-	-	-
OCD*	-	**9.6**	-	-	-	-
LF-MMI*	-	-	-	-	-	4.1

Recognition performance (%WER) on Librispeech

Effect of LM on token level, sequence level and prefix (partial sequence) level training

Model type (%WER)	No RNNLM		Word RNNLM	
	test-clean	test-other	test-clean	test-other
CE	6.7	21.5	4.0	12.7
MBR	5.5	17.4	3.7	11.3
PAPB	4.7	15.1	3.1	9.8
OCD*	4.5	13.3	-	-
LF-MMI*	-	-	3.8	8.7

* https://github.com/kaldi-asr/kaldi/blob/master/egs/librispeech/s5/local/chain/tuning/run_tdnn_1d.sh

Conclusion and Constraints

- Prefix boosting with softmax-margin objective provides considerable gains
- Effective compared to sequence-level MBR objective
- Beam-search is not an efficient method to run with GPU
- 2-fold increase in training time
- Constraint in setting larger training beam-size
- Future work will be to use sampling approach instead of beam-search

References

- S. Wiseman and A. M. Rush, “Sequence-to-sequence learning as beam-search optimization,” arXiv preprint arXiv:1606.02960, 2016
- D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, “Boosted MMI for model and feature-space discriminative training,” in IEEE ICASSP, pp. 4057–4060, IEEE, 2008.
- K. Vesel`y, A. Ghoshal, L. Burget, and D. Povey, “Sequence-discriminative training of deep neural networks.,” in INTERSPEECH, pp. 2345–2349, 2013.
- H. Su, G. Li, D. Yu, and F. Seide, “Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription,” in ICASSP, 2013, pp. 6664–6668, IEEE, 2013
- S. Sabour, W. Chan, and M. Norouzi, “Optimal completion distillation for sequence learning,” arXiv preprint arXiv:1810.01398, 2018