XLNet: Generalized Autoregressive Pretraining for Language Understanding

- ARUN

Content

- Transformer
- Model Comparison
- Unsupervised Pre-training Tasks
- Performance
- Ablation Studies
- References

Transformer Model

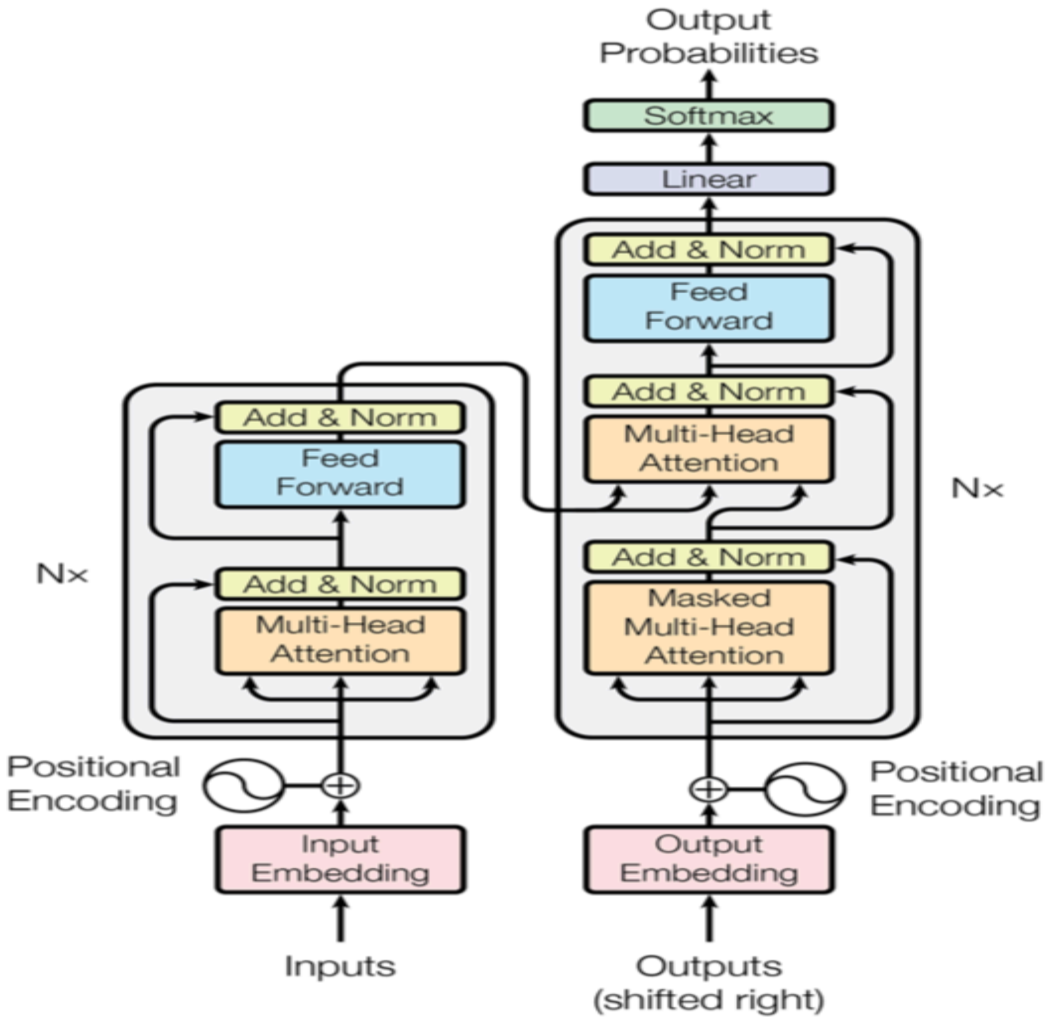


Figure 1: The Transformer - model architecture.

Attention

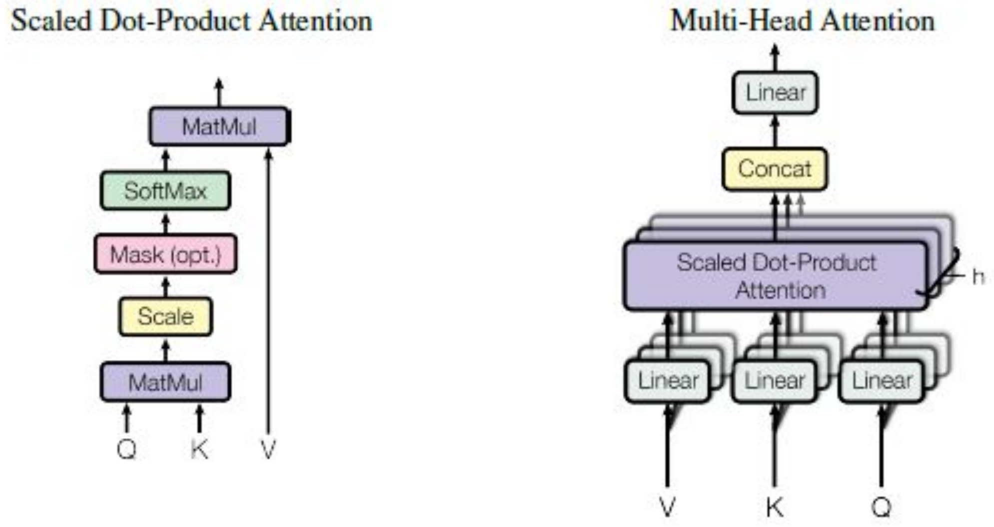
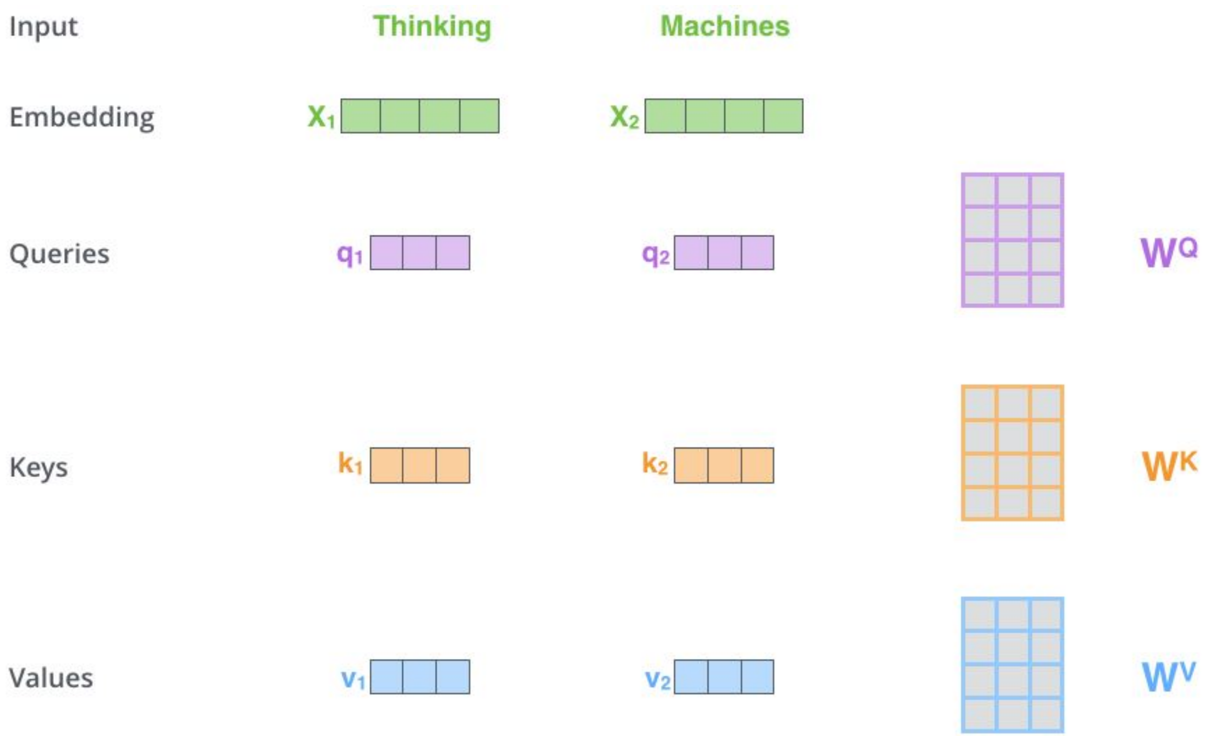


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Key,Value,Query



/lultiplying x1 by the WQ weight matrix produces q1, the "query" vector associated with that word. We end up creating a "query", a "key", and a "value" projection of each word in the input sentence.

Multi head attention

$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

- dmodel = 512 ; dk = dv = dmodel/h
- Jointly attend to information from different representation subspaces at different positions
- Reduced dimension of each head, the total computational cost is similar to single-head attention with full dimensionality

Model

- Embeddings + Positional Encoding
- Encoder

  - Multi head attention + Fully Positionwise FFN

  - Implements residual connections with batch normalization
- Decoder

  - Masked Multi Head Attention : to prevent positions from attending to subsequent positions

  - Encoder Decoder Multi Head Attention + Fully Positionwise FFN
- Two Linear Layers & Softmax

Model Comparison

- ELMo (Peters et al. NAACL 2018)
  - BiLSTM Pretrained using LM
  - Embeddings from Hidden Layers
- ULMFiT (Howard & Ruder ACL 2018)
  - LSTM Pretrained using LM & Fine Tuned on Specific Task
  - Add Classification layer for Predictions
- OpenAI GPT (Radford et al. 2018)
  - Transformer Decoder Pretrained using LM(Left to Right)
  - Add Classification layer for Predictions
- BERT (Devlin et al. 2018)
  - Transformer encoder Pretrained using MLM & NPS
  - Add Classification layer for Predictions

Masked Language Model

- Rather than *always* replacing the chosen words with `[MASK]`, the data generator will do the following:
- 80% of the time: Replace the word with the `[MASK]` token, e.g., `my dog is hairy` → `my dog is [MASK]`
- 10% of the time: Replace the word with a random word, e.g., `my dog is hairy` → `my dog is apple`
- 10% of the time: Keep the word unchanged, e.g., `my dog is hairy` → `my dog is hairy`. The purpose of this is to bias the representation towards the actual observed word.

- Why not <MASK> everywhere ?
  - Might not predict well during fine tuning
  - Might learn good contextual representation of only <MASK>
- Why leave some sentences intact?
  - Biasing to learn masked tokens better
  - Helps model learn representation for all the tokens
- Will random words confuse the model?
  - Yes. Hence, only small percentage
  - Didn’t affect model performance

Next Sentence Prediction

Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]

penguin [MASK] are flight ##less birds [SEP]

Label = NotNext

- Helps understanding the relationship between two text sentences
- 50% of time B is the actual next sentence that follows A, & other 50% random sentence from corpus
- 97%-98% accuracy at this task

Denoising autoencoding approach

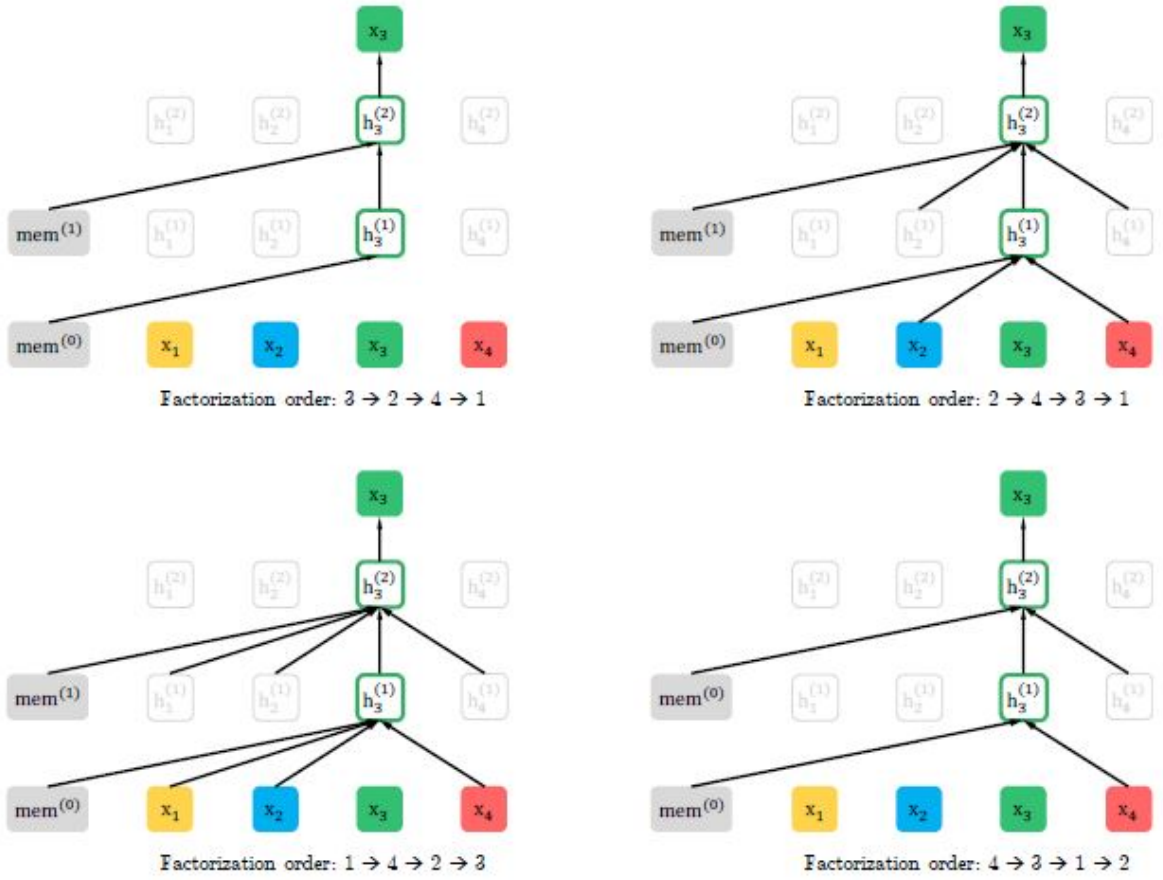
Advantages :

- Context dependency

Disadvantages:

- Input noise -> Pretrain-Finetune discrepancy
- Independence assumption

Permutation Language Modelling



Masked LM Comparison

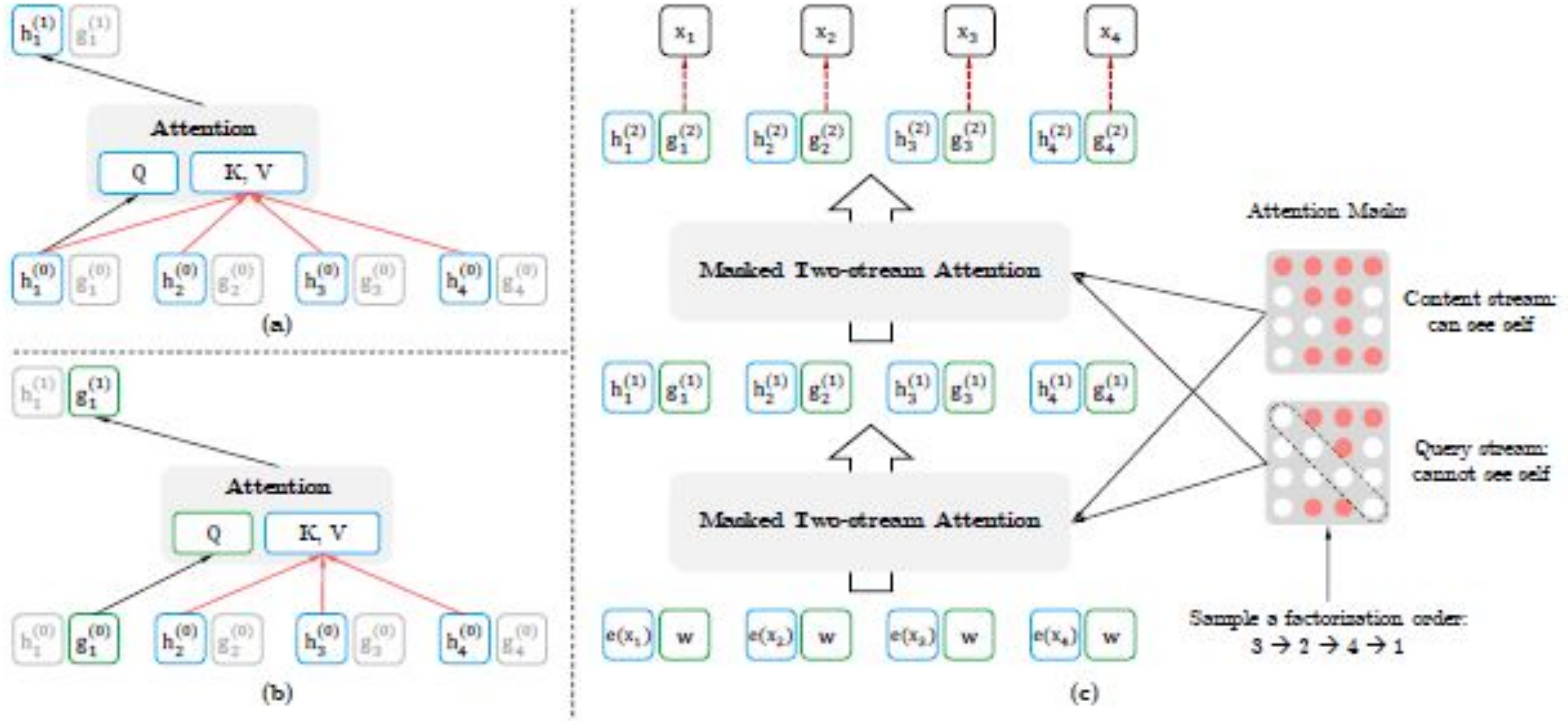
$$\mathcal{J}_{\text{BERT}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{is a city}),$$

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

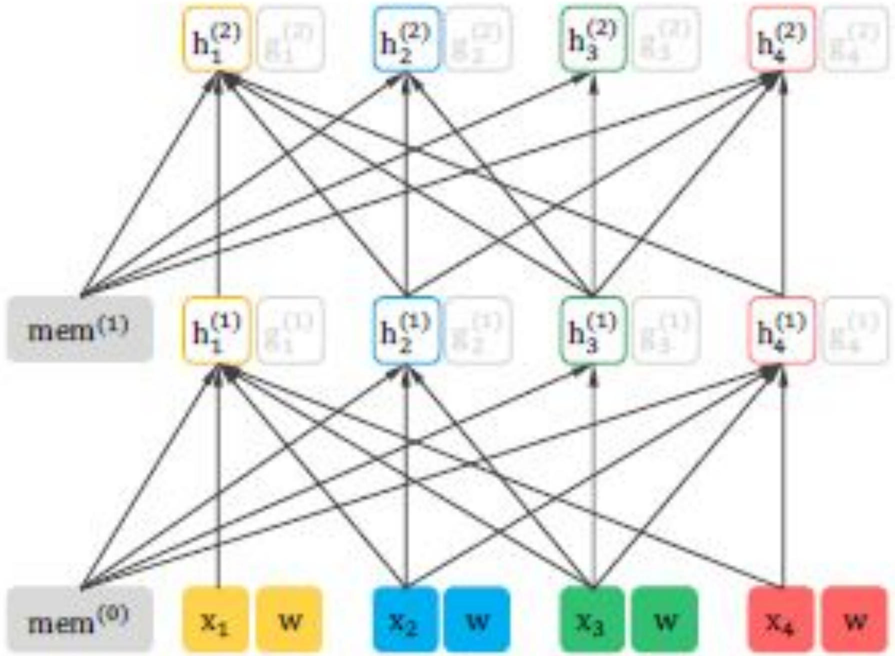
Two Stream Attention Motivation

- Predict using current position & previous context
- But, also encode previous and current content for next token prediction

Two Stream self attention – Target Aware Prediction

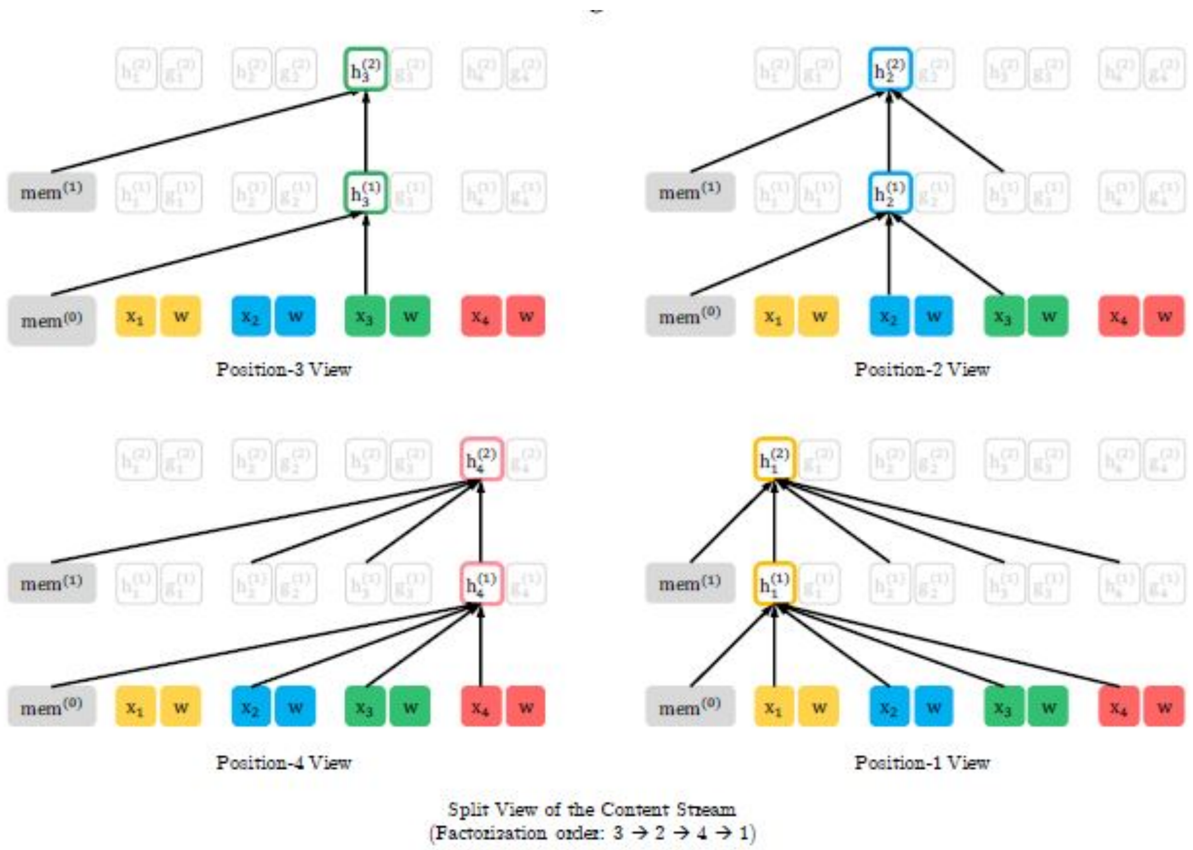


Content Stream visualization

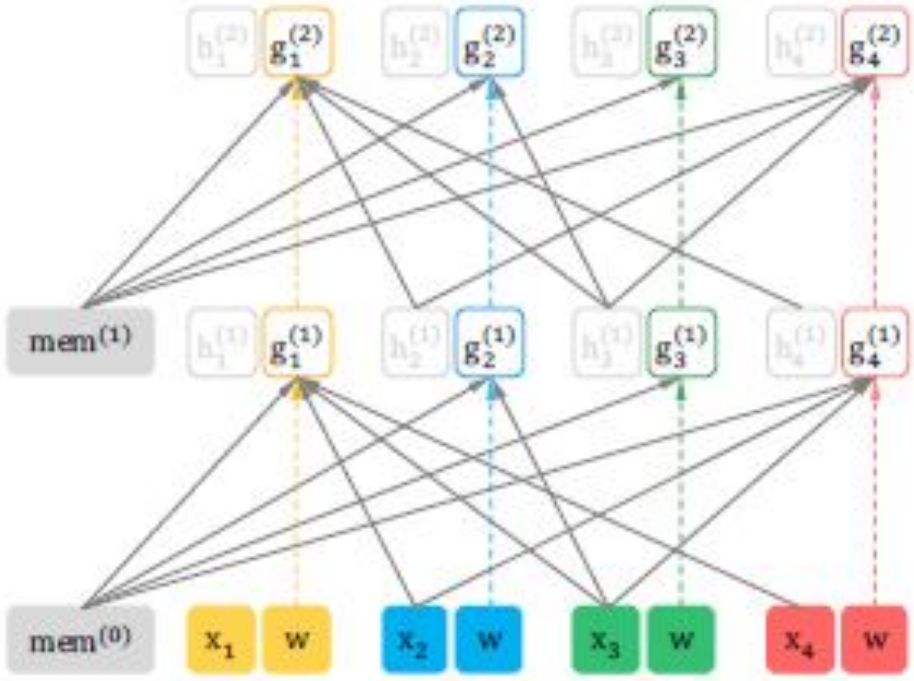


Joint View of the Content Stream
(Factorization order: 3 → 2 → 4 → 1)

Content Stream visualization

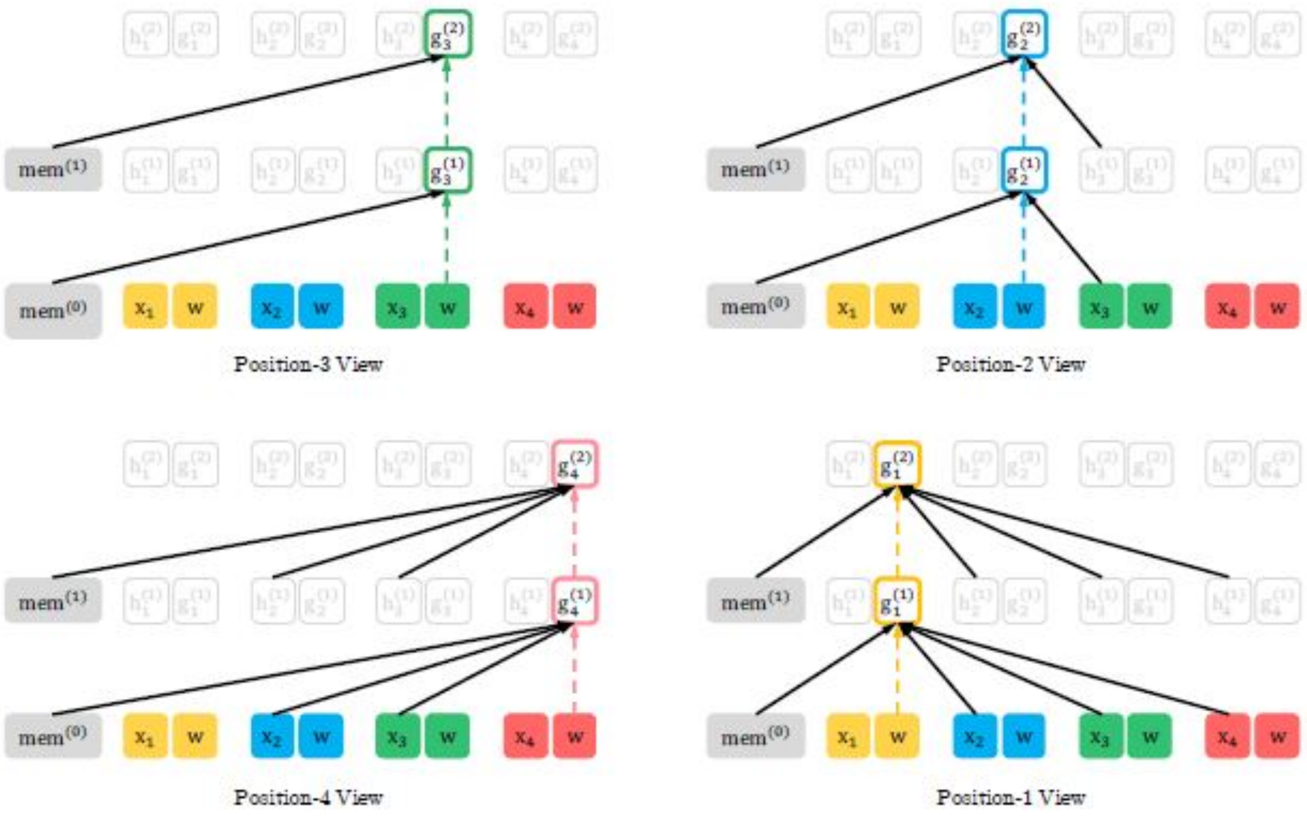


Query Stream visualization



Joint View of the Query Stream
(Factorization order: 3 → 2 → 4 → 1)

Query Stream visualization



Split View of the Query Stream
(Factorization order: 3 → 2 → 4 → 1)

Design choices

- Partial prediction

Transformer-XL

- Relative position encoding
- Recurrence mechanism

Experiments

RACE	Accuracy	Middle	High
GPT [25]	59.0	62.9	57.4
BERT [22]	72.0	76.6	70.1
BERT+OCN* [28]	73.5	78.4	71.5
BERT+DCMN* [39]	74.1	79.5	71.8
XLNet	**81.75**	**85.45**	**80.21**

RACE Dataset
(reading comprehension)

SQuAD1.1	EM	F1	SQuAD2.0	EM	F1
Dev set results without data augmentation					
BERT [10]	84.1	90.9	BERT† [10]	78.98	81.77
XLNet	**88.95**	**94.52**	XLNet	**86.12**	**88.79**
Test set results on leaderboard, with data augmentation (as of June 19, 2019)					
Human [27]	82.30	91.22	BERT+N-Gram+Self-Training [10]	85.15	87.72
ATB	86.94	92.64	SG-Net	85.23	87.93
BERT* [10]	87.43	93.16	BERT+DAE+AoA	85.88	88.62
XLNet	**89.90**	**95.08**	XLNet	**86.35**	**89.13**

SQUAD Dataset
(Question answering)

GLUE Benchmark

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**

Experiments

Model	IMDB	Yelp-2	Yelp-5	DBpedia	AG	Amazon-2	Amazon-5
CNN [14]	-	2.90	32.39	0.84	6.57	3.79	36.24
DPCNN [14]	-	2.64	30.58	0.88	6.87	3.32	34.81
Mixed VAT [30, 20]	4.32	-	-	0.70	4.95	-	-
ULMFiT [13]	4.6	2.16	29.98	0.80	5.01	-	-
BERT [35]	4.51	1.89	29.32	0.64	-	2.63	34.17
XLNet	**3.79**	**1.55**	**27.80**	**0.62**	**4.49**	**2.40**	**32.26**

Model	NDCG@20	ERR@20
DRMM [12]	24.3	13.8
KNRM [8]	26.9	14.9
Conv [8]	28.7	18.1
BERT†	30.53	18.67
XLNet	**31.10**	**20.28**

ClueWeb09-B (Document ranking)

Text Classification

Ablation Studies

#	Model	RACE	SQuAD2.0 F1	SQuAD2.0 EM	MNLI m/mm	SST-2
1	BERT-Base	64.3	76.30	73.66	84.34/84.65	92.78
2	DAE + Transformer-XL	65.03	79.56	76.80	84.88/84.45	92.60
3	XLNet-Base ($K = 7$)	66.05	**81.33**	**78.46**	**85.84/85.43**	92.66
4	XLNet-Base ($K = 6$)	66.66	80.98	78.18	85.63/85.12	**93.35**
5	- memory	65.55	80.15	77.27	85.32/85.05	92.78
6	- span-based pred	65.95	80.61	77.91	85.49/85.02	93.12
7	- bidirectional data	66.34	80.65	77.87	85.31/84.99	92.66
8	+ next-sent pred	**66.76**	79.83	76.94	85.32/85.09	92.89

References

- https://arxiv.org/pdf/1706.03762.pdf
- https://arxiv.org/pdf/1802.05365.pdf
- https://arxiv.org/pdf/1801.06146.pdf
- https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf
- https://medium.com/@hyponymous/paper-summary-bert-12af7c89d2e0
- https://medium.com/dissecting-bert/dissecting-bert-part2-335ff2ed9c73
- https://www.lyrn.ai/2018/11/07/explained-bert-state-of-the-art-language-model-for-nlp/
- http://jalammar.github.io/illustrated-bert/
- http://mlexplained.com/2019/01/07/paper-dissected-bert-pre-training-of-deep-bidirectional-transformers-for-language-understanding-explained/