Efficient Sequence Modeling

Michael Auli

Alexei Baevski, Angela Fan, Edouard Grave, Felix Wu, Jiatao Gu,

Maha Elbayad, Yann Dauphin

Facebook AI Research (FAIR)

Sequence models (NLP) ⟷ ConvNets (Vision)

"*Sequence to sequence learning is the basis for 25% of papers at ACL'17*"

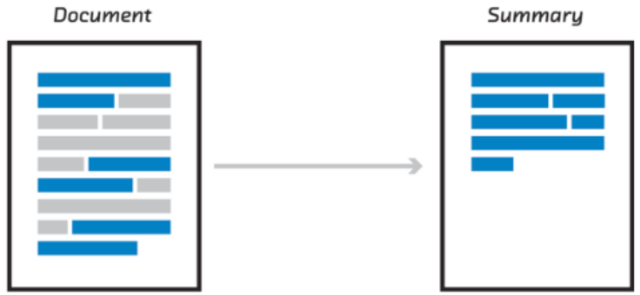
— ACL'17 Keynote by Mirella Lapata

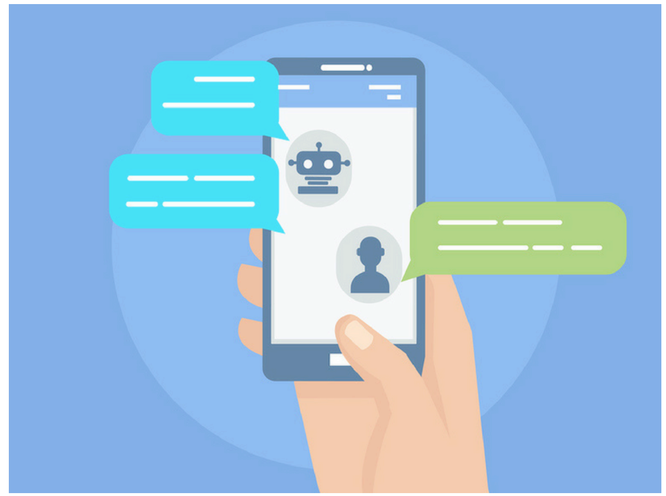
Neural Sequence Modeling

- machine translation
- language modeling
- text summarization
- question answering
- parsing
- dialogue, chatbots
- paraphrasing
- ...

I want some beef noodle soup.

我想來點牛肉麵。





Also for biology (modeling protein sequence), RL (AlphaStar)

Recurrent Continuous Translation Models

Nal Kalchbrenner **Phil Blunsom**
Department of Computer Science
University of Oxford
{nal.kalchbrenner,phil.blunsom}@cs.ox.ac.uk

Joint Language and Translation Modeling with Recurrent Neural Networks

Michael Auli, Michel Galley, Chris Quirk, Geoffrey Zweig
Microsoft Research
Redmond, WA, USA
{michael.auli,mgalley,chrisq,gzweig}@microsoft.com

Published as a conference paper at ICLR 2015

NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal

Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation

Jie Zhou Ying Cao Xuguang Wang Peng Li Wei Xu
Baidu Research - Institute of Deep Learning
Baidu Inc., Beijing, China
{zhoujie01,caoying03,wangxuguang,lipeng17,wei.xu}@baidu.com

Convolutional Sequence to Sequence Learning

Jonas Gehring [1] **Michael Auli** [1] **David Grangier** [1] **Denis Yarats** [1] **Yann N. Dauphin** [1]

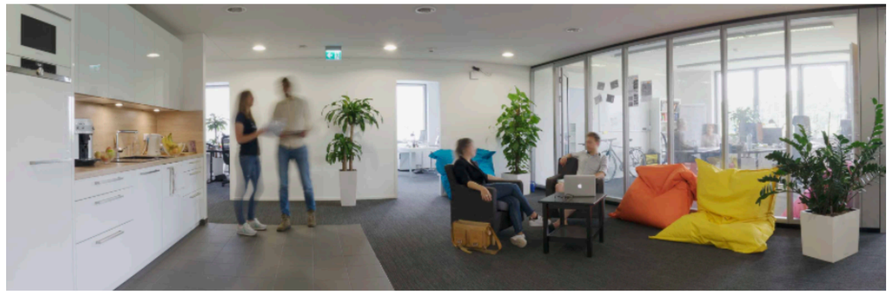
Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez*† University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin*‡ illia.polosukhin@gmail.com

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi
yonghui,schuster,zhifengc,qvl,mnorouzi@google.com
Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean

DeepL



Press Information – DeepL Translator Launch

This talk

- Depth-Adaptive Transformers
  Do you need a huge model to process simple inputs?

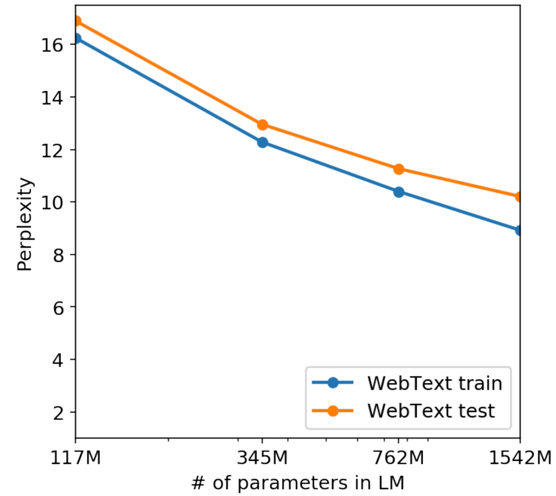
- Lightweight and Dynamic Convolutions
  More efficient architectures for sequence modeling

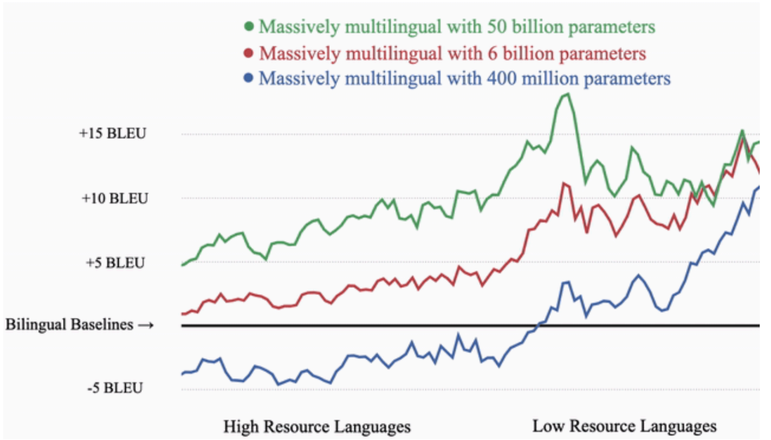
This talk

- Depth-Adaptive Transformers
  Do you need a huge model to process simple inputs?

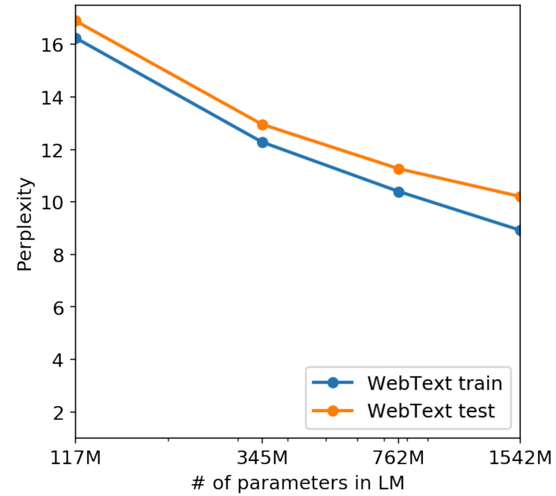
- Lightweight and Dynamic Convolutions
  More efficient architectures for sequence modeling

- Large models are everywhere:
  - GPT2 - 1.5B parameters
  - WMT'19 En-De winner: ensemble of 2B parameters
  - Massively Multilingual NMT - 50B parameters

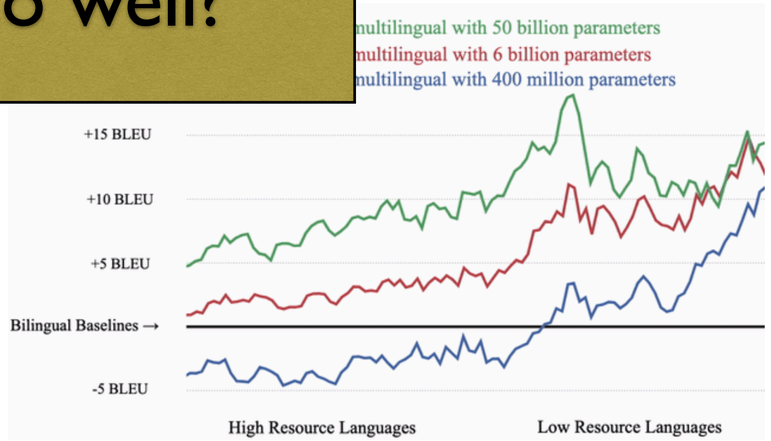




- Large models are everywhere:
  - GPT2 - 1.5B parameters
  - WMT'19 En-De winner: ensemble of 2B parameters
  - Massively Multilingual NMT - 50B parameters



Do you need a huge model to do well?



- Common ways to reduce model size
  - Distillation
  - Parameter pruning

    e.g., Reducing Transformer Depth on Demand with Structured Dropout. A Fan, E Grave, A Joulin. arXiv, 2019.

- Common ways to reduce model size
  - Distillation
  - Parameter pruning

    e.g., Reducing Transformer Depth on Demand with Structured Dropout. A Fan, E Grave, A Joulin. arXiv, 2019.

- Reduces model size for all inputs equally

- Some inputs / classification decisions easier than others, e.g., translating ”thanks” vs. a long sentence etc.

- Common ways to reduce model size
  - Distillation
  - Parameter pruning

    e.g., Reducing Transformer Depth on Demand with Structured Dropout. A Fan, E Grave, A Joulin. arXiv, 2019.

- Reduces model size for all inputs equally

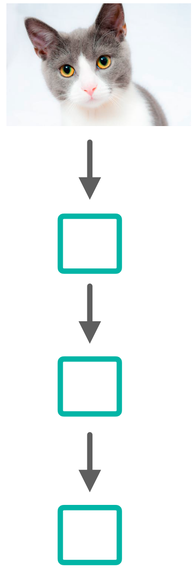
- Some inputs / classification decisions easier than others, e.g., translating ″thanks″ vs. a long sentence etc.

This work: 'autosize' model to tackle the current problem!

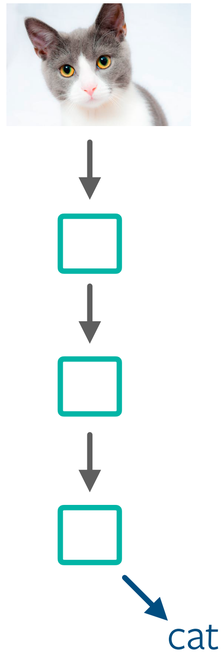
Key Ideas

- Make number of network layers input dependent.
- Attach output classifier at multiple locations of network.
- Predict how many layers based on the input.

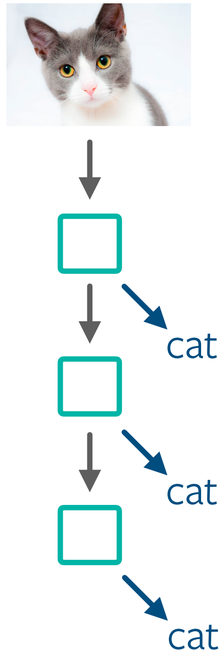
Computer Vision
Image classification



Computer Vision
Image classification



Computer Vision
Image classification

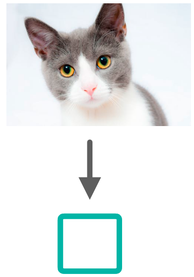


Huang et al., ’18

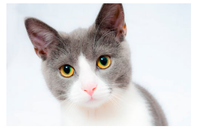
multiple output classifiers

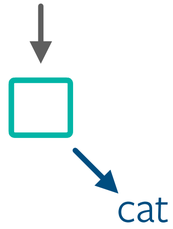
Computer Vision
Image classification

Computer Vision
Image classification

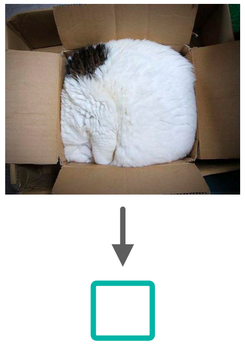


Computer Vision
Image classification

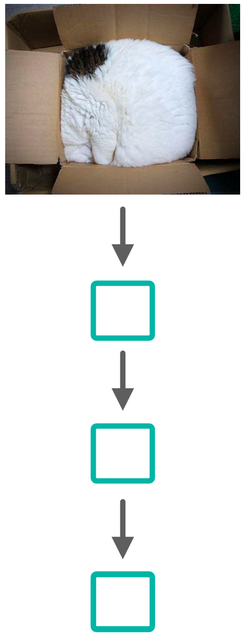




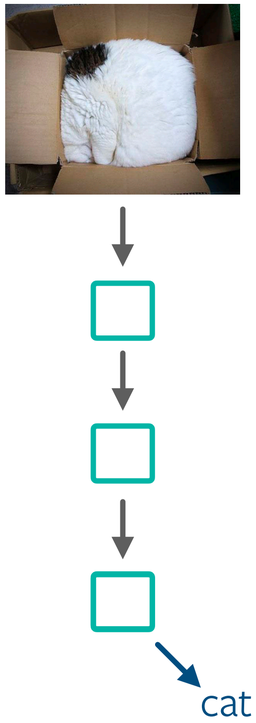
Computer Vision
Image classification



Computer Vision
Image classification

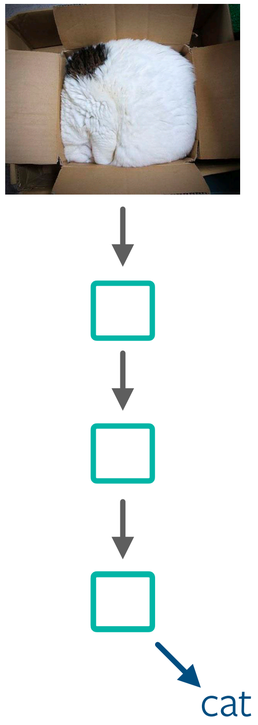


Computer Vision
Image classification



Huang et al., ’18

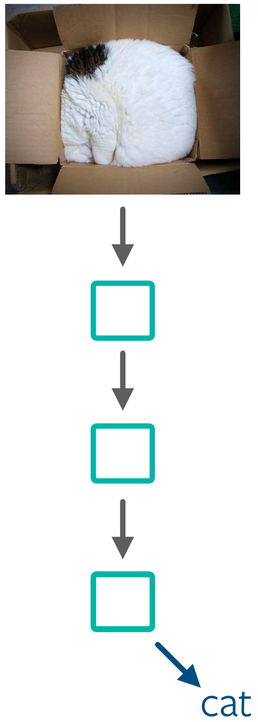
Computer Vision
Image classification



Huang et al., ’18

Stop when output classifier reaches threshold.

Computer Vision
Image classification



Huang et al., ’18

Stop when output classifier reaches threshold.

Multiple output classifiers can be an issue:

ImageNet 1K	1,000 classes
NMT	32,000 classes
Billion Word LM	800,000 classes

Compute those at every layer!

NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!

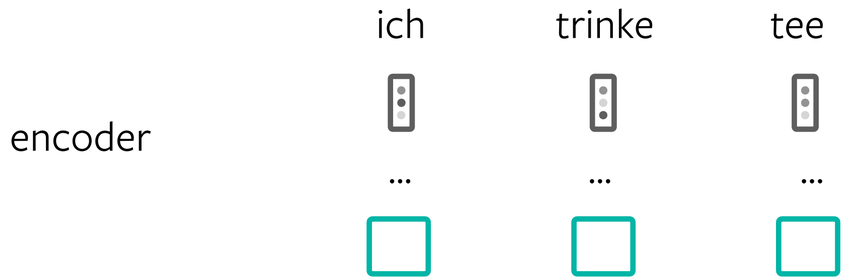
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!

ich trinke tee

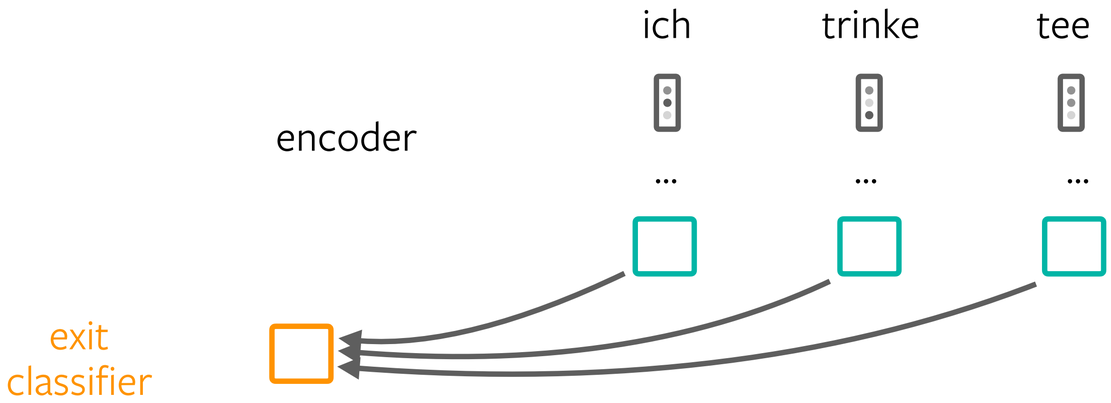
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



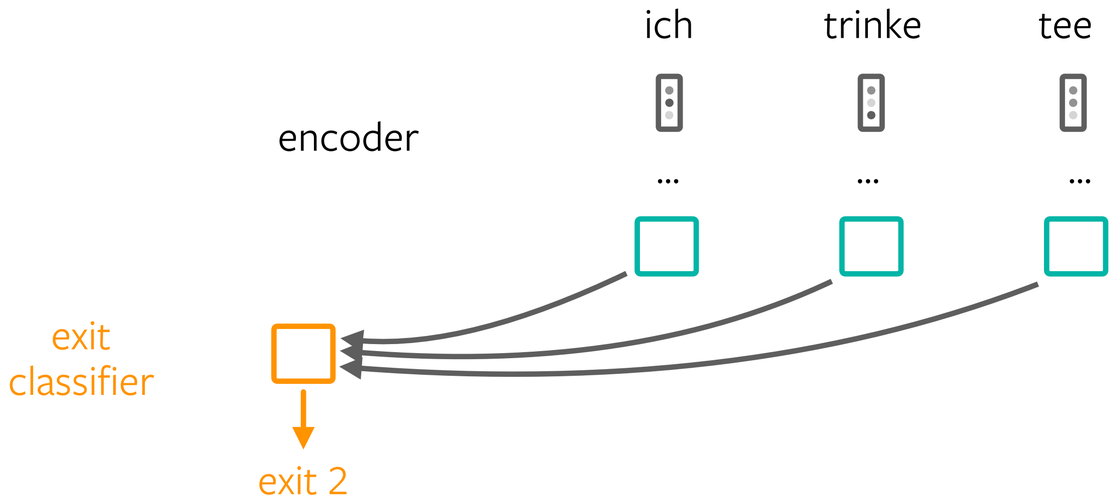
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



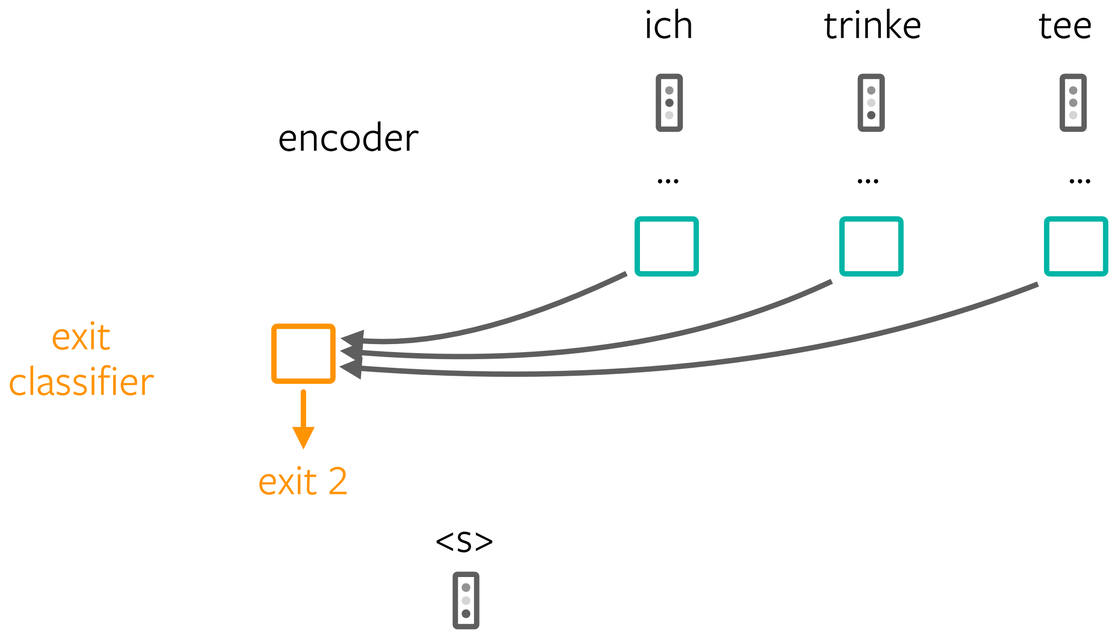
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



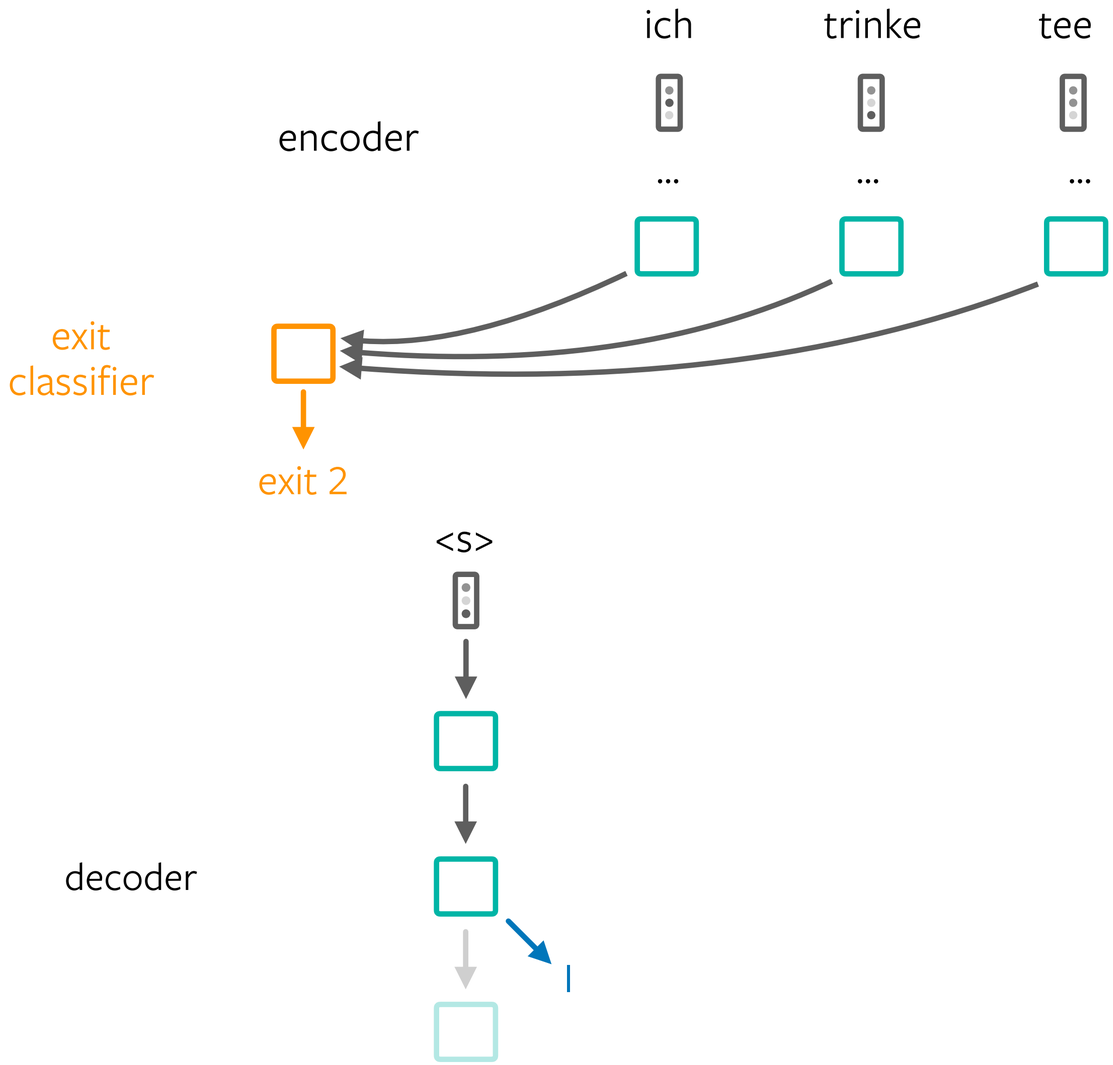
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



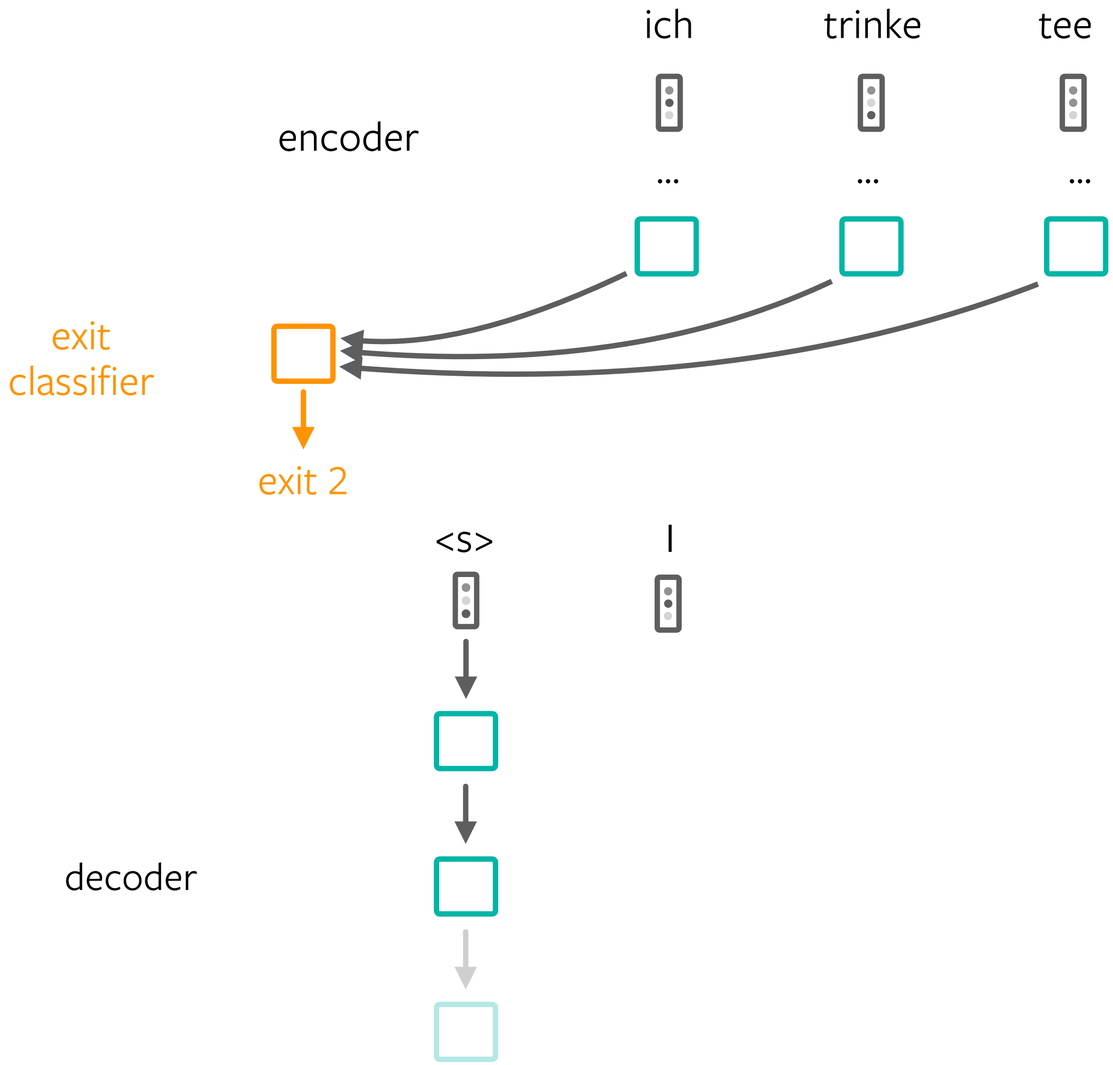
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



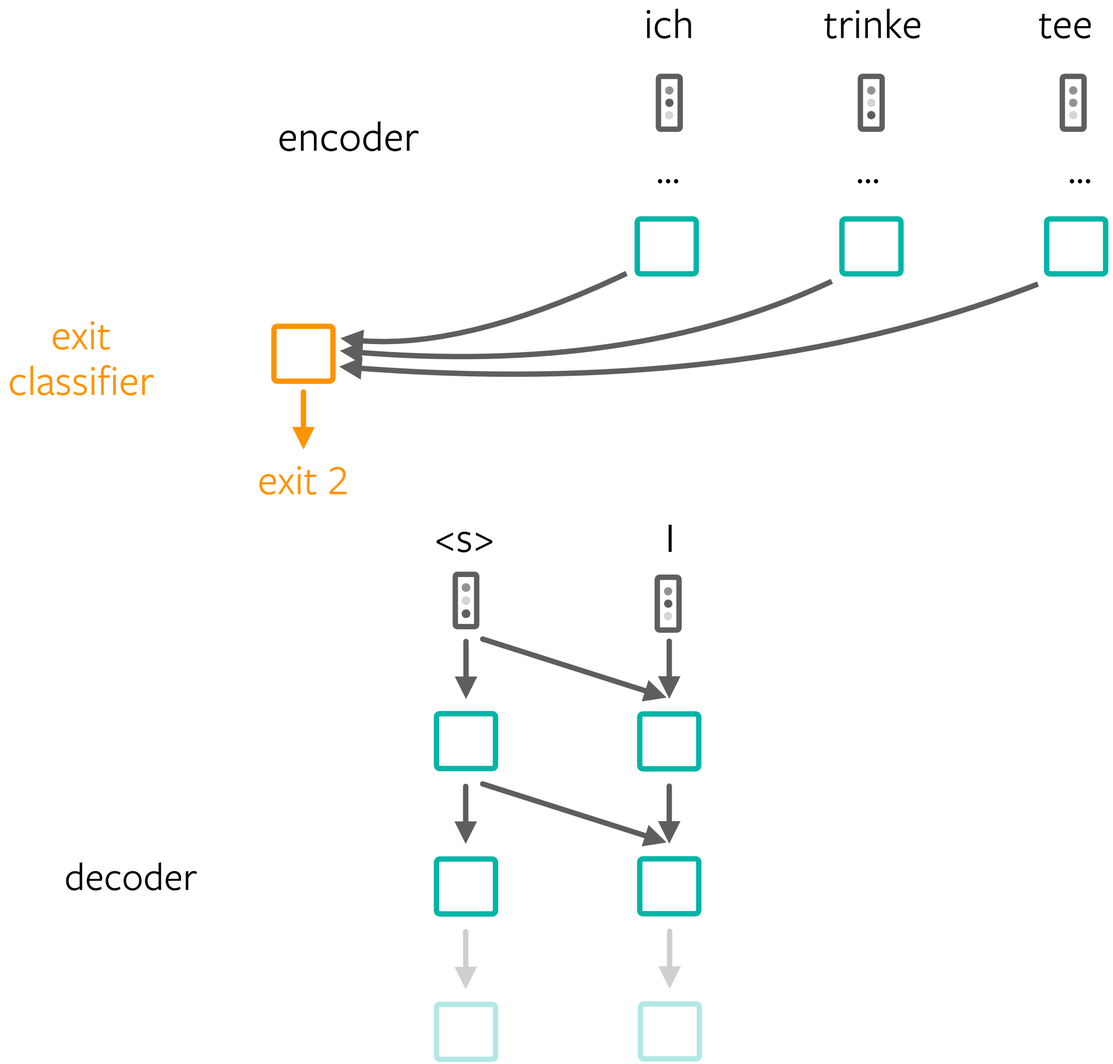
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



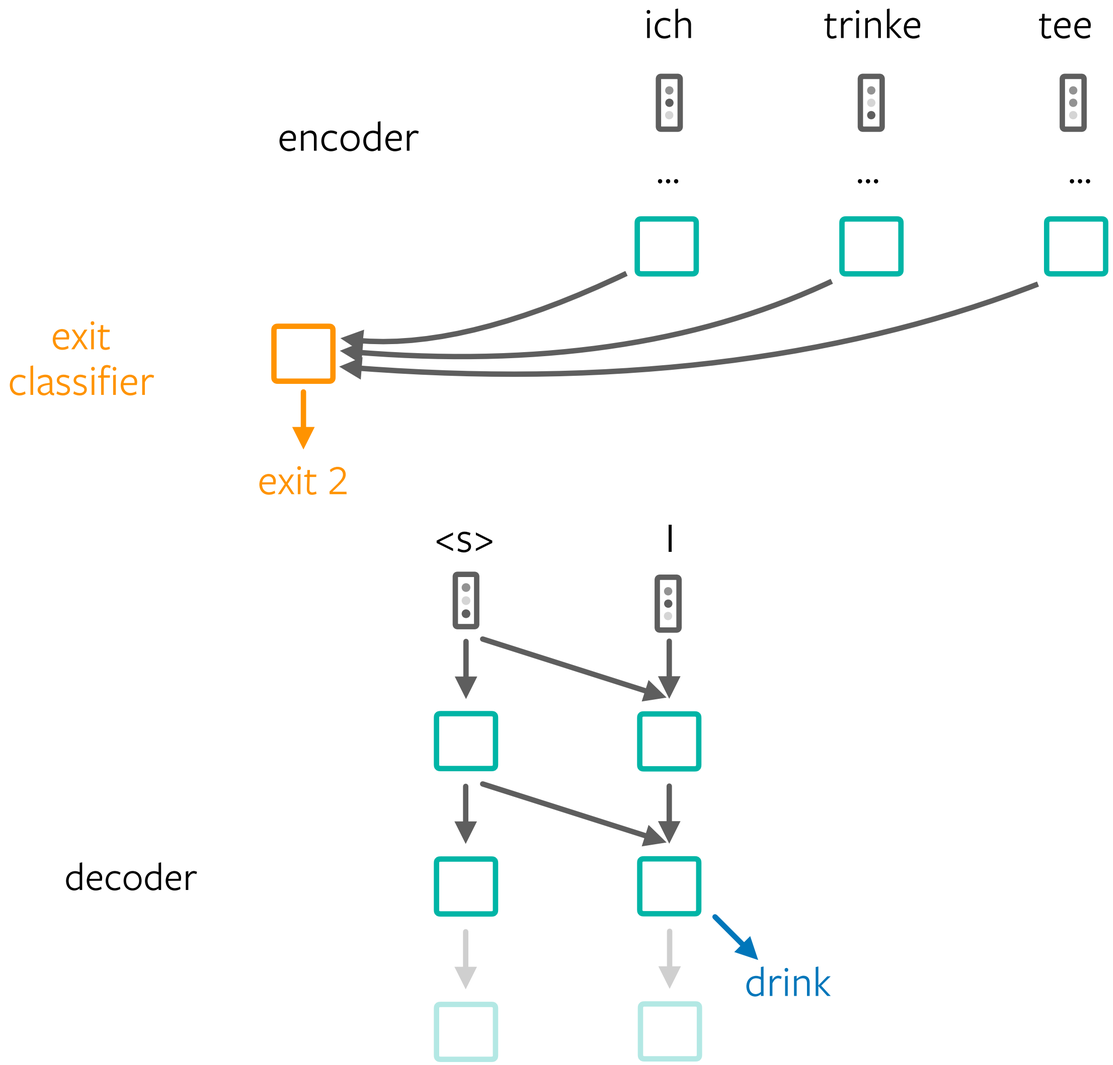
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



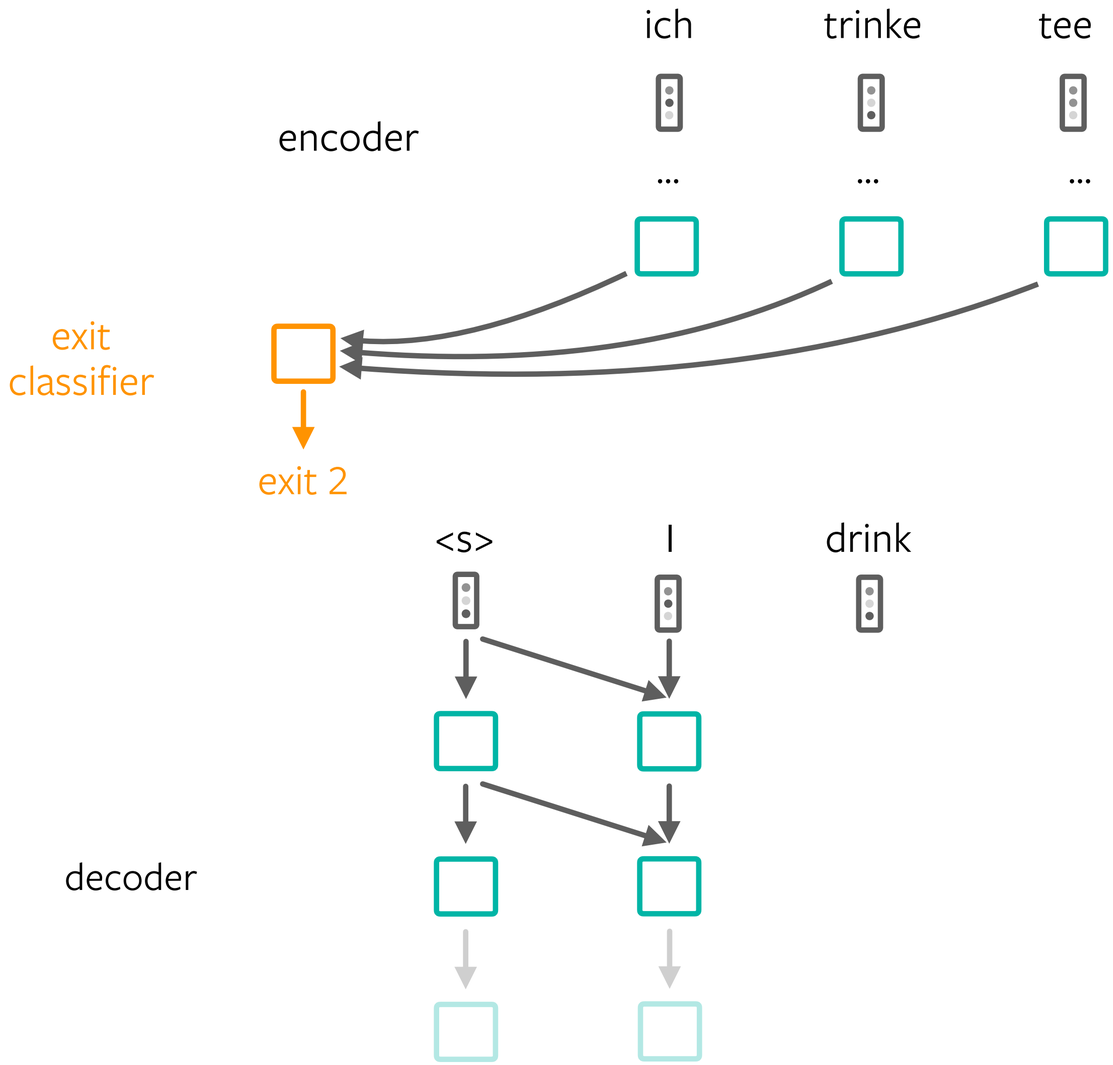
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



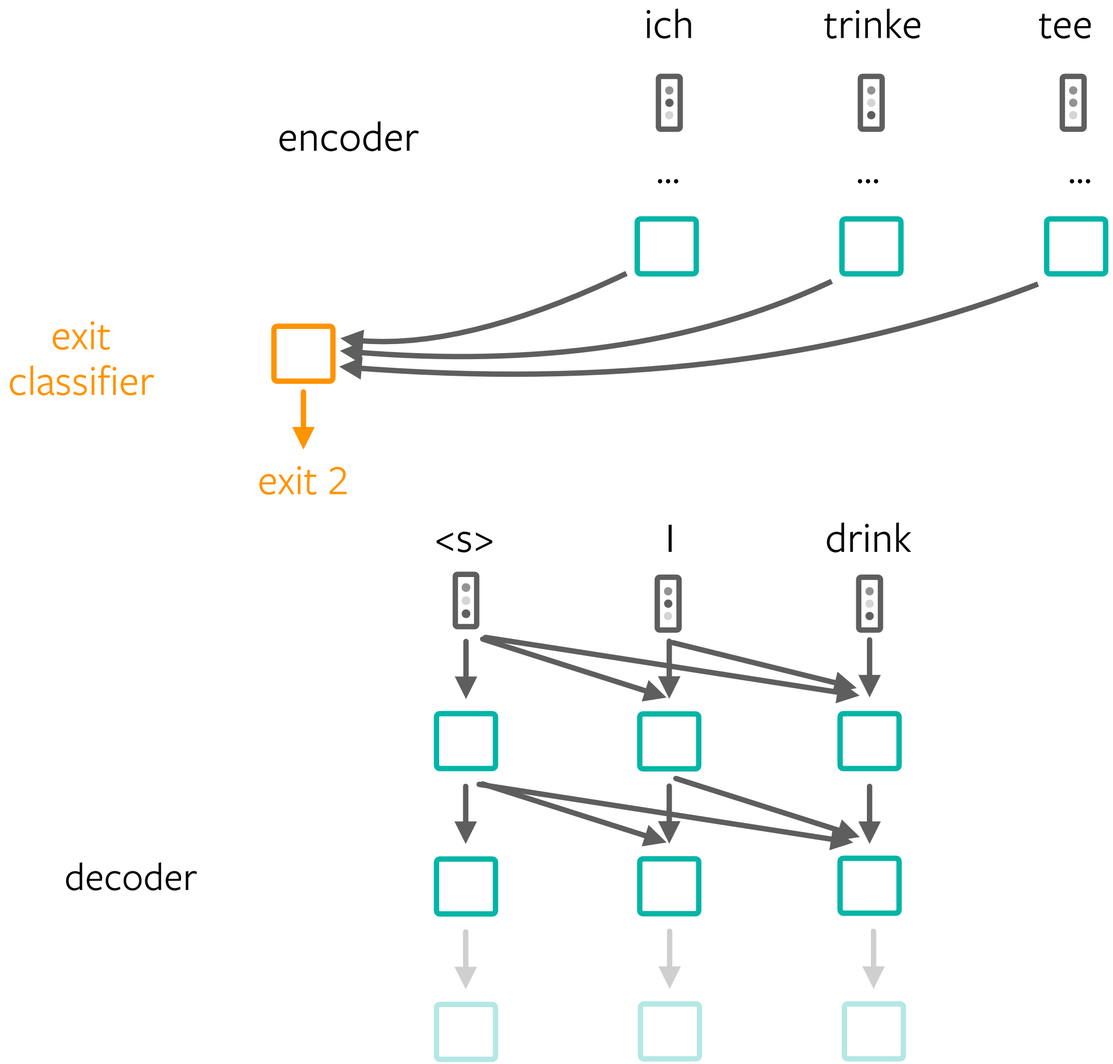
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



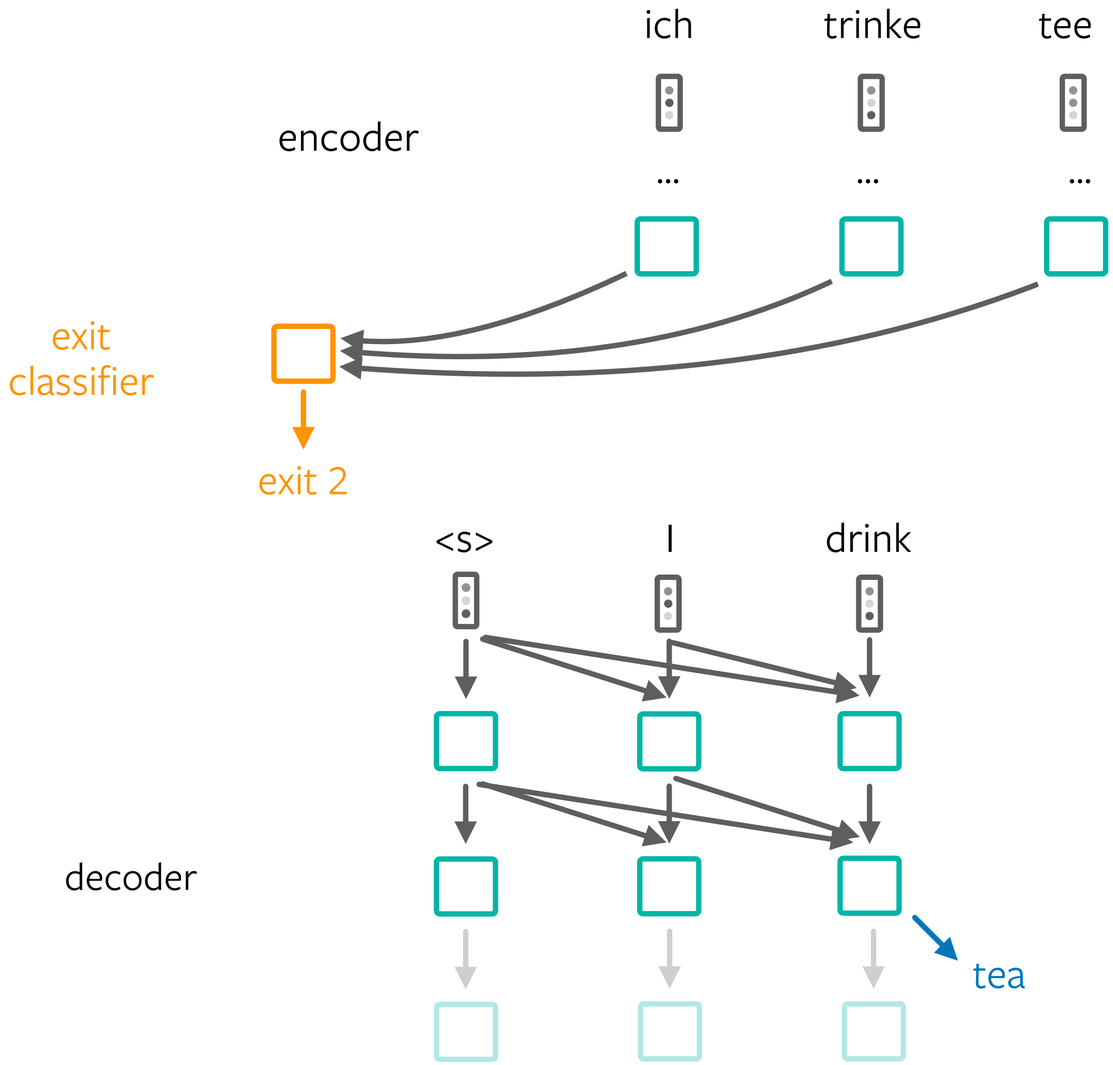
NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



NLP: Adaptive-Depth on Sequence Level

- Focus on decoder network
- Exit classifier predicts depth for all tokens
- Attach output classifier after every block
- Use output classifier once per token!



NLP: Adaptive-Depth on Token Level

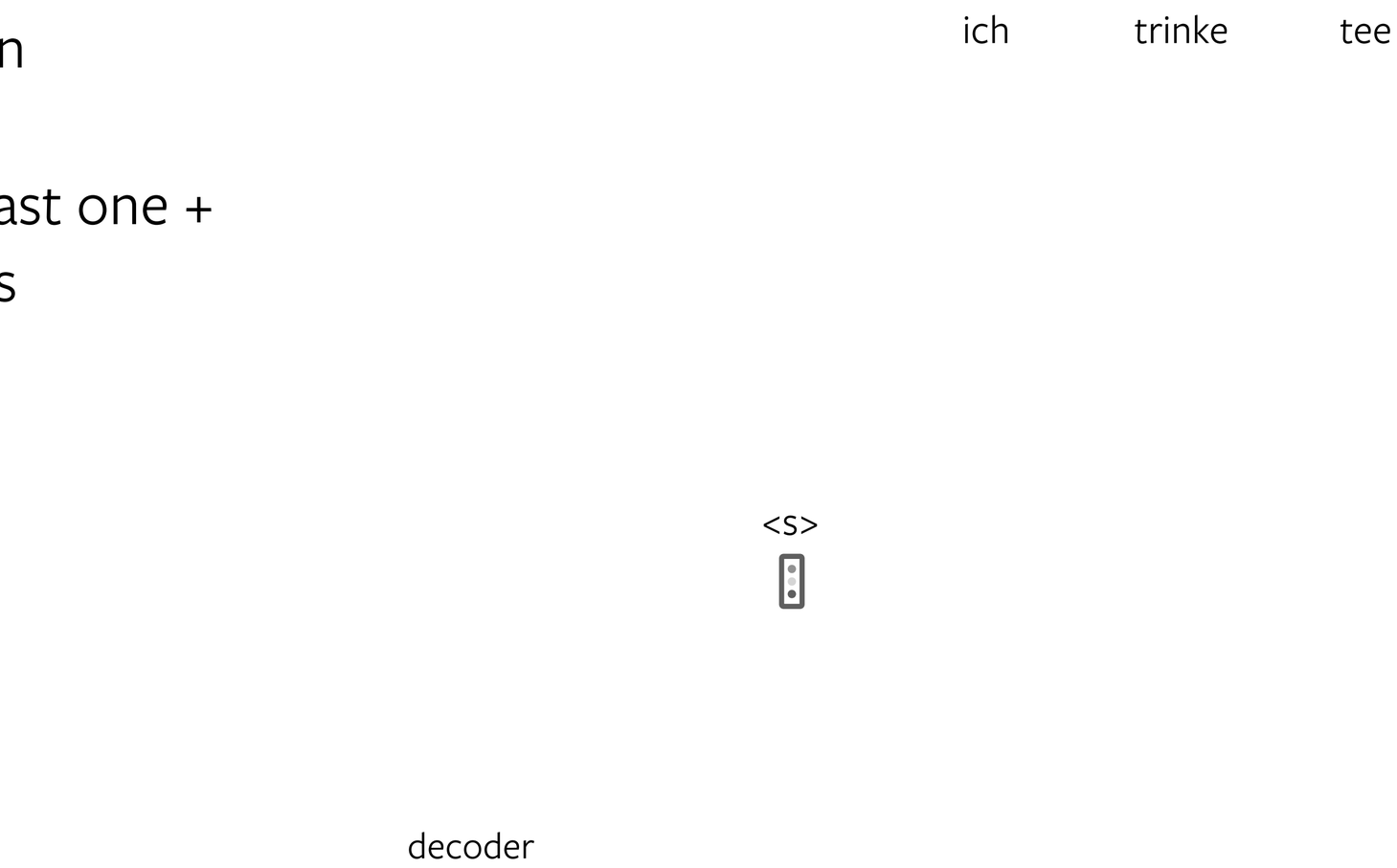
- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

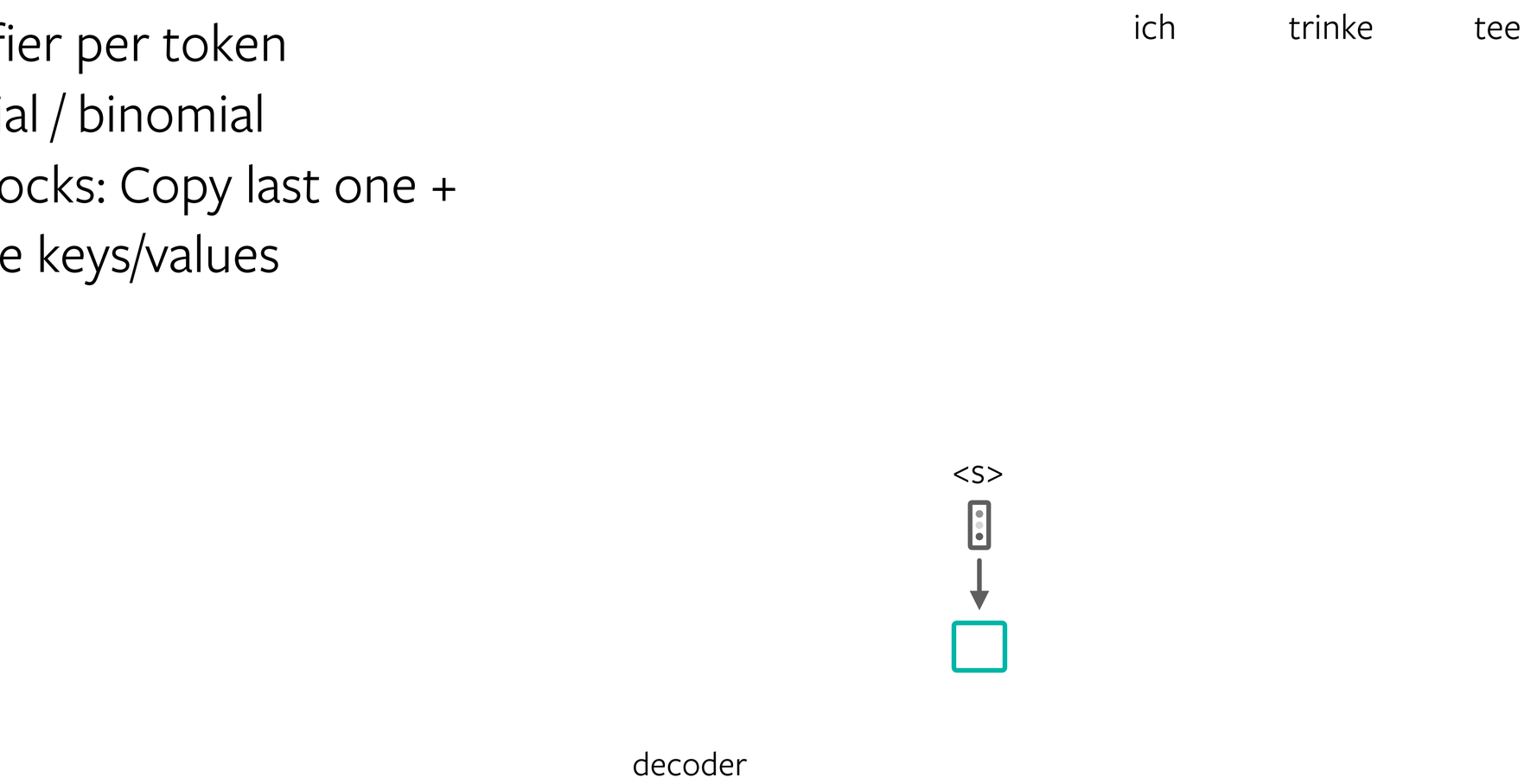
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



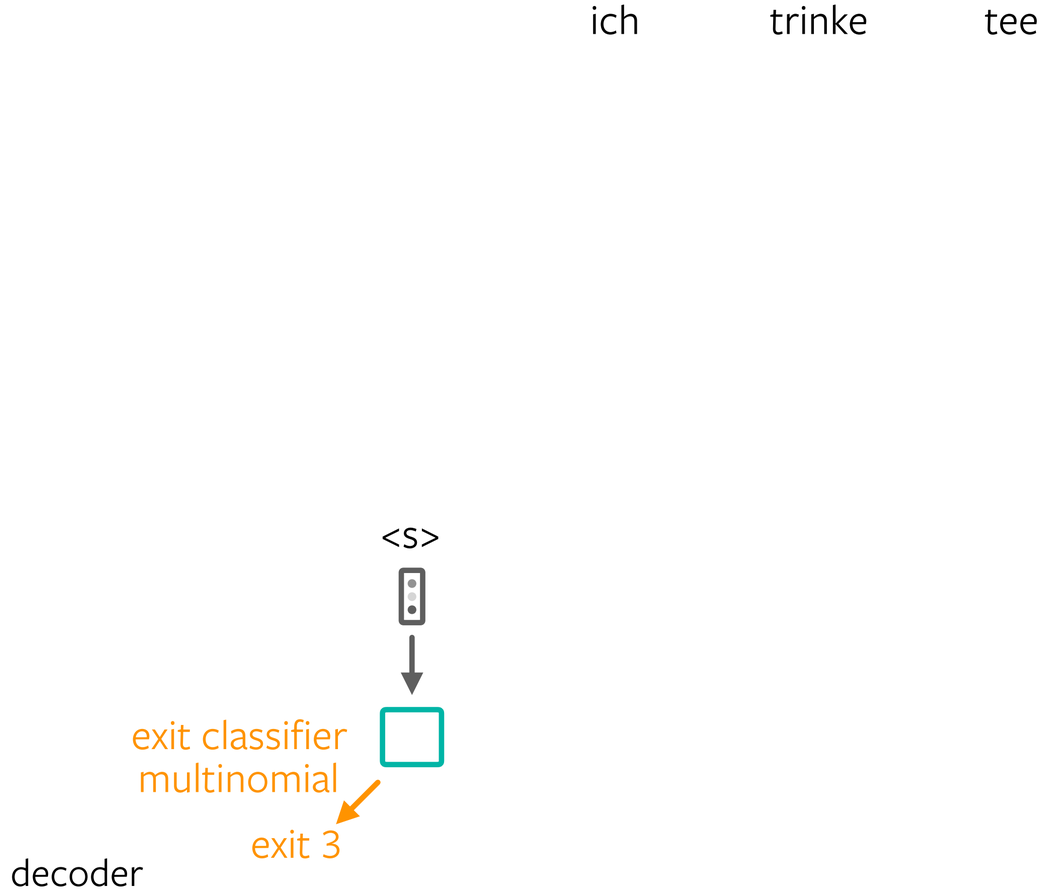
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



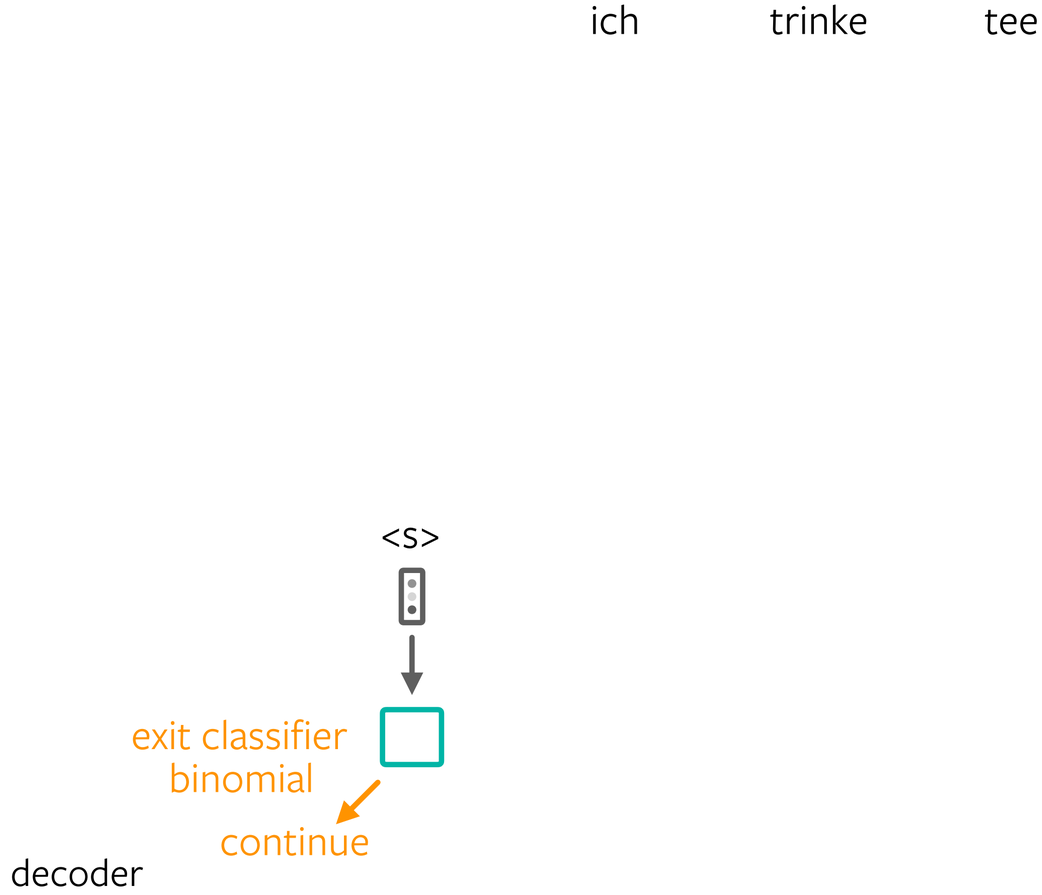
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



NLP: Adaptive-Depth on Token Level

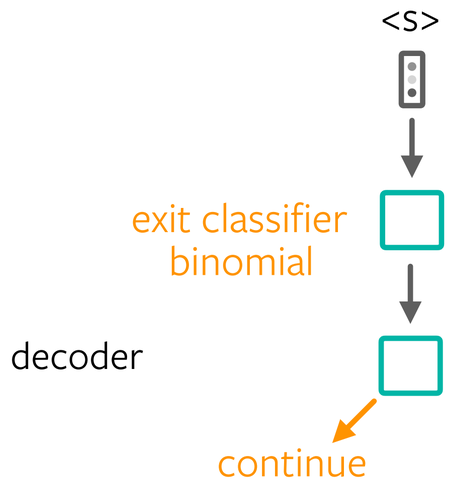
- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

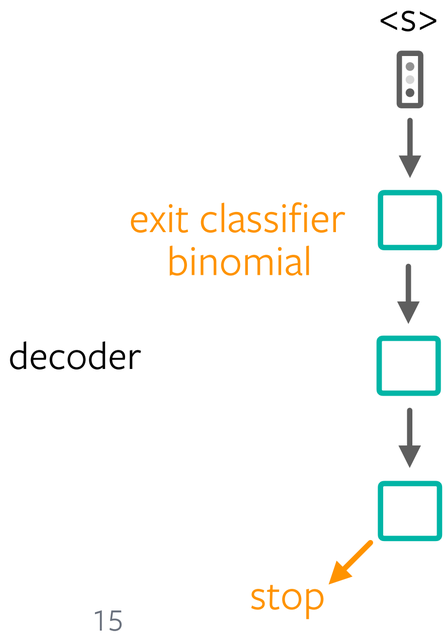
ich trinke tee



NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

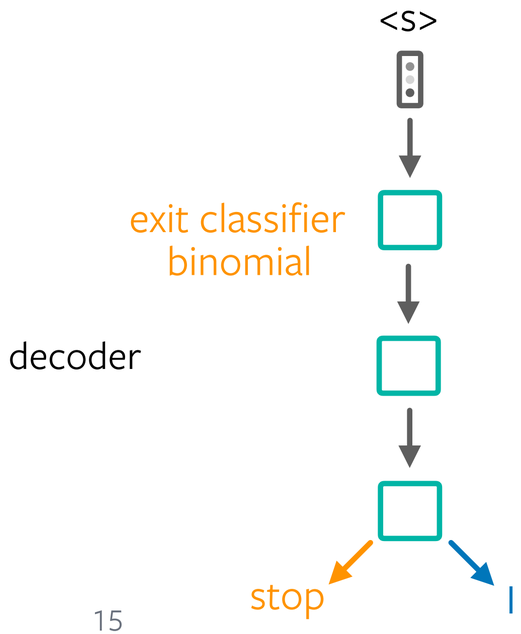
ich trinke tee



NLP: Adaptive-Depth on Token Level

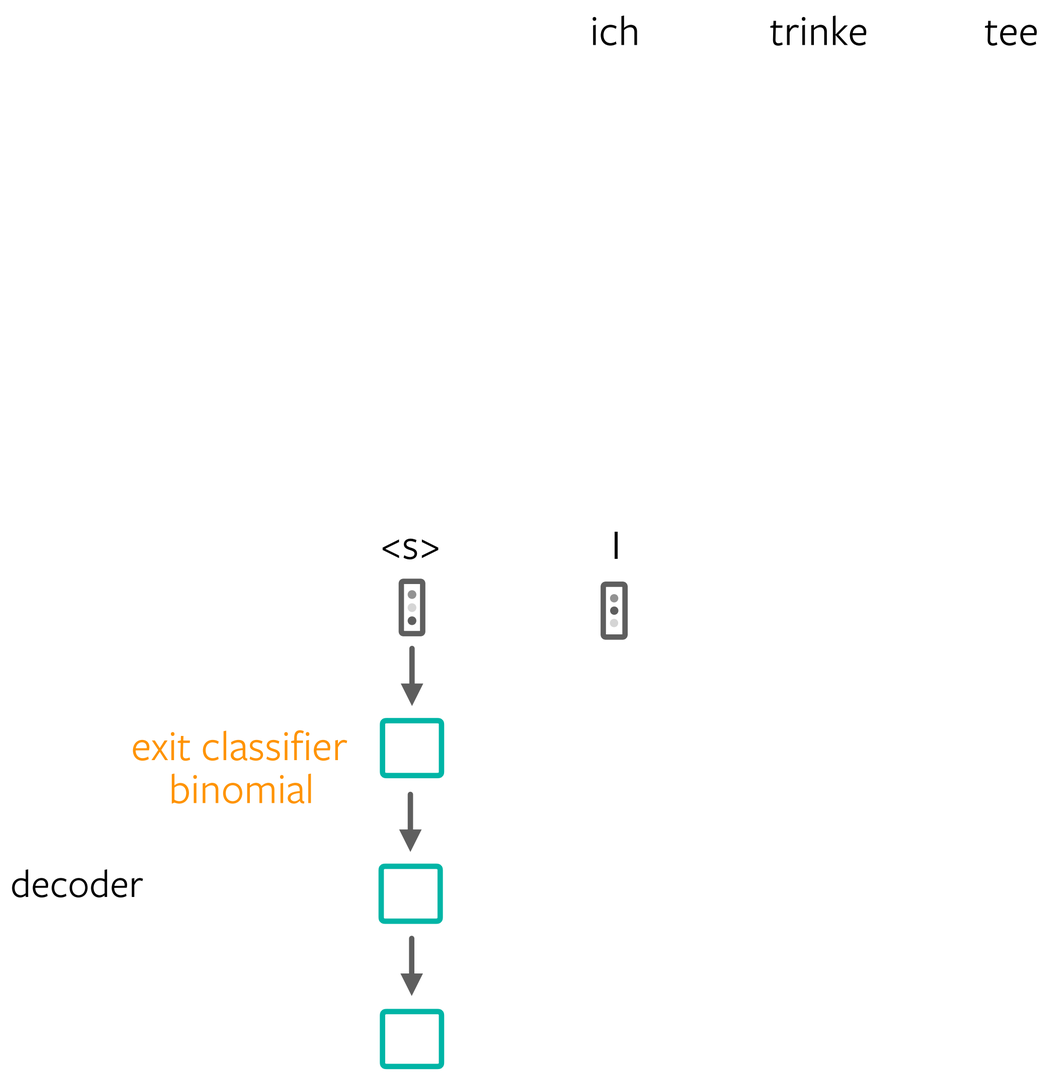
- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

ich trinke tee



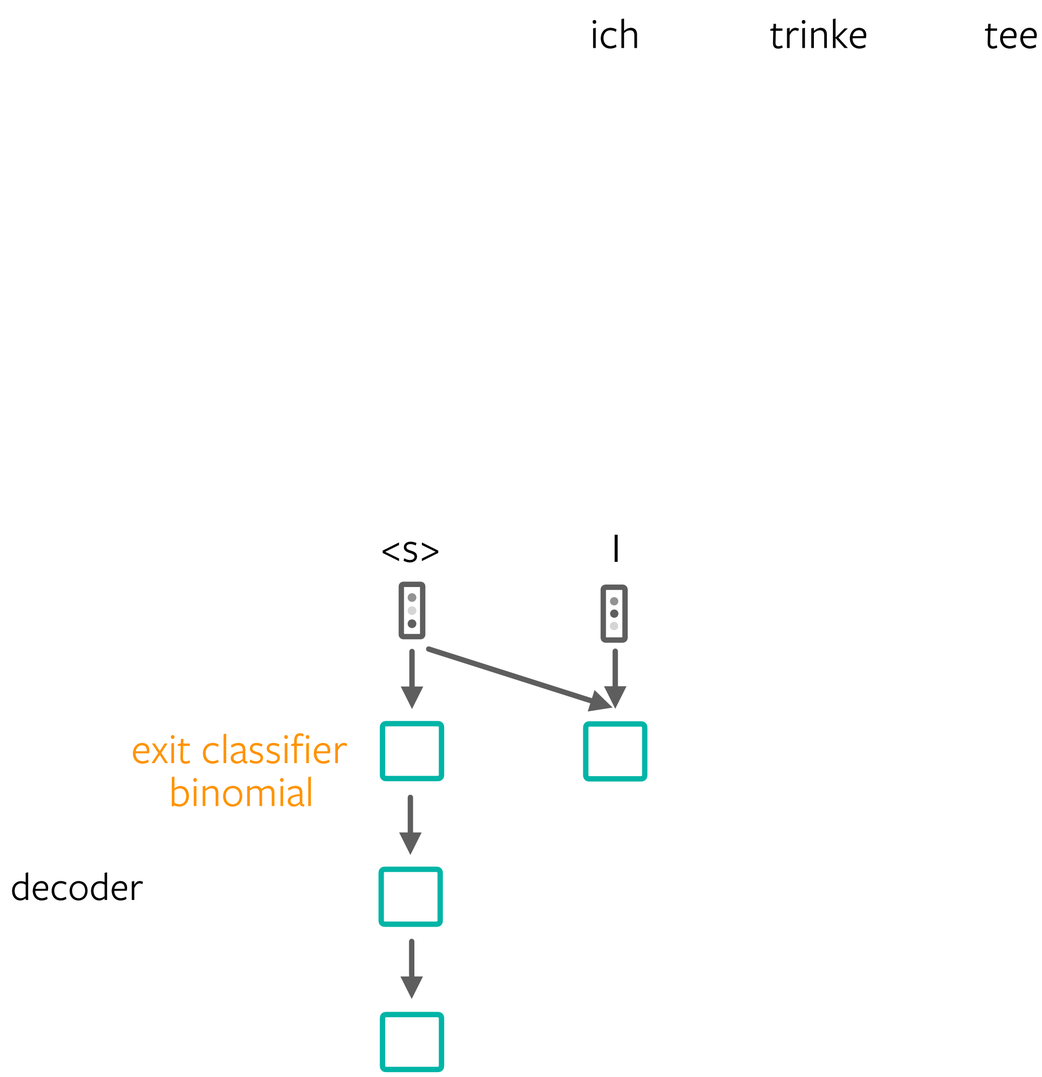
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



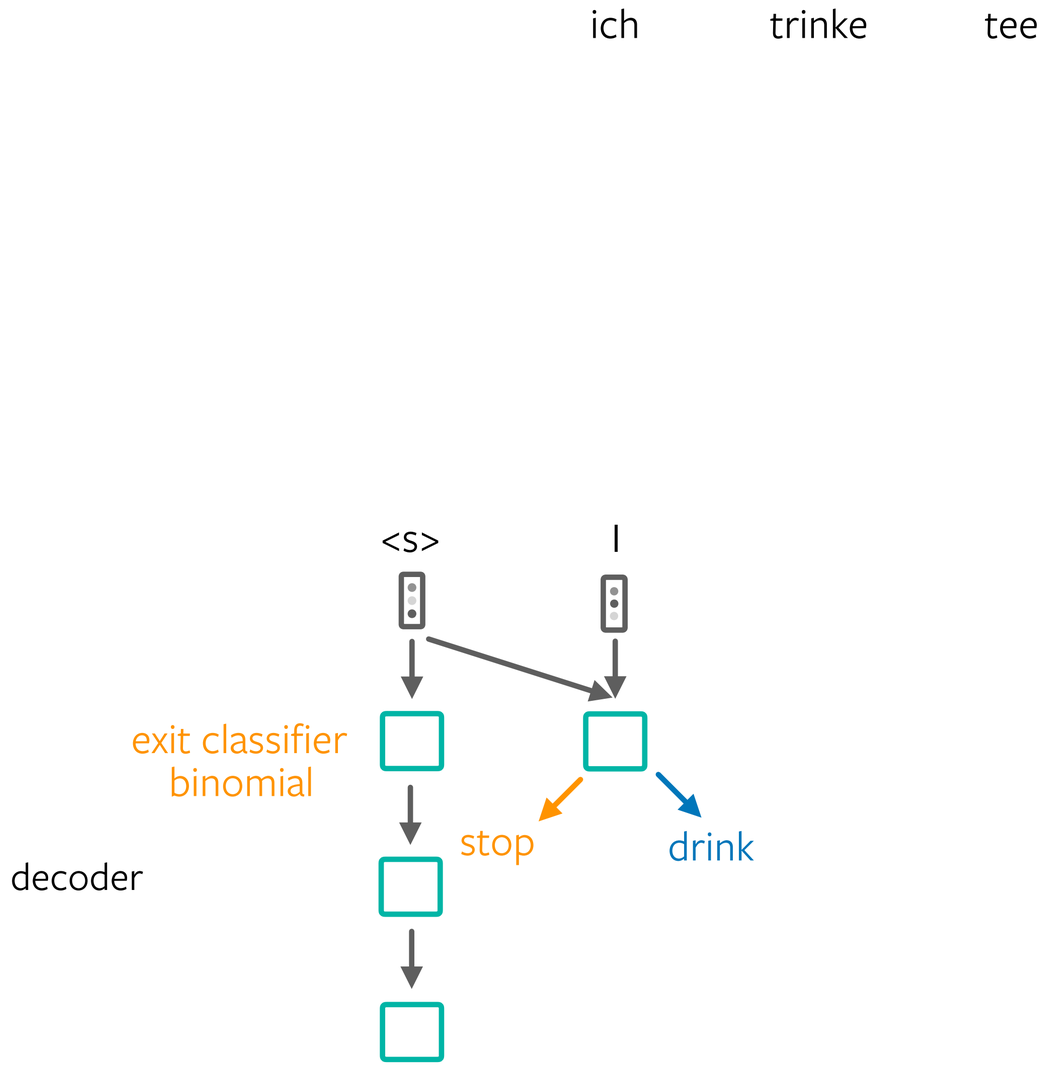
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



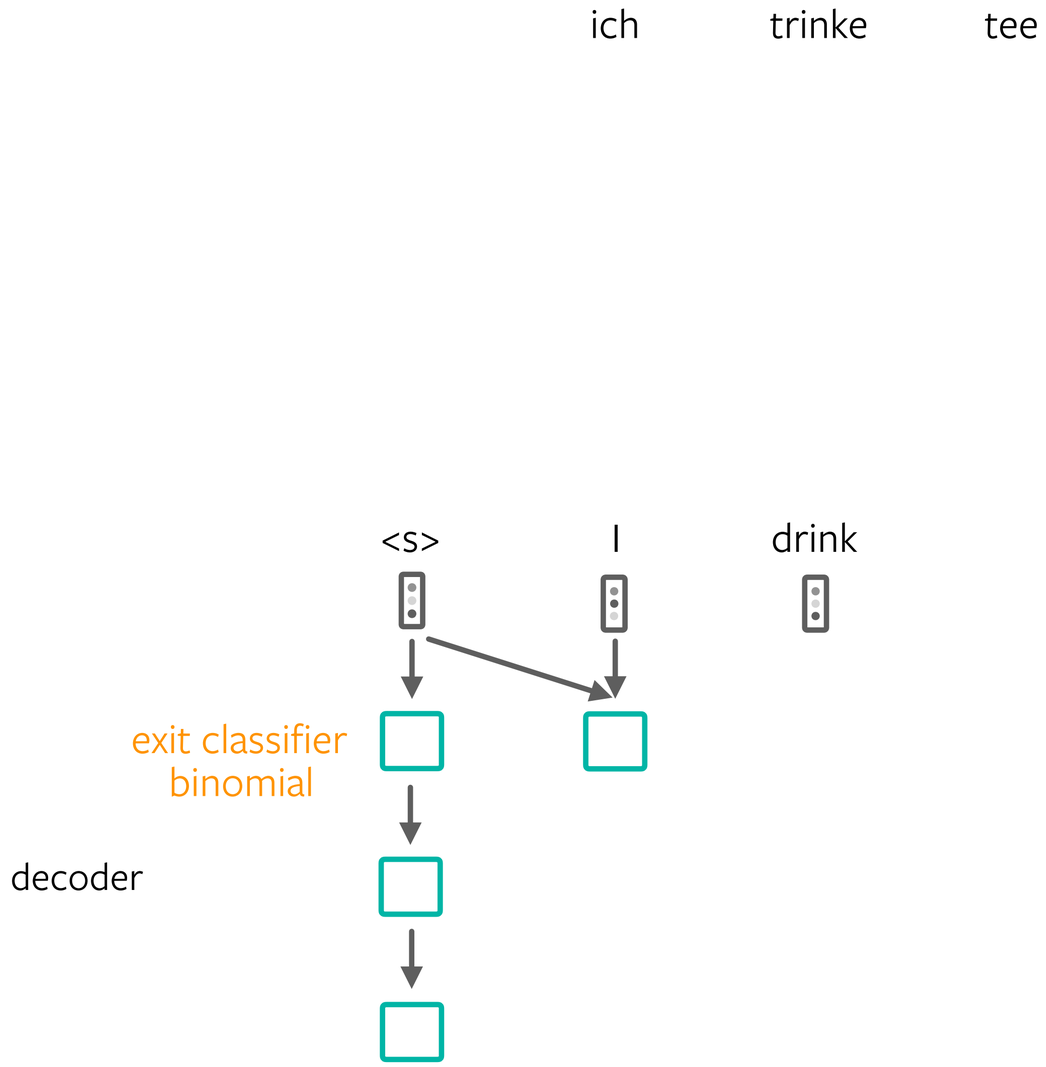
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



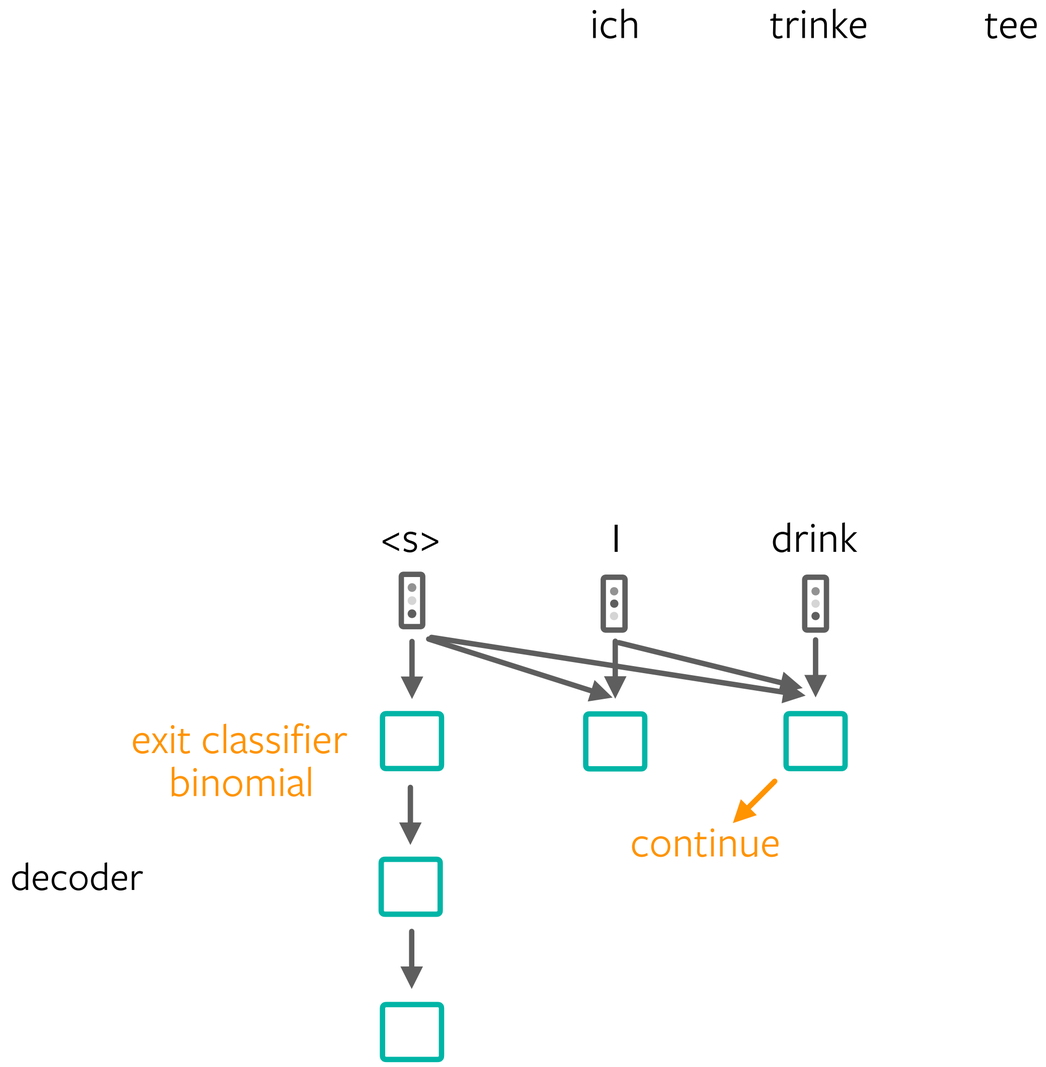
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



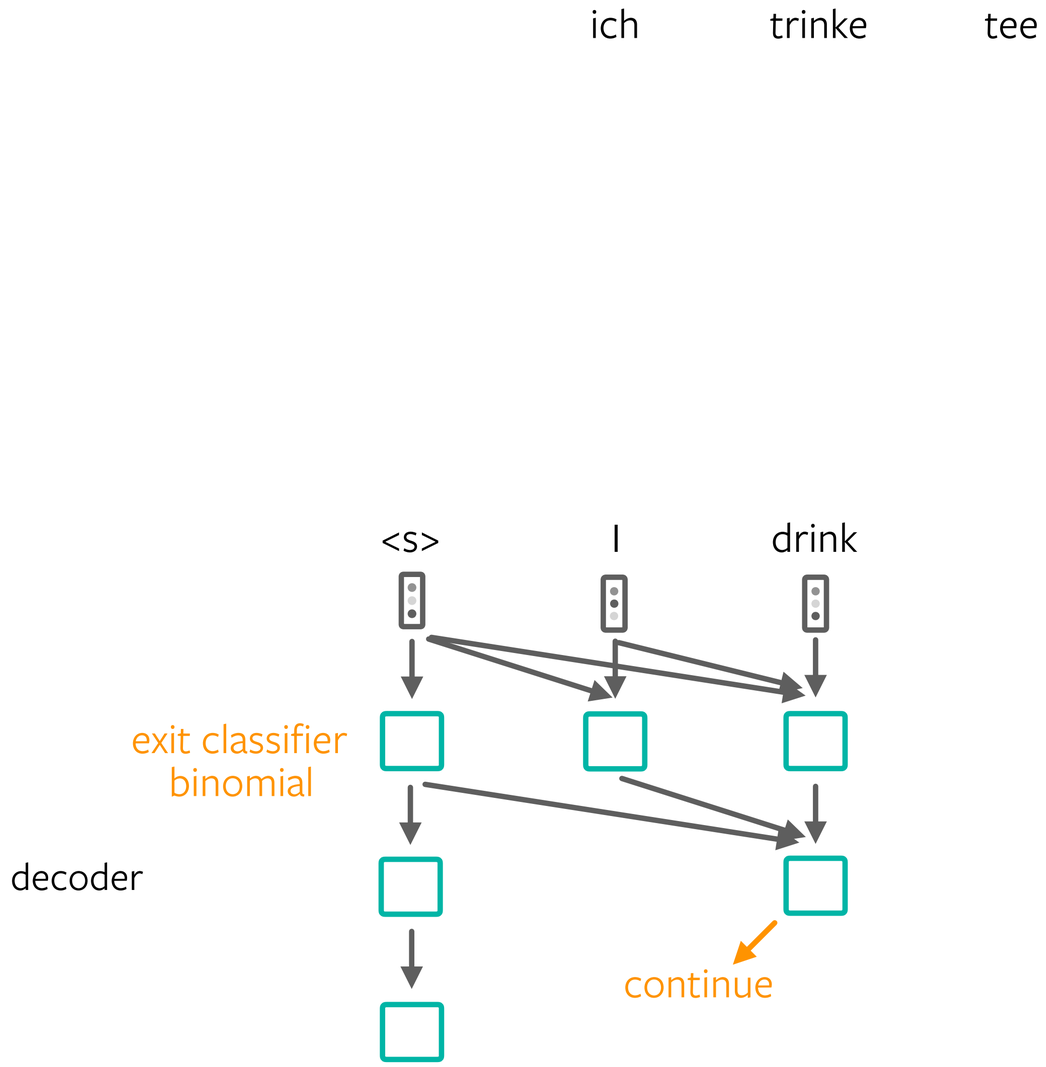
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



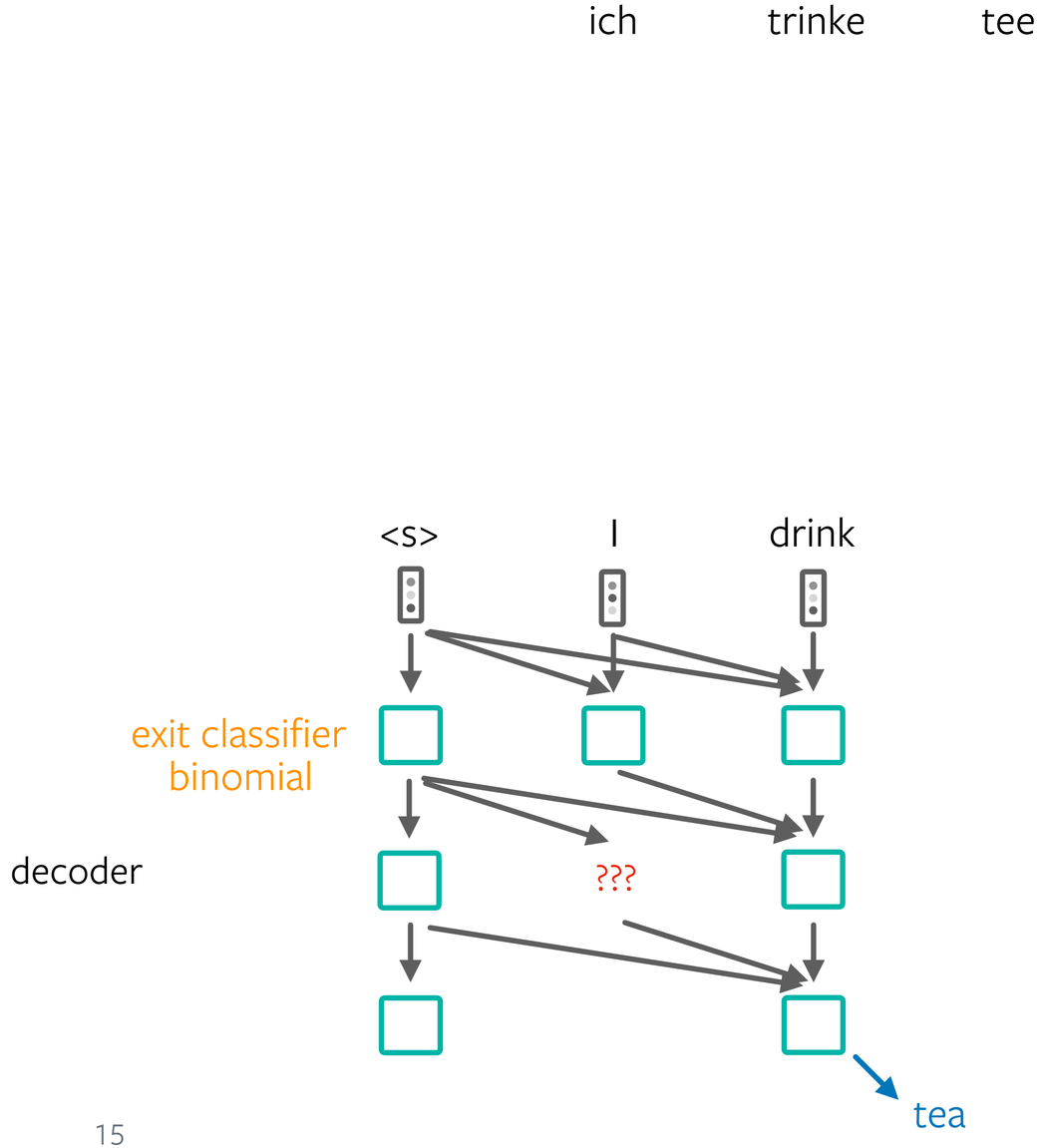
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



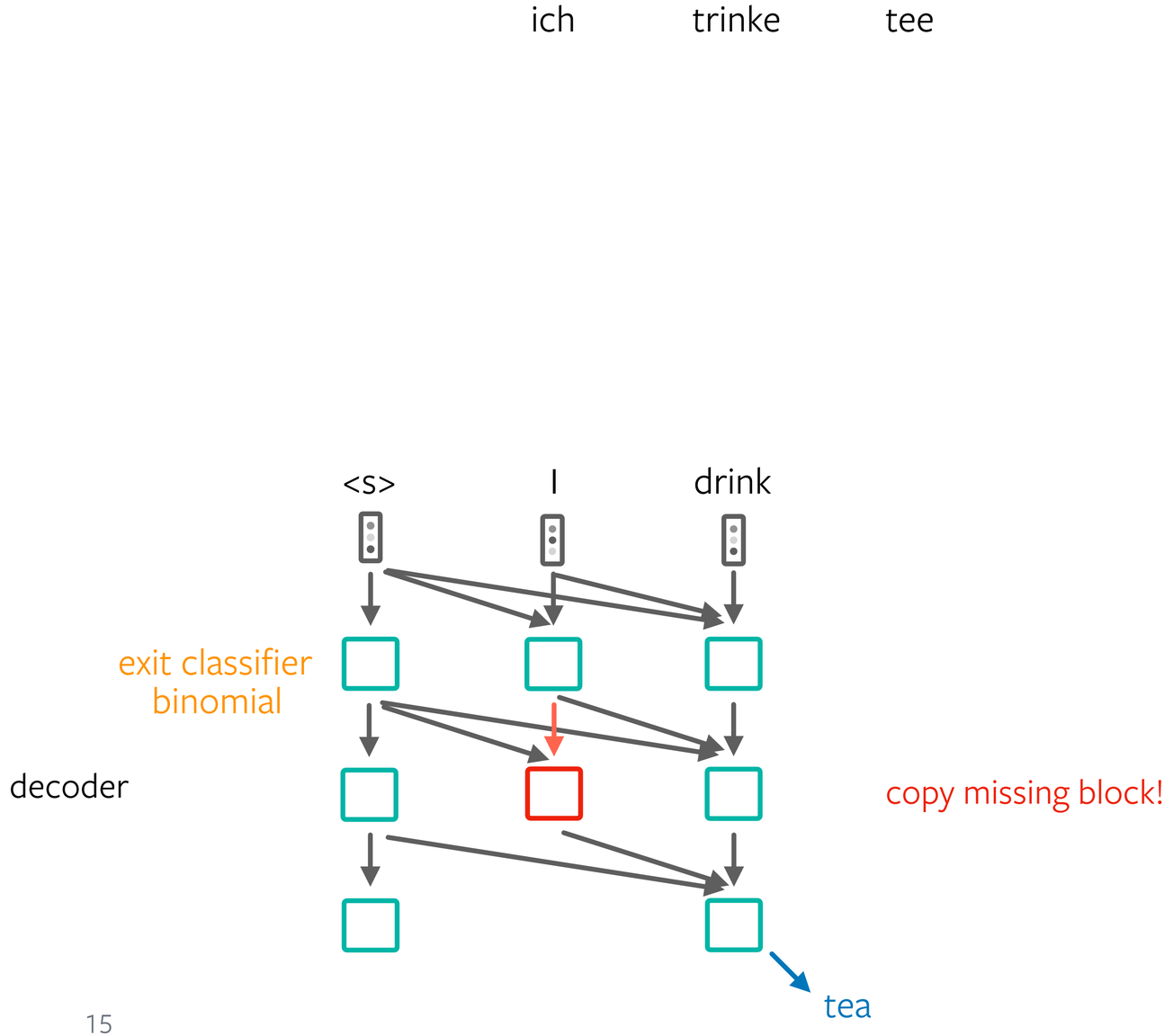
NLP: Adaptive-Depth on Token Level

- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



NLP: Adaptive-Depth on Token Level

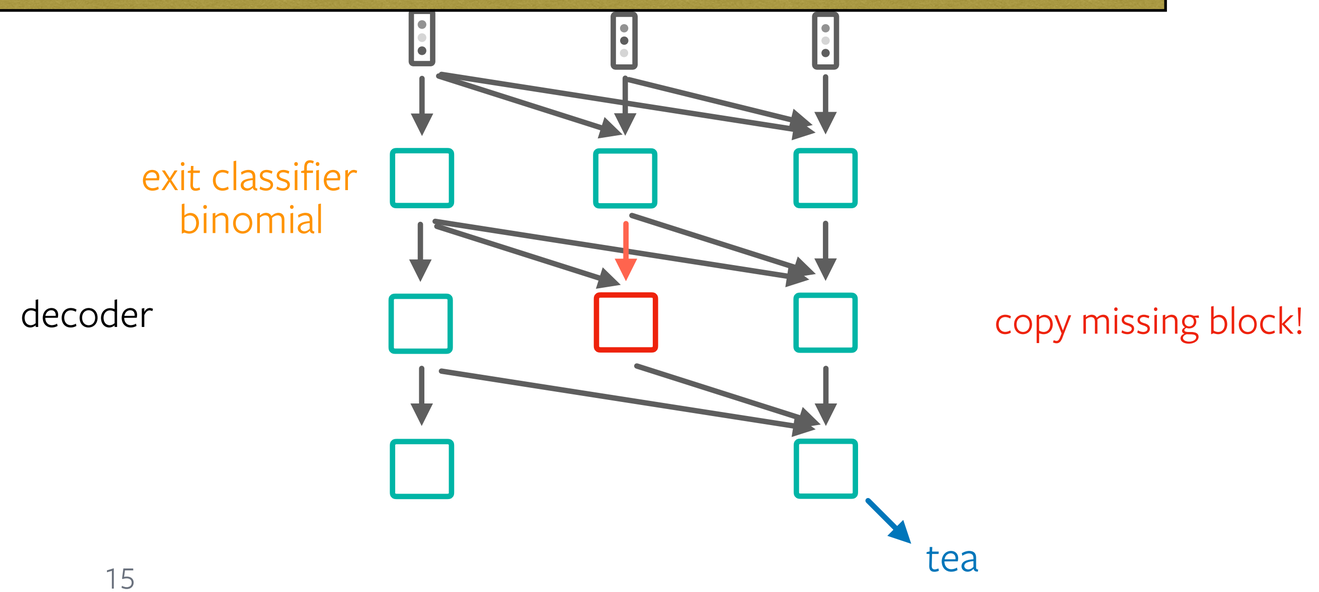
- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values



NLP: Adaptive-Depth on Token Level

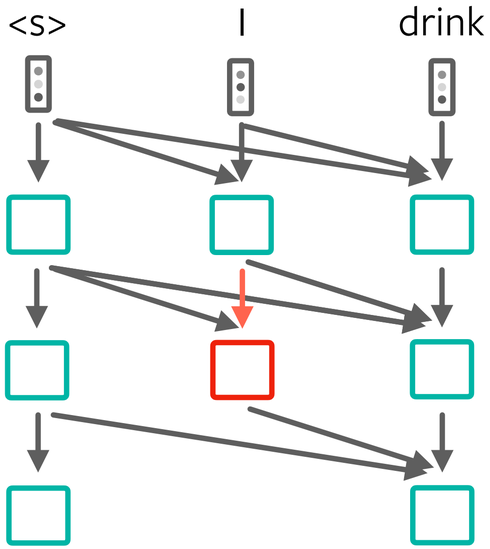
- Exit classifier per token
- Multinomial / binomial
- Missing blocks: Copy last one + recompute keys/values

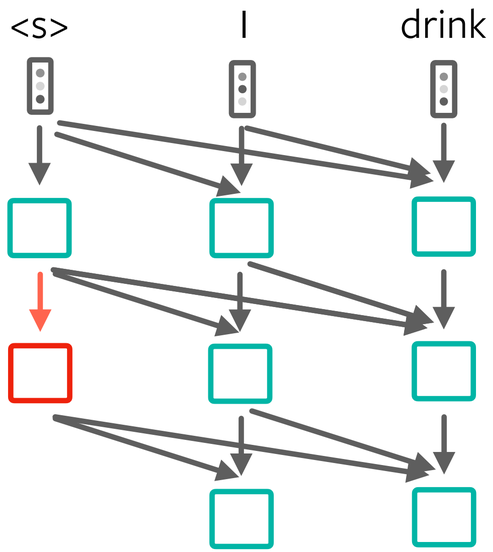
Copied states never seen in training.
Train/test mismatch!

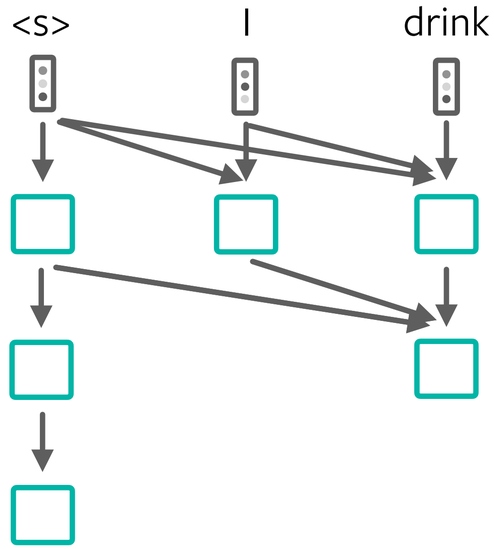


Mixed State Training: Mitigate train/test mismatch

- Sample M paths, copy if needed, and train on this data

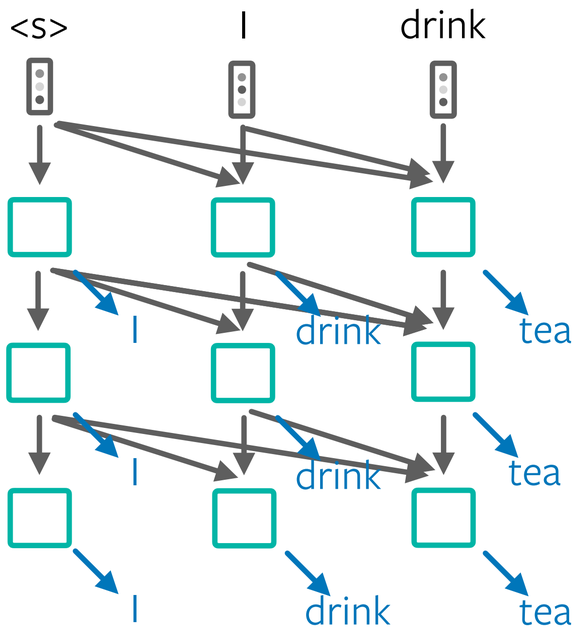






Aligned State Training

- Does not address train/test mismatch
- Simply trains with all states (‘aligned’) as usual.



Mixed State Training: Mitigate train/test mismatch

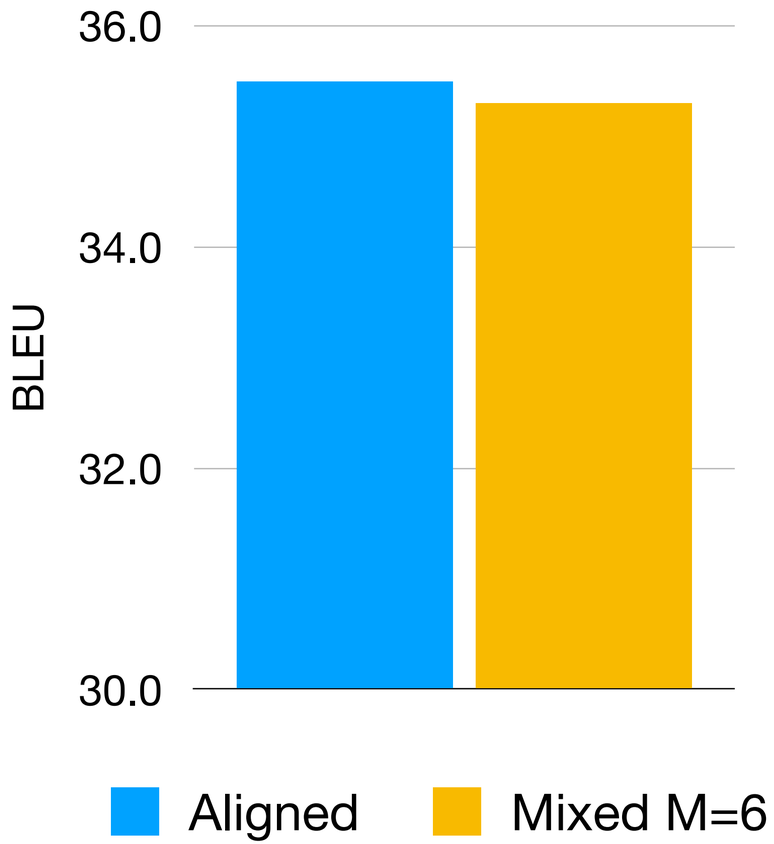
Experiment #1
Generate with **random** exit

Experiment #2
Generate with **fixed** exit

IWSLT'14 German-English

Mixed State Training: Mitigate train/test mismatch

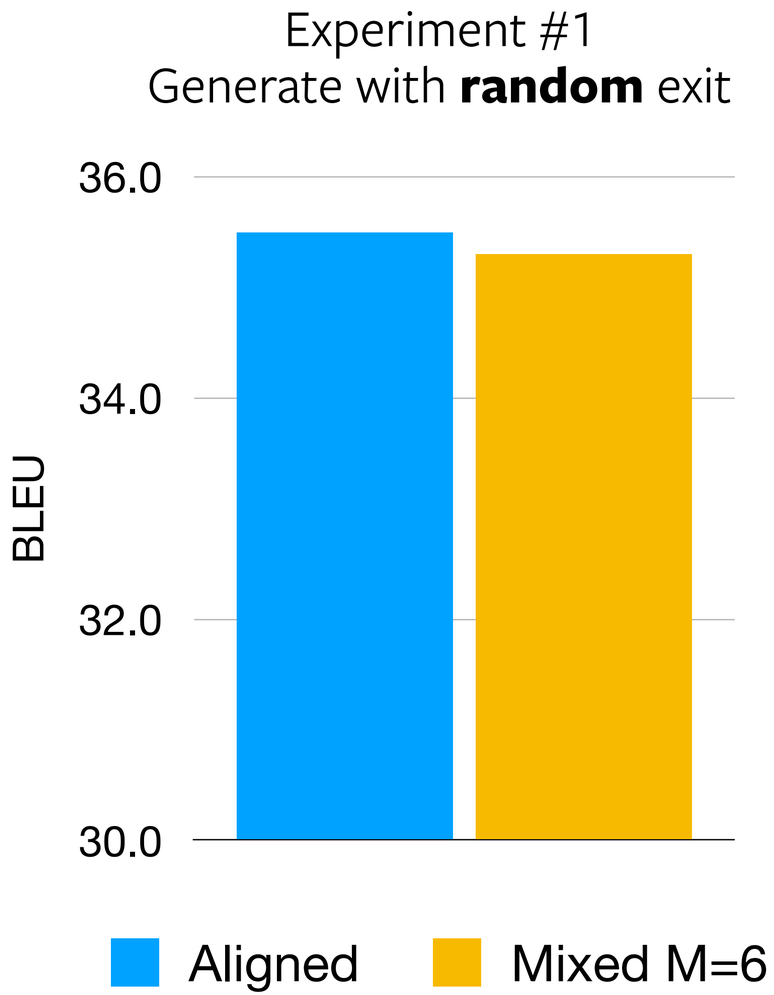
Experiment #1
Generate with **random** exit

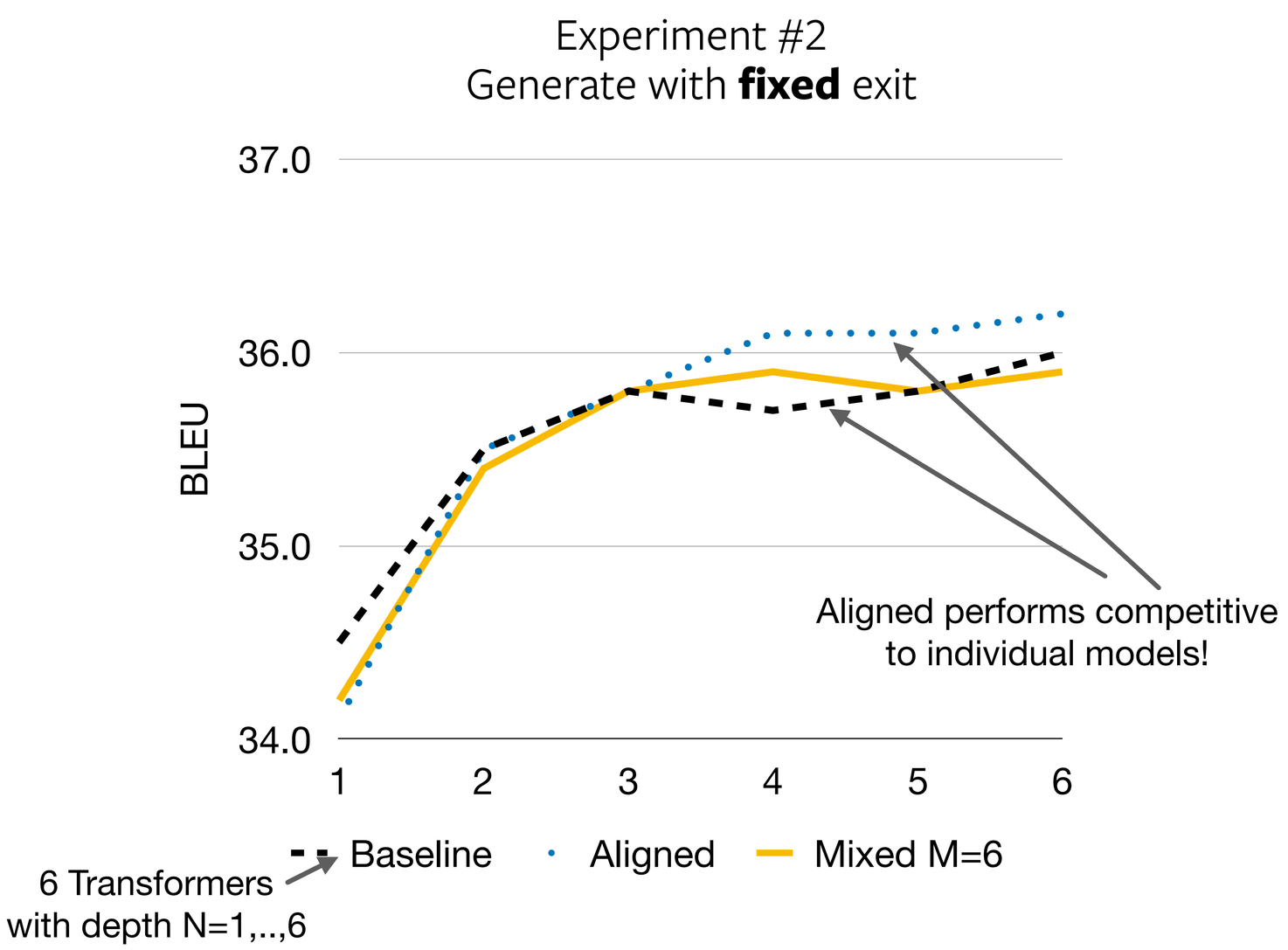


Experiment #2
Generate with **fixed** exit

IWSLT'14 German-English

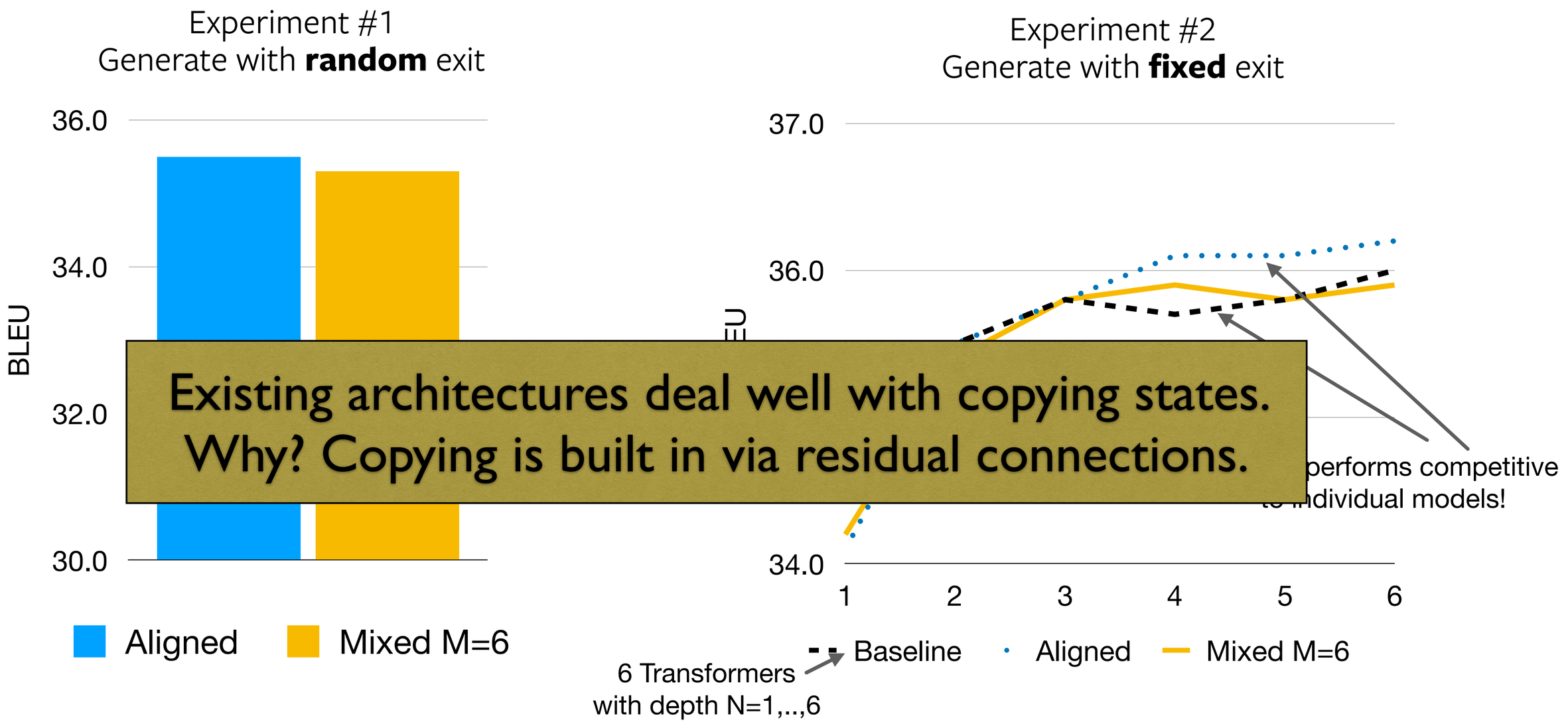
Mixed State Training: Mitigate train/test mismatch





IWSLT'14 German-English

Mixed State Training: Mitigate train/test mismatch



Existing architectures deal well with copying states.
Why? Copying is built in via residual connections.

IWSLT'14 German-English

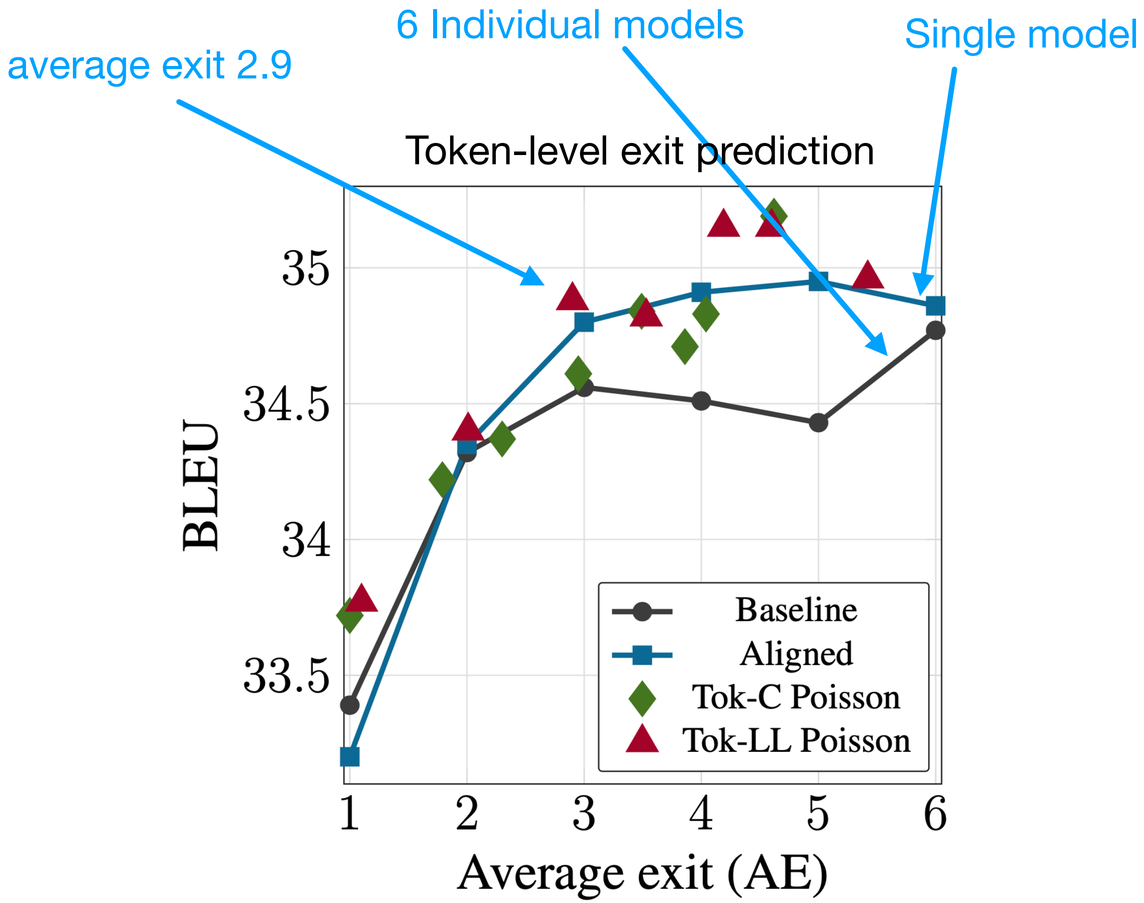
Training

- Exit classifier supervision
  - Exit with highest likelihood (of correct token)
  - Lowest exit at which the correct token is the argmax
  - Penalty to encourage lower exits
  - For sequence-level or token-level

- Phase 1: Aligned training w/o exit classifier
- Phase 2: Add exit classifier

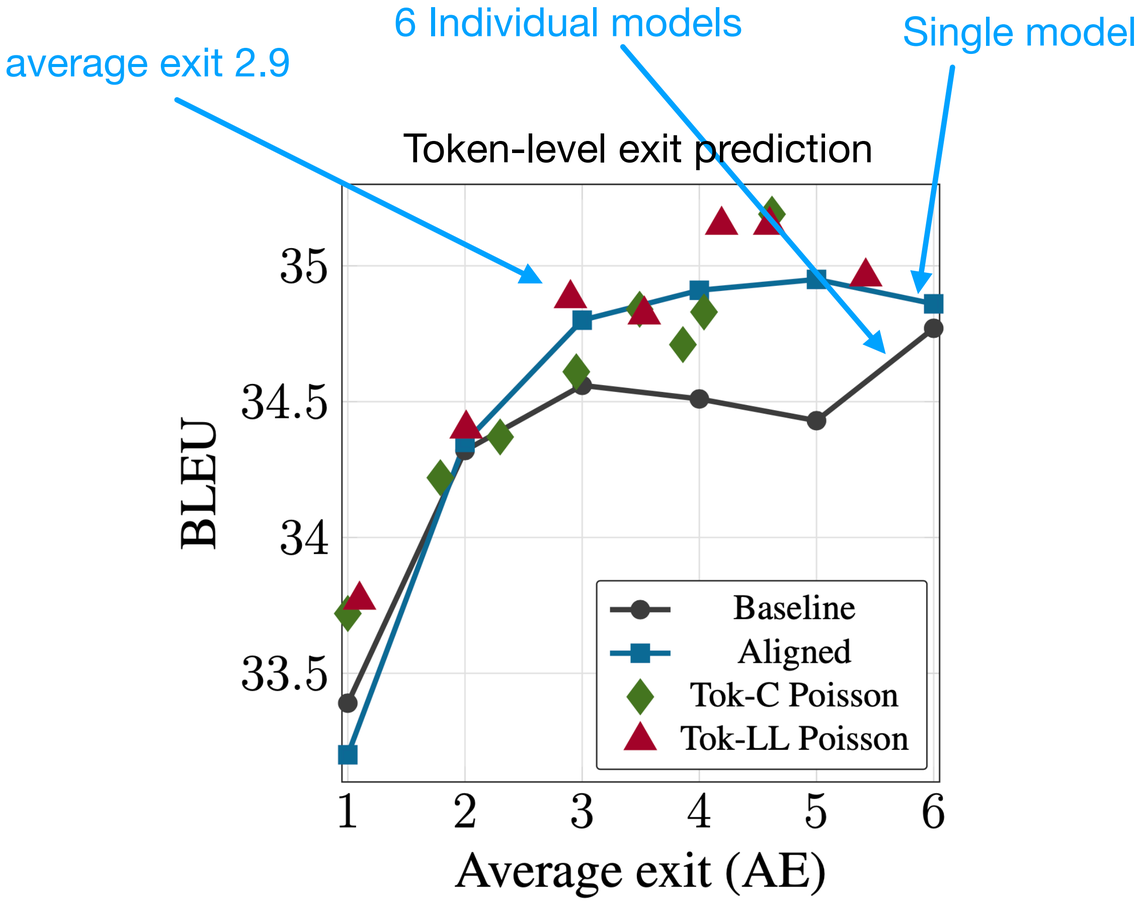
Experiments

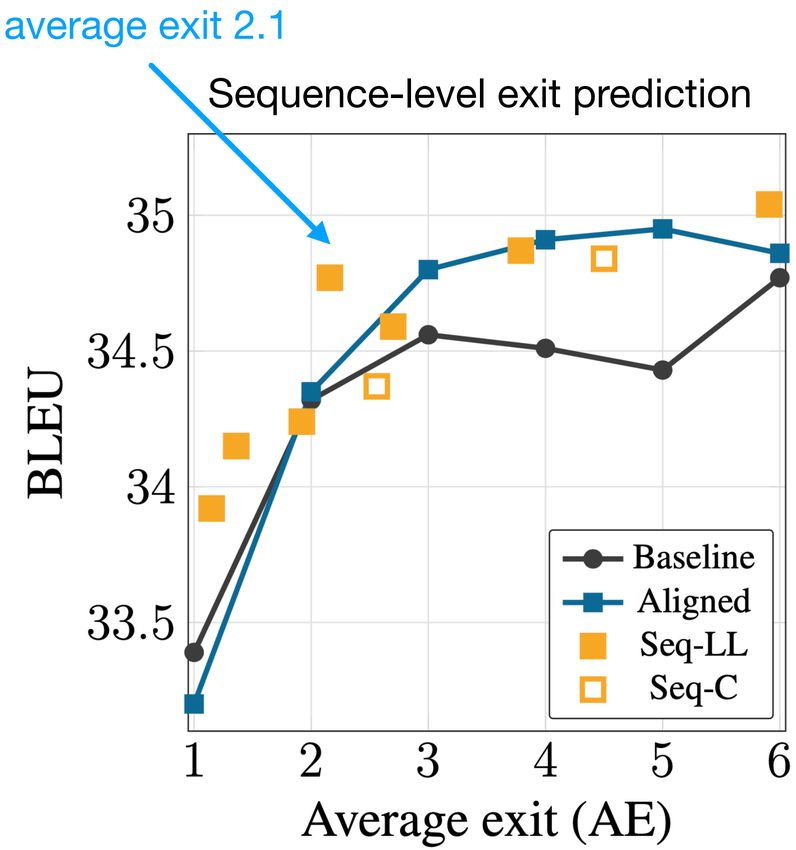
Results



IWSLT'14 German-English

Results

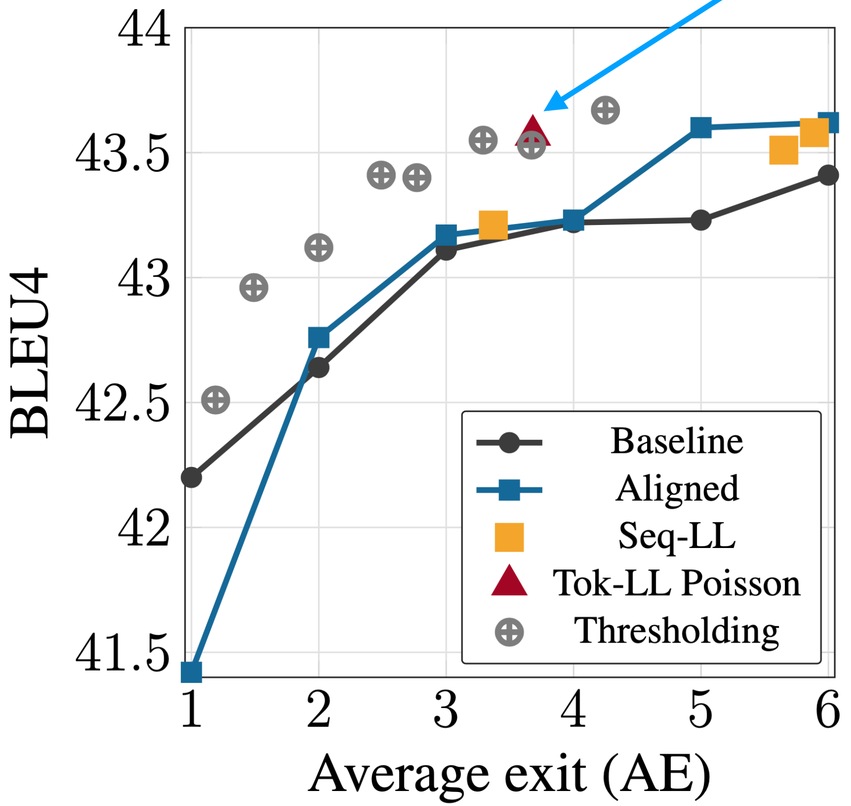


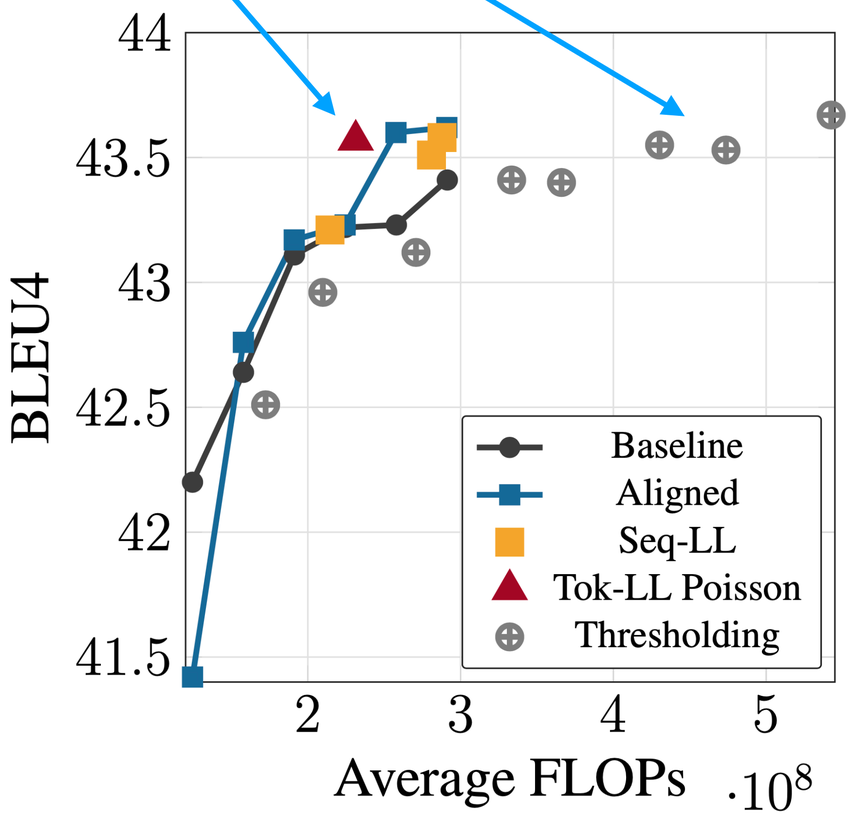


IWSLT'14 German-English

WMT’14 English-French

Thresholding vs. Exit Classifier!





WMT'14 English-French

Related Work

- Adaptive Computation Time (Grave ′16): RNNs
- Universal Transformers (Dehgani et al., ′18): repeated application of the same layer, different goal.
- Multi-scale DenseNets (Huang et al., ′18): classification / structured prediction

Summary

- We can half the effectively used model at no loss in accuracy.
- Modern sequence models resilient: models can naturally deal with copied states.
- Future work: Applications to other sequence modeling tasks!

This talk

- Depth-Adaptive Transformers
  Do you need a huge model to process simple inputs?

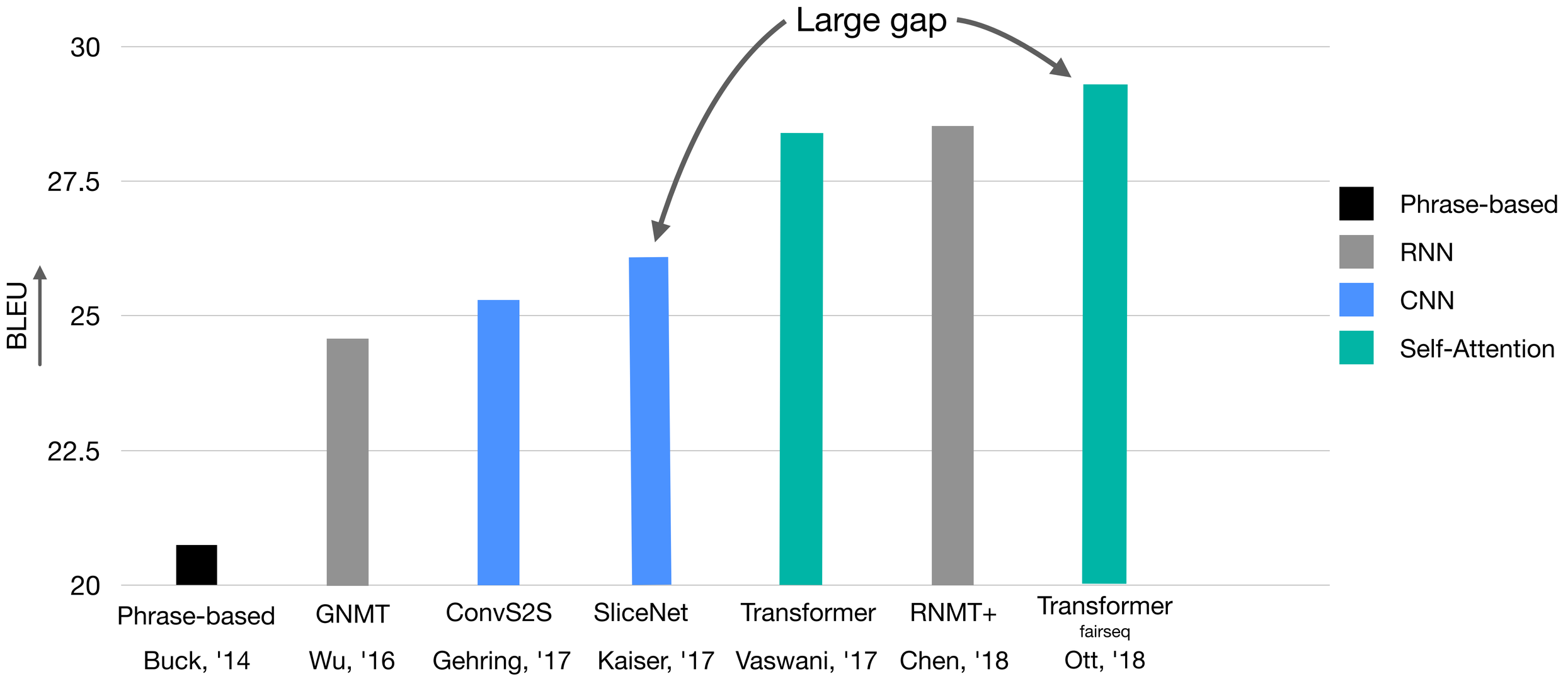
- Lightweight and Dynamic Convolutions
  More efficient architectures for sequence modeling

Research questions

- Neural sequence models make heavy use of self-attention (Transformer)
- Q1: Is self-attention critical to get good performance?
- Q2: Can we do well on a range of NLP tasks with limited context (with a more efficient mechanism)?

WMT newstest2014
English to German

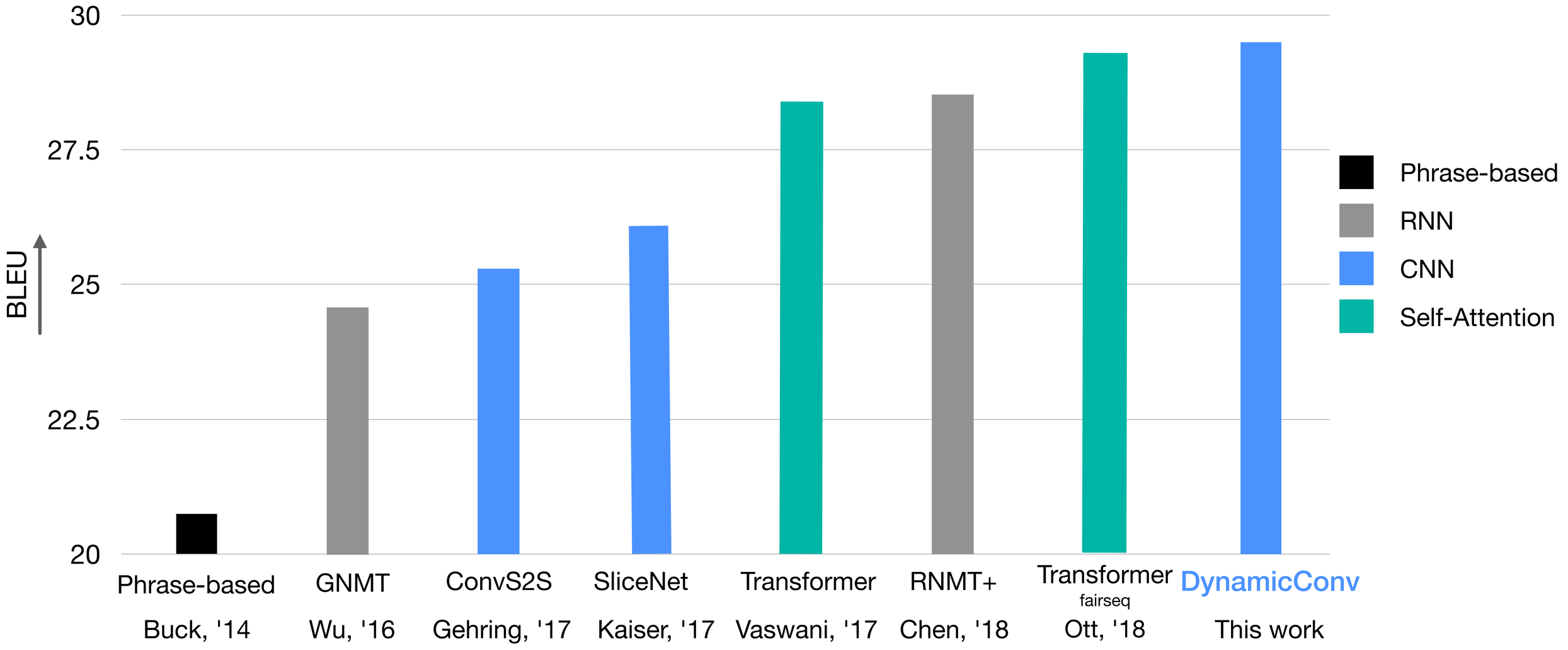
Progress in Neural Machine Translation



WMT newstest2014
English to German

Progress in Neural Machine Translation

Also faster!



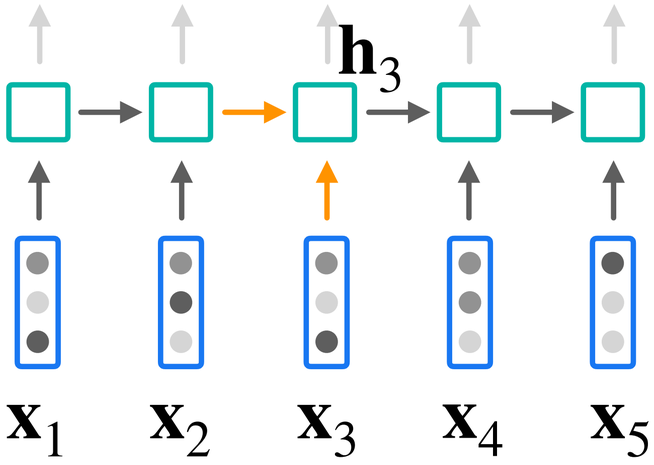
Architectures for Sequence Modeling

n: sequence length
k: kernel size

RNNs

Rumelhart et al., 1986
Hochreiter and Schmidhuber, 1997
Cho et al., 2014

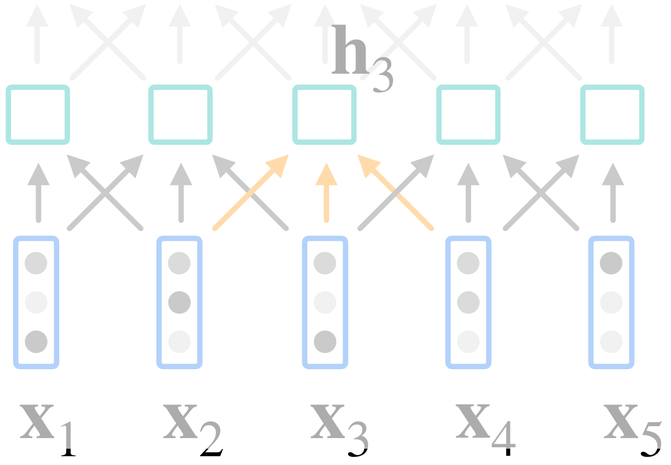
$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$$



CNNs

Waibel et al., 1987
LeCun et al., 1989
Sifre et al., 2014

$$\mathbf{h}_t = f(\mathbf{x}_{t-\lfloor k/2 \rfloor}, \ldots, \mathbf{x}_{t-\lfloor k/2 \rfloor+k})$$

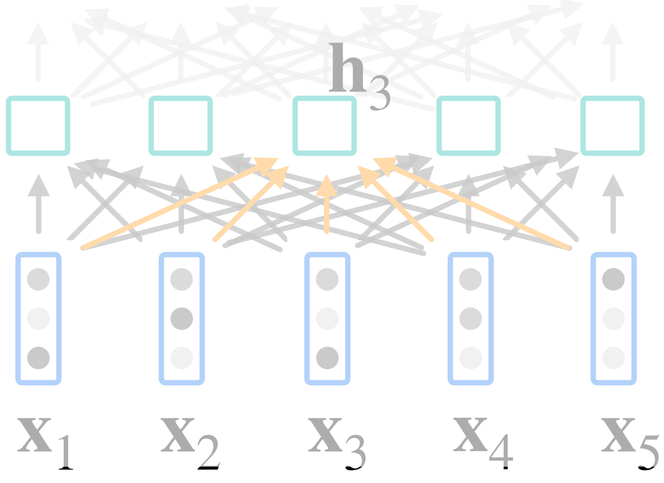


Self-Attention

Cheng et al., 2016
Parikh et al., 2016
Vaswani et al., 2017

$$\mathbf{h}_t = \sum_{i=1}^{n} a_{t,i} \cdot \mathbf{x}_i$$

$$a_{t,i} = \frac{\exp(f(x_t, x_i))}{\sum_{j=1}^{n} \exp(f(x_t, x_j))}$$



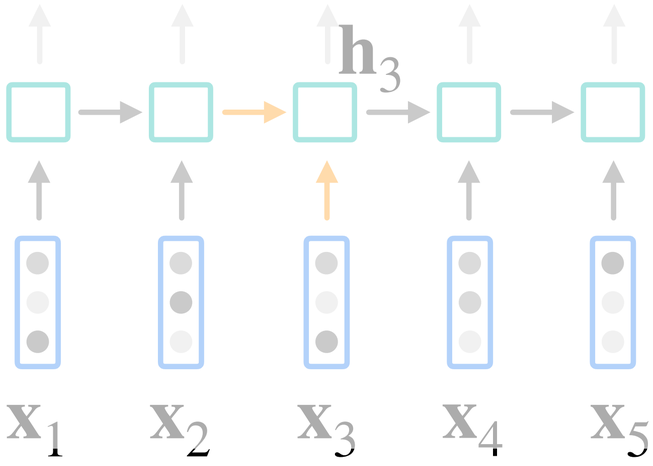
Architectures for Sequence Modeling

n: sequence length
k: kernel size

RNNs

Rumelhart et al., 1986
Hochreiter and Schmidhuber, 1997
Cho et al., 2014

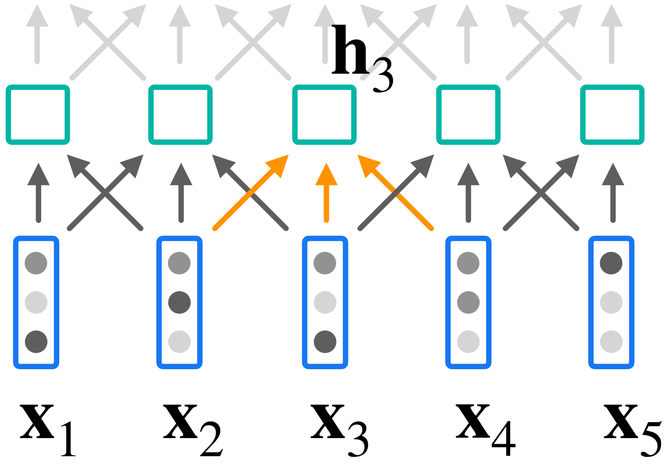
$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$$



CNNs

Waibel et al., 1987
LeCun et al., 1989
Sifre et al., 2014

$$\mathbf{h}_t = f(\mathbf{x}_{t-\lfloor k/2 \rfloor}, \ldots, \mathbf{x}_{t-\lfloor k/2 \rfloor+k})$$

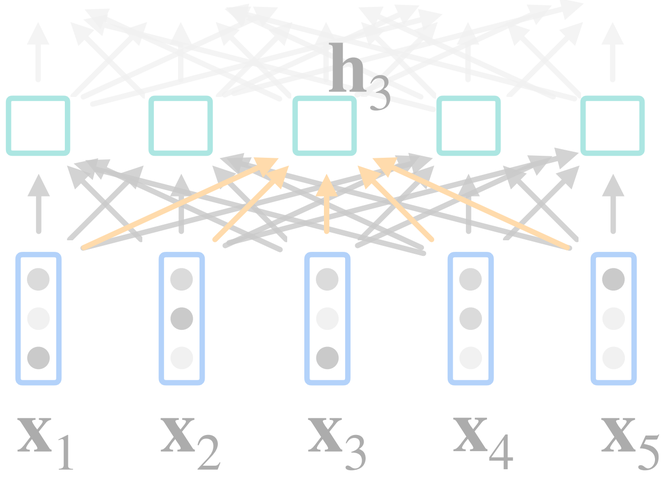


Self-Attention

Cheng et al., 2016
Parikh et al., 2016
Vaswani et al., 2017

$$\mathbf{h}_t = \sum_{i=1}^{n} a_{t,i} \cdot \mathbf{x}_i$$

$$a_{t,i} = \frac{\exp(f(x_t, x_i))}{\sum_{j=1}^{n} \exp(f(x_t, x_j))}$$



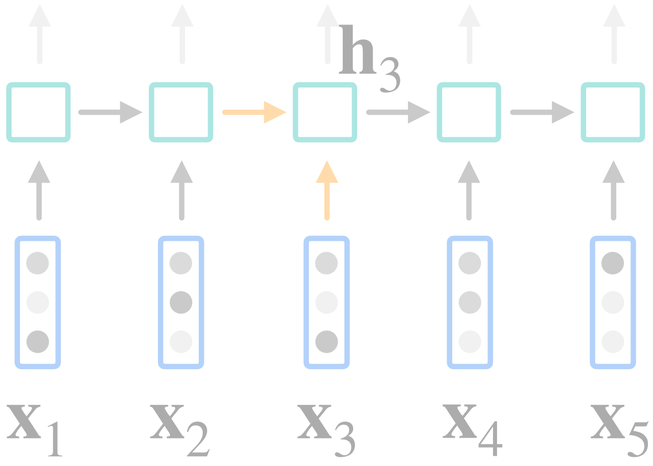
Architectures for Sequence Modeling

n: sequence length
k: kernel size

RNNs

Rumelhart et al., 1986
Hochreiter and Schmidhuber, 1997
Cho et al., 2014

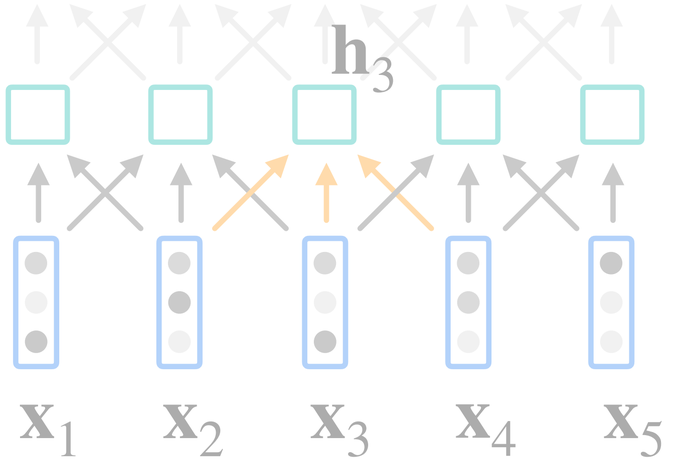
$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$$



CNNs

Waibel et al., 1987
LeCun et al., 1989
Sifre et al., 2014

$$\mathbf{h}_t = f(\mathbf{x}_{t-\lfloor k/2 \rfloor}, \ldots, \mathbf{x}_{t-\lfloor k/2 \rfloor+k})$$

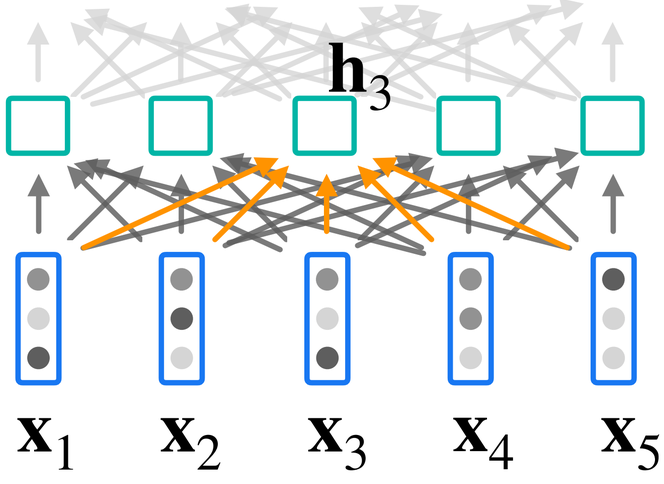


Self-Attention

Cheng et al., 2016
Parikh et al., 2016
Vaswani et al., 2017

$$\mathbf{h}_t = \sum_{i=1}^{n} a_{t,i} \cdot \mathbf{x}_i$$

$$a_{t,i} = \frac{\exp(f(x_t, x_i))}{\sum_{j=1}^{n} \exp(f(x_t, x_j))}$$



n: sequence length
k: kernel size

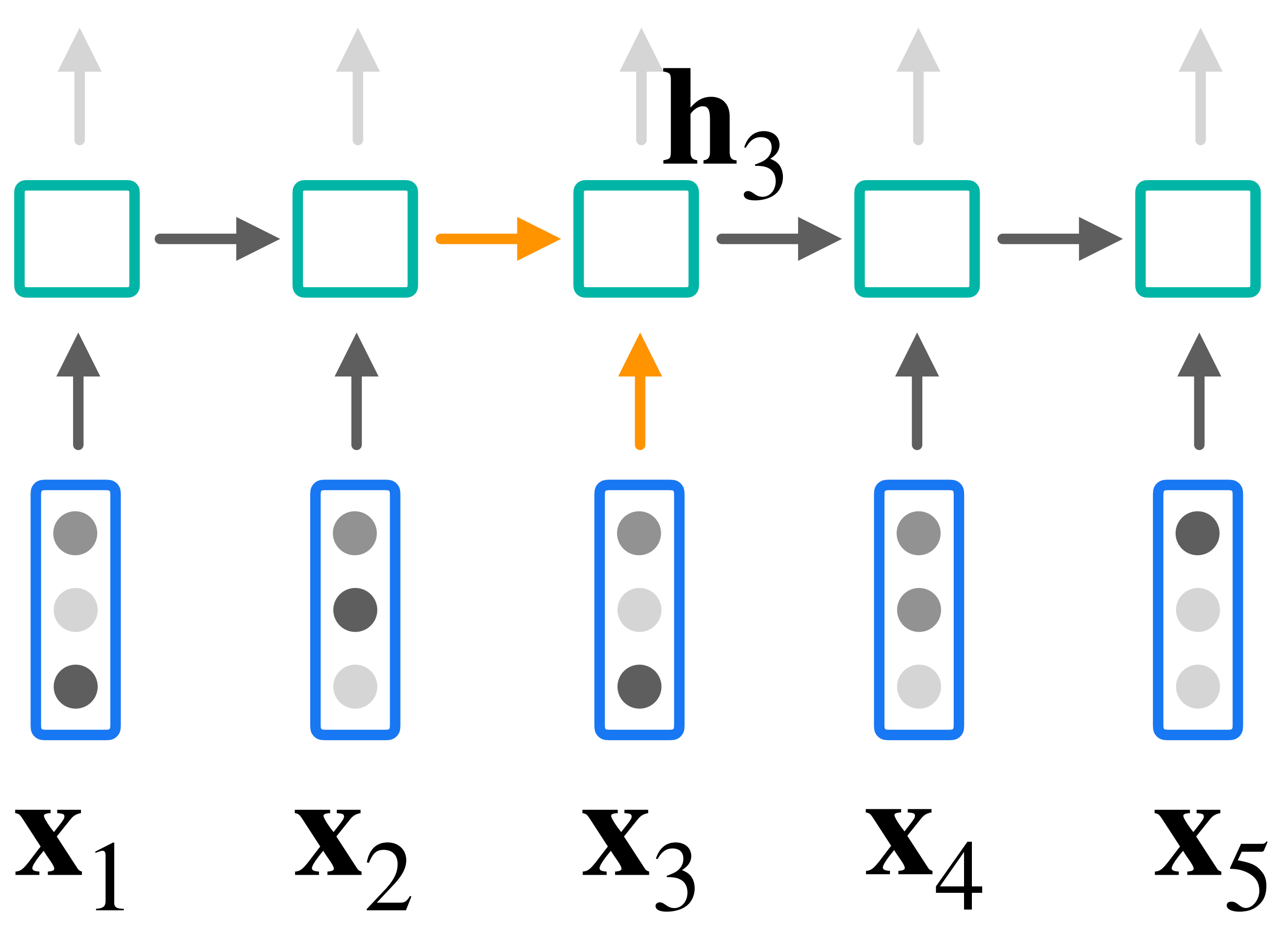
Architectures for Sequence Modeling

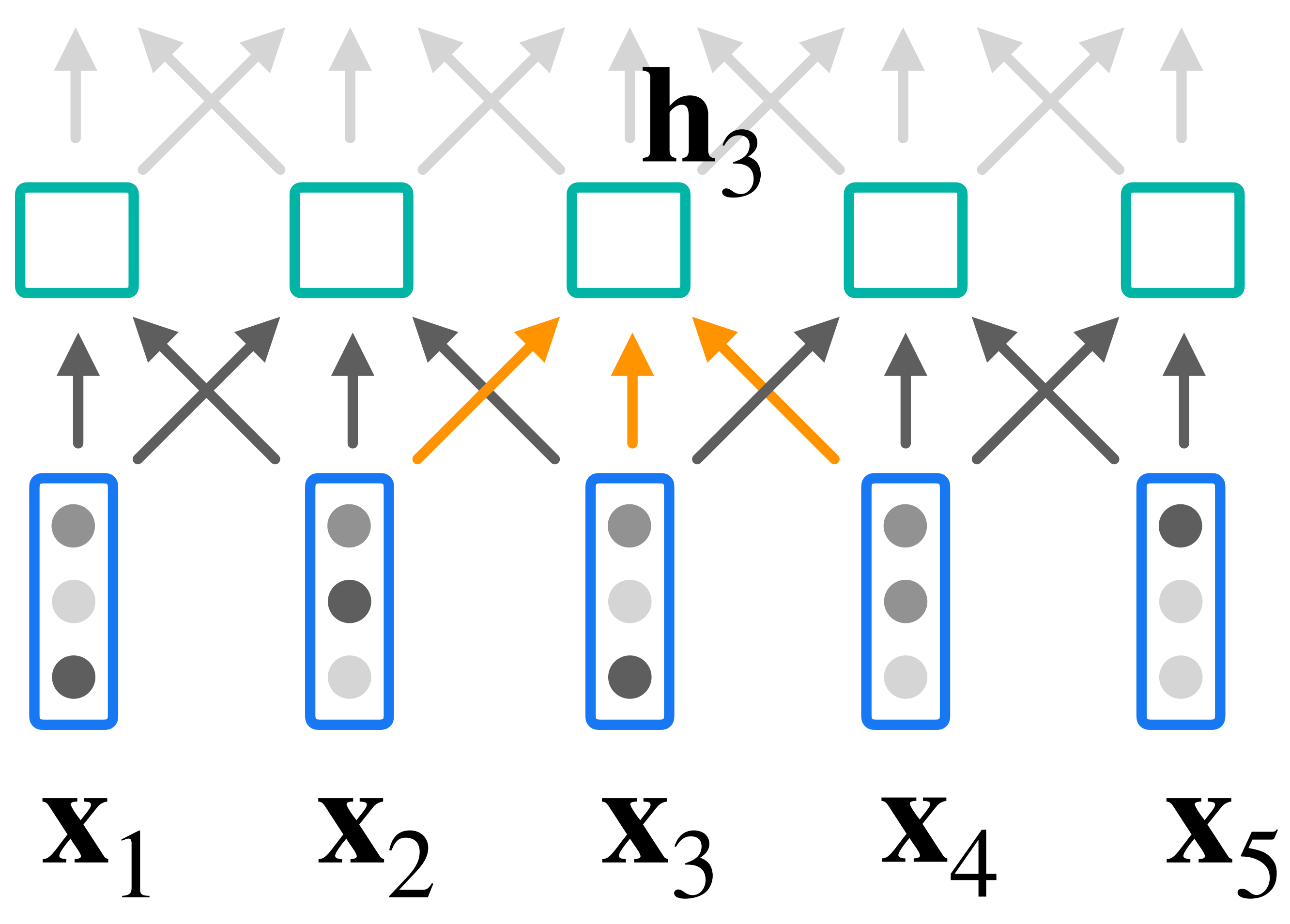
RNNs

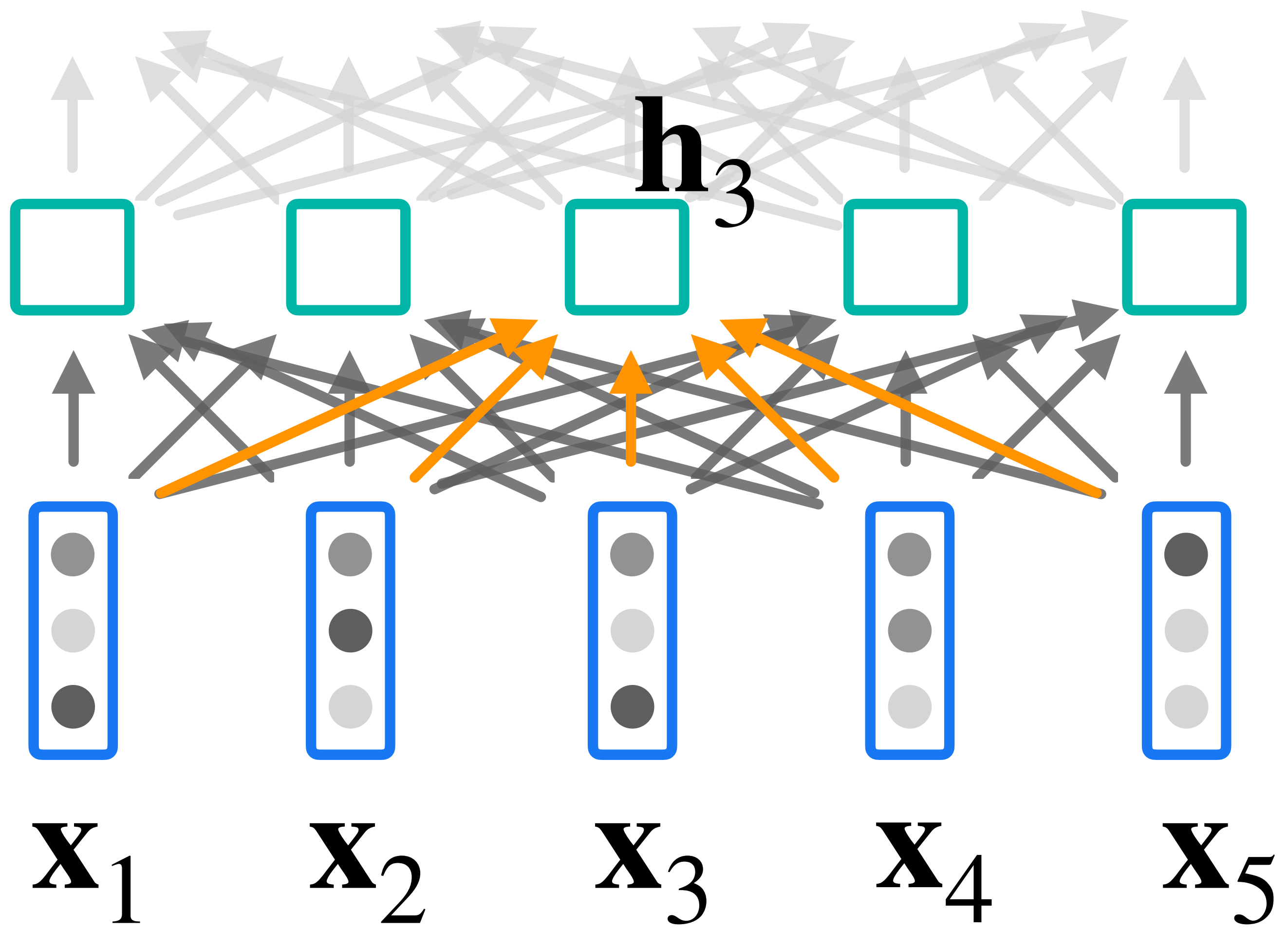
CNNs

Self-Attention

Parallel



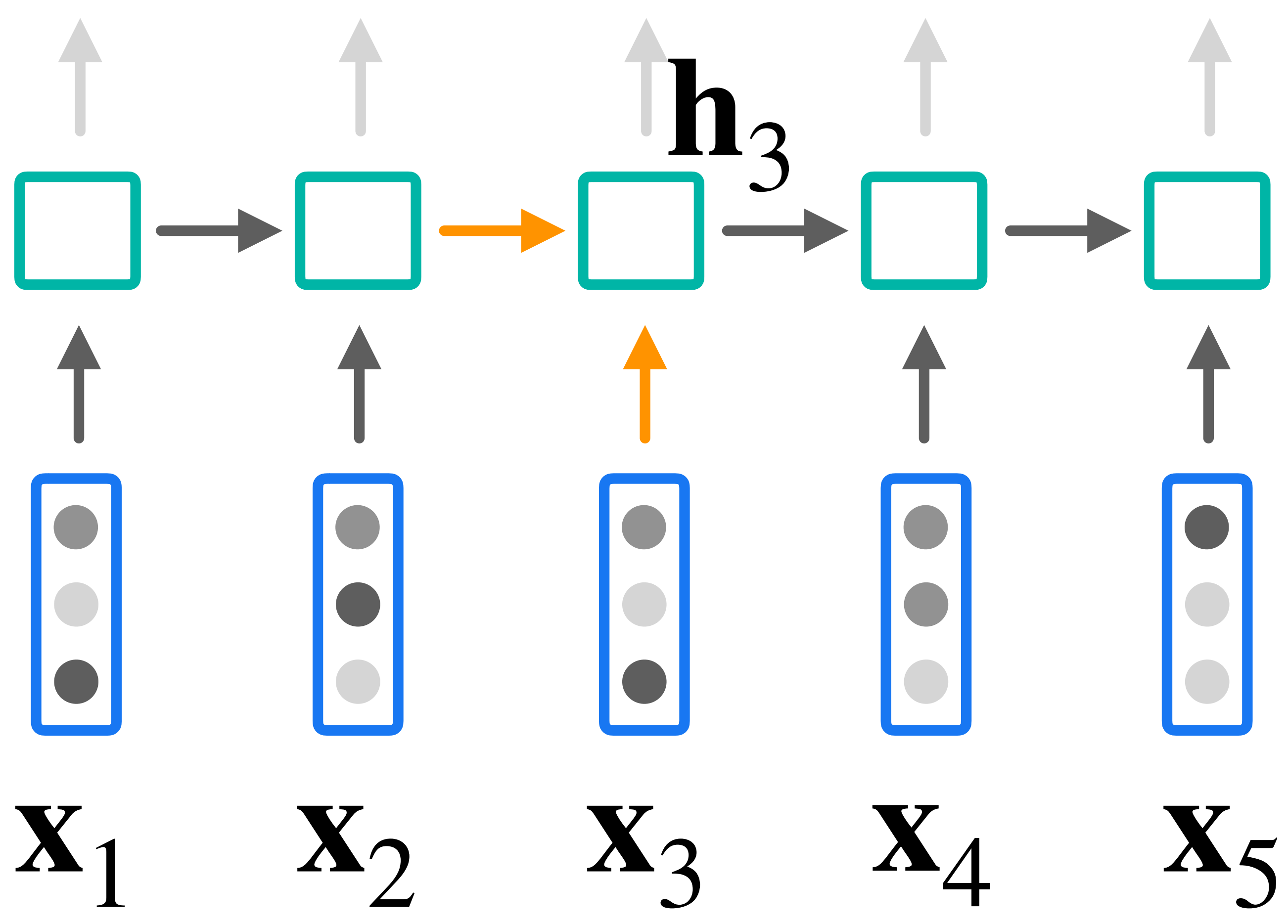


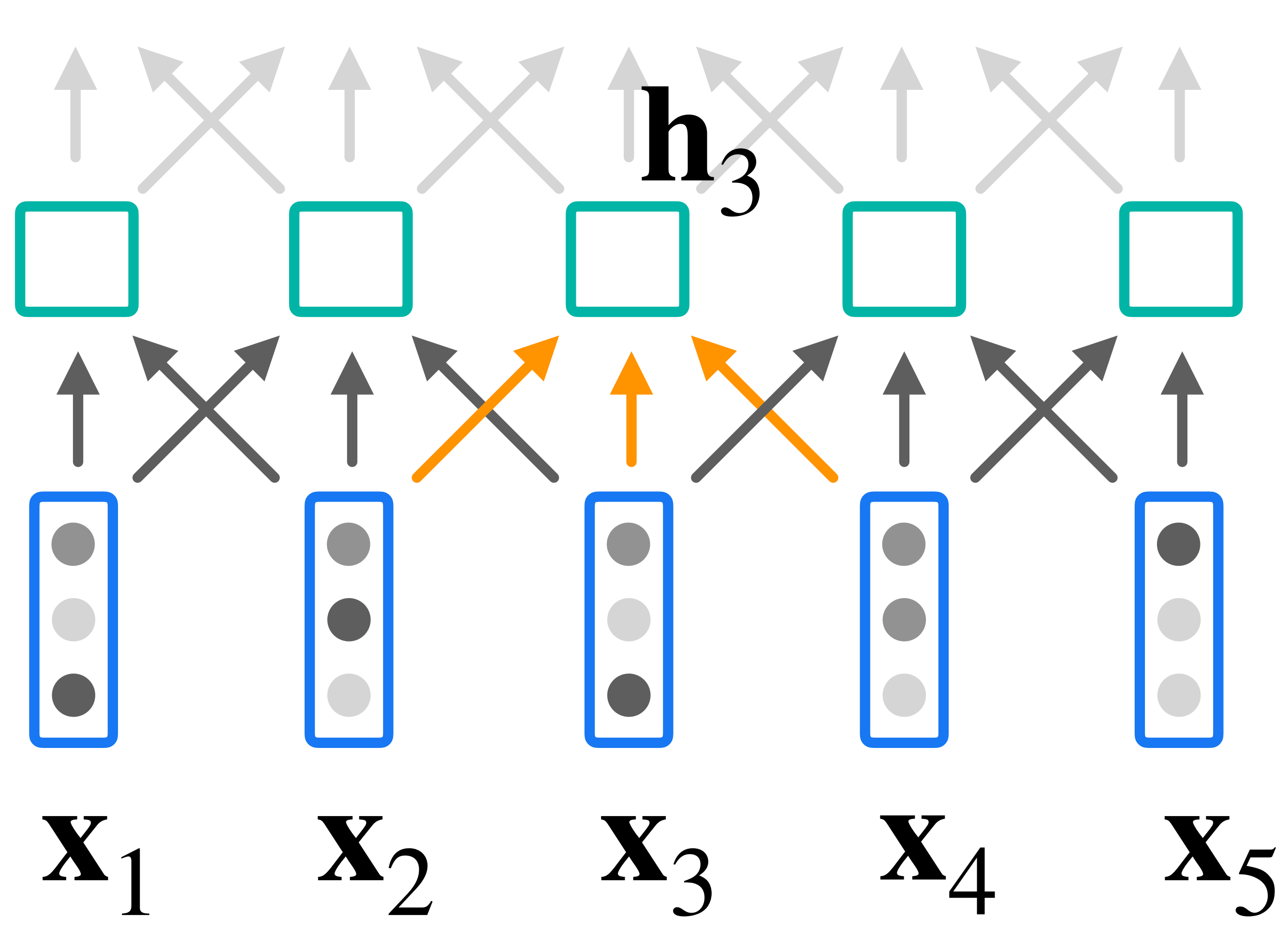


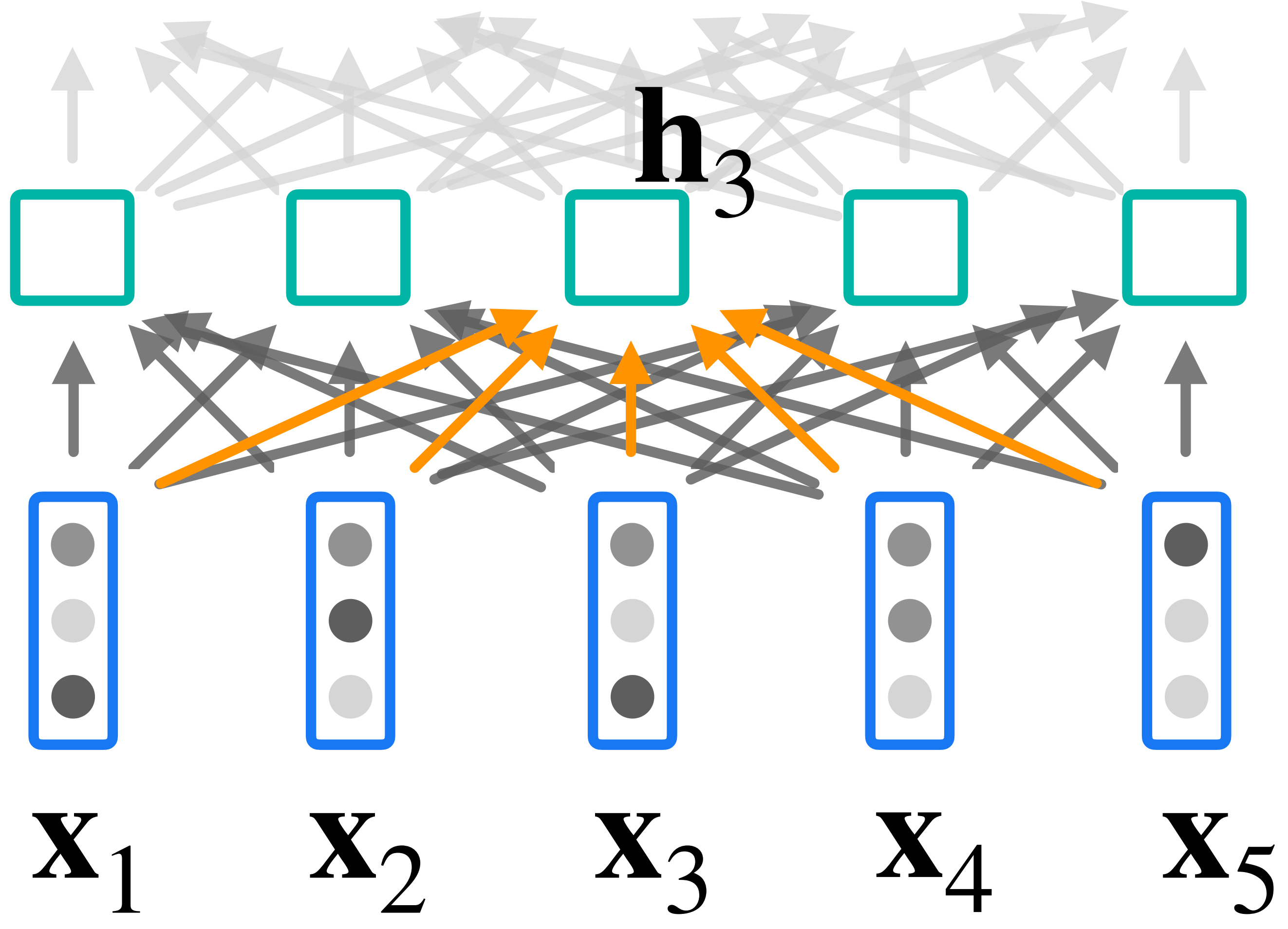
Architectures for Sequence Modeling

n: sequence length
k: kernel size

	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓



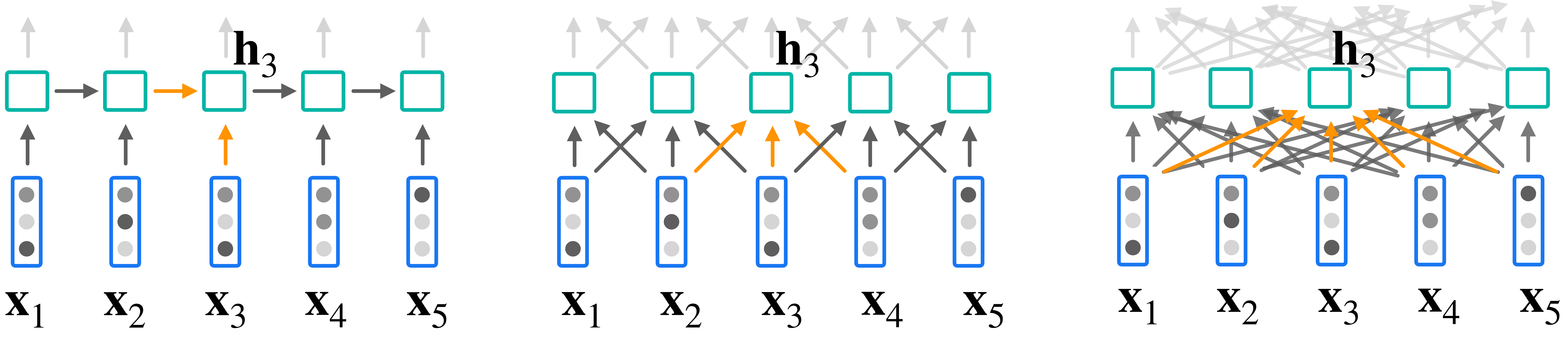




Architectures for Sequence Modeling

n: sequence length
k: kernel size

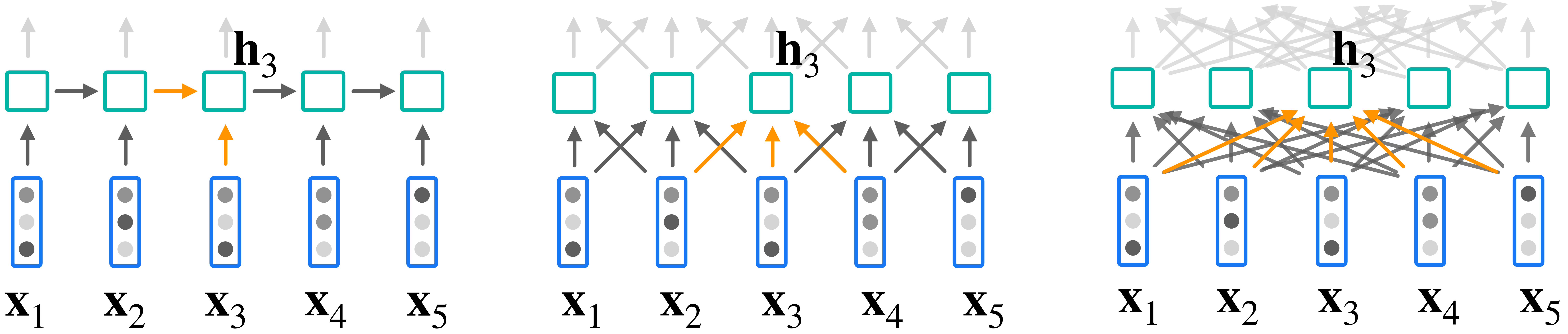
	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context			



Architectures for Sequence Modeling

n: sequence length
k: kernel size

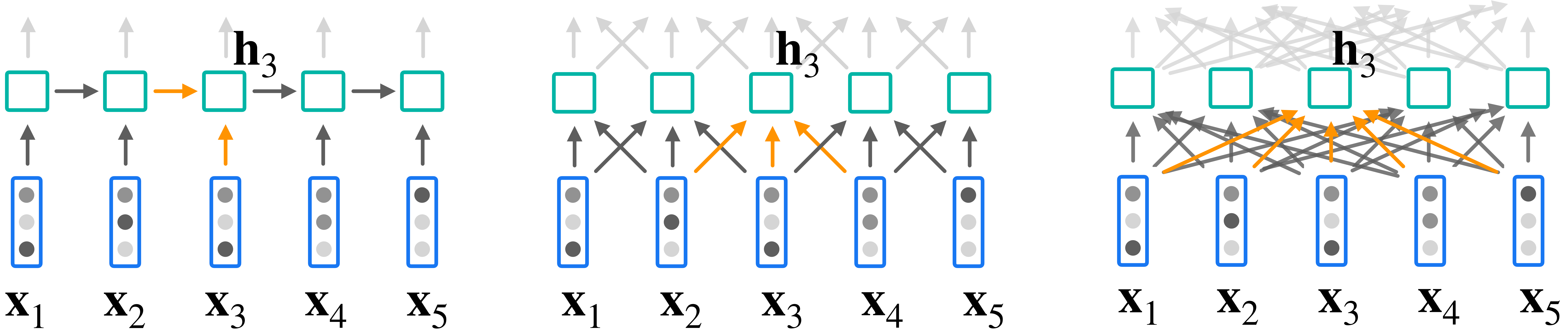
	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓



Architectures for Sequence Modeling

n: sequence length
k: kernel size

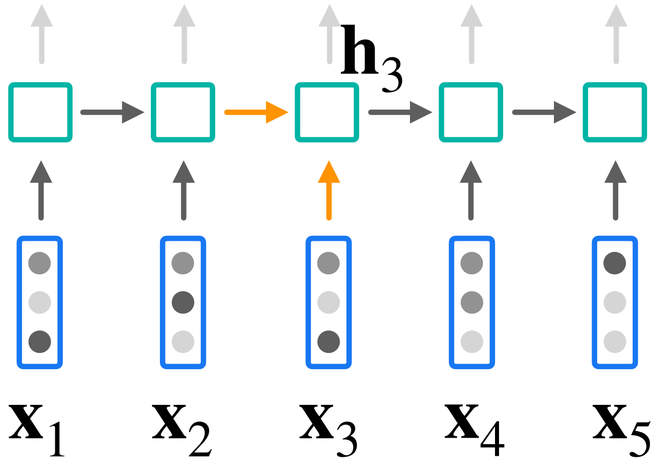
	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓
Time Complexity			

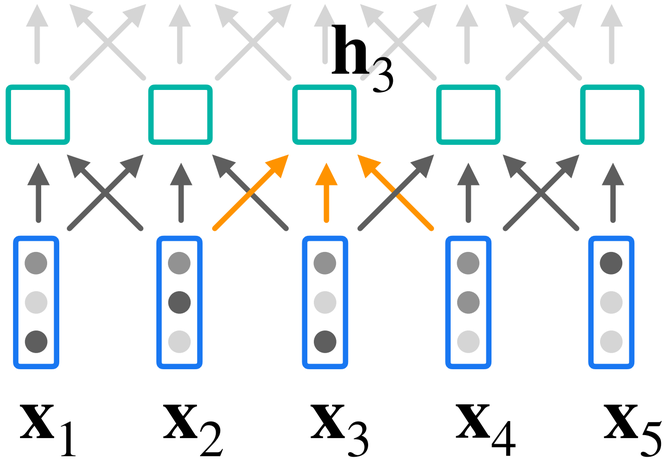


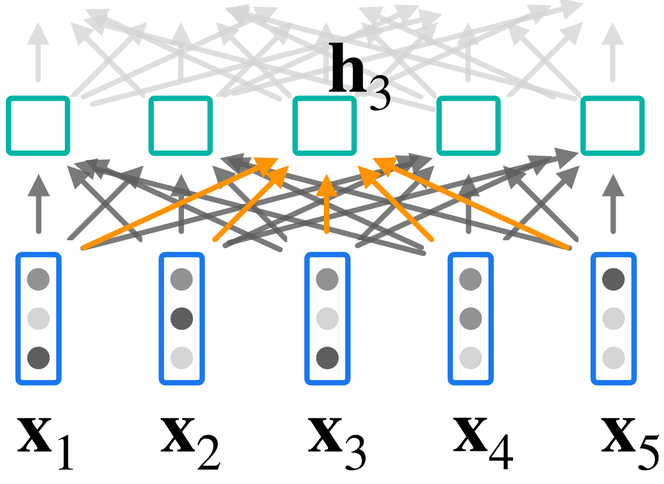
Architectures for Sequence Modeling

n: sequence length
k: kernel size

	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓
Time Complexity	$O(n)$	$O(kn)$	$O(n^2)$





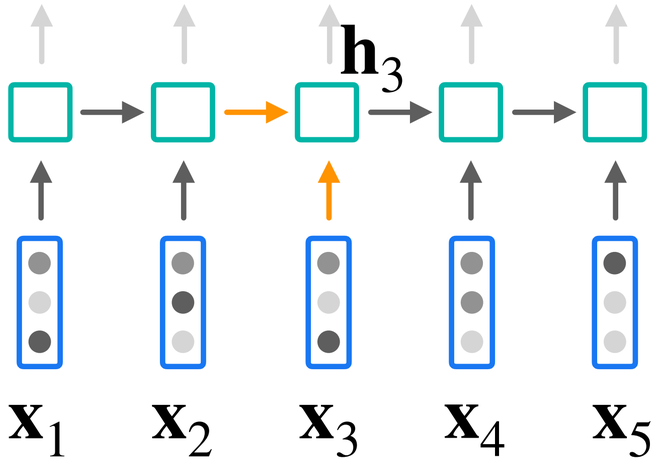


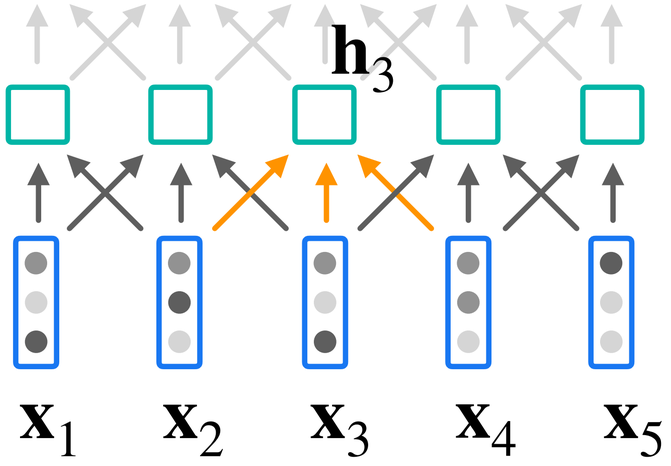
Architectures for Sequence Modeling

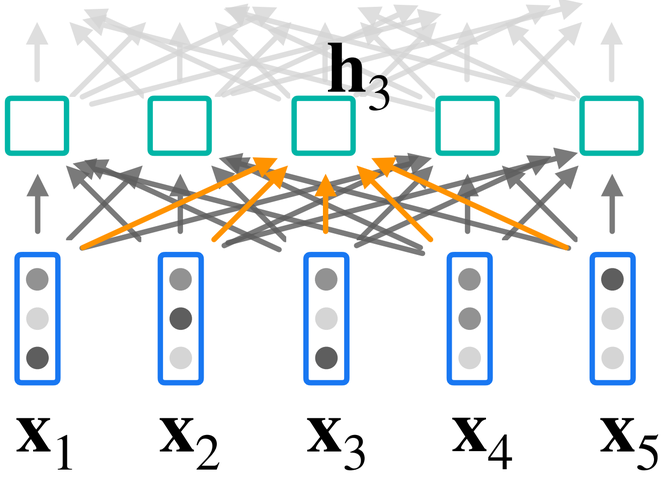
n: sequence length
k: kernel size

	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓
Time Complexity	$O(n)$	$O(kn)$	$O(n^2)$

Dynamic Weights



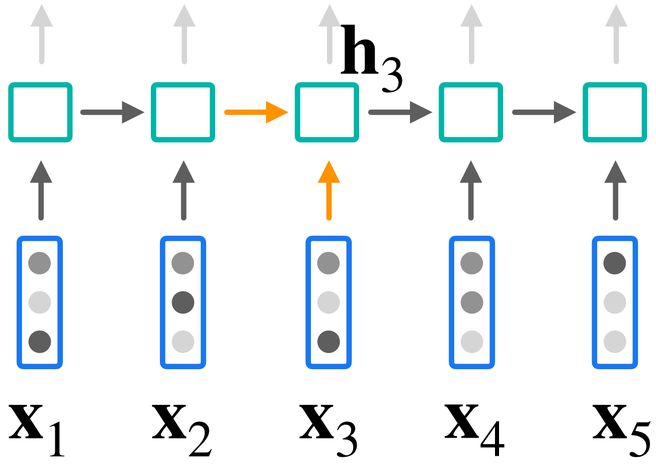


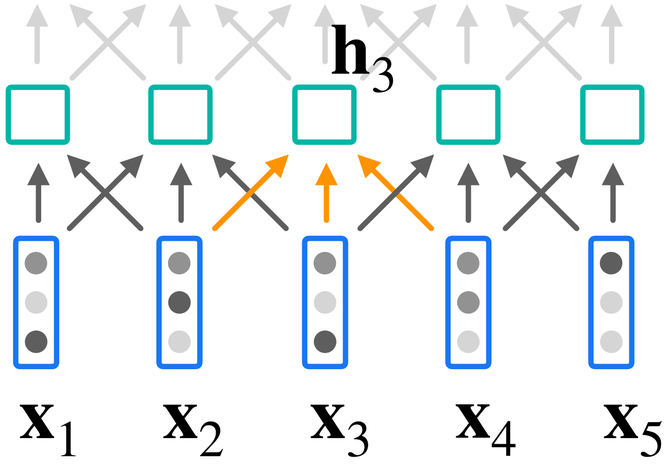


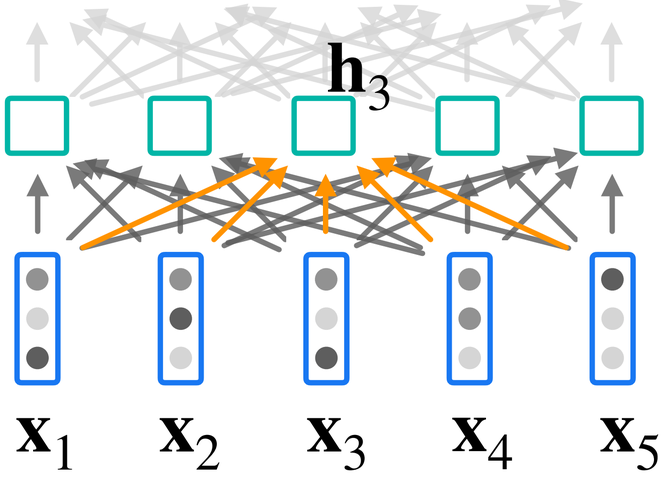
Architectures for Sequence Modeling

n: sequence length
k: kernel size

	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓
Time Complexity	$O(n)$	$O(kn)$	$O(n^2)$
Dynamic Weights	✓	✗	✓



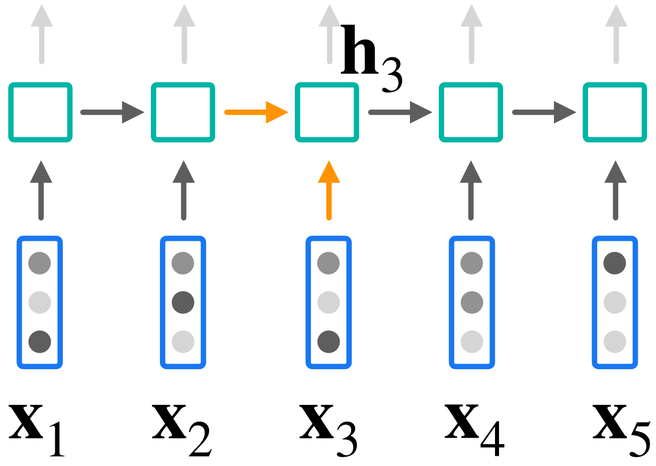


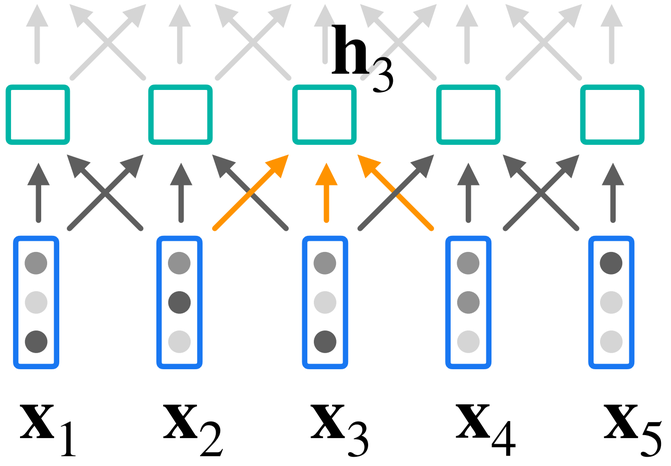


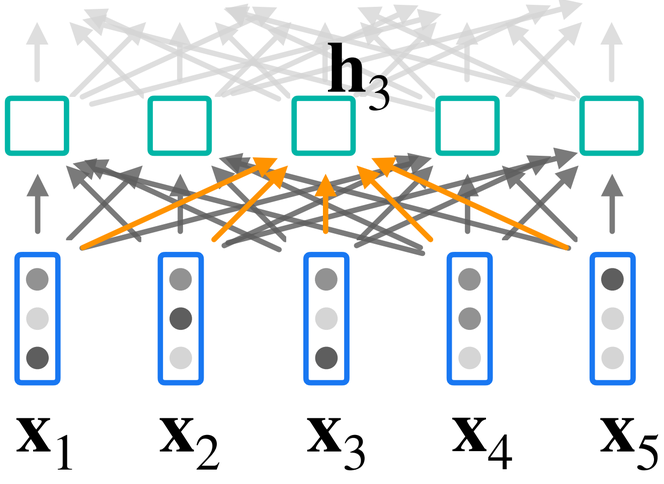
Architectures for Sequence Modeling

n: sequence length
k: kernel size

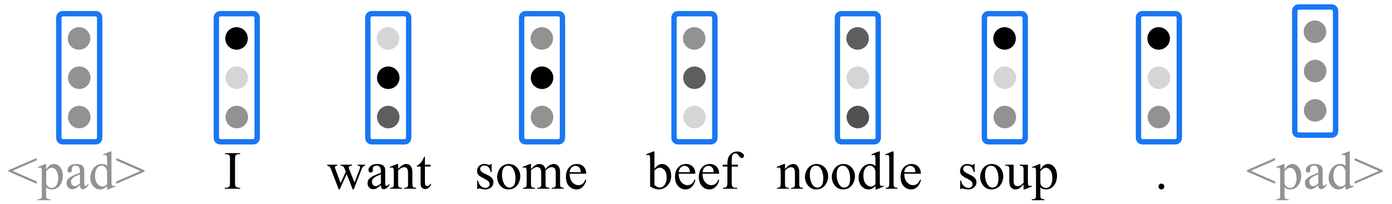
	RNNs	CNNs	Self-Attention
Parallel	✗	✓	✓
Infinite Context	✓	✗	✓
Time Complexity	$O(n)$	$O(kn)$	$O(n^2)$
Dynamic Weights	✓	✓ **This work**	✓





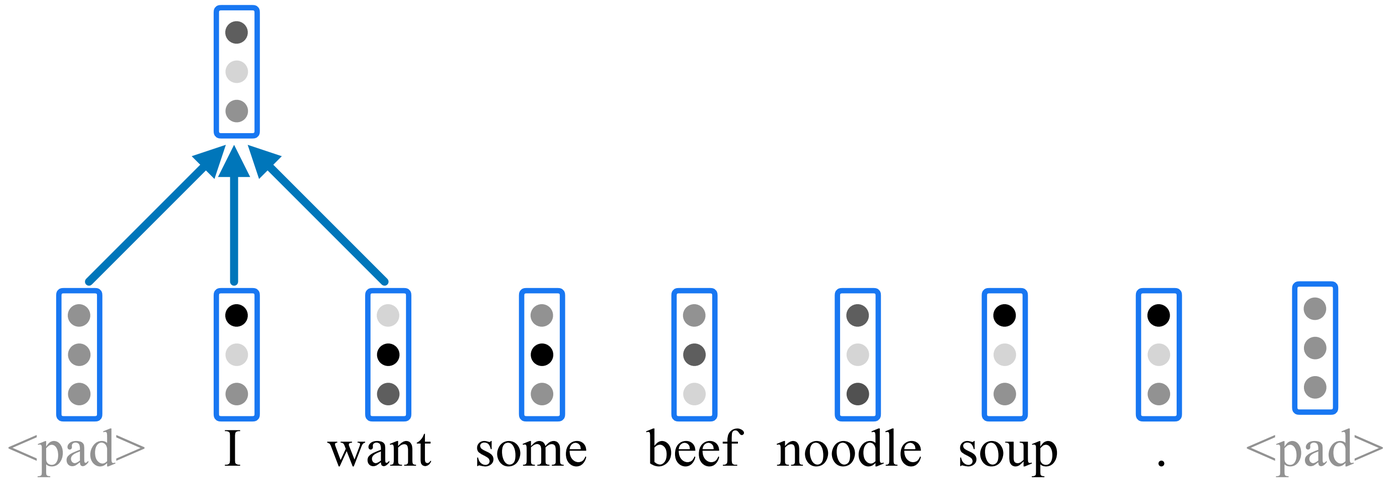


Normal Convolution:



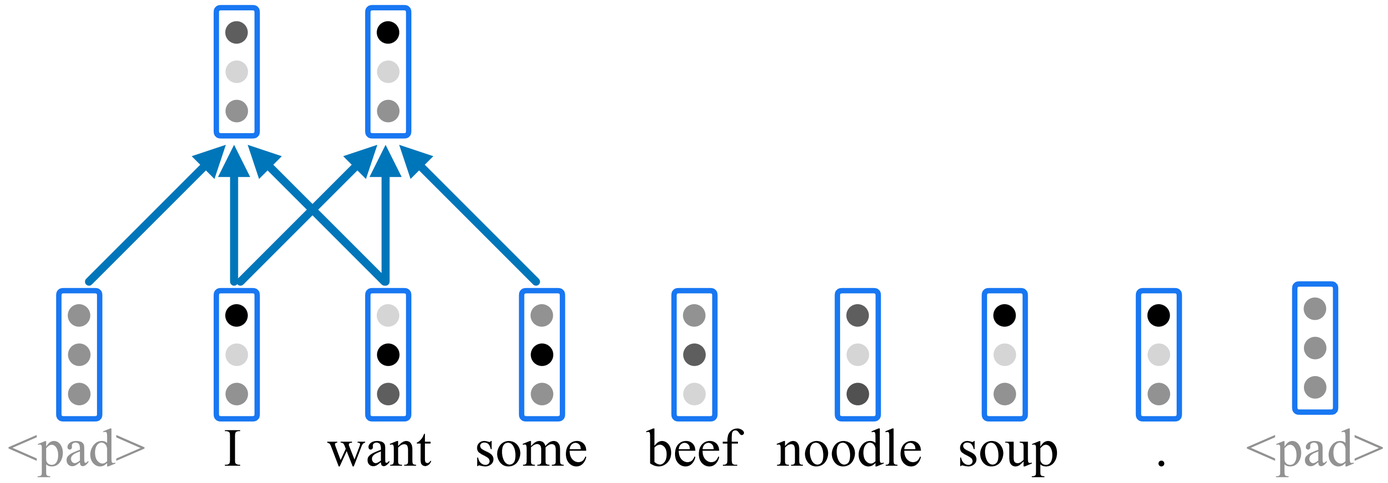
Dynamic Convolution:

Normal Convolution:



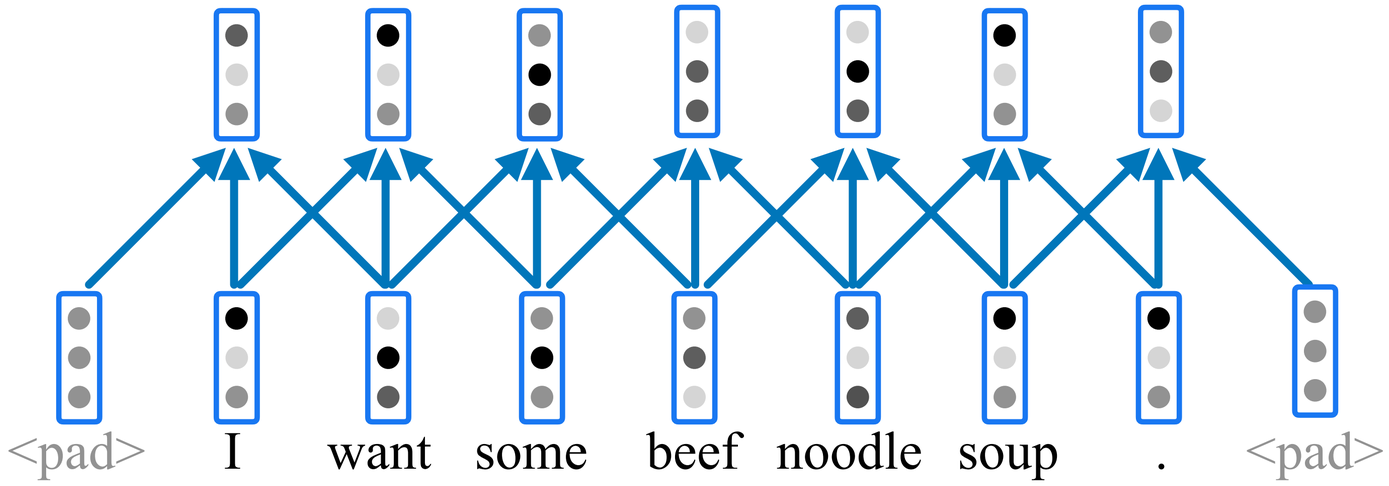
Dynamic Convolution:

Normal Convolution:



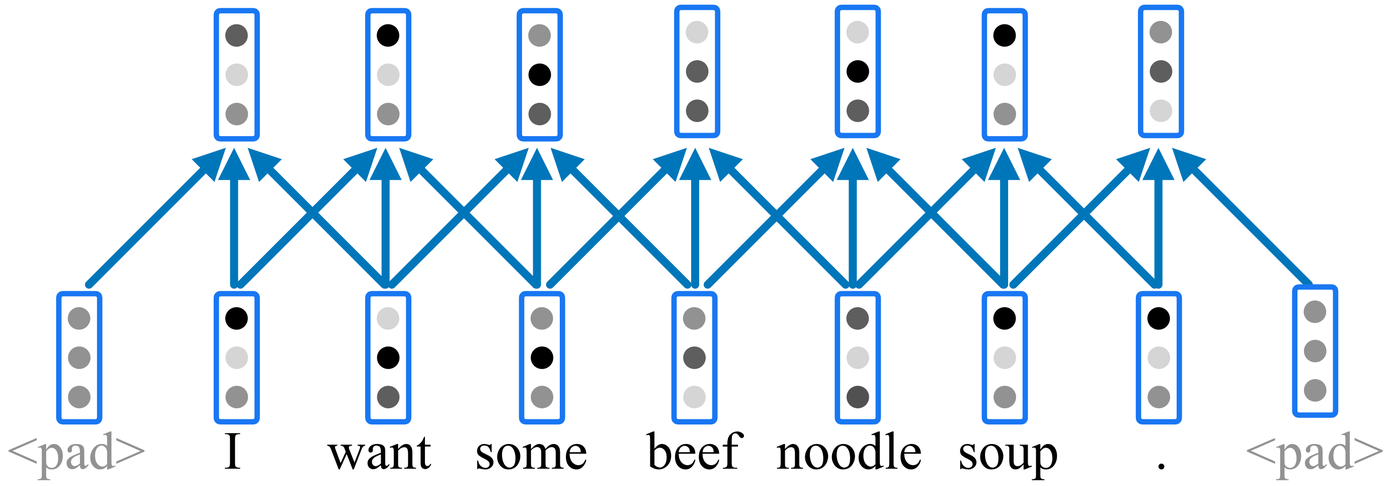
Dynamic Convolution:

Normal Convolution:

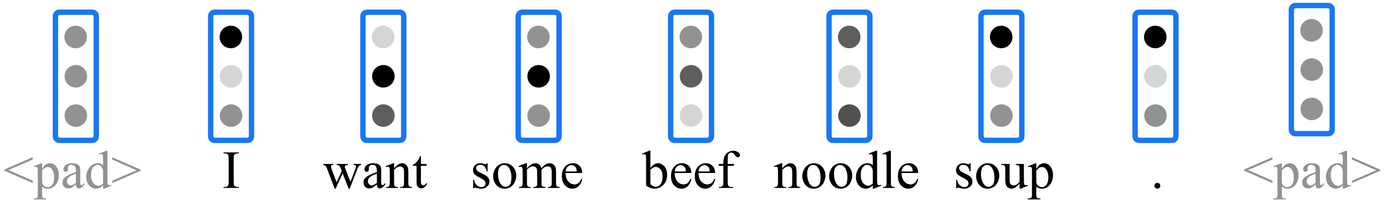


Dynamic Convolution:

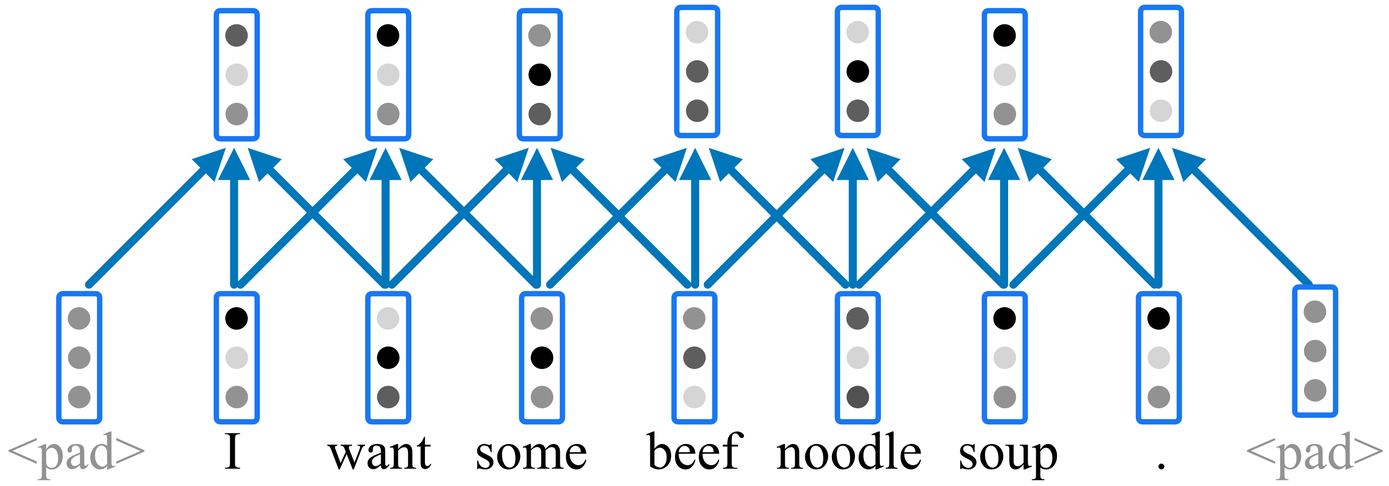
Normal Convolution:



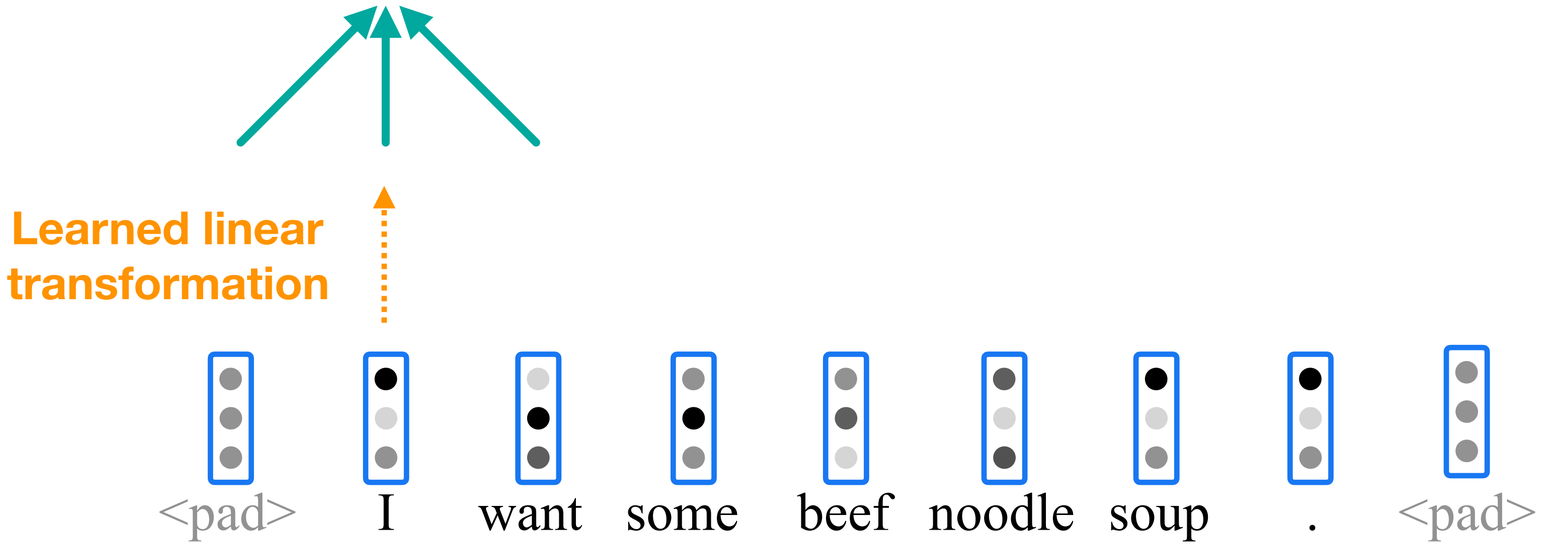
Dynamic Convolution:



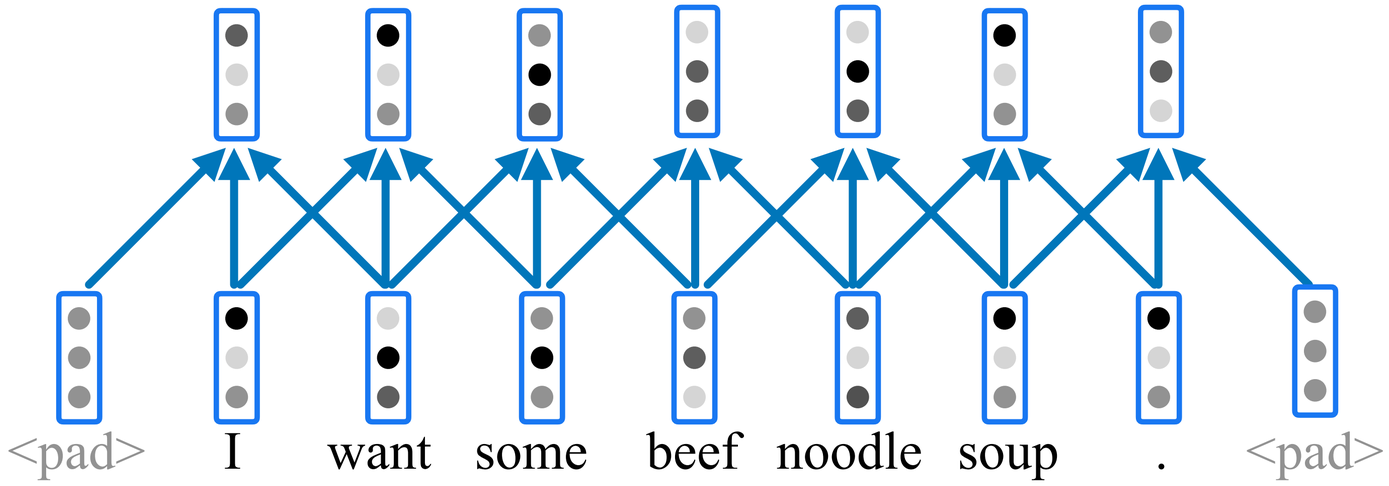
Normal Convolution:



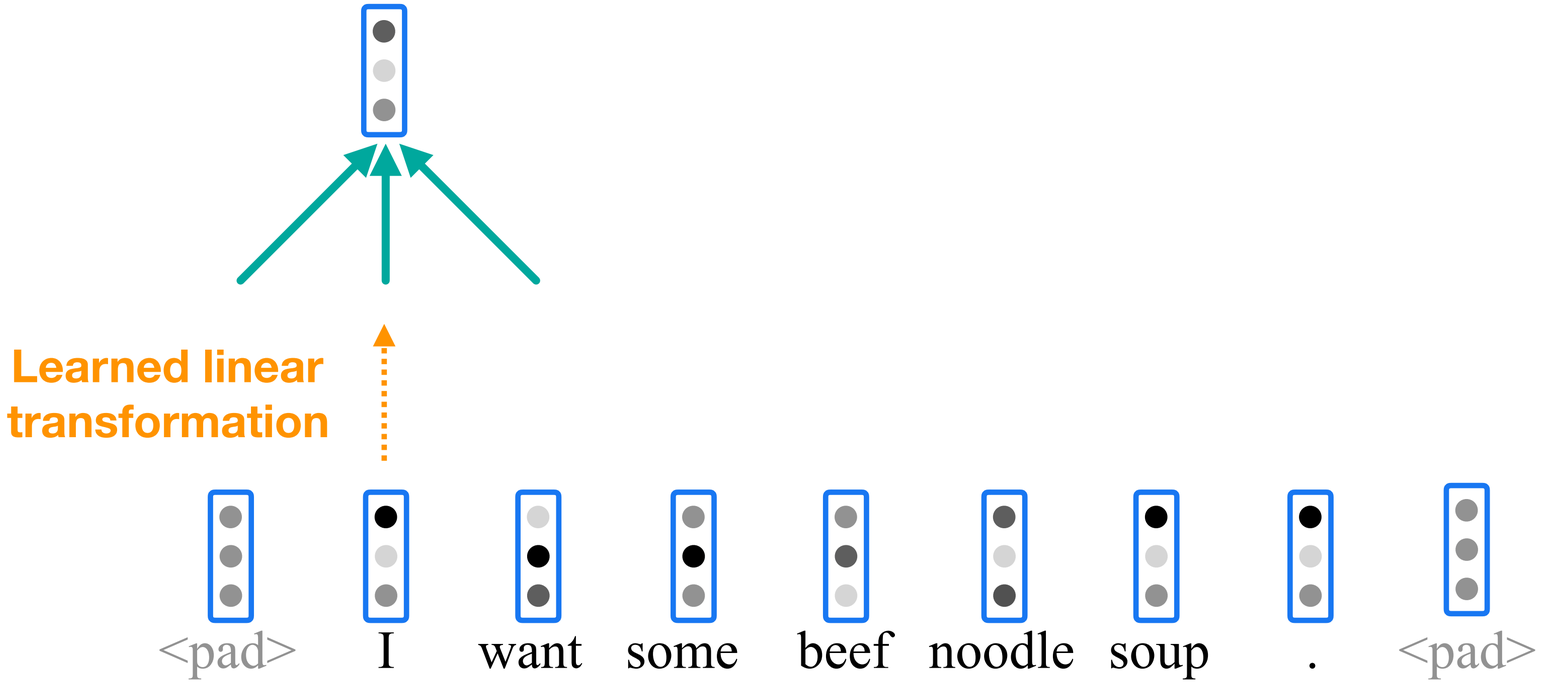
Dynamic Convolution:



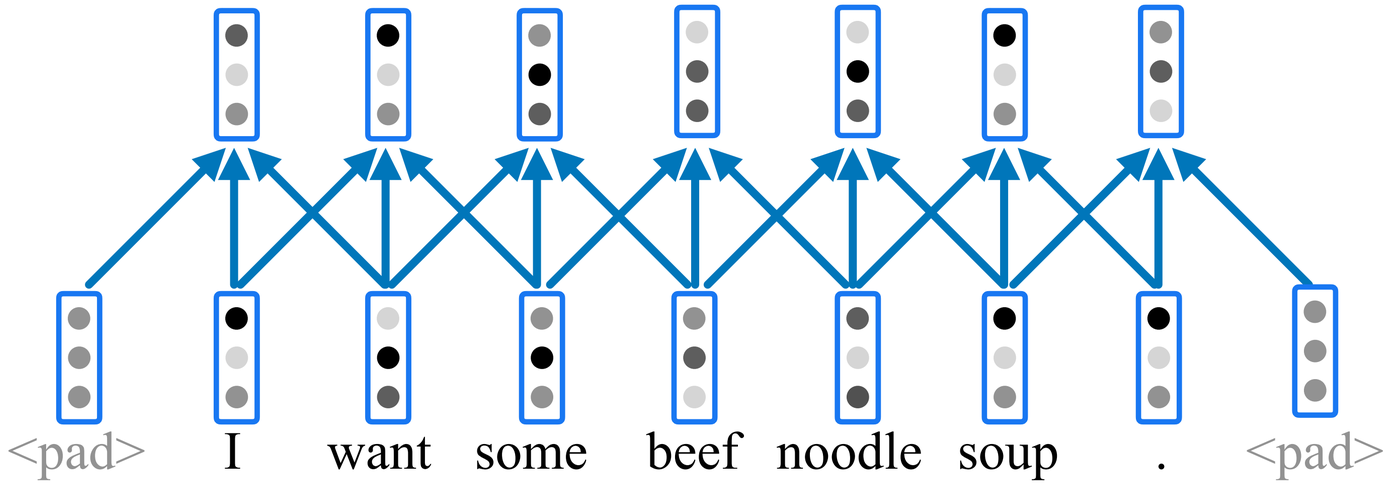
Normal Convolution:



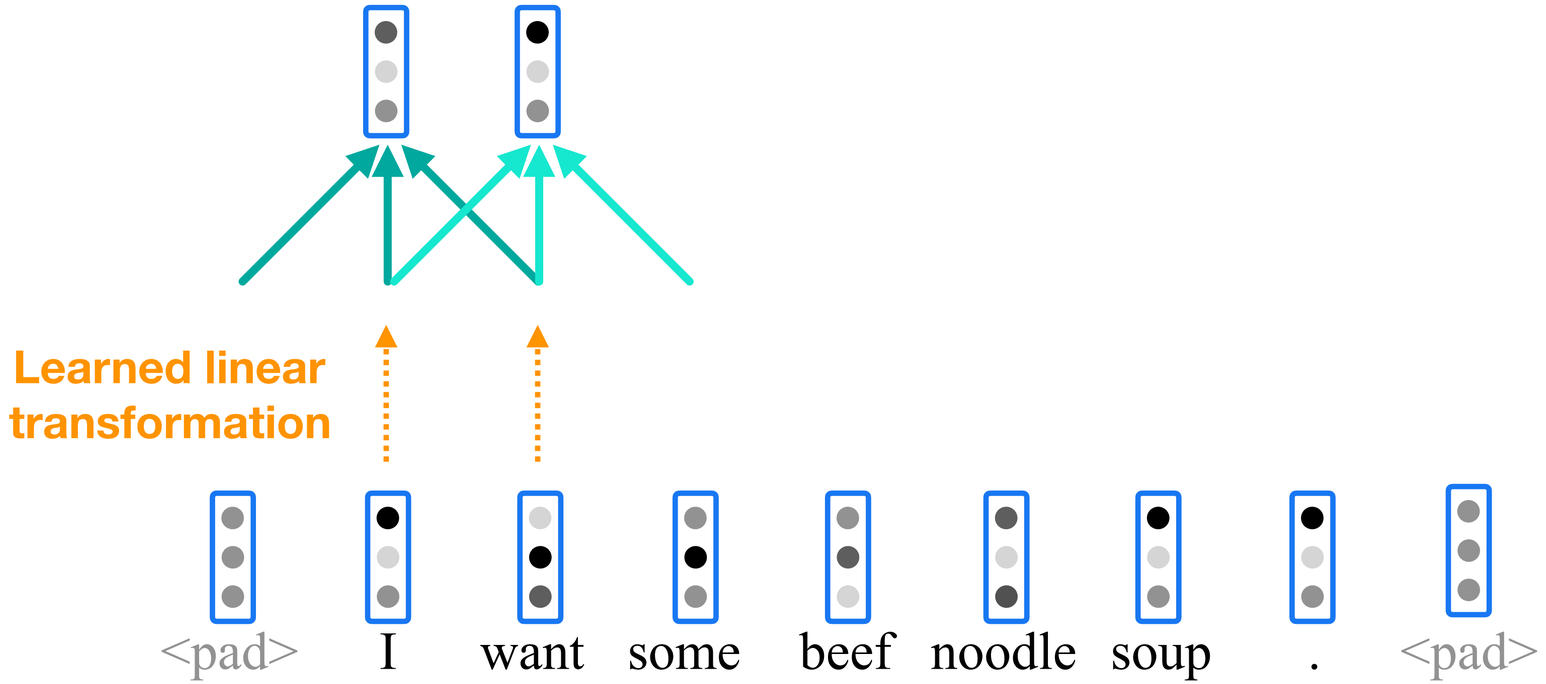
Dynamic Convolution:



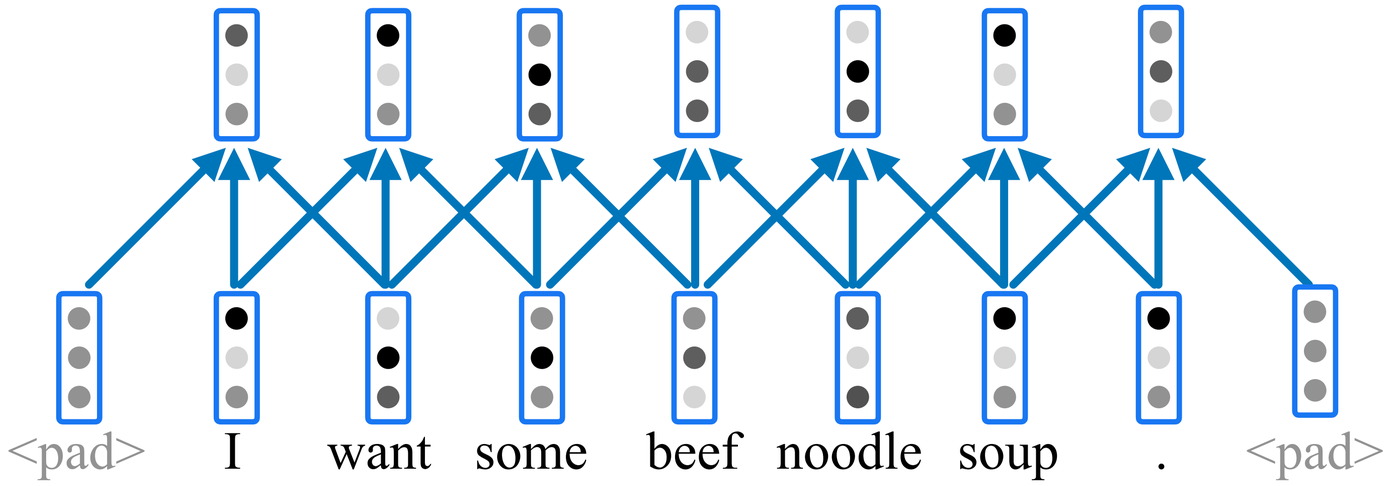
Normal Convolution:



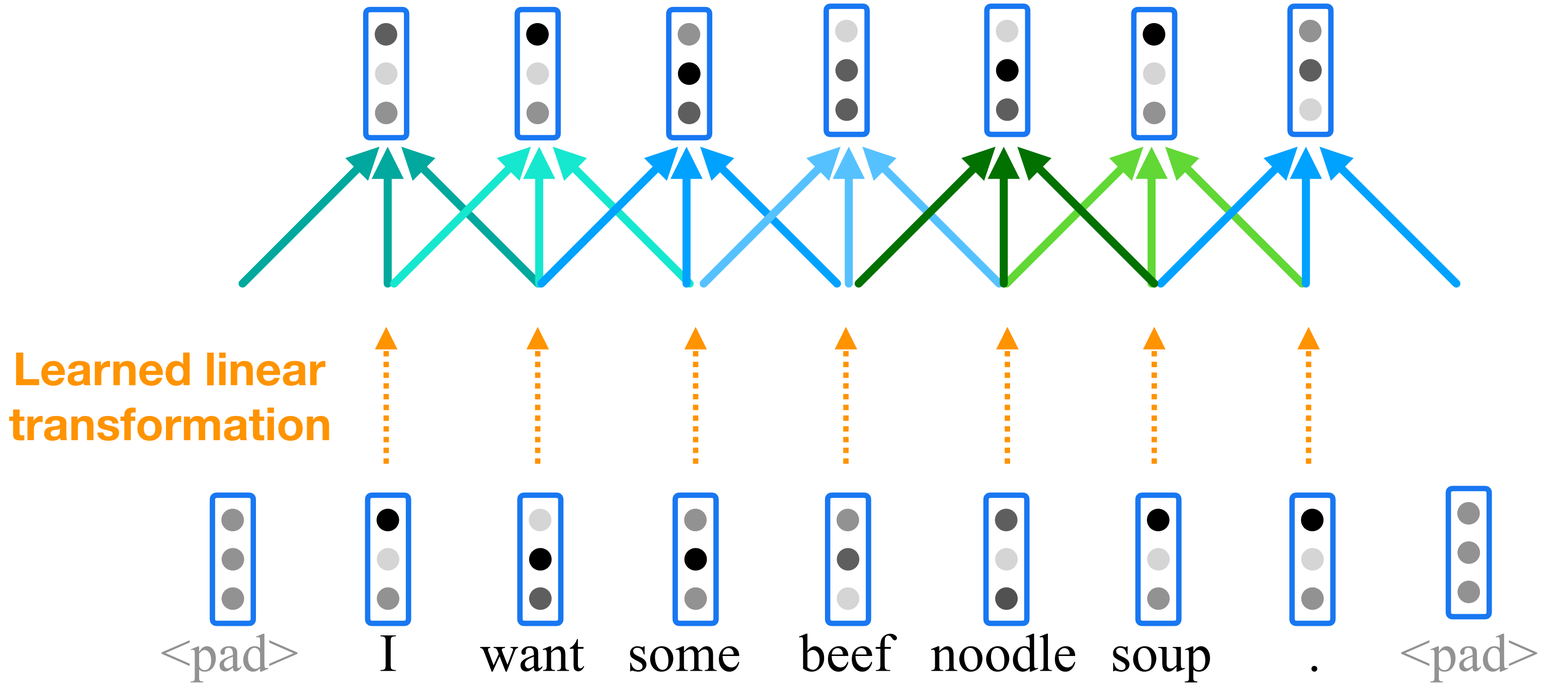
Dynamic Convolution:



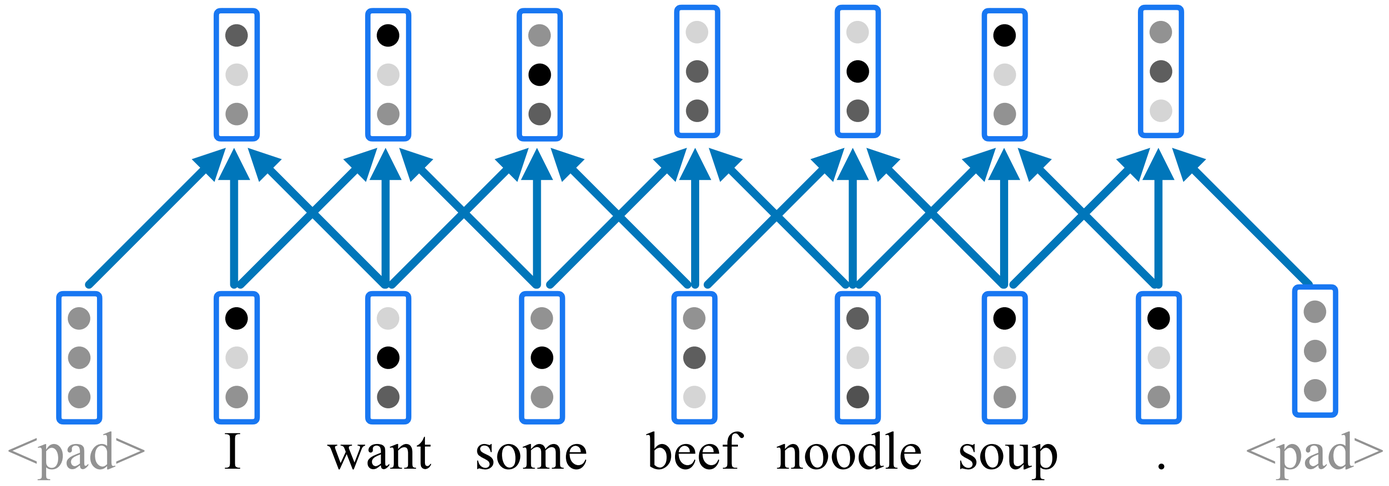
Normal Convolution:



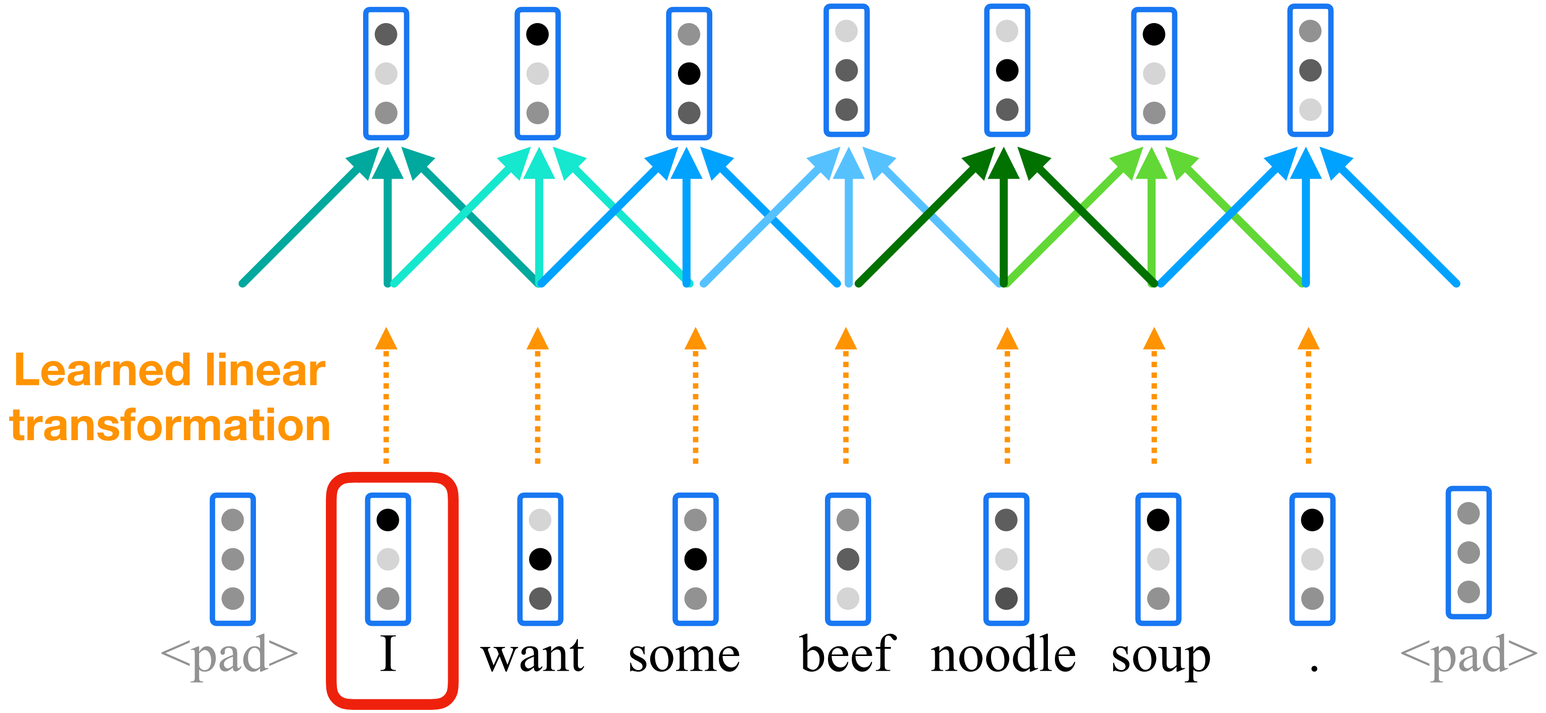
Dynamic Convolution:



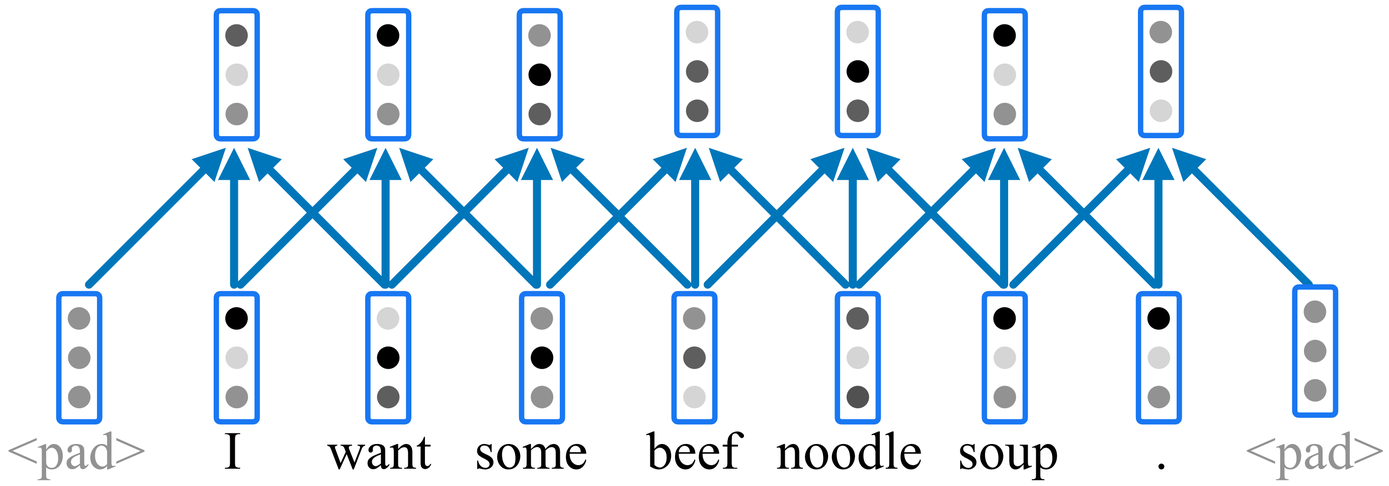
Normal Convolution:



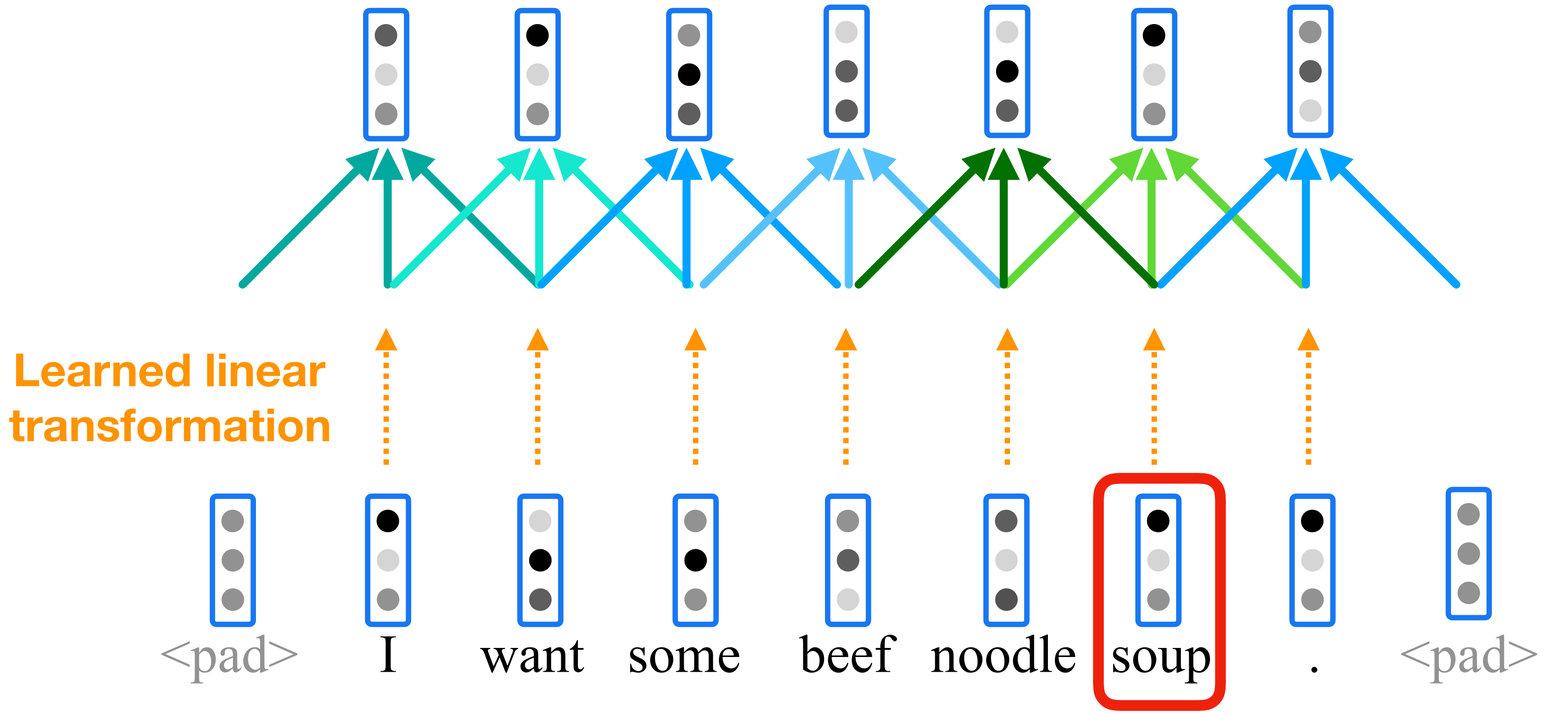
Dynamic Convolution:



Normal Convolution:



Dynamic Convolution:

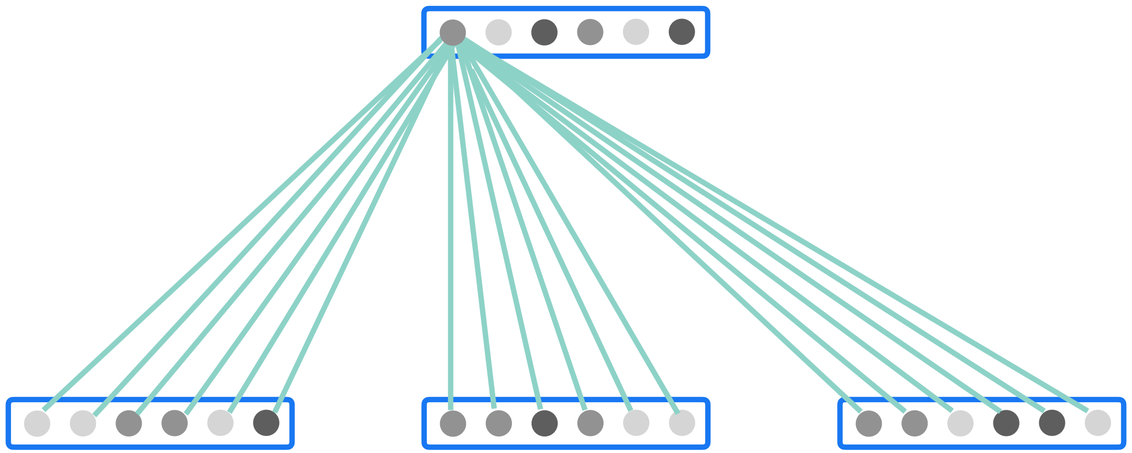


Challenges with Dynamic Convolutions

- Too many weight parameters to predict

# of parameters per layer: $n \times (d \times d \times k) = 128 \times (512 \times 512 \times 3) \approx 100M$

sequence length hidden size kernel size



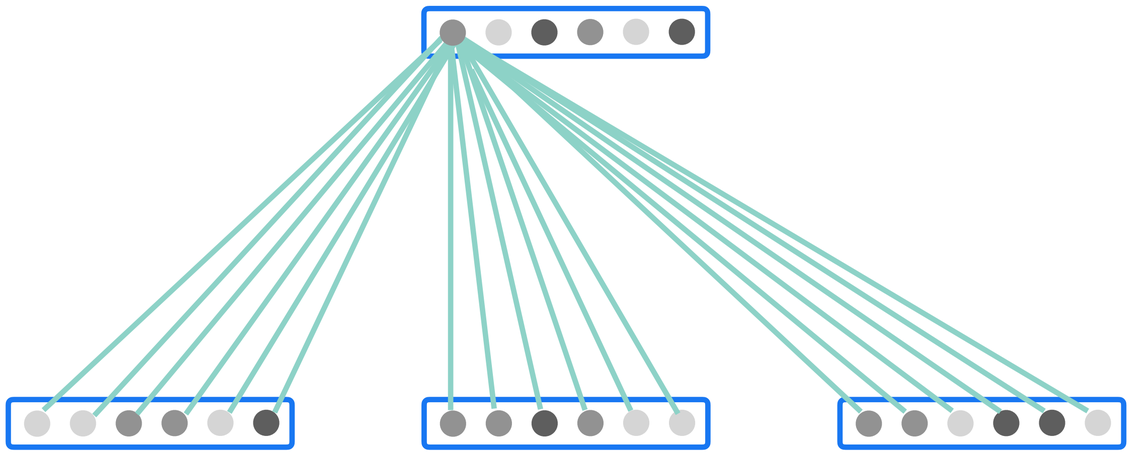
Challenges with Dynamic Convolutions

- Too many weight parameters to predict

# of parameters per layer: $n \times (d \times d \times k) = 128 \times (512 \times 512 \times 3) \approx 100M$

vs. $(512 \times 512 \times 3) \approx 0.8M$

sequence length hidden size kernel size



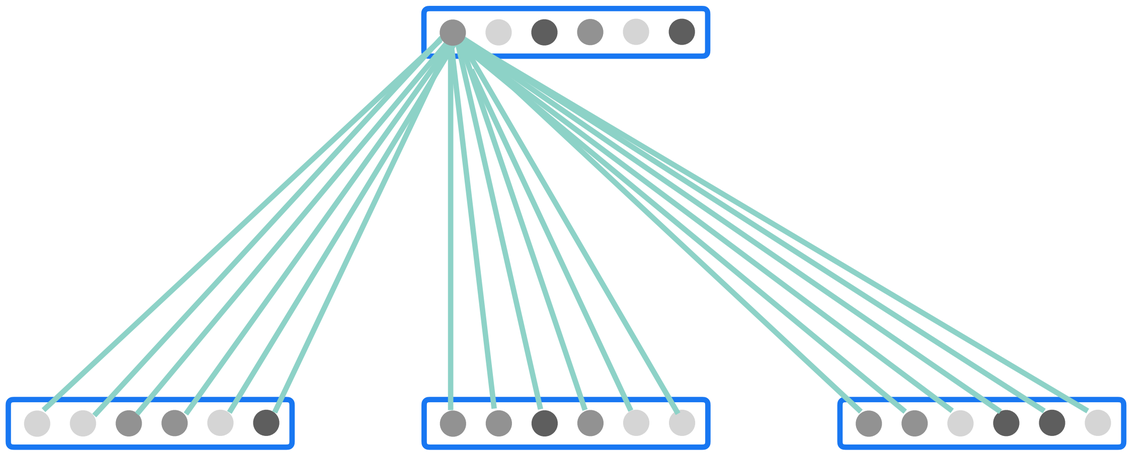
Challenges with Dynamic Convolutions

- Too many weight parameters to predict

# of parameters per layer: $n \times (d \times d \times k) = 128 \times (512 \times 512 \times 3) \approx 100M$

vs. $(512 \times 512 \times 3) \approx 0.8M$

sequence length hidden size kernel size

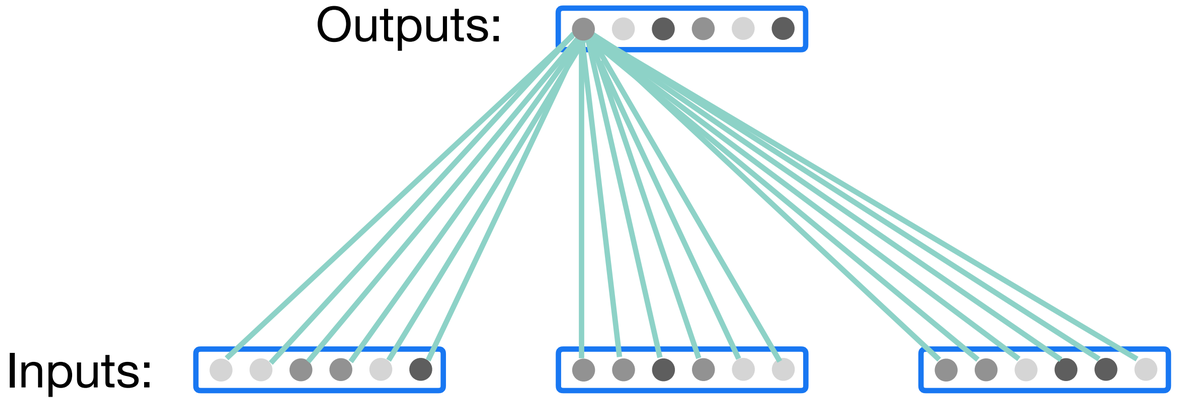


Solution: **Lightweight Convolution**

Lightweight Convolutions reduce parameters

- Convolution: hidden size, kernel size

$$d \times d \times k = 512 \times 512 \times 3 \approx 786K$$



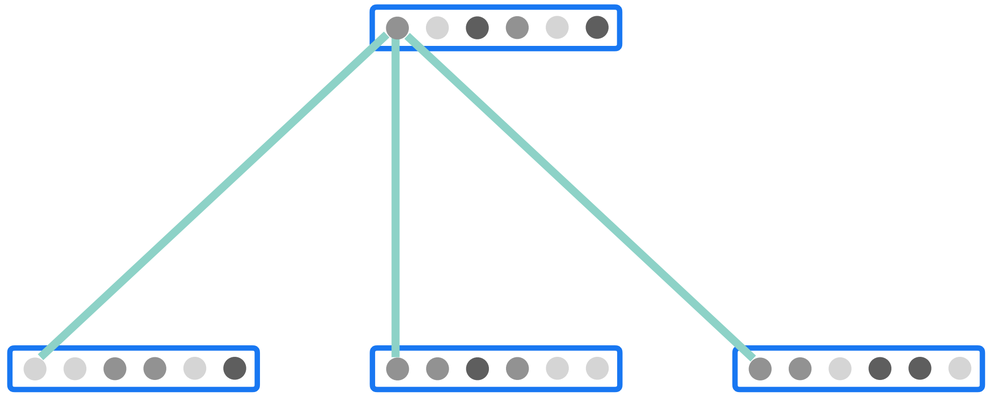
Lightweight Convolutions reduce parameters

- Convolution:

$$d \times d \times k = 512 \times 512 \times 3 \approx 786K$$

- Depthwise Convolution (Sifre, 2014):

$$d \times 1 \times k = 512 \times 1 \times 3 \approx 1.5K$$



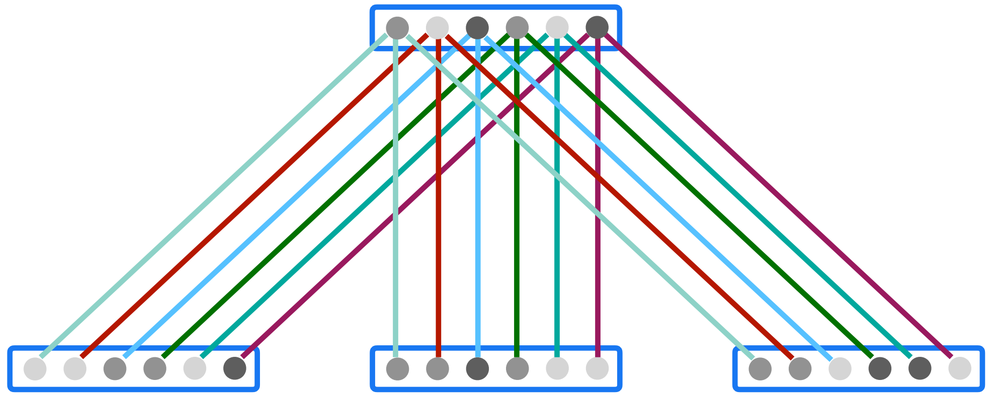
Lightweight Convolutions reduce parameters

- Convolution:

$$d \times d \times k = 512 \times 512 \times 3 \approx 786K$$

- Depthwise Convolution (Sifre, 2014):

$$d \times 1 \times k = 512 \times 1 \times 3 \approx 1.5K$$



One color corresponds to a convolution kernel with 3 parameters

Lightweight Convolutions reduce parameters

- Convolution:

$$d \times d \times k = 512 \times 512 \times 3 \approx 786K$$

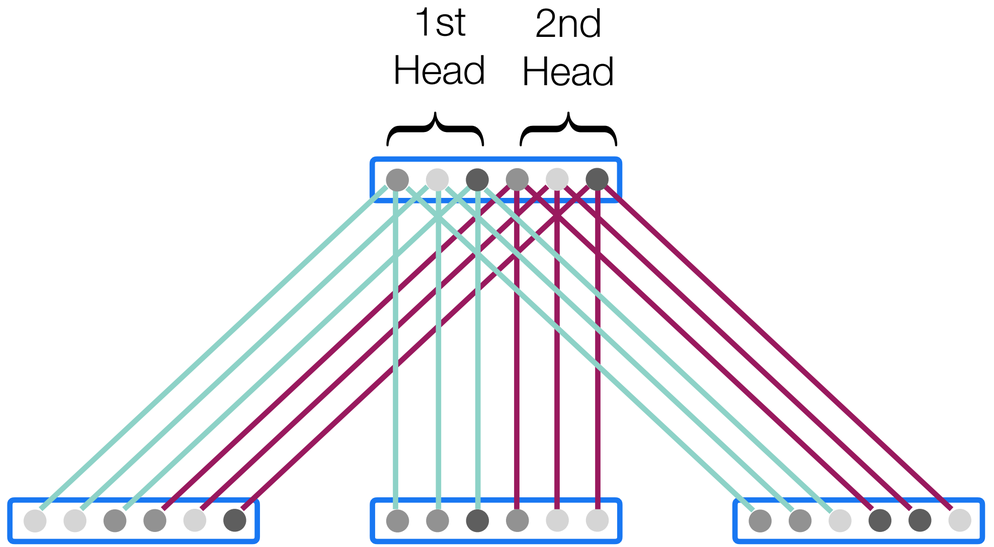
- Depthwise Convolution (Sifre, 2014):

$$d \times 1 \times k = 512 \times 1 \times 3 \approx 1.5K$$

- Lightweight Convolution

$$H \times 1 \times k = 4 \times 1 \times 3 = 12$$

H: # of heads



One color corresponds to a convolution kernel with 3 parameters

Lightweight Convolutions reduce parameters

- Convolution:

$$d \times d \times k = 512 \times 512 \times 3 \approx 786K$$

- Depthwise Convolution (Sifre, 2014):

$$d \times 1 \times k = 512 \times 1 \times 3 \approx 1.5K$$

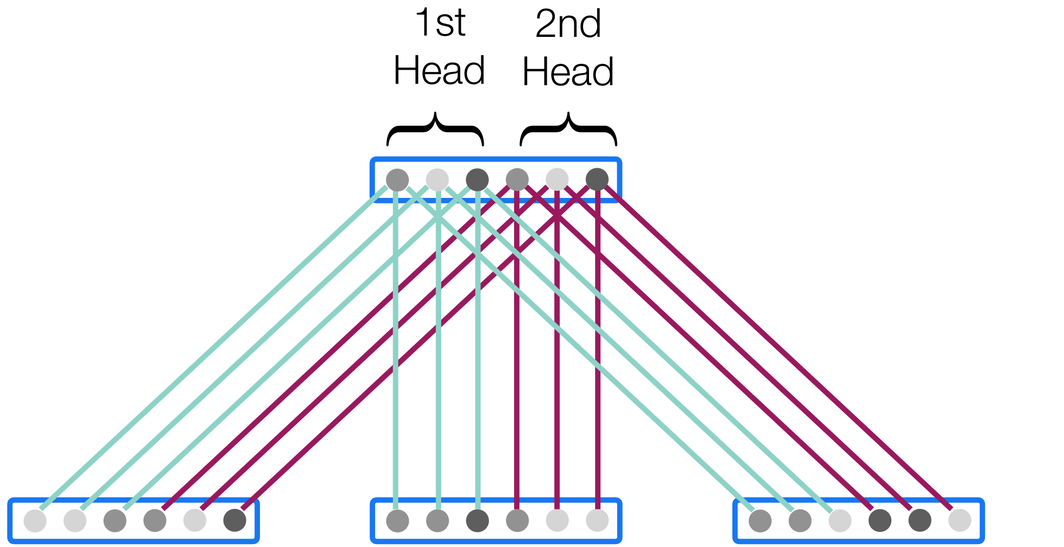
- Lightweight Convolution

$$H \times 1 \times k = 4 \times 1 \times 3 = 12$$

- Dynamic Convolution

$$n \times H \times 1 \times k = 4 \times 1 \times 3 = n \times 12$$

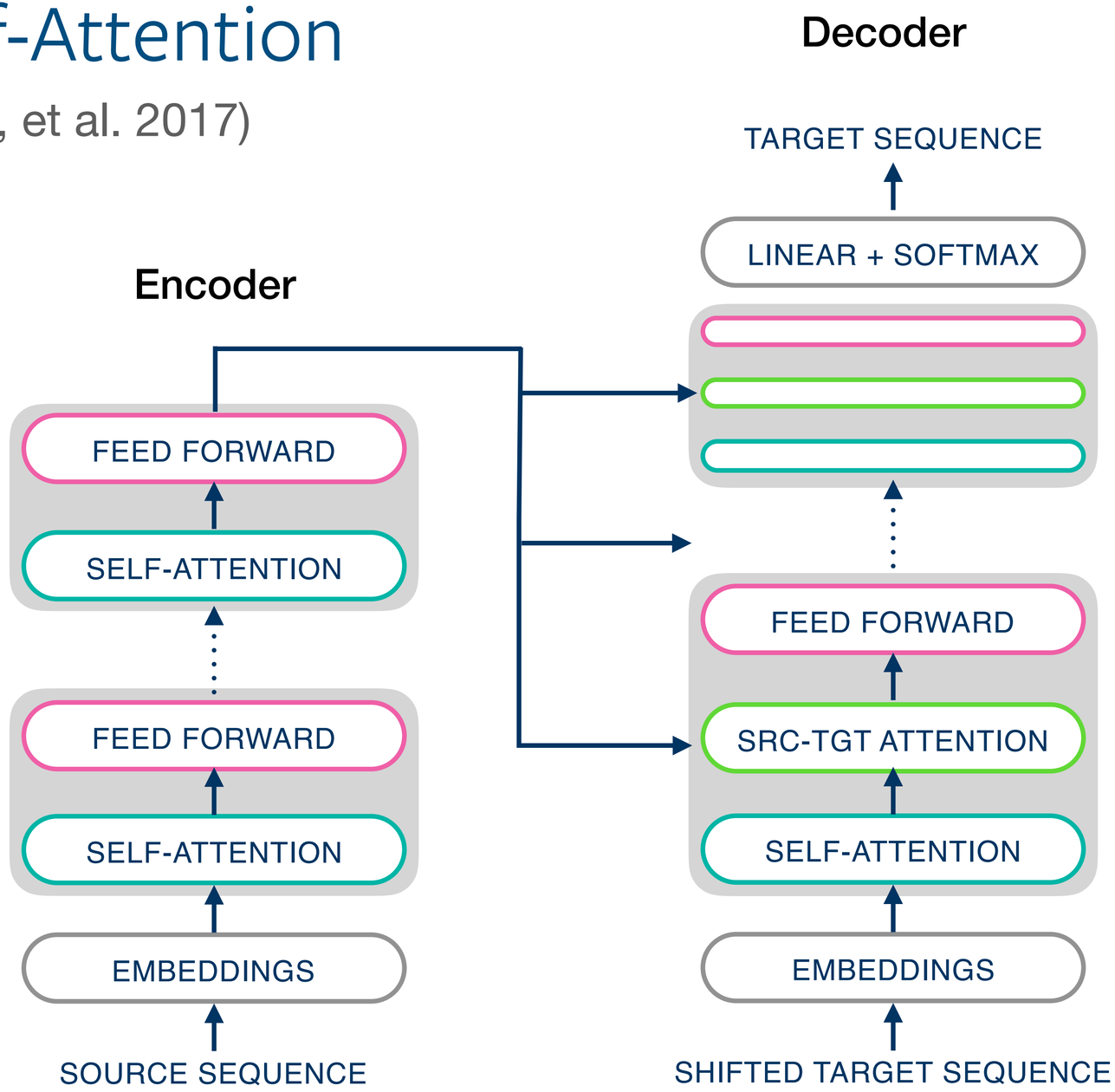
n: # of time steps



One color corresponds to a convolution kernel with 3 parameters

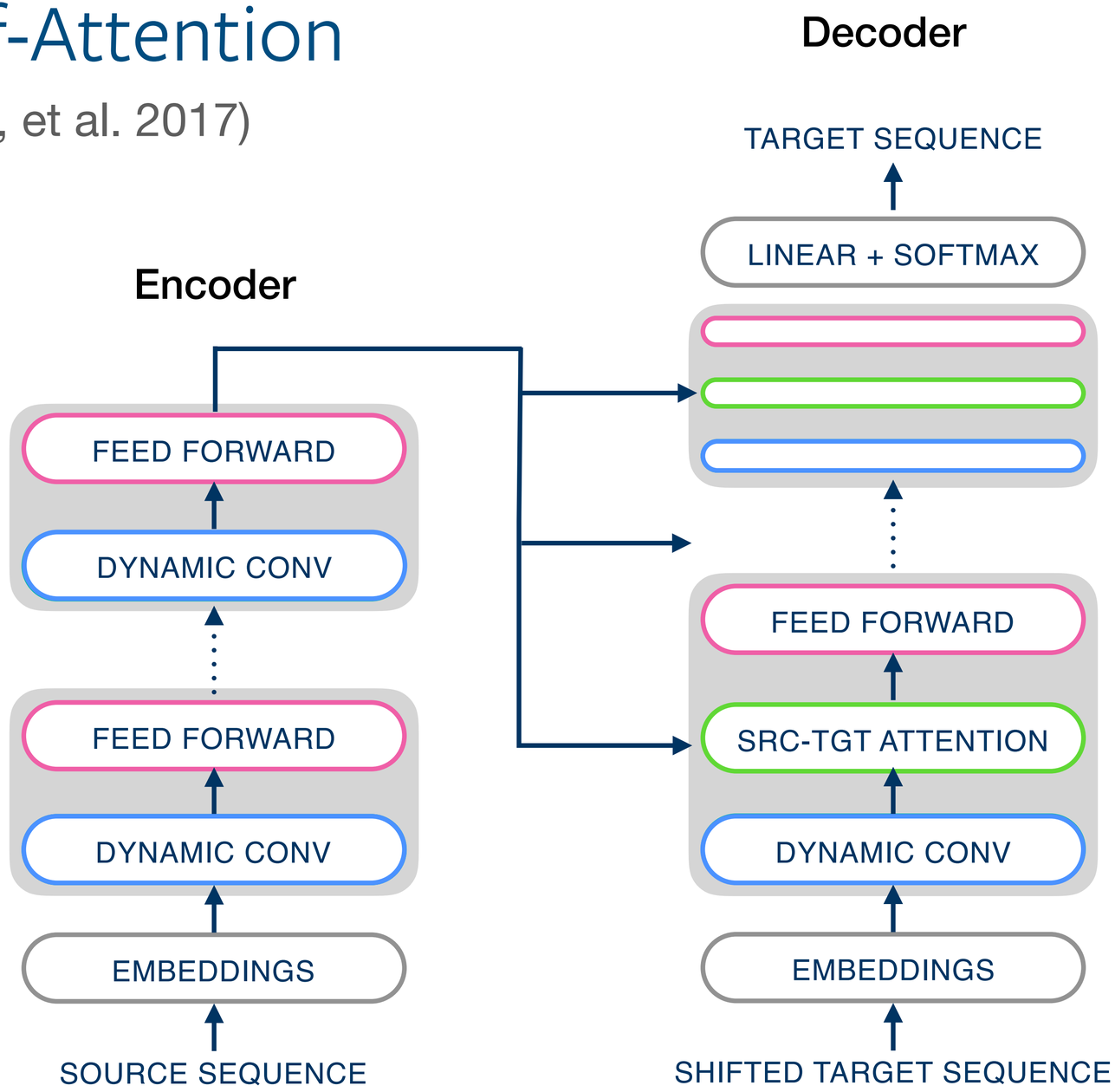
Replacing Self-Attention

Transformer (Vaswani, et al. 2017)



Replacing Self-Attention

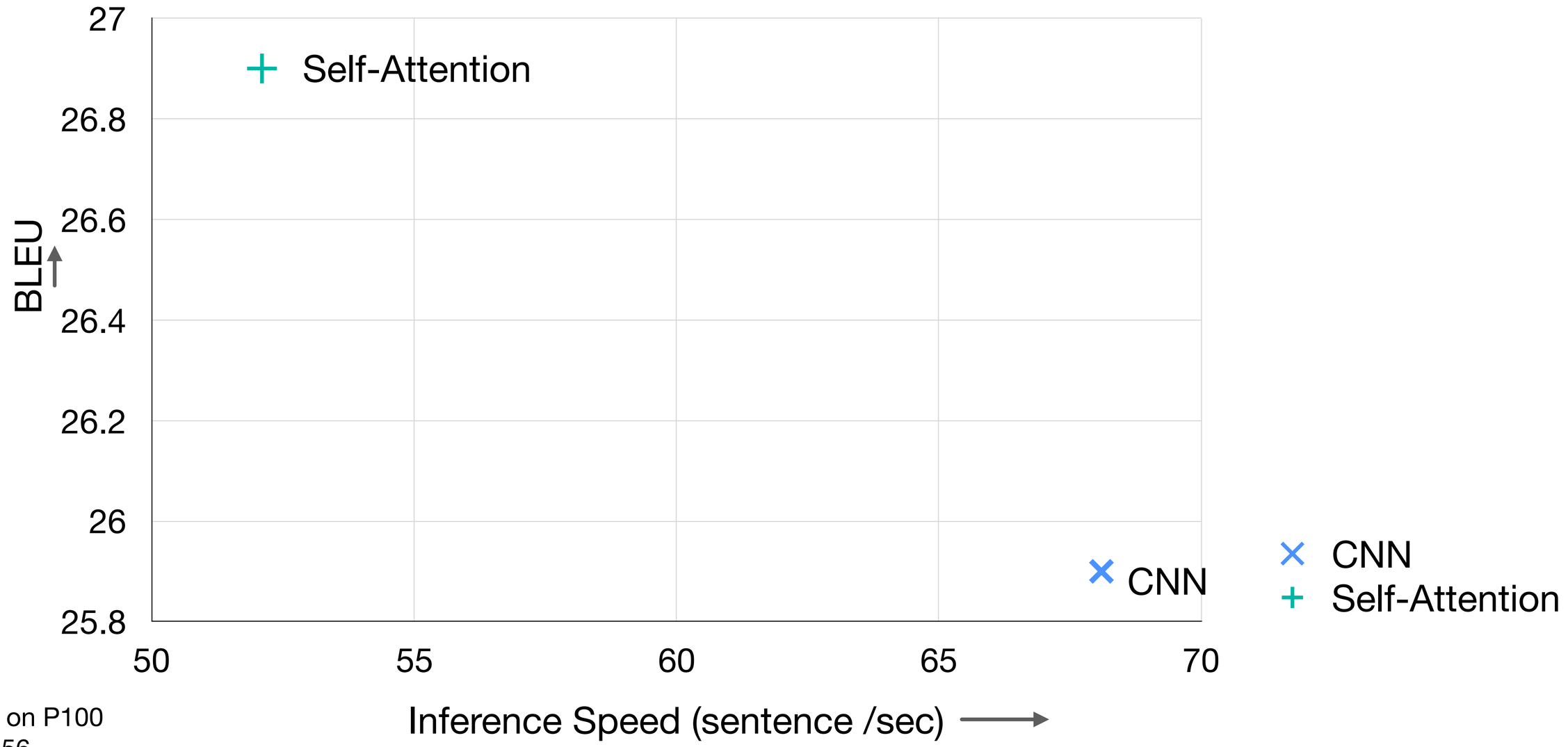
Transformer (Vaswani, et al. 2017)



Experiments

Performance vs. Speed

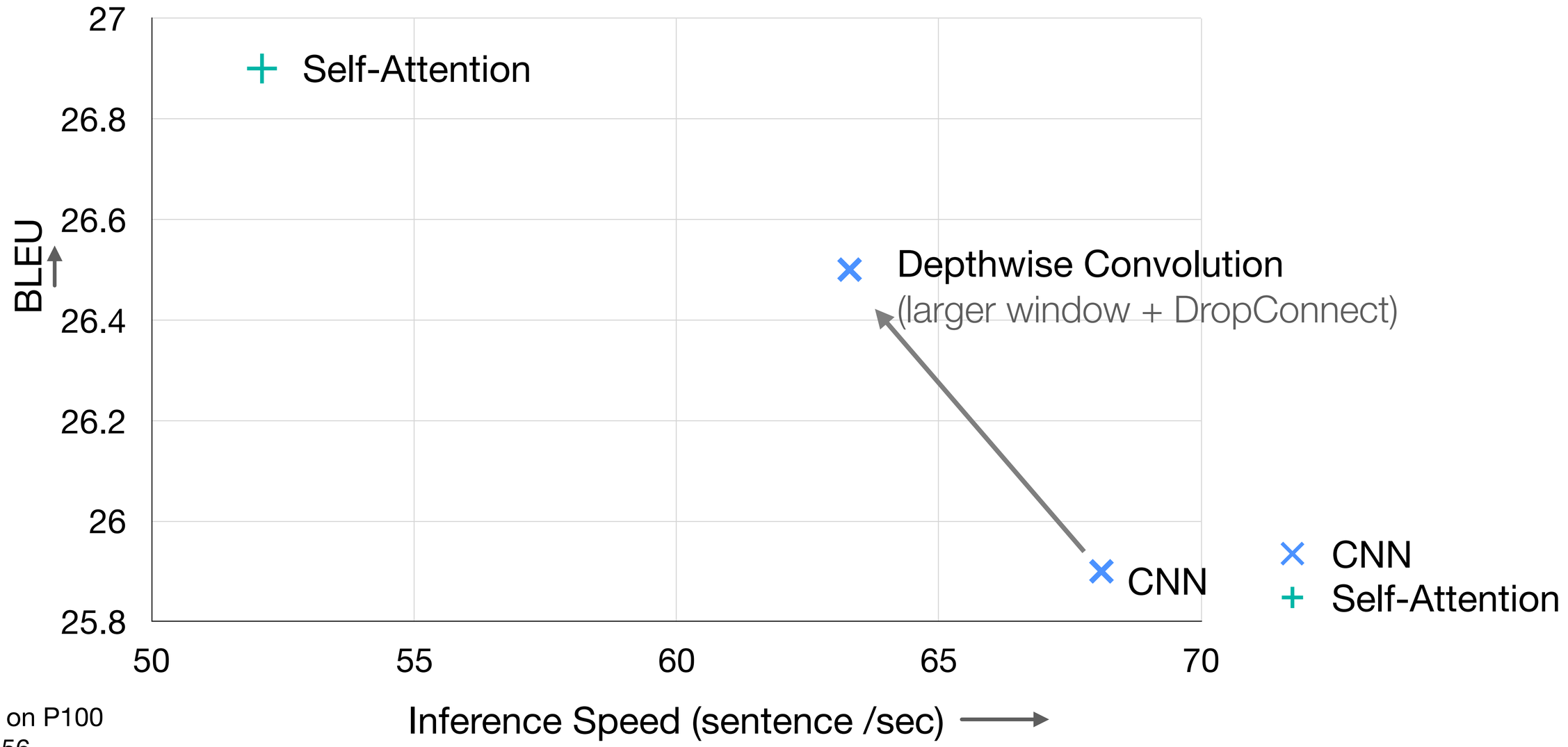
On WMT newstest2013
En-De Translation



Measured on P100
with bsz 256

Performance vs. Speed

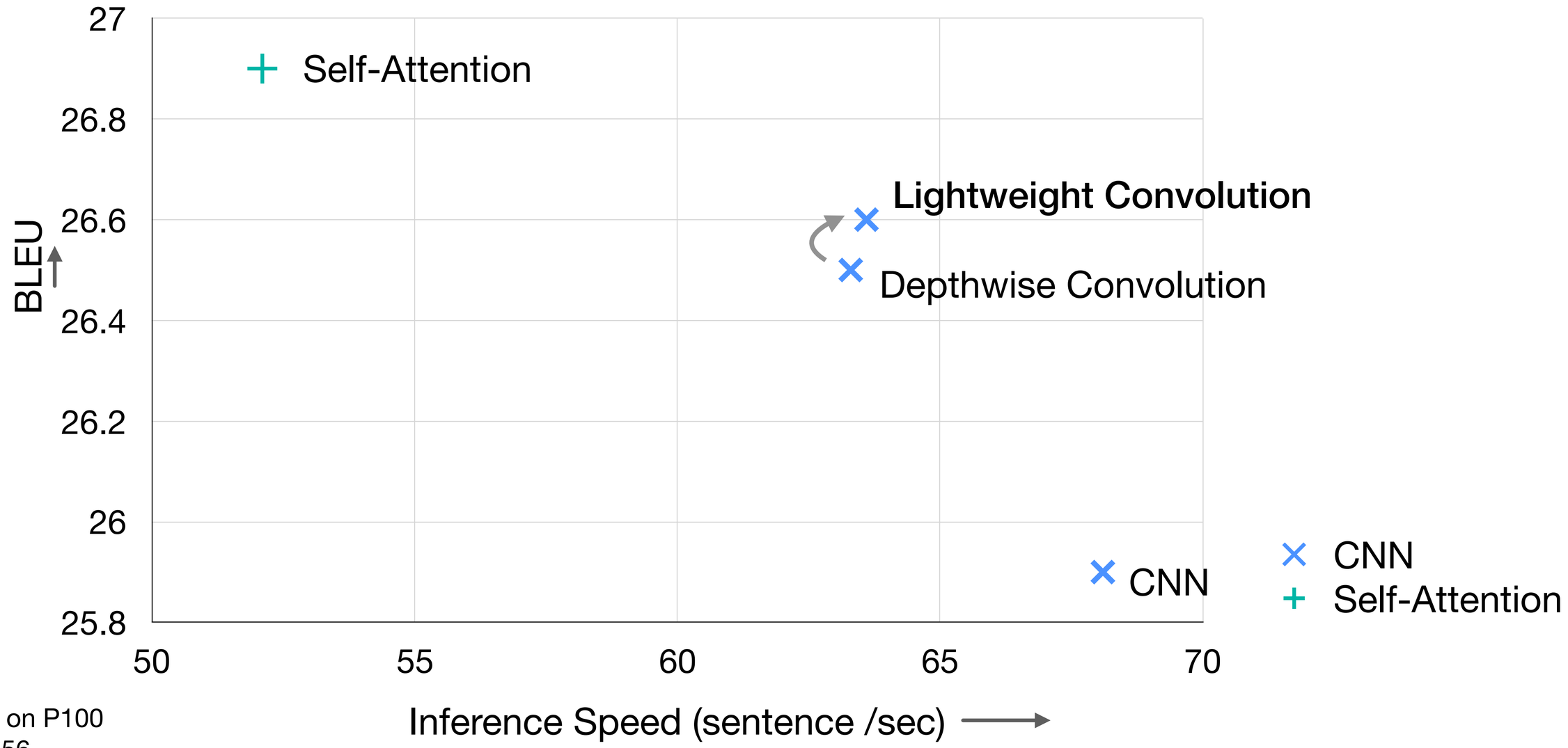
On WMT newstest2013
En-De Translation



Measured on P100
with bsz 256

Performance vs. Speed

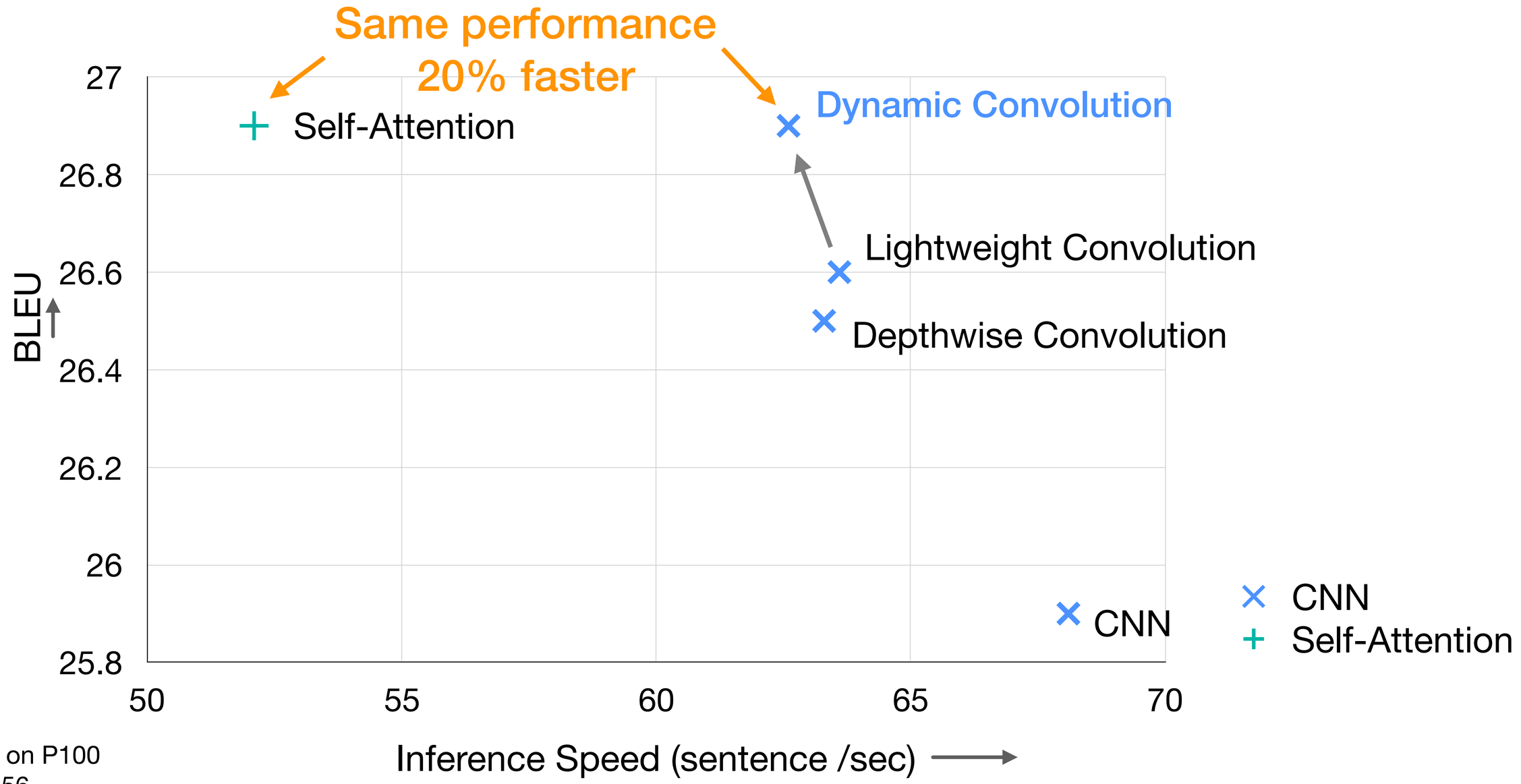
On WMT newstest2013
En-De Translation



Measured on P100
with bsz 256

Performance vs. Speed

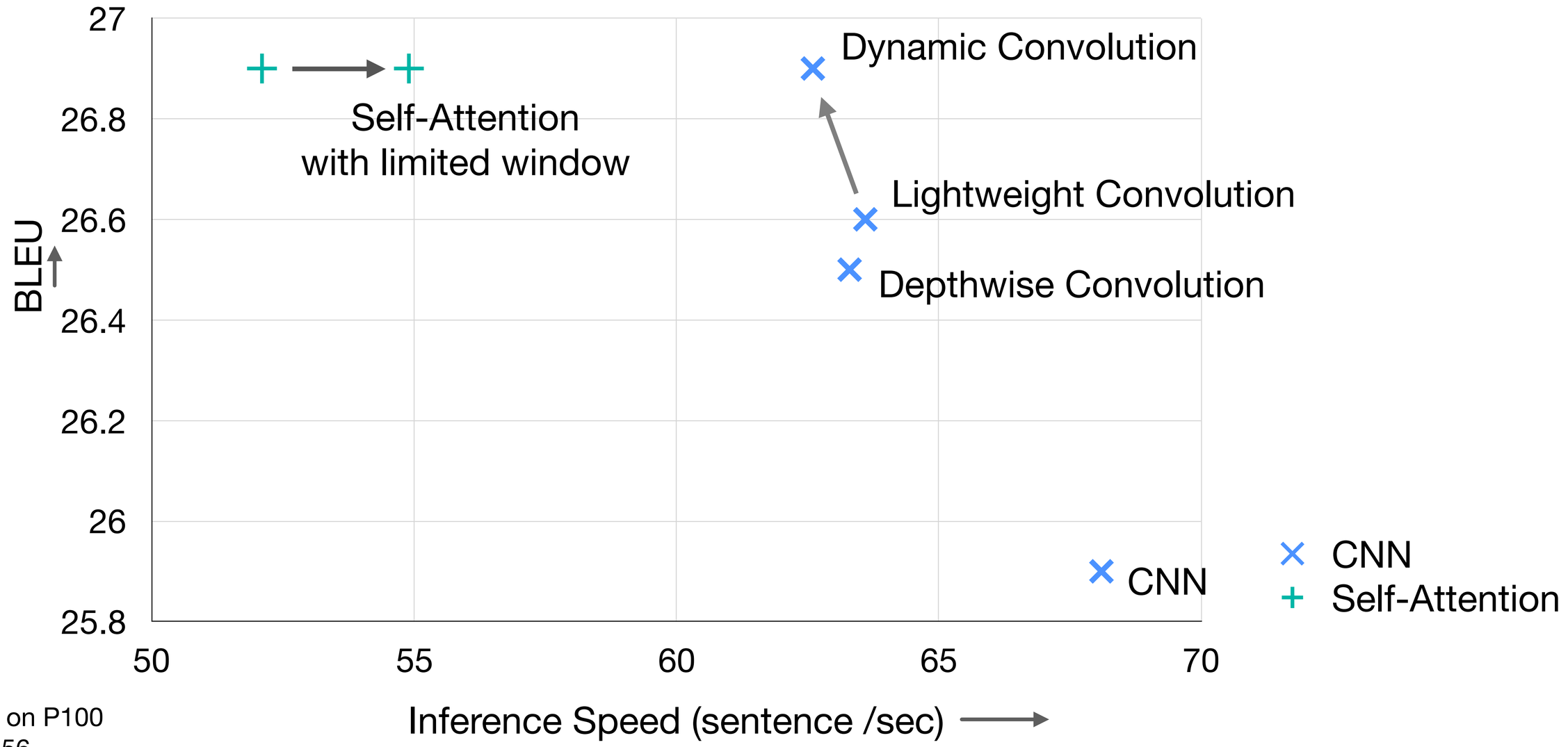
On WMT newstest2013
En-De Translation



Measured on P100
with bsz 256

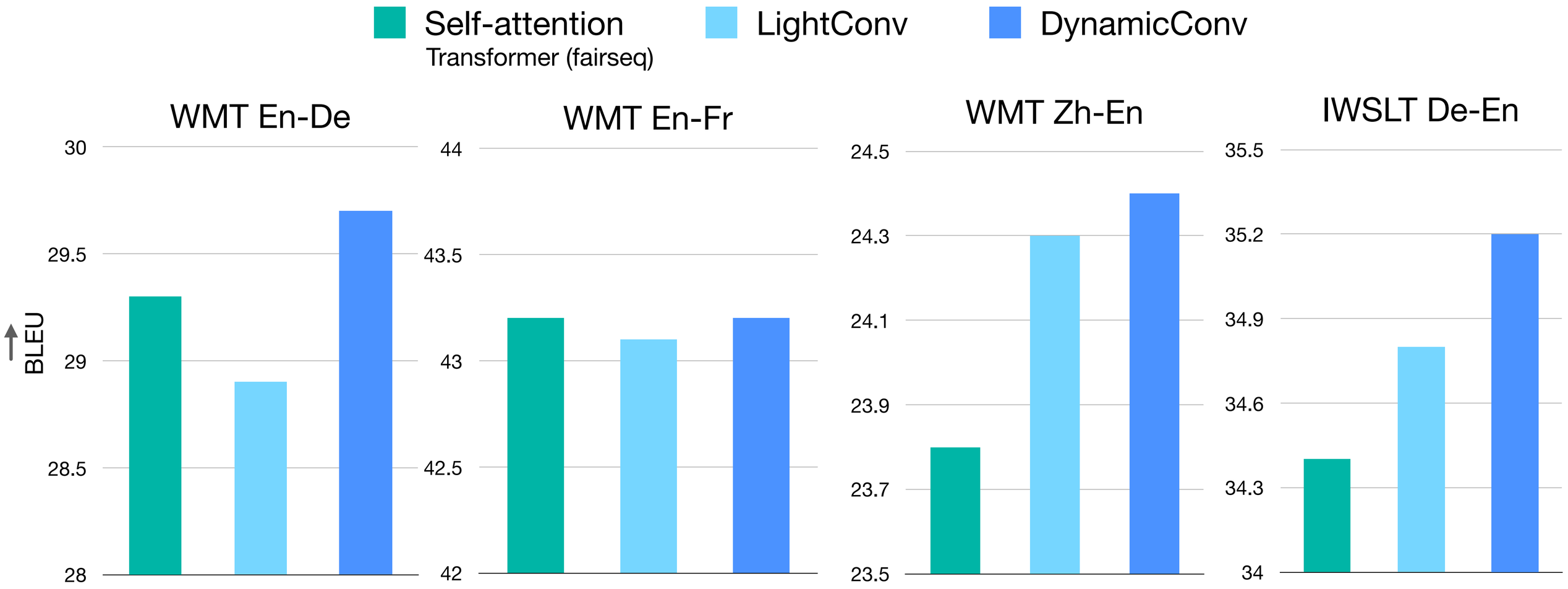
Performance vs. Speed

On WMT newstest2013
En-De Translation

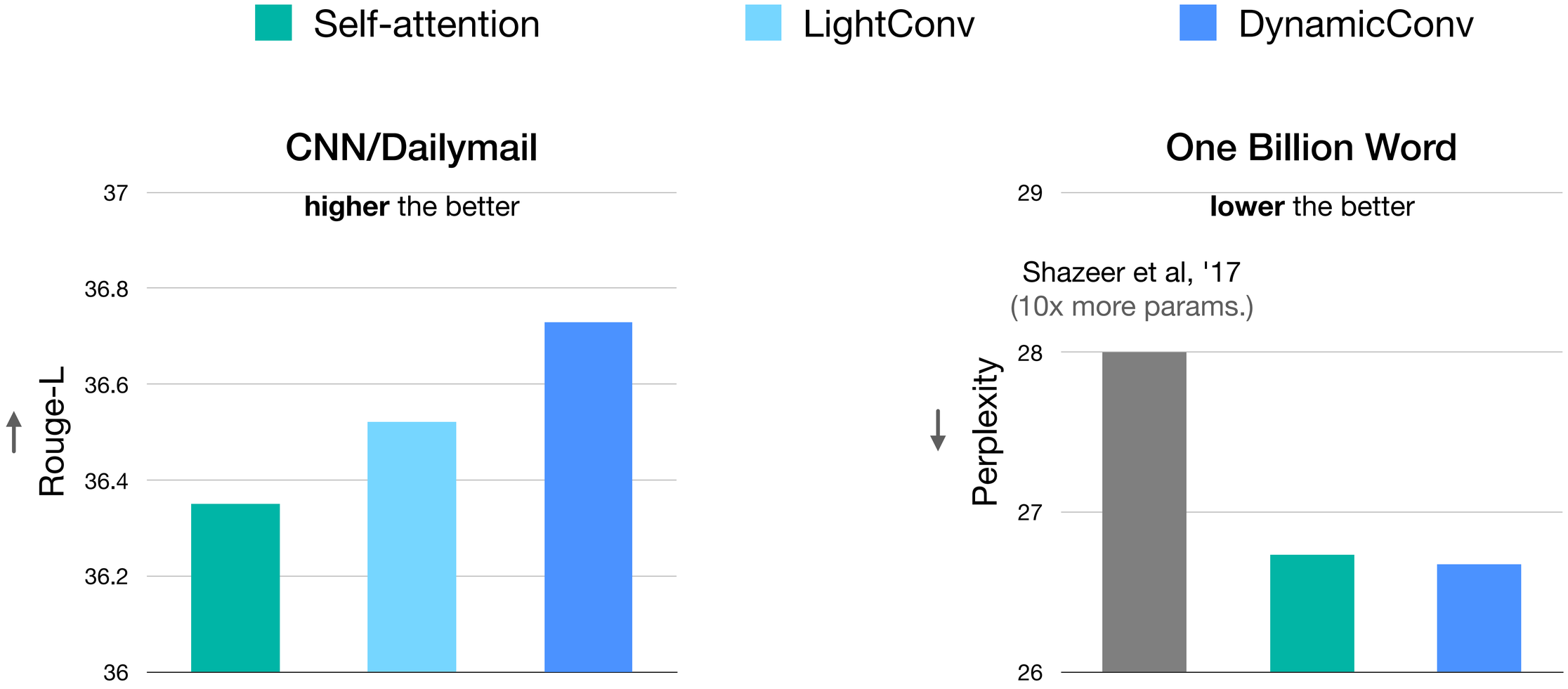


Measured on P100
with bsz 256

Machine Translation
(BLEU score, the higher the better)



Text Summarization & Language Modeling



Summary

- Local information (limited window) is sufficient for several NLP tasks
- Dynamic Convolution: context-specific kernels
- Lightweight Convolution: fewer convolution weights still work well

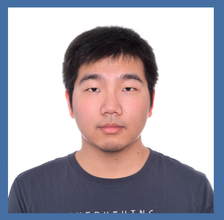
Code available in

FAIRSEQ

Questions


Alexei Baevski


Angela Fan


Jiatao Gu

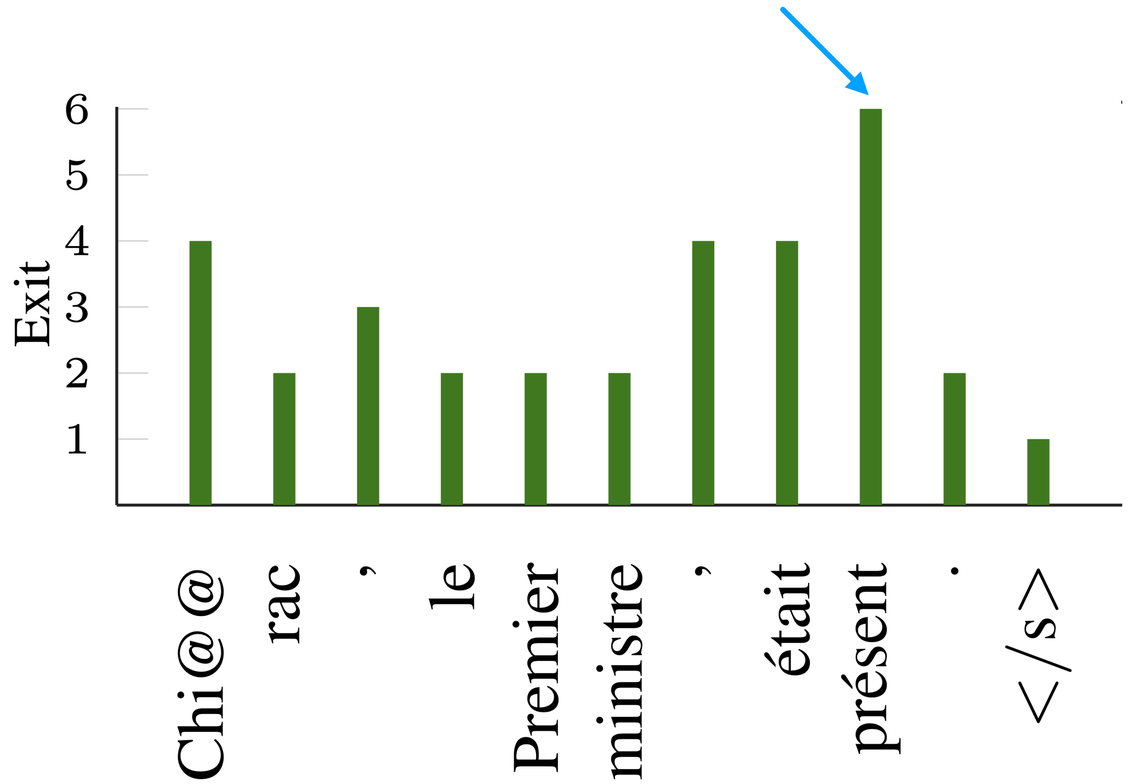

Edouard Grave


Felix Wu


Maha Elbayad


Yann Dauphin

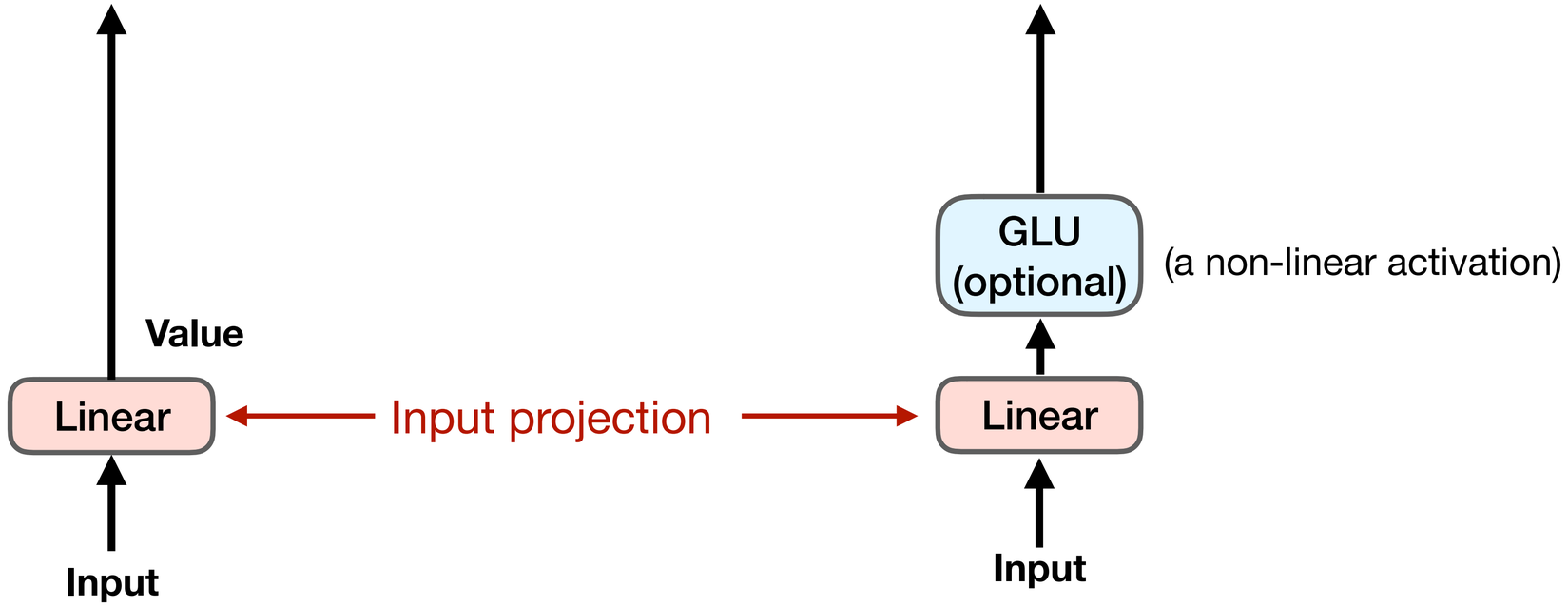
Backup slides



(a) **Src:** Chi@@rac , the Prime Minister , was there .
Ref: Chi@@rac , Premier ministre , est là .

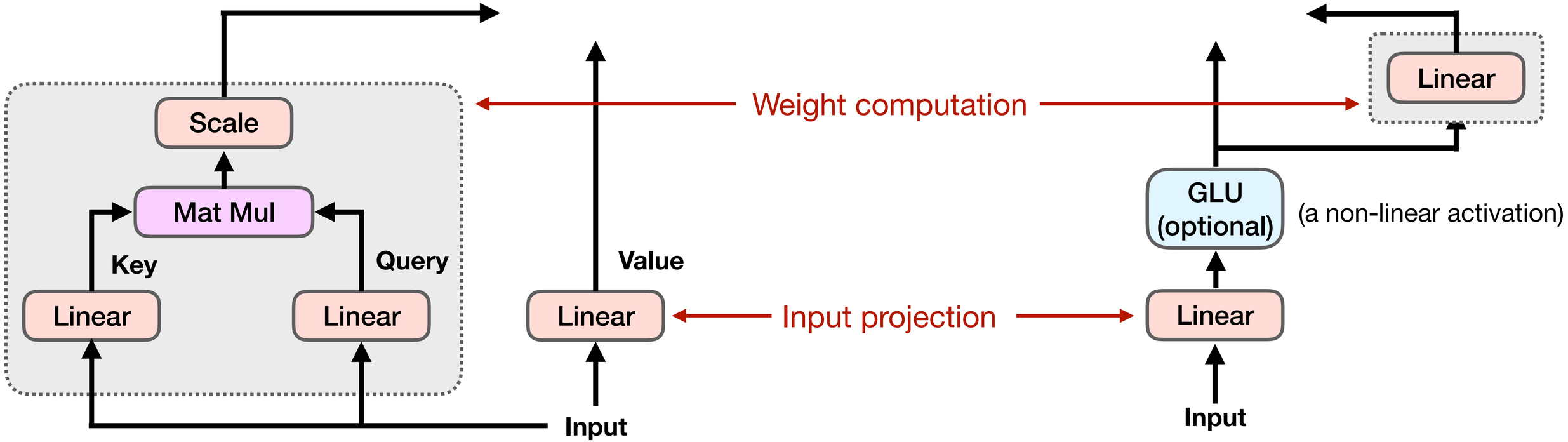
Self-Attention Block

Dynamic Convolution Block



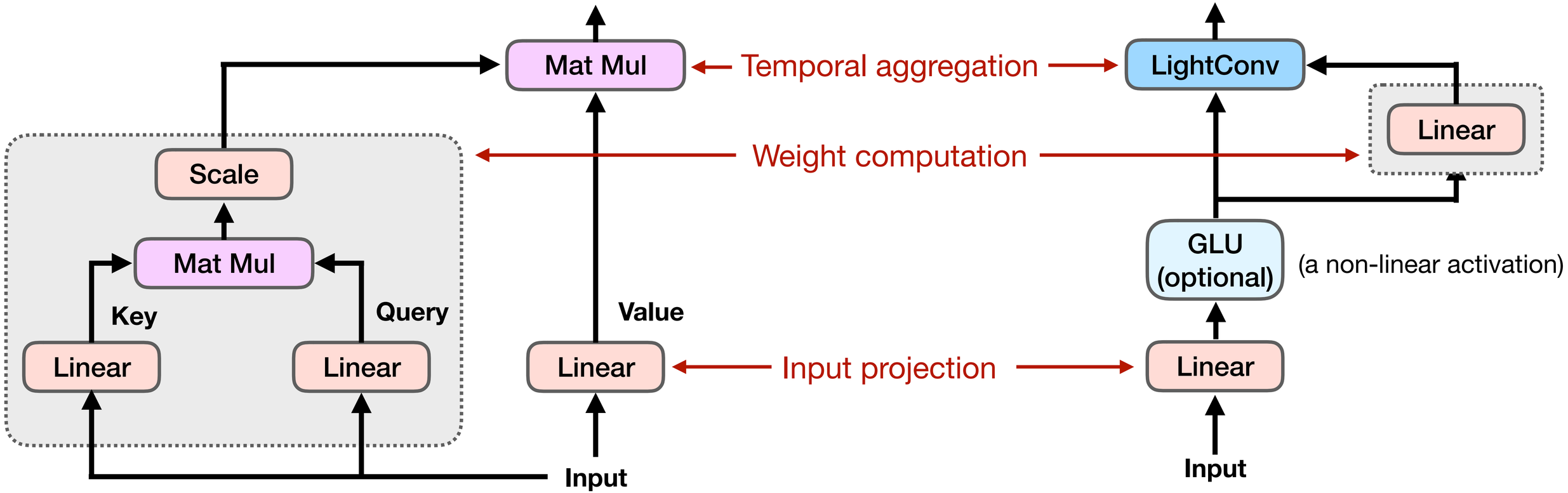
Self-Attention Block

Dynamic Convolution Block



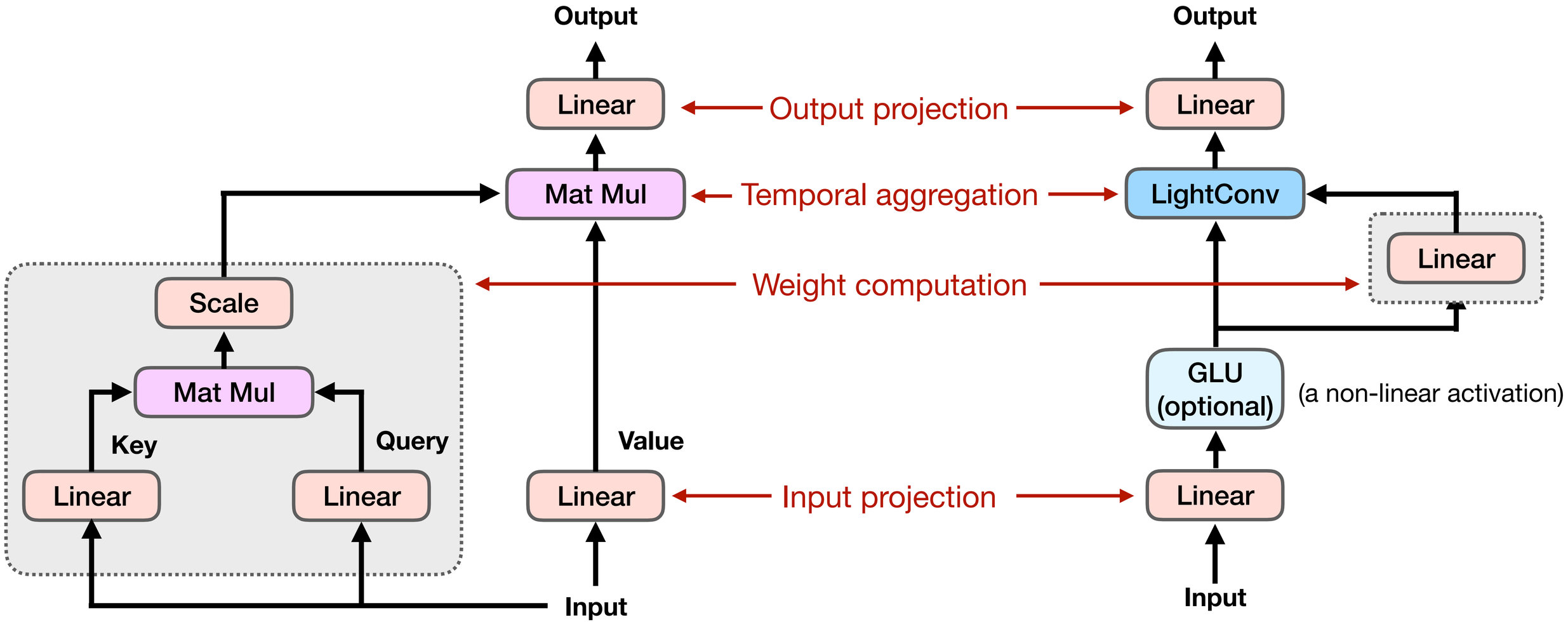
Self-Attention Block

Dynamic Convolution Block



Self-Attention Block

Dynamic Convolution Block



How important is softmax-normalization?

Method	BLEU
W (No normalization)	diverges
$\mathrm{softmax}(W)$	26.9 ± 0.2
$\sigma(W)$	26.6 ± 0.3
$\tanh(W)$	25.6 ± 0.2
$\frac{W}{\|W\|_1+\epsilon}$	diverges
$\frac{W}{\|W\|_2+\epsilon}$	26.8 ± 0.2
$\mathrm{power}(W, 2)$	diverges
$\mathrm{abs}(W)$	diverges
$\frac{\mathrm{abs}(W)}{\|W\|_1+\epsilon}$	diverges
$\frac{\mathrm{abs}(W)}{\|W\|_2+\epsilon}$	26.7 ± 0.2

Table 6: Alternatives to softmax-normalization in DynamicConv on WMT English-German newstest2013 ($\epsilon = 10^{-6}$).