Applied Deep Learning

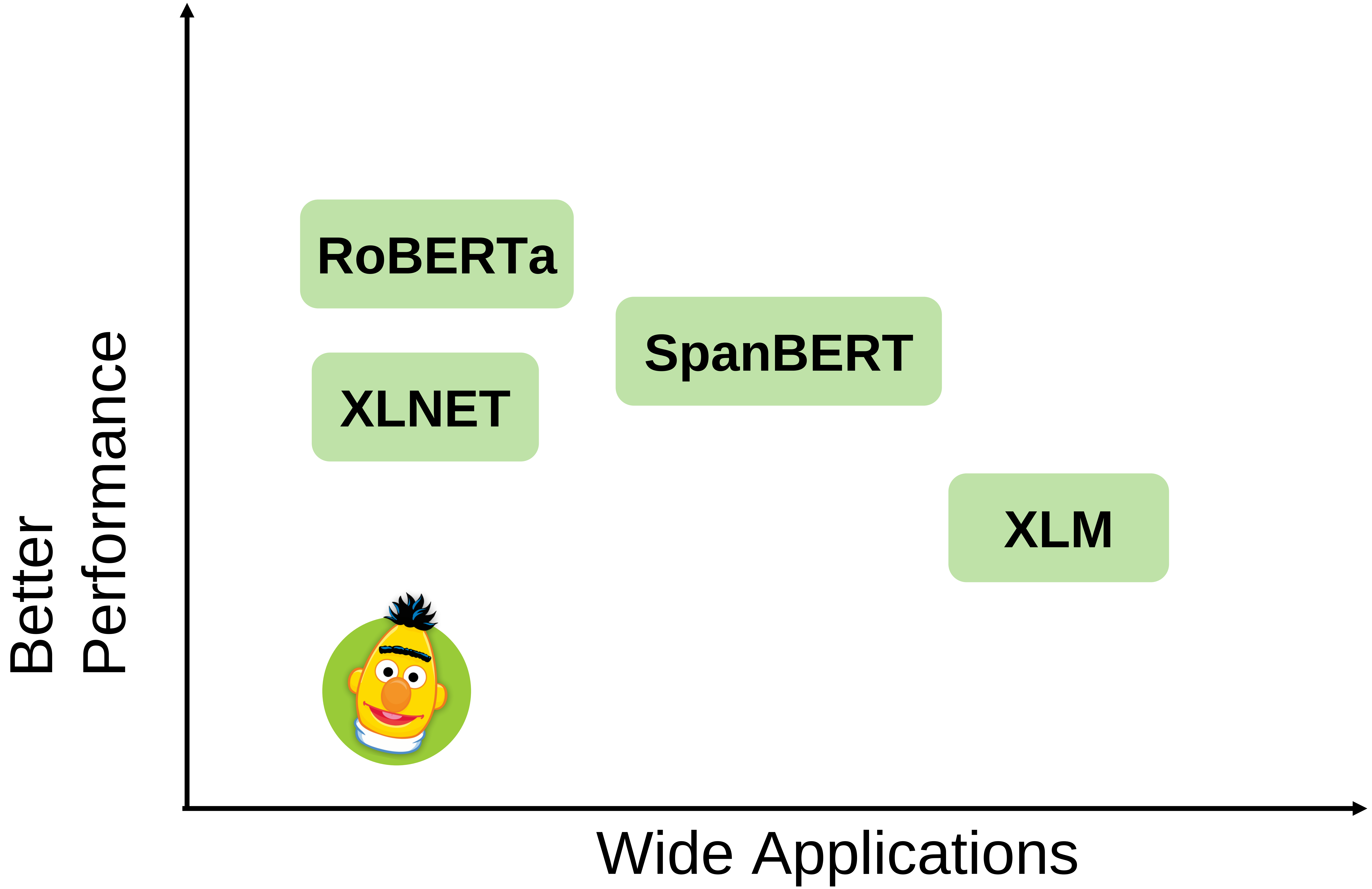
More BERT

April 14th, 2020 http://adl.miulab.tw

國立臺灣大學 National Taiwan University

2

Beyond BERT



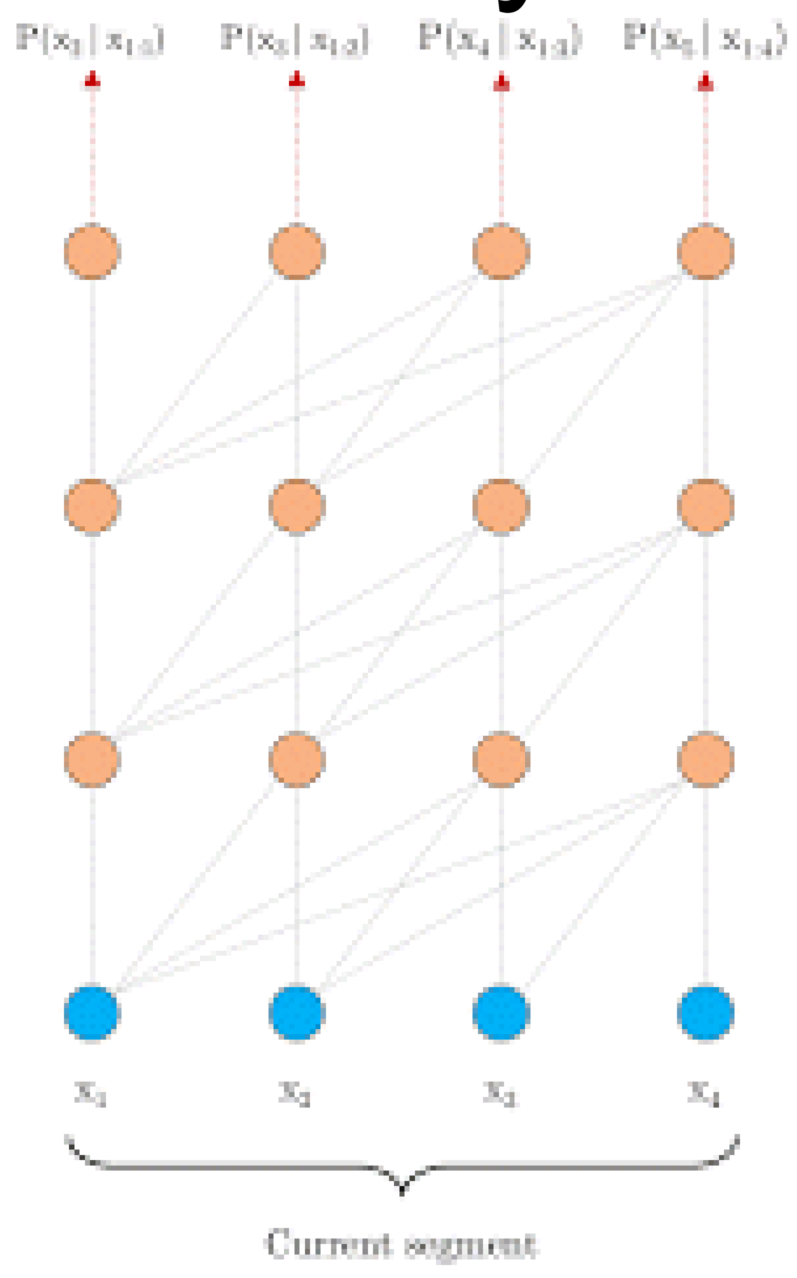
3

Transformer-XL

(Dai et al, 2019)

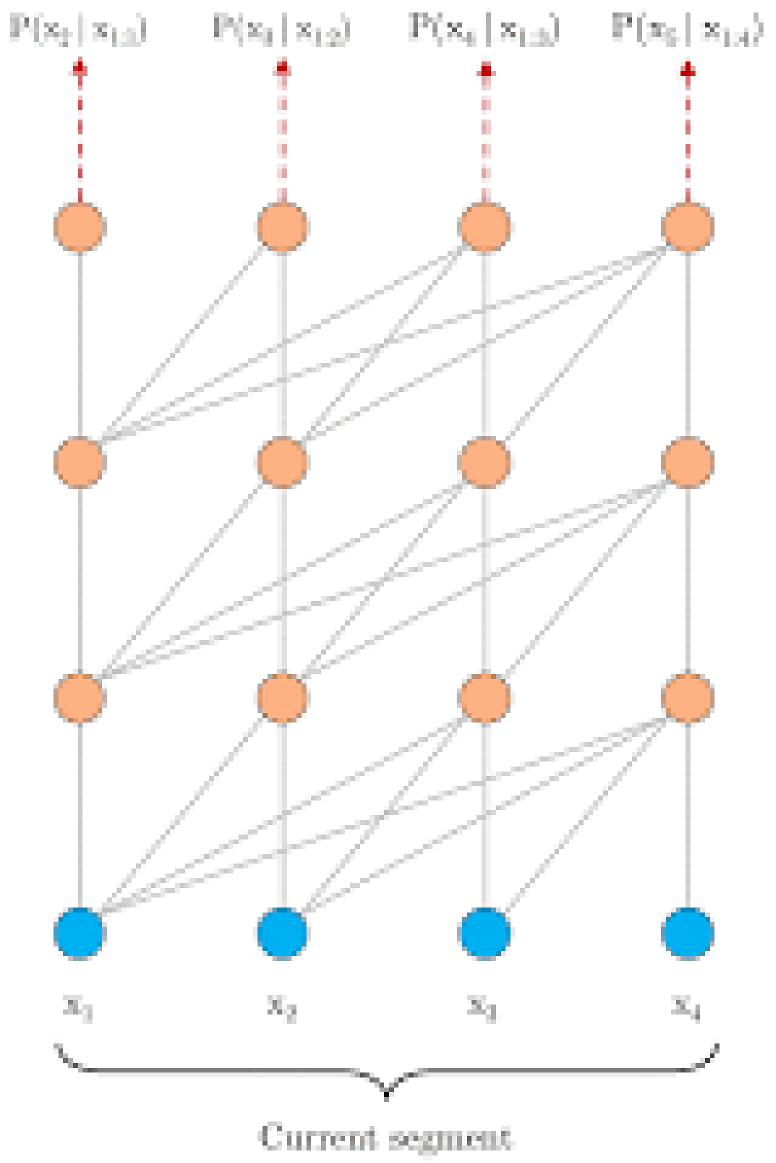
4 Transformer

- Issue: context fragmentation
  - Long dependency: unable to model dependencies longer than a fixed length
  - Inefficient optimization: ignore sentence boundaries
    - particularly troublesome even for short sequences



Transformer-XL (extra-long)

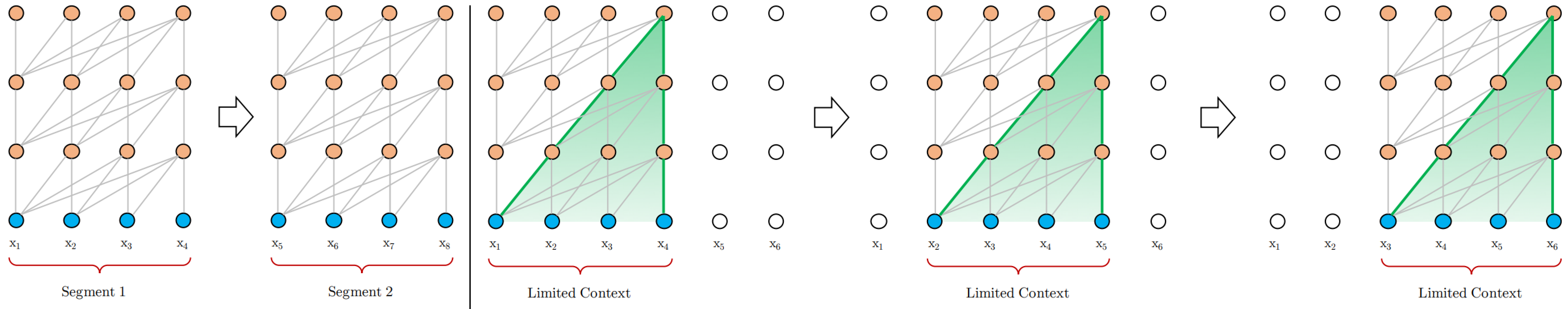
- Idea: segment-level recurrence
  - Previous segment embeddings are **fixed** and **cached** to be reused when training the next segment
  - → increases the largest dependency length by N times (N: network depth)



resolve the context fragmentation issue and makes the dependency longer

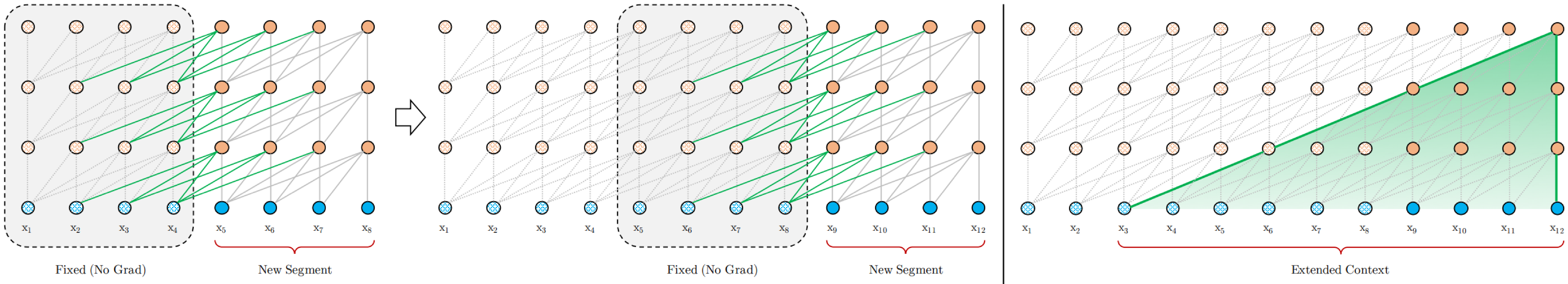
6 State Reuse for Segment-Level Recurrence

- Vanilla



(a) Train phase. (b) Evaluation phase.

- State Reuse



(a) Training phase. (b) Evaluation phase.

7

Incoherent Positional Encoding

- Issue: naively applying segment-level recurrence can’t work
  - positional encodings are *incoherent* when reusing

$$[0, 1, 2, 3] \longrightarrow [0, 1, 2, 3, 0, 1, 2, 3]$$

Relative Positional Encoding

- Idea: relative positional encoding
  - learnable embeddings → fixed embeddings with learnable transformations
    - the query vector is the same for all query positions
    - the attentive bias towards different words should remain the same

$$\mathbf{A}_{i,j}^{\text{abs}} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

absolute (b), absolute (d)

$$\mathbf{A}_{i,j}^{\text{rel}} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)} + \underbrace{u^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{v^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

content (a), position (b), content (c), position (d)

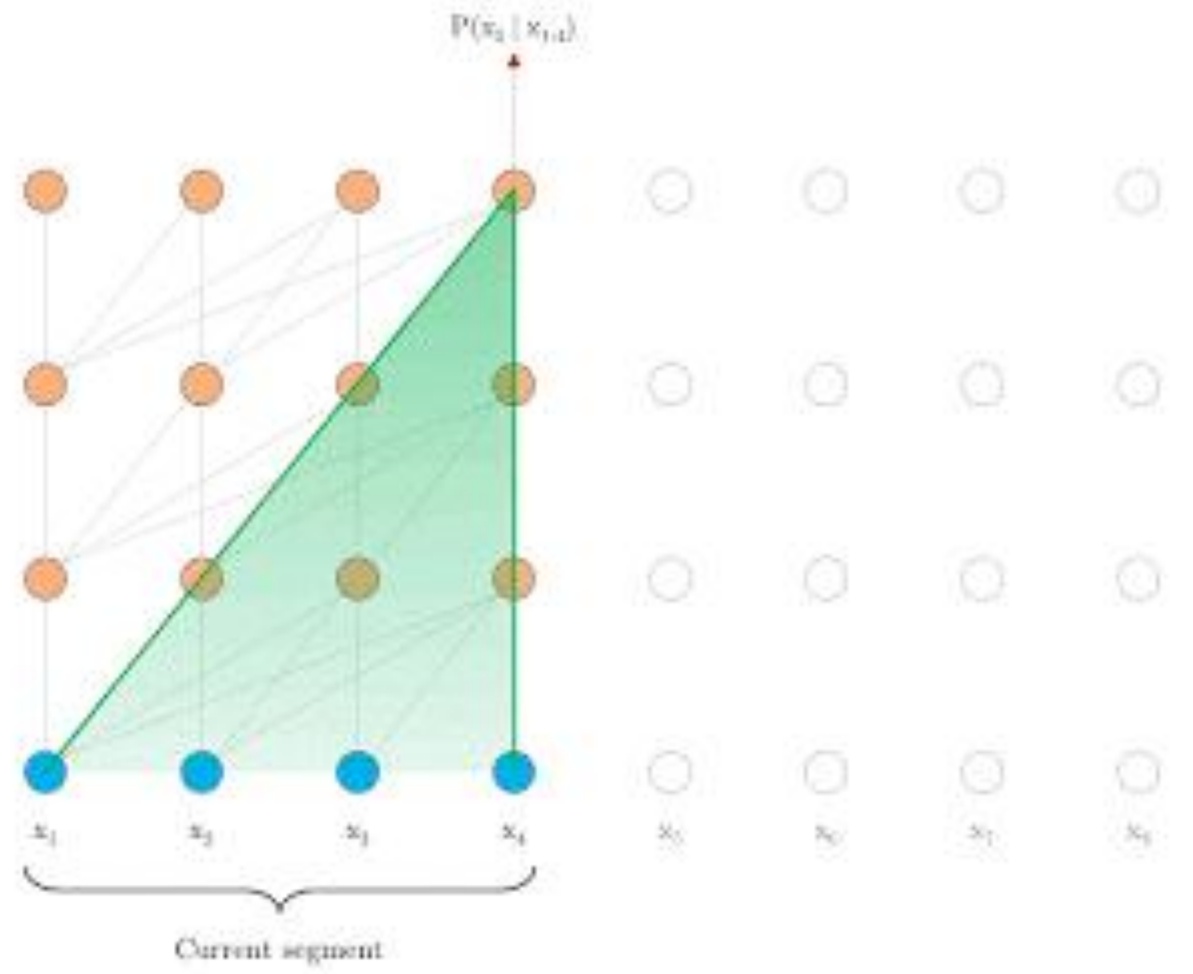
relative (b), trainable parameters (c), trainable parameters (d), relative (d)

much longer effective contexts than a vanilla model during evaluation

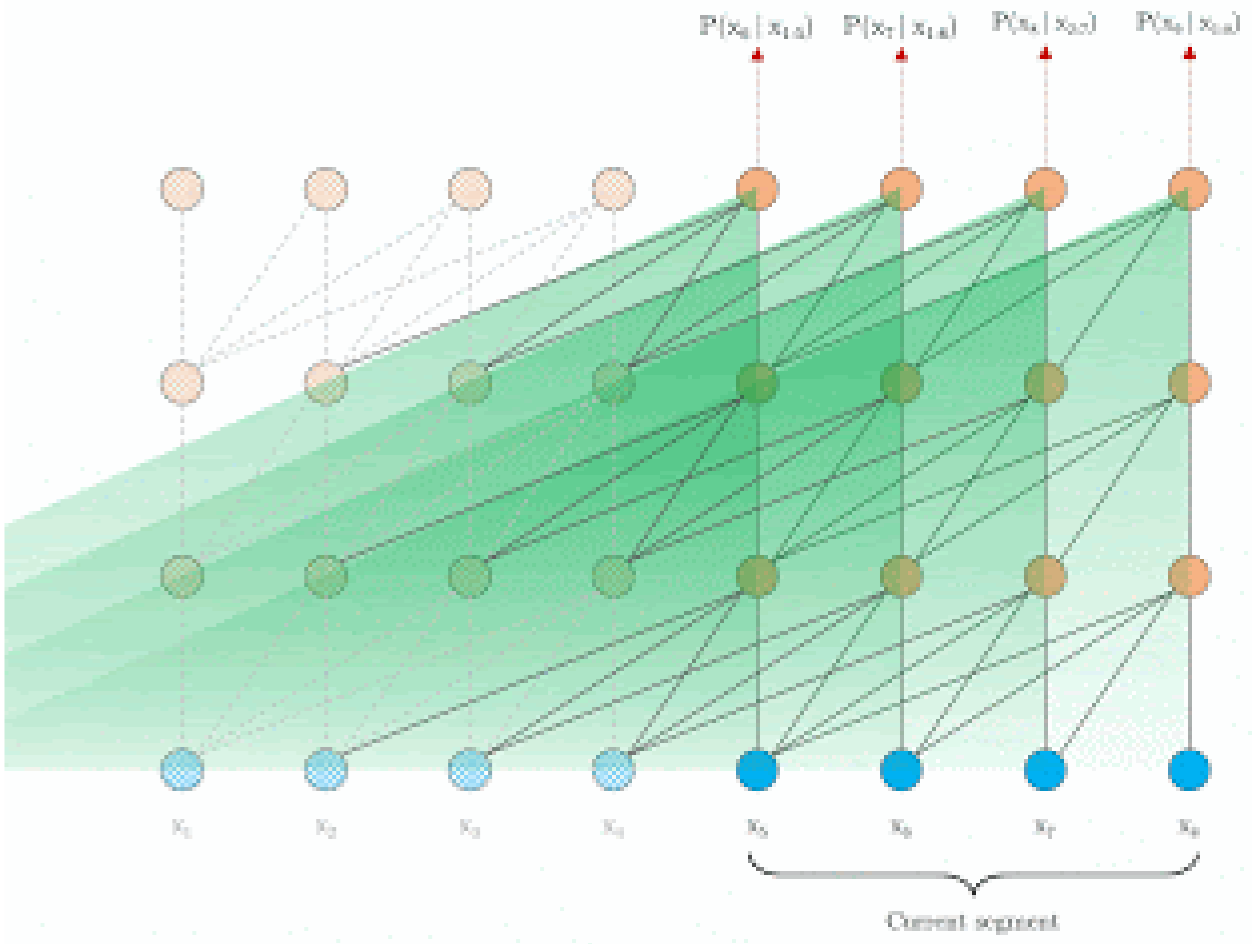
better generalizability to longer sequences

Segment-Level Recurrence in Inference

- Vanilla



- State Reuse



Contributions

- Longer context dependency
  - Learn dependency about 80% longer than RNNs and
    450% longer than vanilla Transformers
  - Better perplexity on long sequences
  - Better perplexity on short sequences by addressing the fragmentation issue
- Speed increase
  - Process new segments without recomputation
  - Achieve up to 1,800+ times faster than a vanilla Transformer during evaluation on LM tasks

11

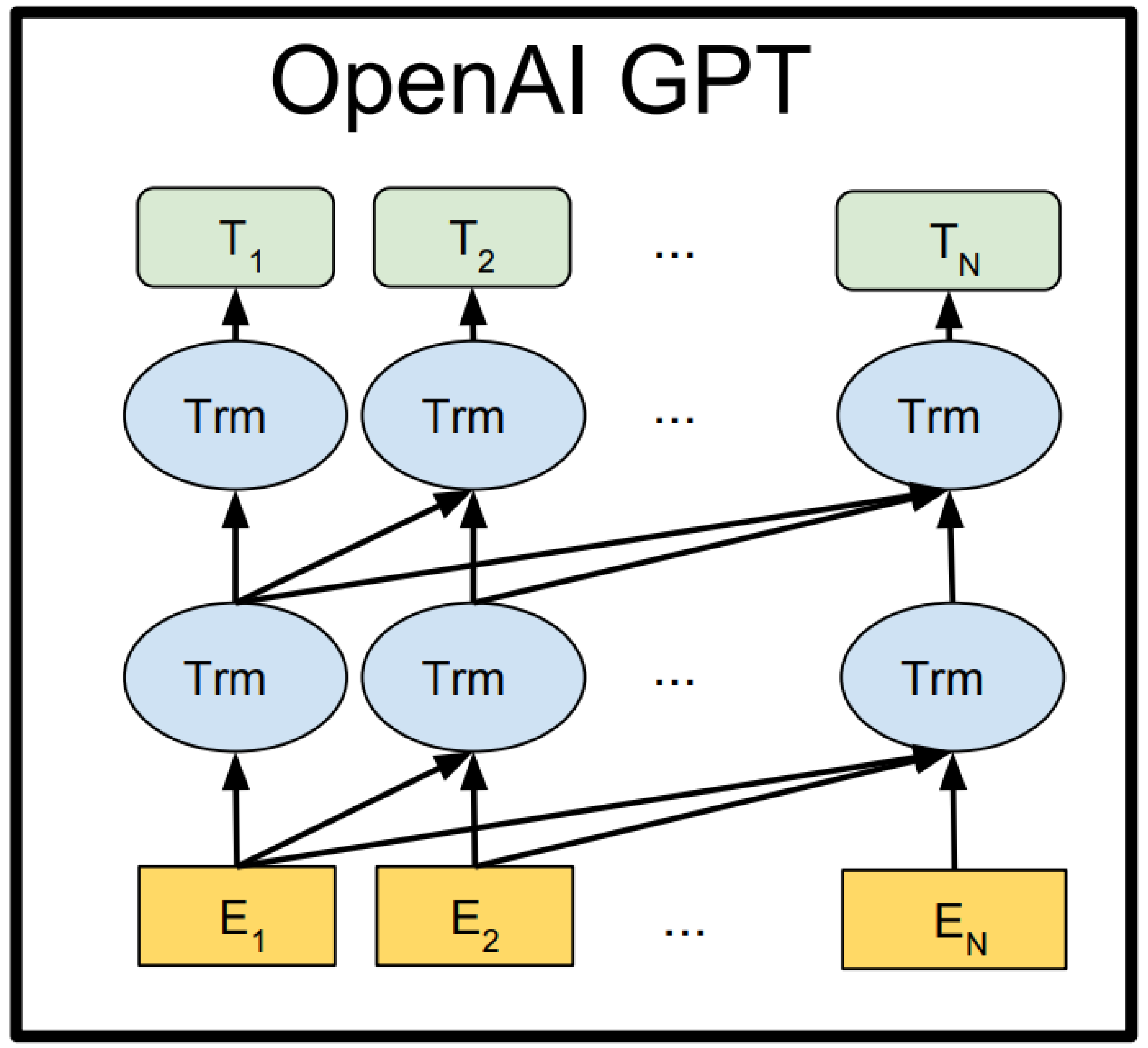
XLNet

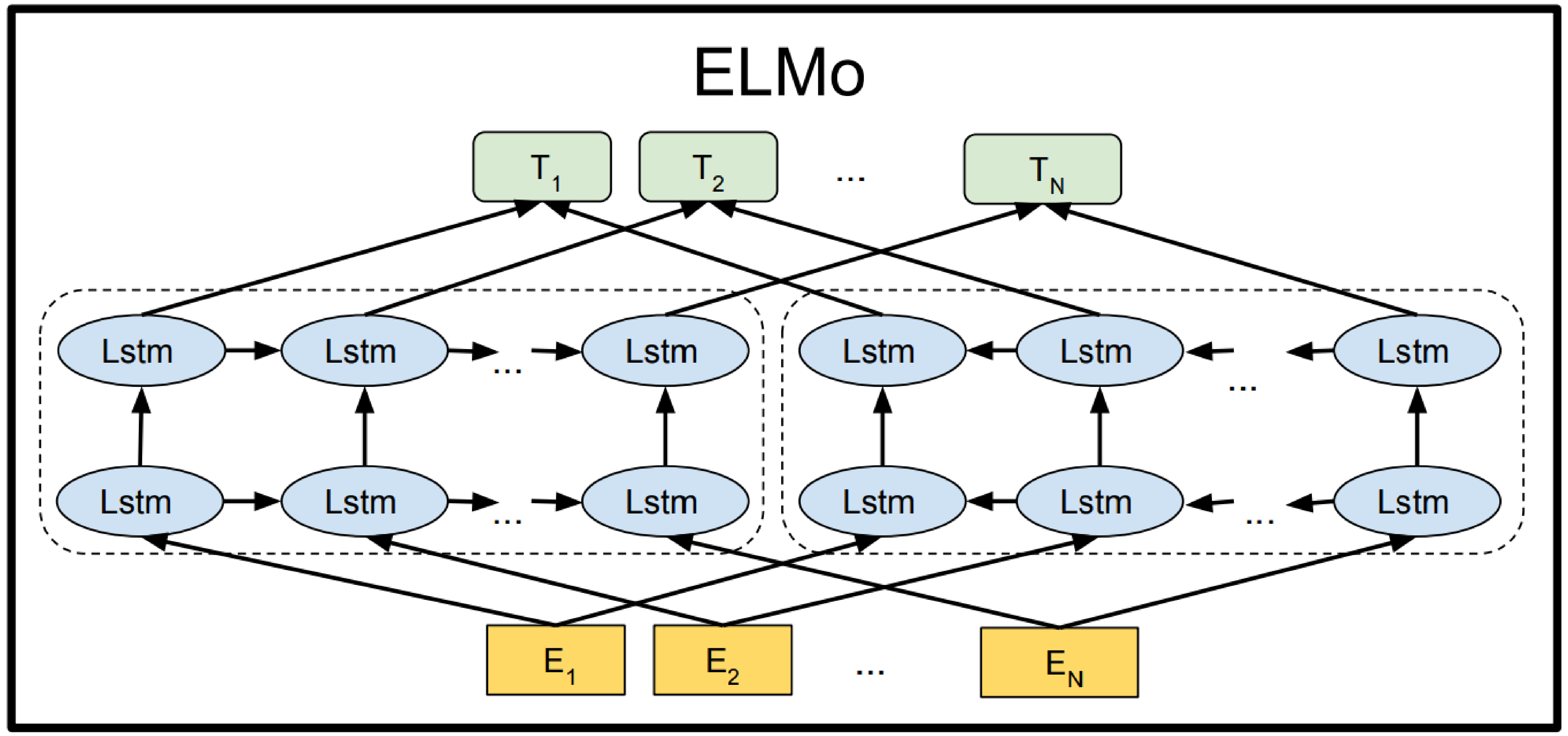
(Yang et al., 2019)

Auto-Regressive (AR)

- Objective: modeling information based on either previous or following contexts

$$\max_{\theta} \quad \log p_\theta(\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(x_t \mid \mathbf{x}_{<t}) = \sum_{t=1}^{T} \log \frac{\exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x_t)\right)}{\sum_{x'} \exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x')\right)}$$



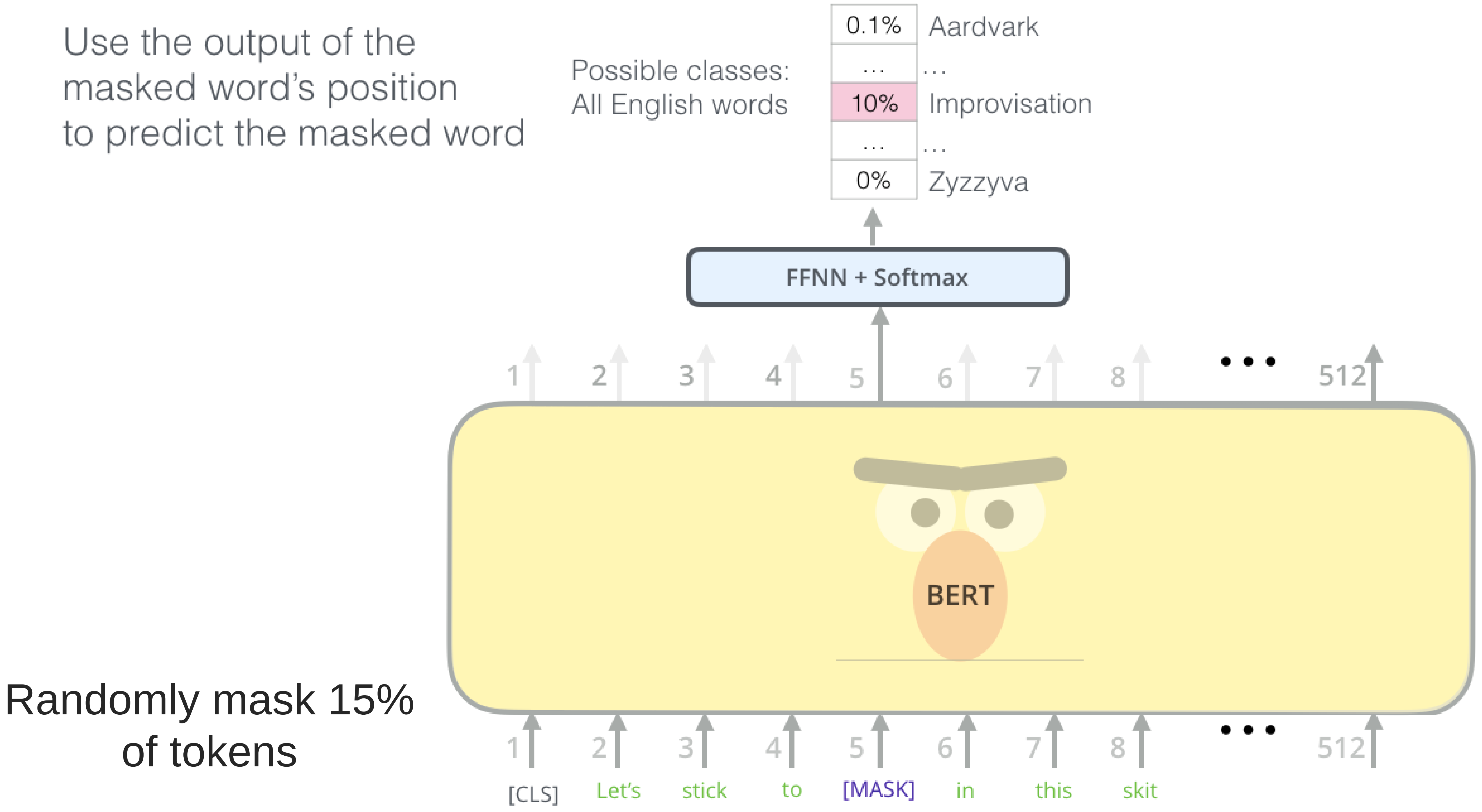


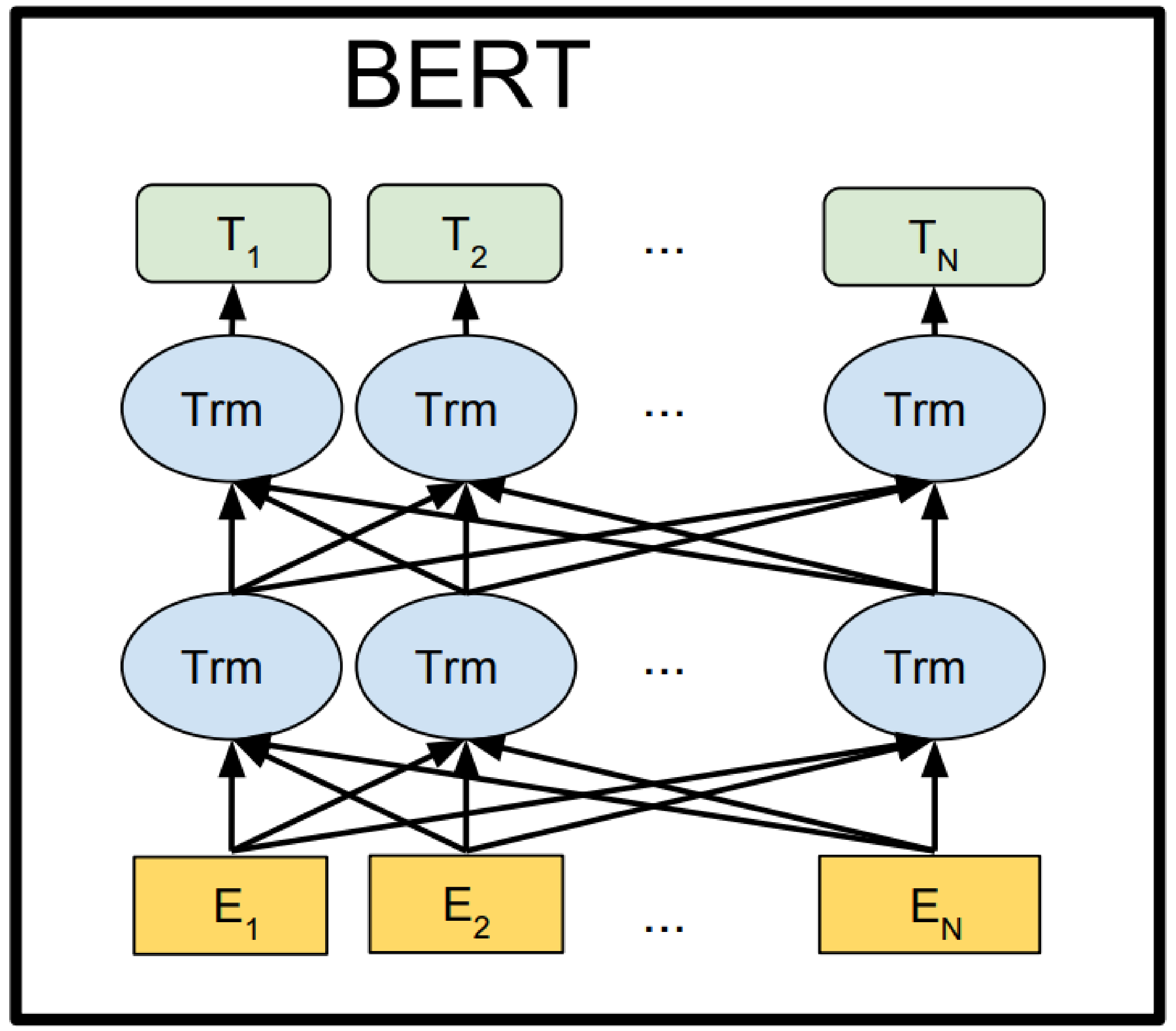
Auto-Encoding (AE)

- Objective: reconstructing $\bar{x}$ from $\hat{x}$

$$\max_{\theta} \quad \log p_\theta(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_\theta(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x_t)\right)}{\sum_{x'} \exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x')\right)}$$

  - dimension reduction or denoising (masked LM)





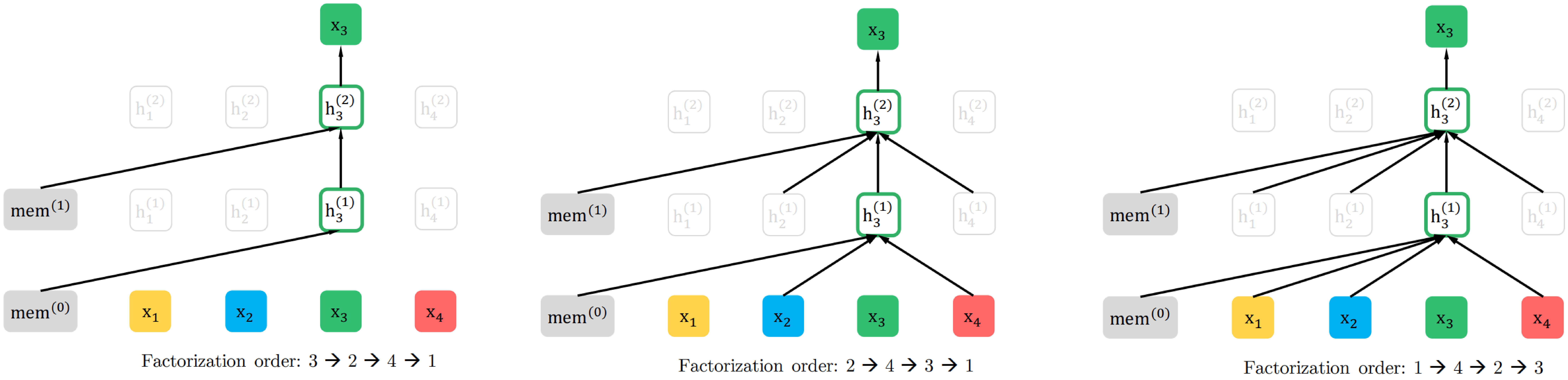
Auto-Encoding (AE)

- Issues
  - *Independence assumption*: ignore the dependency between masks
  - *Input noise*: discrepancy between pre-training and fine-tuning

    (w/ [MASK]) (w/o [MASK])

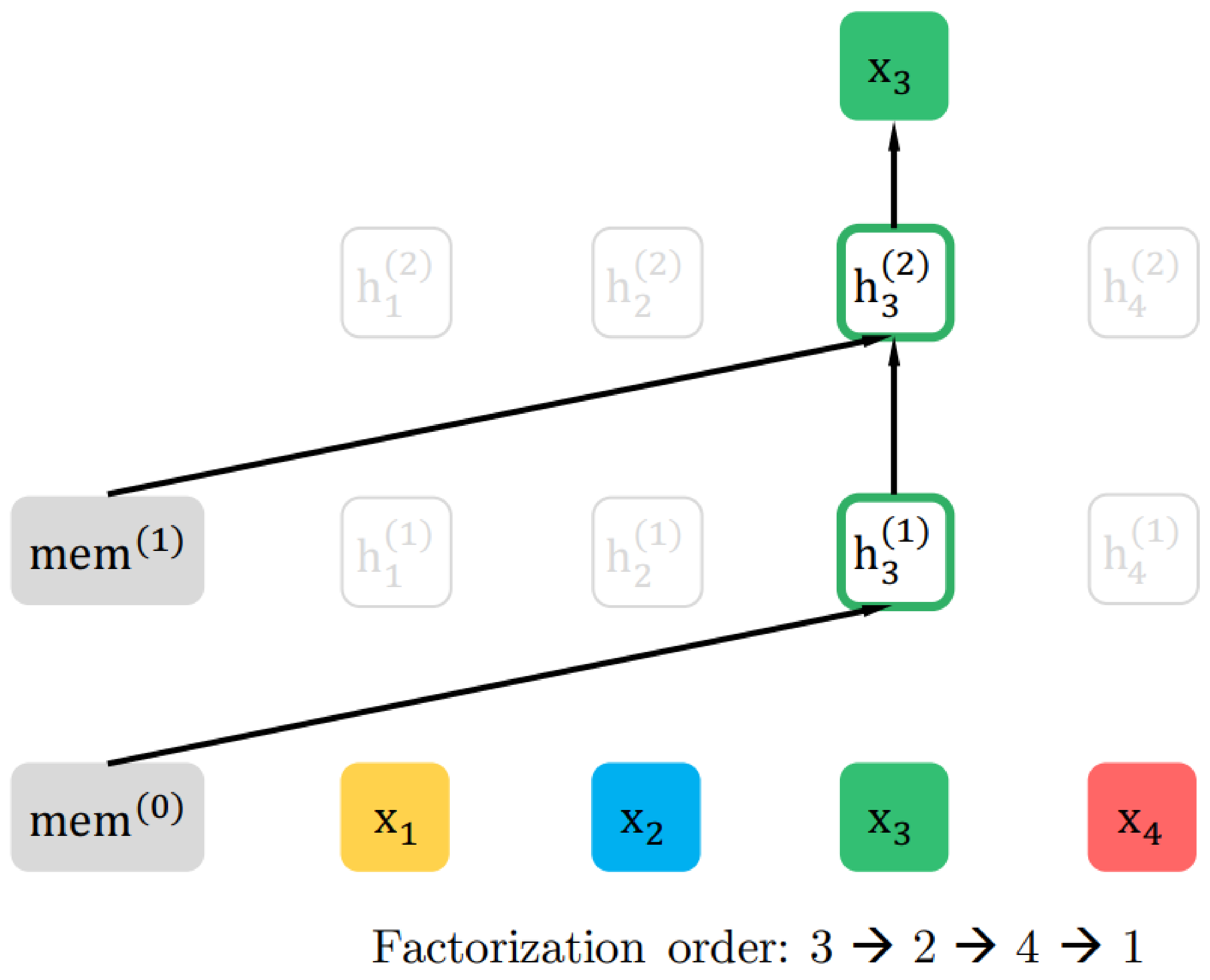
Permutation Language Model

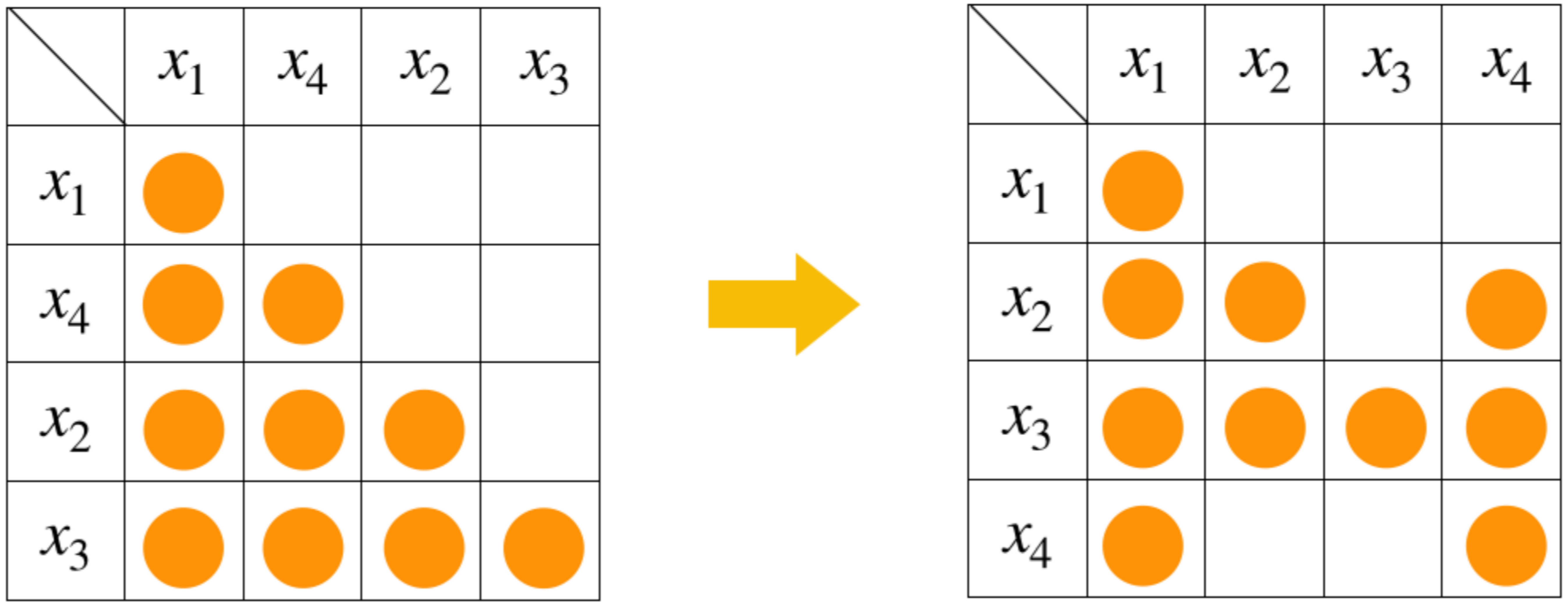
- Goal: use AR and bidirectional contexts for prediction
- Idea: parameters shared across all factorization orders in expectation
  - *T*! different orders to a valid AR factorization for a sequence of length *T*
  - Pre-training on sequences sampled from all possible permutations



Permutation Language Model

- Implementation: only permute the factorization order
  - Remain original positional encoding
  - Rely on proper attention masks in Transformers





resolve independence assumption and pretrain-finetune discrepancy issues

Formulation Reparameterizing

- Issue: naively applying permutation LM does not work
- Original formulation

$$p_\theta(X_{z_t} = x \mid \mathbf{x}_{z_{<t}}) = \frac{\exp\left(e(x)^\top h_\theta(\mathbf{x}_{\mathbf{z}_{<t}})\right)}{\sum_{x'} \exp\left(e(x')^\top h_\theta(\mathbf{x}_{\mathbf{z}_{<t}})\right)}$$

  - [MASK] indicates the target position
  - $h_\theta(x_{z_{<t}})$ does not depend on predicted position

$x_1, x_2, x_3, x_4 \rightarrow P(x_3|x_1, x_2)$

$x_1, x_2, x_4, x_3 \rightarrow P(x_4|x_1, x_2)$

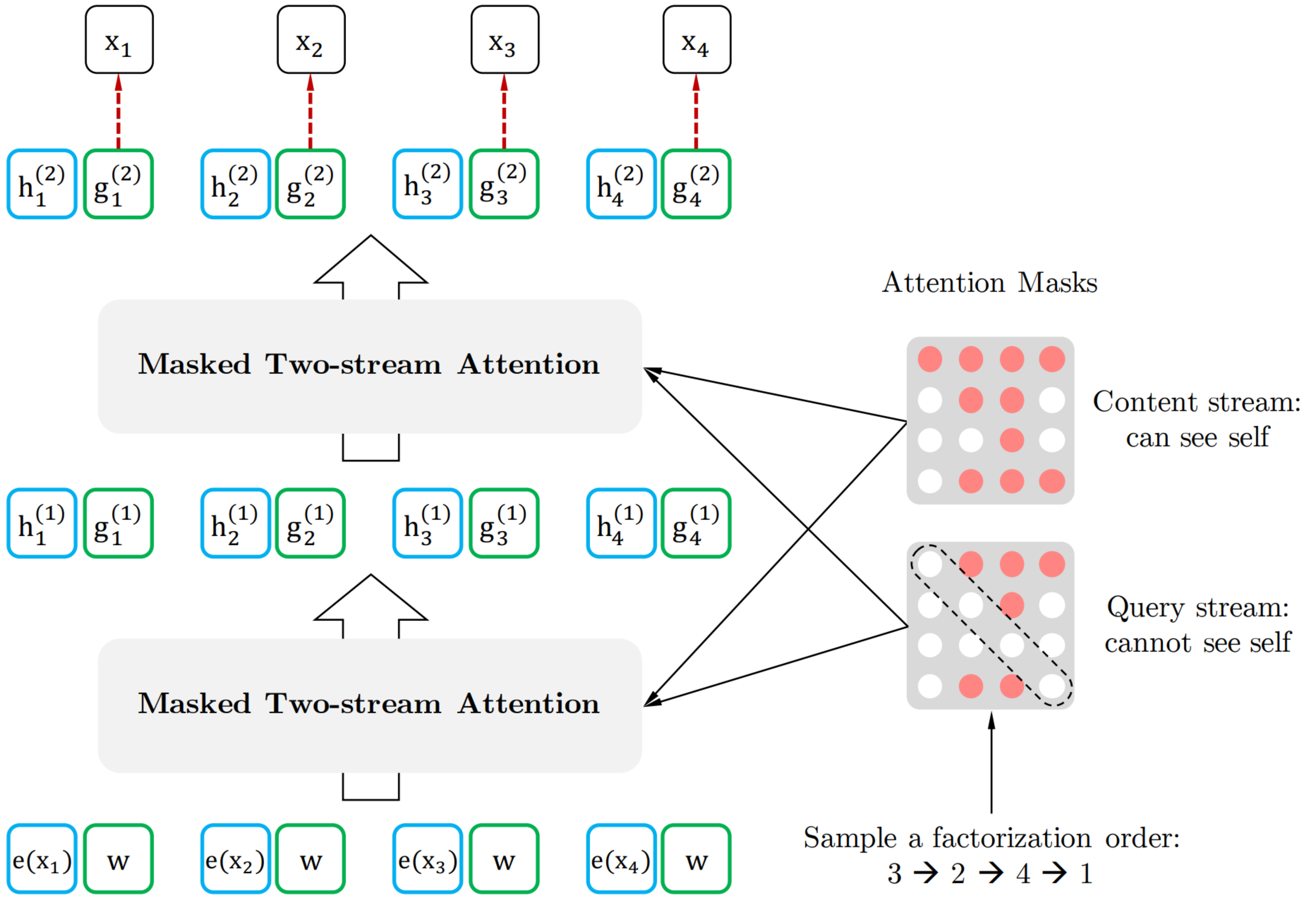
- Reparameterization

$$p_\theta(X_{z_t} = x \mid \mathbf{x}_{z_{<t}}) = \frac{\exp\left(e(x)^\top g_\theta(\mathbf{x}_{\mathbf{z}_{<t}}, z_t)\right)}{\sum_{x'} \exp\left(e(x')^\top g_\theta(\mathbf{x}_{\mathbf{z}_{<t}}, z_t)\right)}$$

  - $g_\theta(x_{z_{<t}}, z_t)$ is a new embedding considering the target position z_t

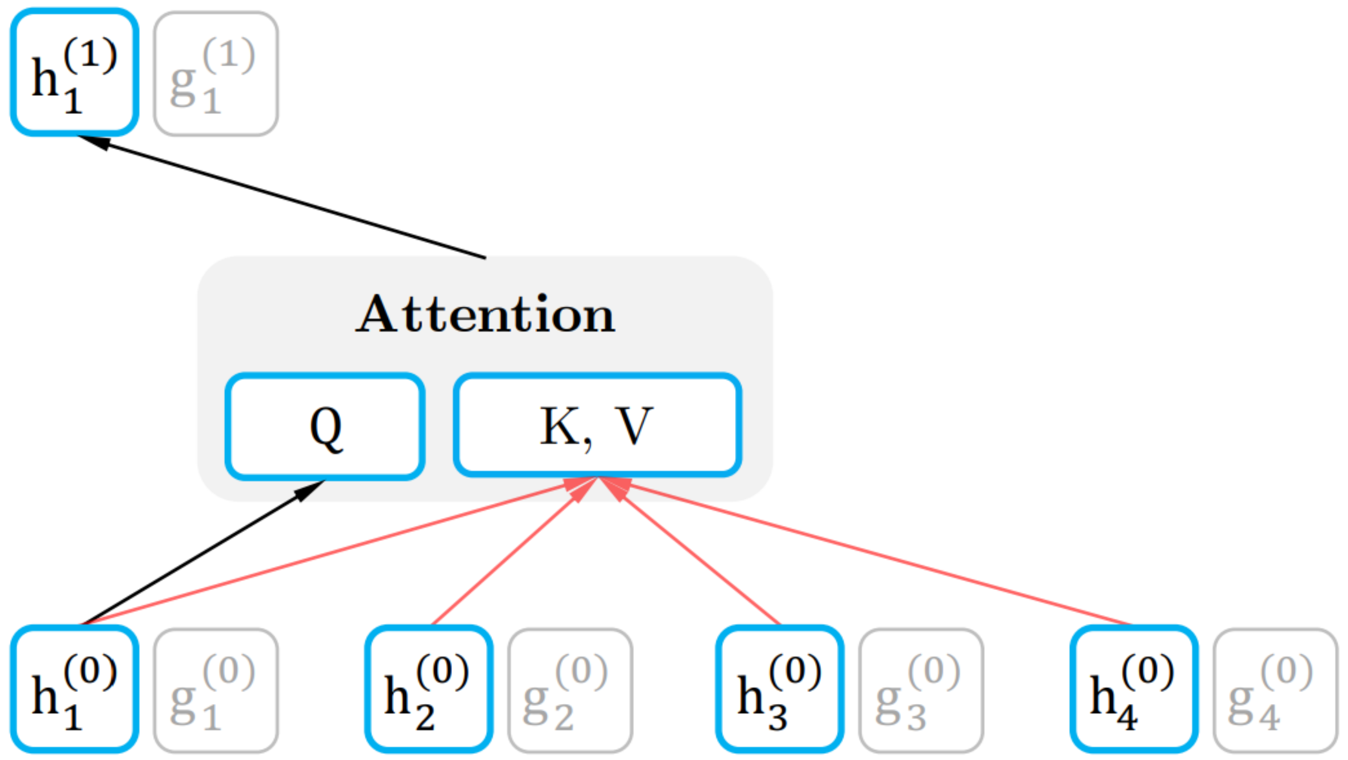
Two-Stream Self-Attention

- Formulation of $g(x_{z_{<t}}, z_t)$
  1) Predicting the token x_{z_t} should only use the position z_t and not the content x_{z_t}
  2) Predicting other tokens x_{z_j} $(j > t)$ should encode the content x_{z_t}
- Idea: two sets of hidden representations
  - Content stream: can see self
  - Query stream: cannot see self

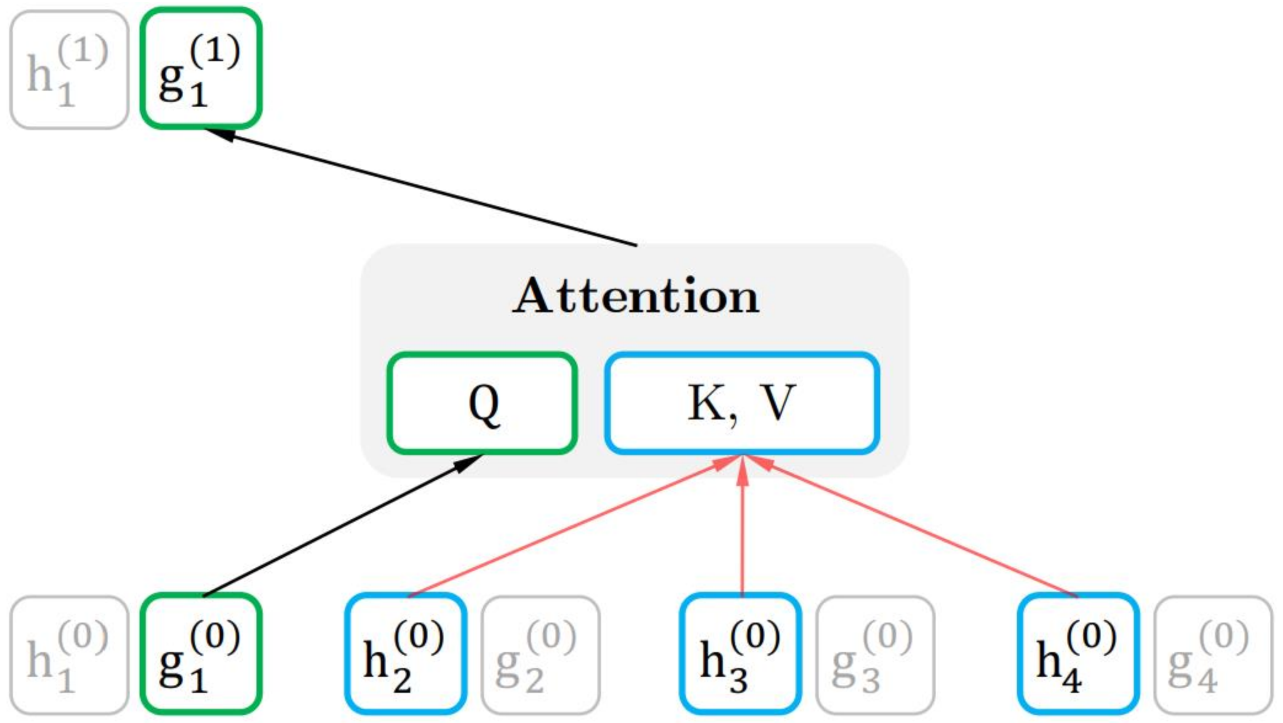


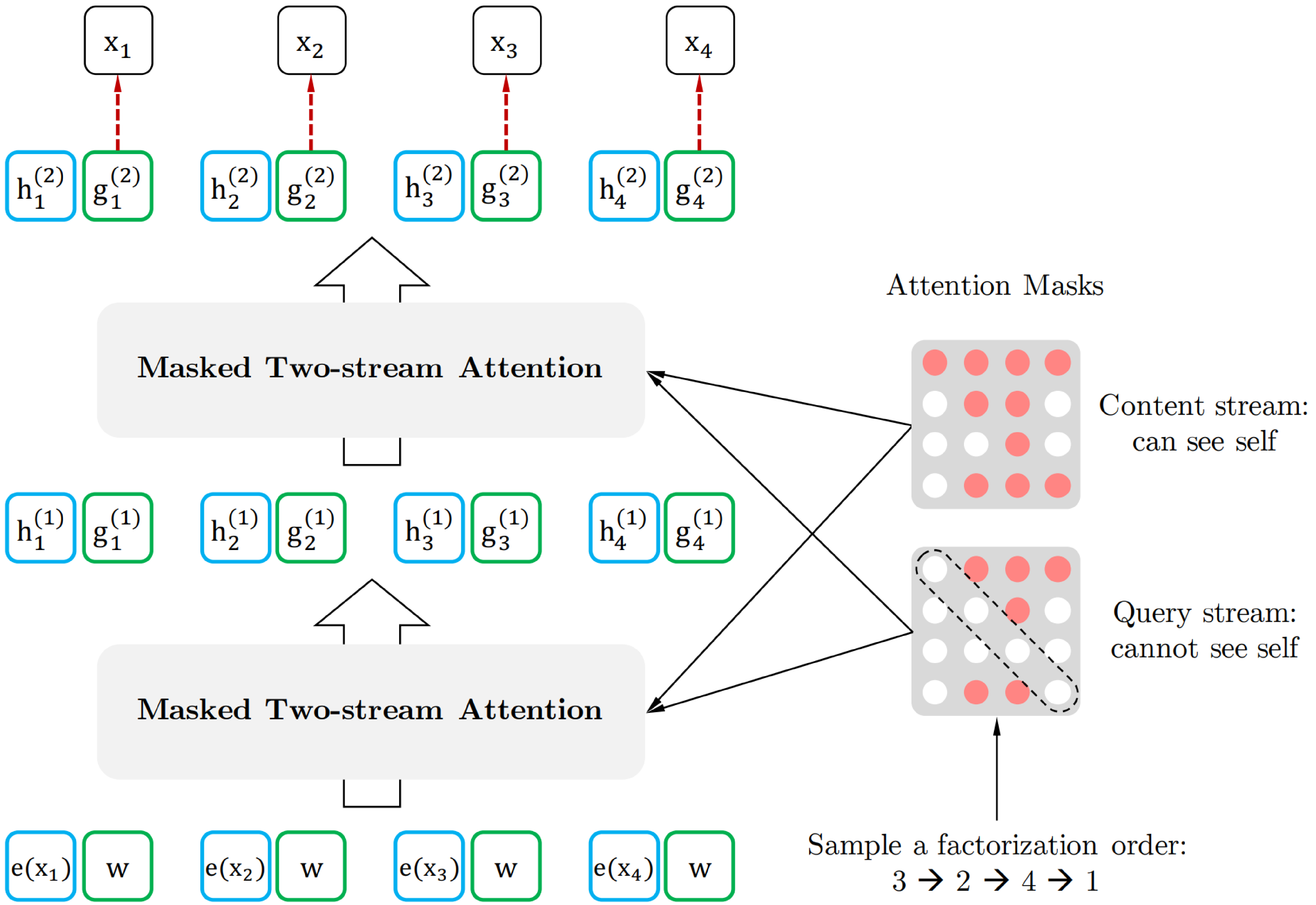
Two-Stream Self-Attention

- Content stream
  - Predict other tokens



- Query stream
  - Predict the current token





GLUE Results

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	WNLI
Single-task single models on dev									
BERT [2]	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-
XLNet	**89.8/-**	**93.9**	**91.8**	**83.8**	**95.6**	**89.2**	**63.6**	**91.8**	-
Single-task single models on test									
BERT [10]	86.7/85.9	91.1	89.3	70.1	94.9	89.3	60.5	87.6	65.1
Multi-task ensembles on test (from leaderboard as of June 19, 2019)									
Snorkel* [29]	87.6/87.2	93.9	89.9	80.9	96.2	91.5	63.8	90.1	65.1
ALICE*	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8
MT-DNN* [18]	87.9/87.4	96.0	89.9	**86.3**	96.5	92.7	68.4	91.1	89.0
XLNet*	**90.2/89.7**†	**98.6**†	90.3†	**86.3**	**96.8**†	**93.0**	67.8	**91.6**	**90.4**

Contributions

- AR for addressing independence assumption

$$\mathcal{J}_{\text{BERT}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{is a city})$$

$$\mathcal{J}_{\text{XLNet}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{New}, \text{is a city})$$

- AE for addressing the pretrain-finetune discrepancy

$$\mathcal{J}_{\text{BERT}} = \sum_{x \in \mathcal{T}} \log p(x \mid \mathcal{N}); \quad \mathcal{J}_{\text{XLNet}} = \sum_{x \in \mathcal{T}} \log p(x \mid \mathcal{N} \cup \mathcal{T}_{<x})$$

RoBERTa

(Liu et al., 2019)

RoBERTa

- Dynamic masking
  - each sequence is masked in 10 different ways over the 40 epochs of training
    - Original masking is performed during data preprocessing
- Optimization hyperparameters
  - peak learning rate and number of warmup steps tuned separately for each setting
    - Training is very sensitive to the Adam epsilon term
    - Setting β_2 = 0.98 improves stability when training with large batch sizes
- Data
  - not randomly inject short sequences
  - train only with full-length sequences
    - Original model trains with a reduced sequence length for first 90% of updates
  - BookCorpus, CC-News, OpenWebText, Stories

GLUE Results

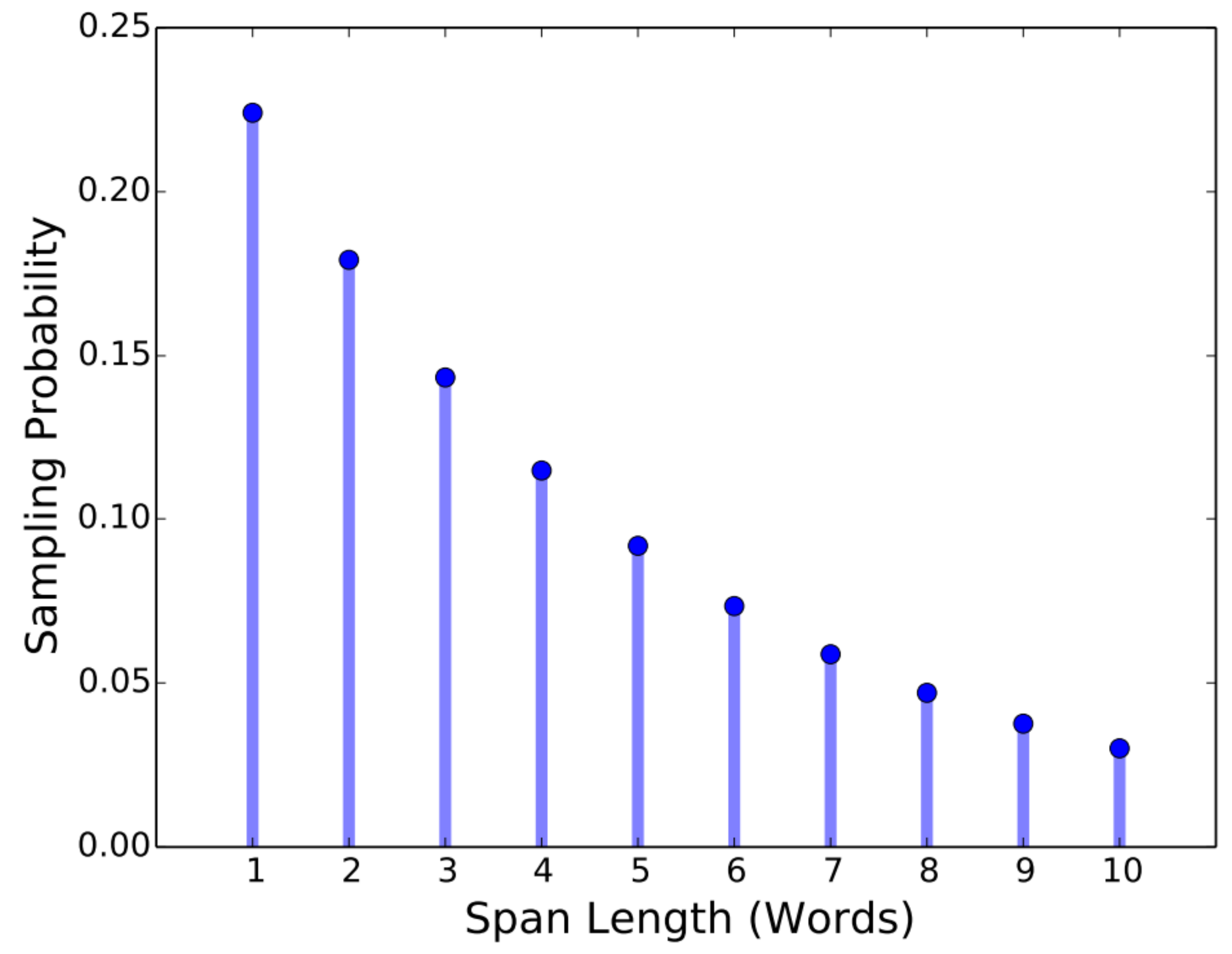
	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$BERT_{LARGE}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$XLNet_{LARGE}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

25

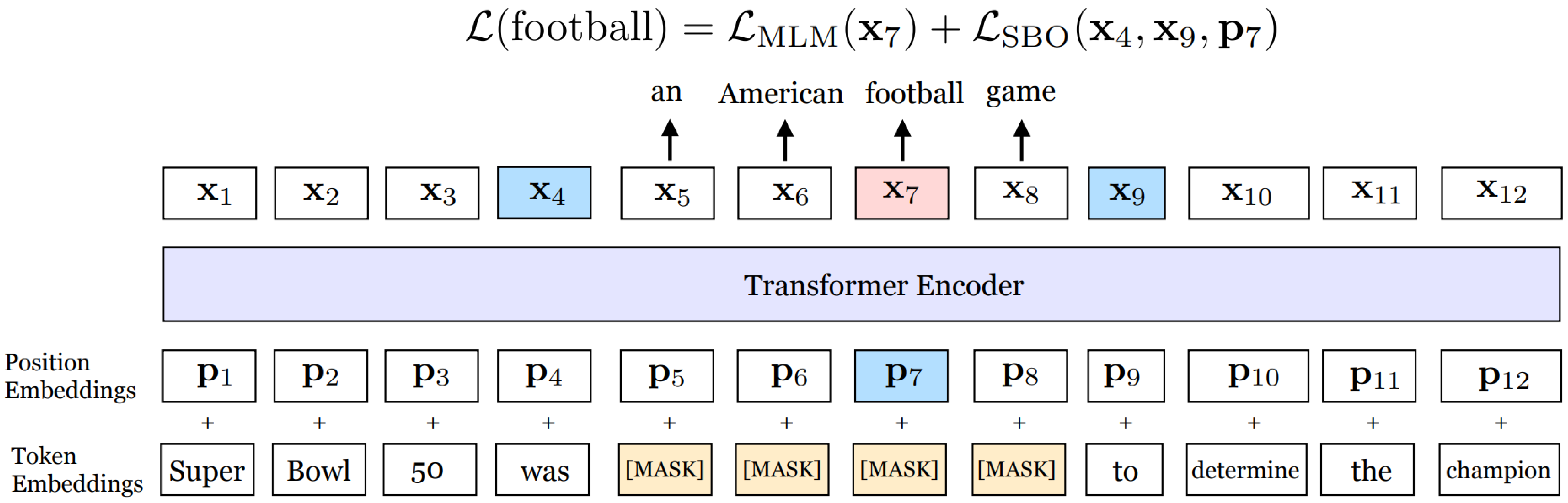
SpanBERT

(Joshi et al., 2019)

SpanBERT



- Span masking
  - A random process to mask spans of tokens
- Single sentence training
  - a single contiguous segment of text for each training sample (instead of two)
- Span boundary objective (SBO)
  - predict the entire masked span using only the span's boundary



Results

- Masking scheme

	SQuAD 2.0	NewsQA	TriviaQA	Coreference	MNLI-m	QNLI
Subword Tokens	83.8	72.0	76.3	**77.7**	86.7	92.5
Whole Words	84.3	72.8	77.1	76.6	86.3	92.8
Named Entities	84.8	72.7	78.7	75.6	86.0	93.1
Noun Phrases	85.0	**73.0**	77.7	76.7	86.5	93.2
Random Spans	**85.4**	**73.0**	**78.8**	76.4	**87.0**	**93.3**

- Auxiliary objective

	SQuAD 2.0	NewsQA	TriviaQA	Coreference	MNLI-m	QNLI
Span Masking (2seq) + NSP	85.4	73.0	78.8	76.4	87.0	93.3
Span Masking (1seq)	86.7	73.4	80.0	76.3	87.3	93.8
Span Masking (1seq) + SBO	**86.8**	**74.1**	**80.3**	**79.0**	**87.6**	**93.9**

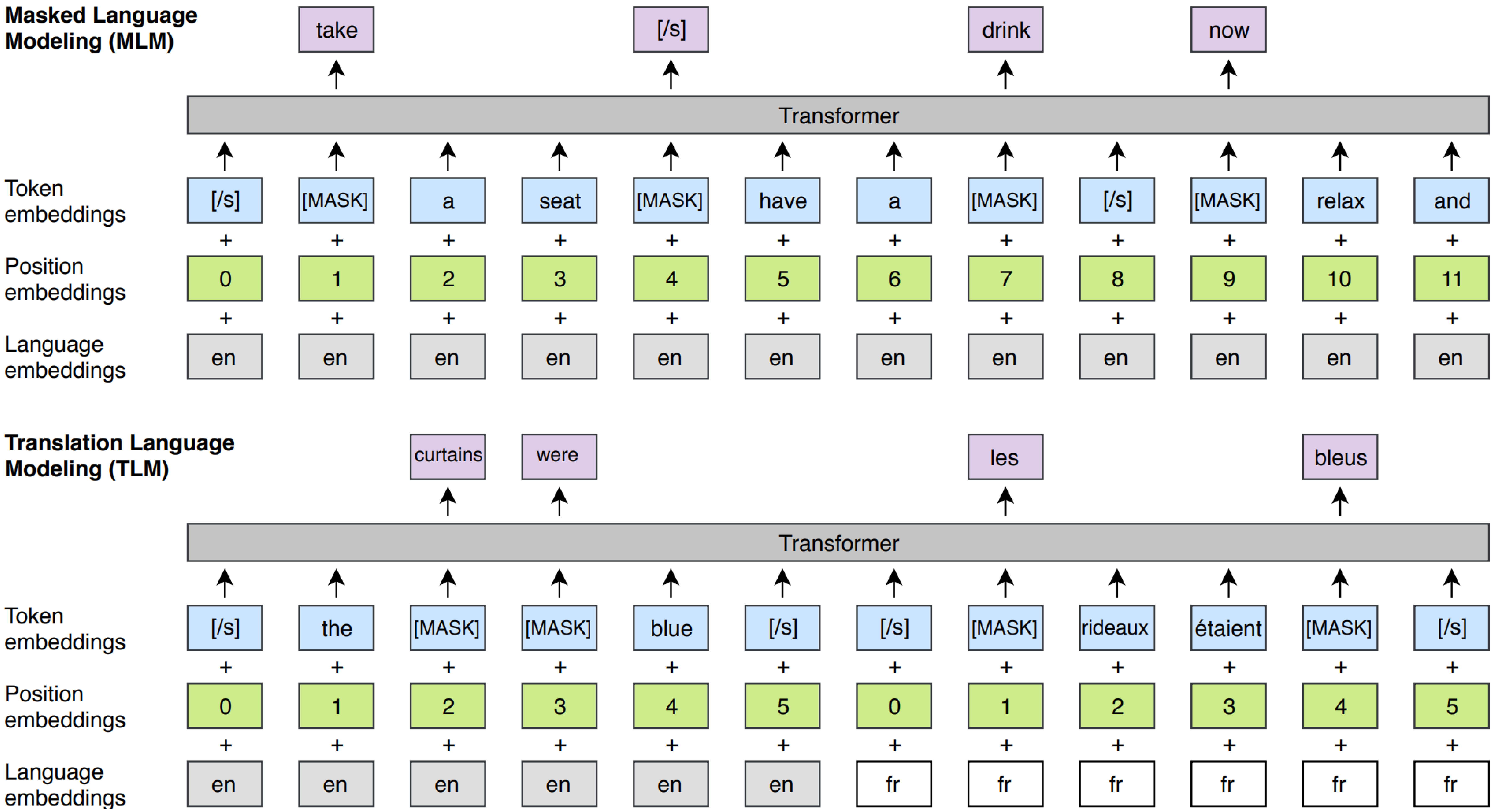
28

XLM

(Lample & Connueau, 2019)

XLM

- Masked LM + Translation LM

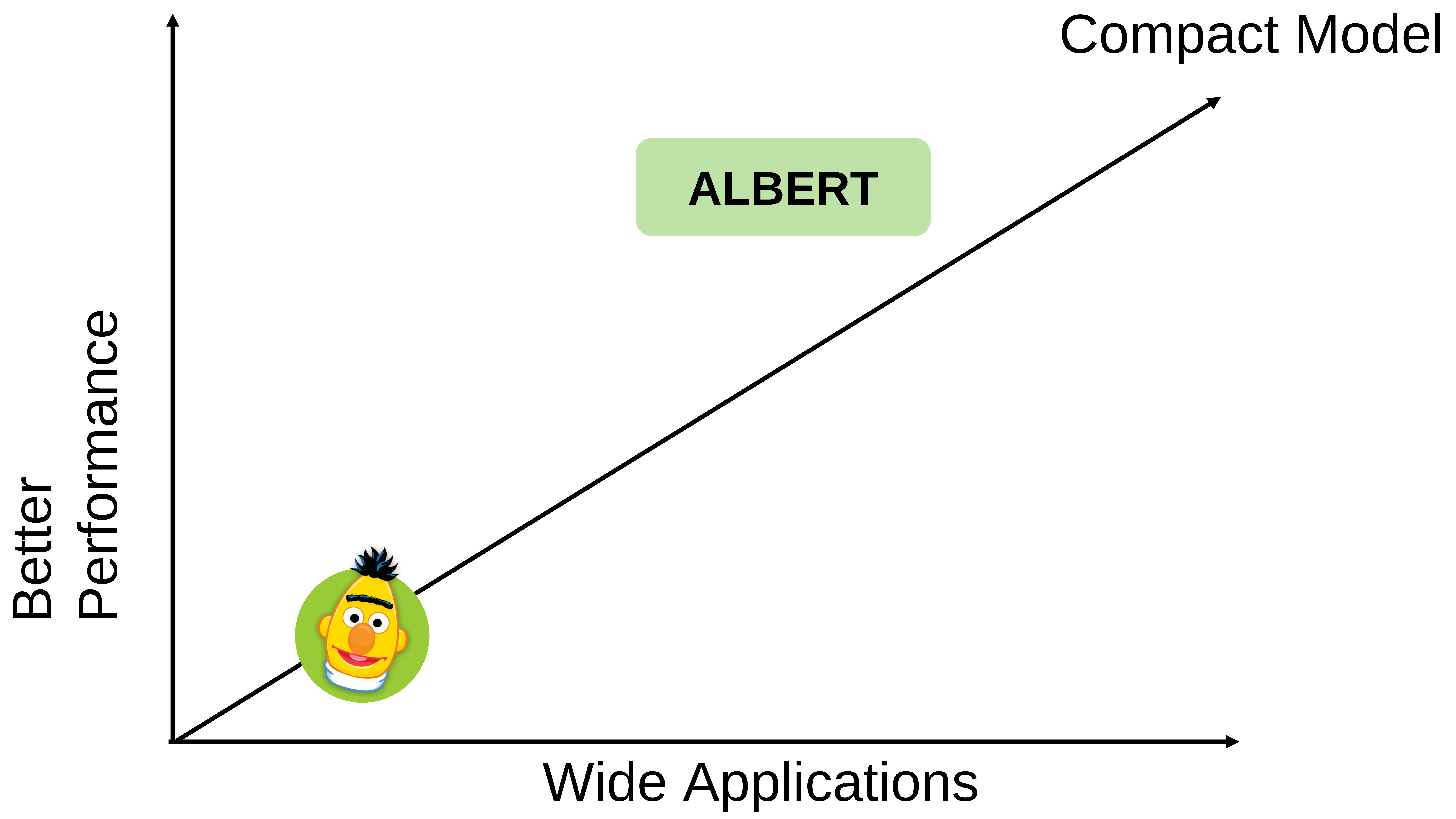


30

ALBERT

(Lan et al., 2020)

Beyond BERT

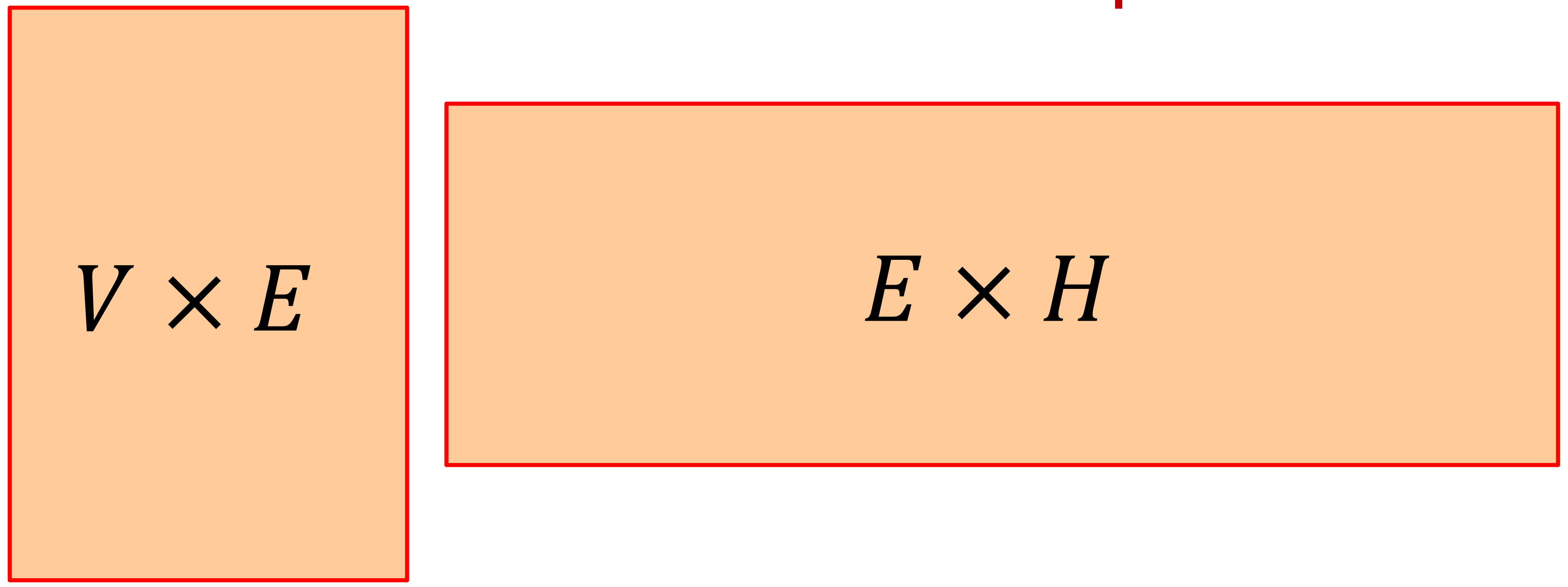

Compact Model
ALBERT
Better
Performance
Wide Applications

ALBERT: A Lite BERT

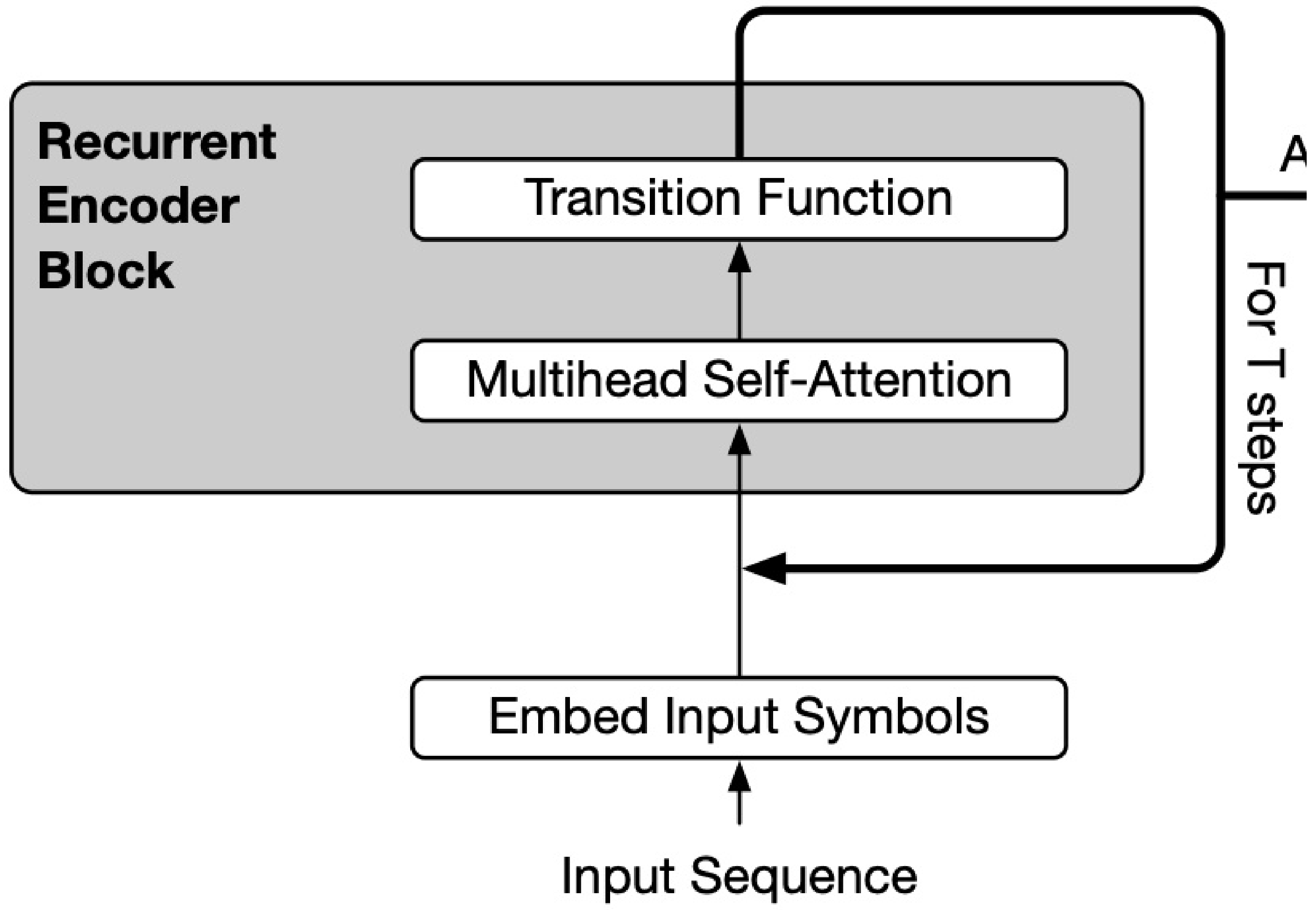
1. Factorized embedding parameterization
   - WordPiece embedding size E is tied with the hidden layer size H → $E \equiv H$

context-independent

context-dependent → $E << H$



2. Cross-layer sharing



ALBERT: A Lite BERT

Model	E	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base not-shared	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base all-shared	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base E=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base E=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

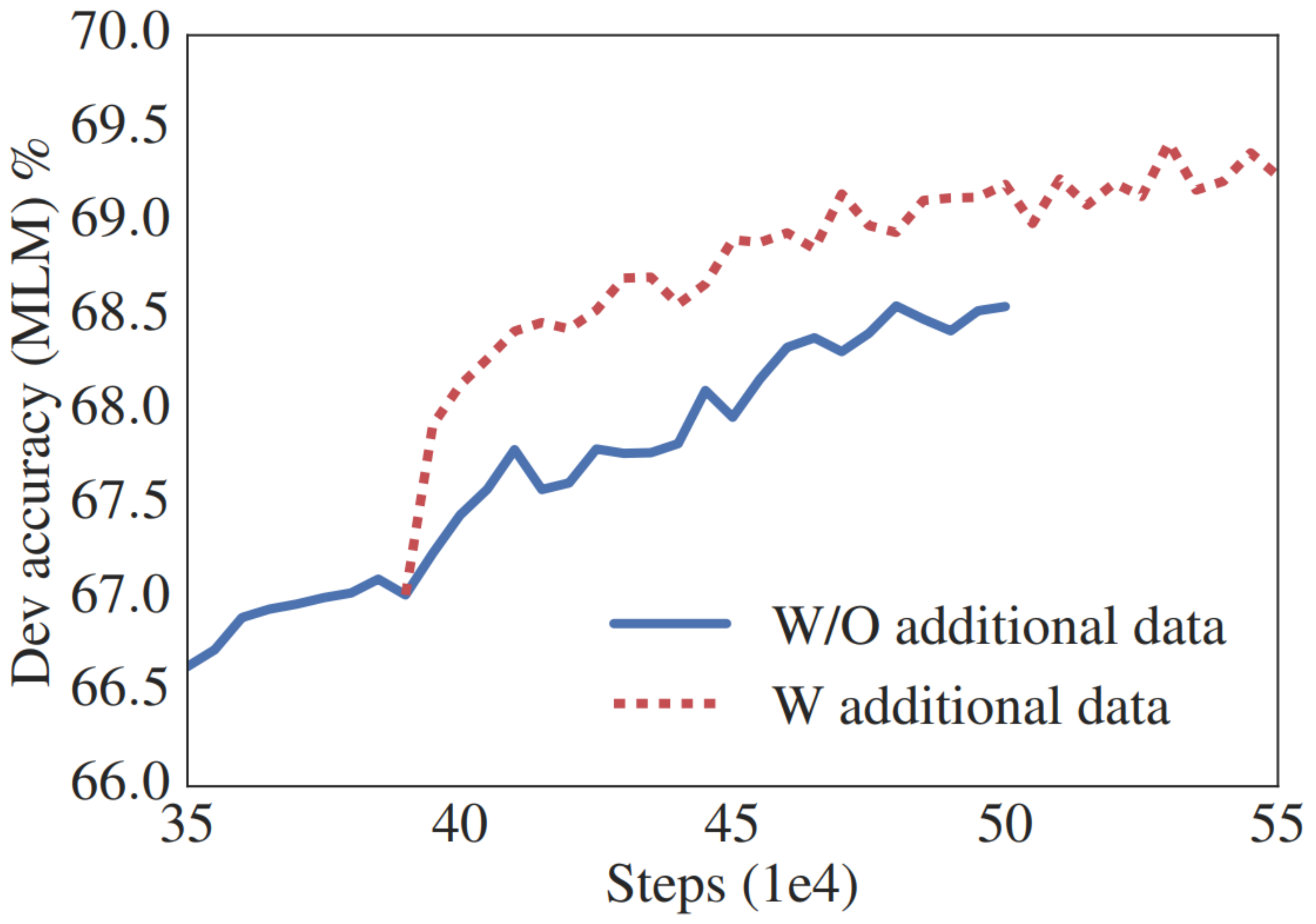
ALBERT: A Lite BERT

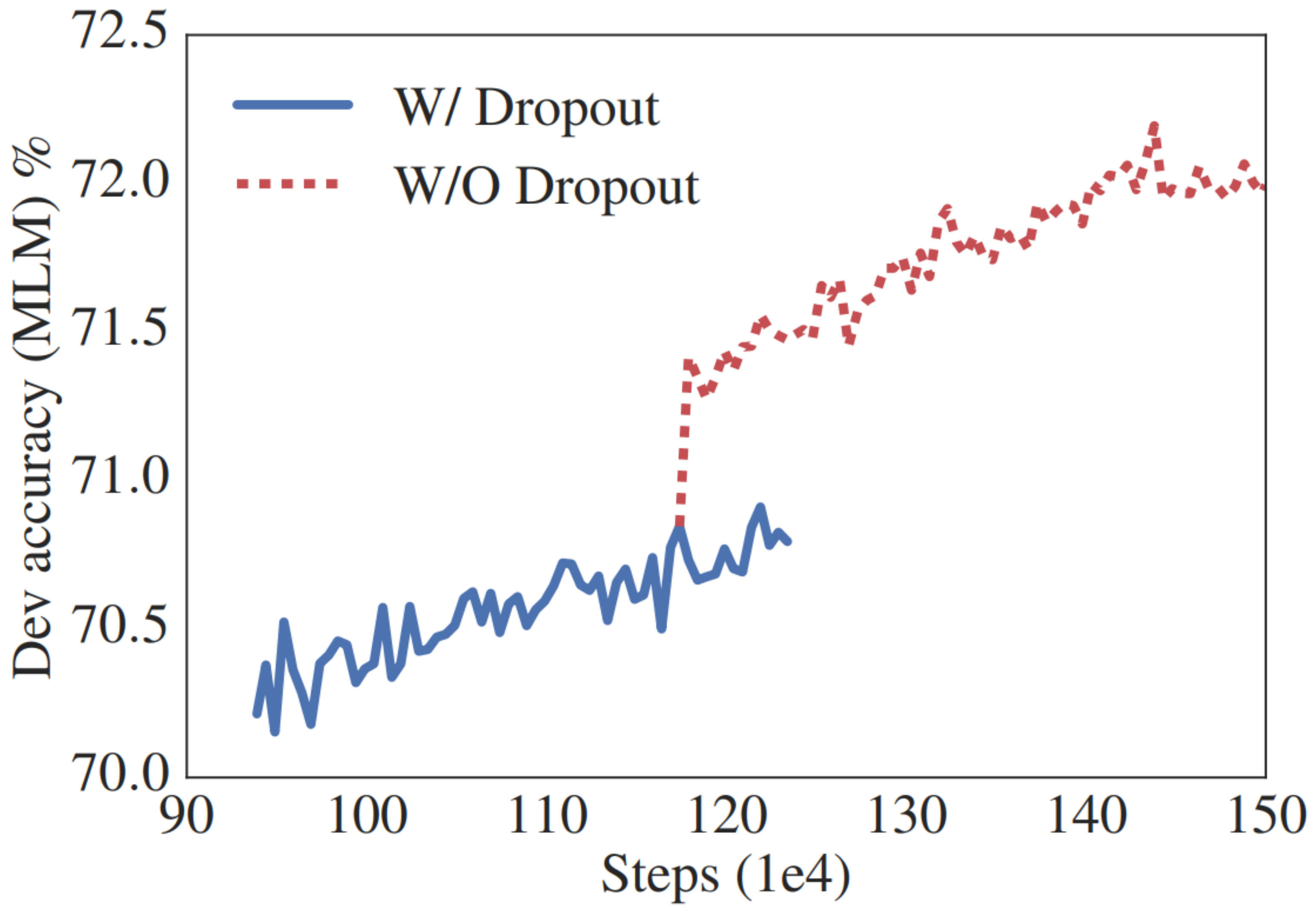
3. Inter-sentence coherence loss
    - NSP (next sentence prediction) contains both topical and ordering information
    - Topical cues help more → model utilizes more
    - SOP (sentence order prediction) focuses on ***ordering*** not topical cues

SP tasks	Intrinsic Tasks			Downstream Tasks					
	MLM	NSP	SOP	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
None	54.9	52.4	53.3	88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0
NSP	54.5	90.5	52.0	88.4/81.5	77.2/74.6	81.6	**91.1**	62.3	79.2
SOP	54.0	78.9	86.5	**89.3/82.3**	**80.0/77.1**	**82.0**	90.3	**64.0**	**80.1**

ALBERT: A Lite BERT

4. Additional data and removing dropout





	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
No additional data	**89.3/82.3**	**80.0/77.1**	81.6	90.3	64.0	80.1
With additional data	88.8/81.7	79.1/76.3	**82.4**	**92.8**	**66.0**	**80.8**

	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
With dropout	94.7/89.2	89.6/86.9	90.0	96.3	85.7	90.4
Without dropout	**94.8/89.5**	**89.9/87.2**	**90.4**	**96.5**	**86.1**	**90.7**

GLUE Results

Models	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
BERT-large	86.6	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
XLNet-large	89.8	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa-large	90.2	94.7	**92.2**	86.6	96.4	**90.9**	68.0	92.4	-	-
ALBERT (1M)	90.4	95.2	92.0	88.1	96.8	90.2	68.7	92.7	-	-
ALBERT (1.5M)	**90.8**	**95.3**	**92.2**	**89.2**	**96.9**	**90.9**	**71.4**	**93.0**	-	-
Ensembles on test (from leaderboard as of Sept. 16, 2019)										
ALICE	88.2	95.7	**90.7**	83.5	95.2	92.6	**69.2**	91.1	80.8	87.0
MT-DNN	87.9	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2	98.6	90.3	86.3	96.8	93.0	67.8	91.6	90.4	88.4
RoBERTa	90.8	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0	88.5
Adv-RoBERTa	91.1	98.8	90.3	88.7	96.8	93.1	68.0	92.4	89.0	88.8
ALBERT	**91.3**	**99.2**	90.5	**89.2**	**97.1**	**93.4**	69.1	**92.5**	**91.8**	**89.4**

Concluding Remarks

- Transformer-XL (https://github.com/kimiyoung/transformer-xl)
  - Longer context dependency
- XLNet (https://github.com/zihangdai/xlnet)
  - AR + AE
  - No pretrain-finetune discrepancy
- RoBERTa (http://github.com/pytorch/fairseq)
  - Optimization details & data
- SpanBERT
  - Better for QA, NLI, coreference
- XLM (https://github.com/facebookresearch/XLM)
  - Zero-shot scenarios
- ALBERT (https://github.com/google-research/google-research/tree/master/albert / https://github.com/brightmart/albert_zh)
  - Compact model, faster training/fine-tuning