XLNET: GENERALIZED AUTOREGRESSIVE PRETRAINING FOR LANGUAGE UNDERSTANDING

XLNet, one of the natural language processing tools, has had more repercussions than many tools, although it has recently joined the family. Researchers from Carnegie Mellon University and Google AI Brain team presented XLNet in a recent NeurIPS 2019 conference paper, leaving quite an impression on the NLP community. XLNet outperforms BERT on 20 NLP benchmark tasks, usually with large margin, and thus becomes not only exciting for researchers, but also important for NLP practitioners.

XLNet leverages the best of both autoregressive (AR) language modeling and autoencoding (AE), the two most well-known pretraining objectives, while avoiding their limitations. The method can be applied to a variety of NLP downstream language tasks including question answering, sentiment analysis, natural language inference, document ranking and so on.

Considered as one of the 2019's most important developments in NLP, XLNet combines the autoregressive language model, Transformer-XL, and bidirectional capability of BERT to unleash the power of this important language modeling tool.

HIGH LEVEL TECHNOLOGY

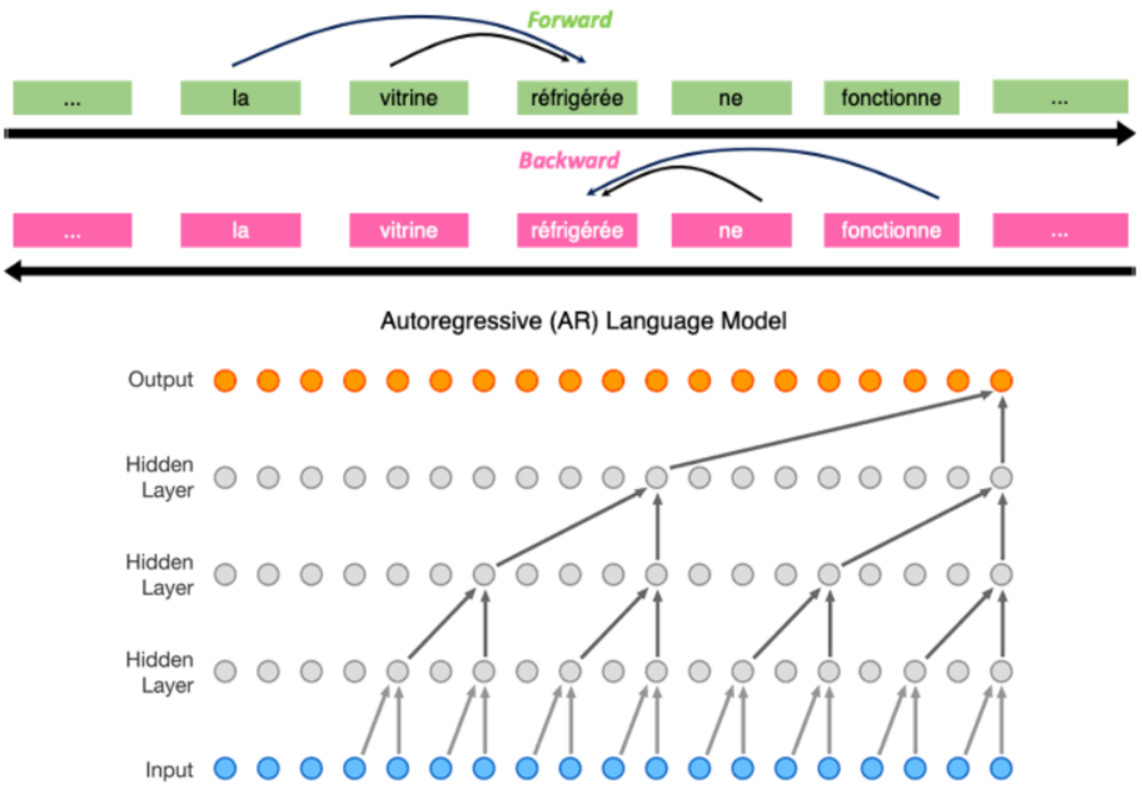
Natural language processing (NLP) tools are becoming increasingly important in machine translation, reading comprehension and summarization, question answering, and sentiment analysis. The typical approach has been using supervised learning on datasets that are task-specific. In recent years, unsupervised representation learning methods represented by BERT (Bidirectional Encoder Representations from Transformers), CoVe (Context Vector), ELMo (Embeddings from Language Models) have gained increasing attention. Language modeling is essentially predicting the next word in a sentence given previous words. These language modeling methods pretrain neural networks on large-scale unlabeled text corpora before fine-tuning the model for subsequent tasks. In other words, language modeling includes two phases, the pretraining phase and fine-tuning phase.

XLNET COMBİNES THE ADVANTAGE of AR and AE

For the pretraining phase, the two most successful architectures are autoregressive (AR) language modeling and autoencoding (AE). Before seeing how XLNet achieves unprecedented performances, we will dive into the two aforementioned pretraining perspectives whose advantages XLNet combines. Here, we will see how they work and what are the limitations:

AUTOREGRESSİVE (AR) LANGUAGE MODELİNG

In conventional AR models, unidirectional context either in the forward or backward direction in a text sequence is encoded. It is useful for generative NLP tasks that generate context in the forward direction. However, AR falls short in case when bidirectional context needs to be utilized simultaneously. This could become problematic especially with downstream language understanding task where bidirectional context information is required.



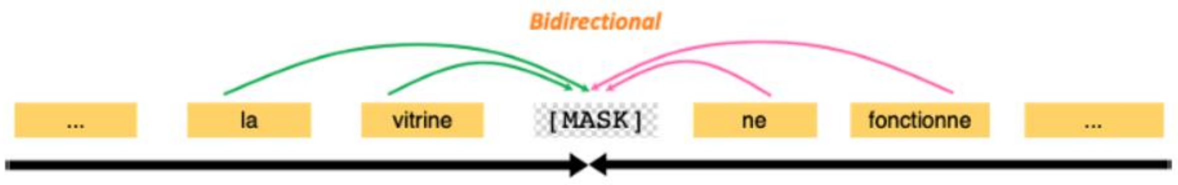
Google DeepMind’s WaveNet illustrates the feed-forward fashion of an autoregressive model.

In AR, a parametric model such as a neural network is trained to model the joint probability distribution of a text corpus, for either a forward product or a backward product conditioned on the words before or after the predicted token.

AUTOENCODİNG (AE) LANGUAGE MODEL

An Autoencoding based model has the capability of modeling bidirectional contexts by reconstructing the original text from corrupted input ([MASK]). AE model is thus better than AR model when it comes to better capturing bidirectional context.

A notable example of AE is BERT that is based on denoising autoencoding. However, it suffers from a pretrain-finetune discrepancy arising from the dependency between the masked tokens and unmasked ones. In particular, [MASK] used in the pretraining stage is absent from the real data used at downstream tasks including the fine-tuning stage. For high-order, long-range dependency characteristics in natural language, BERT oversimplifies the problem by assuming predicted tokens (masked in the input) are independent of each other as long as the unmasked tokens are given.

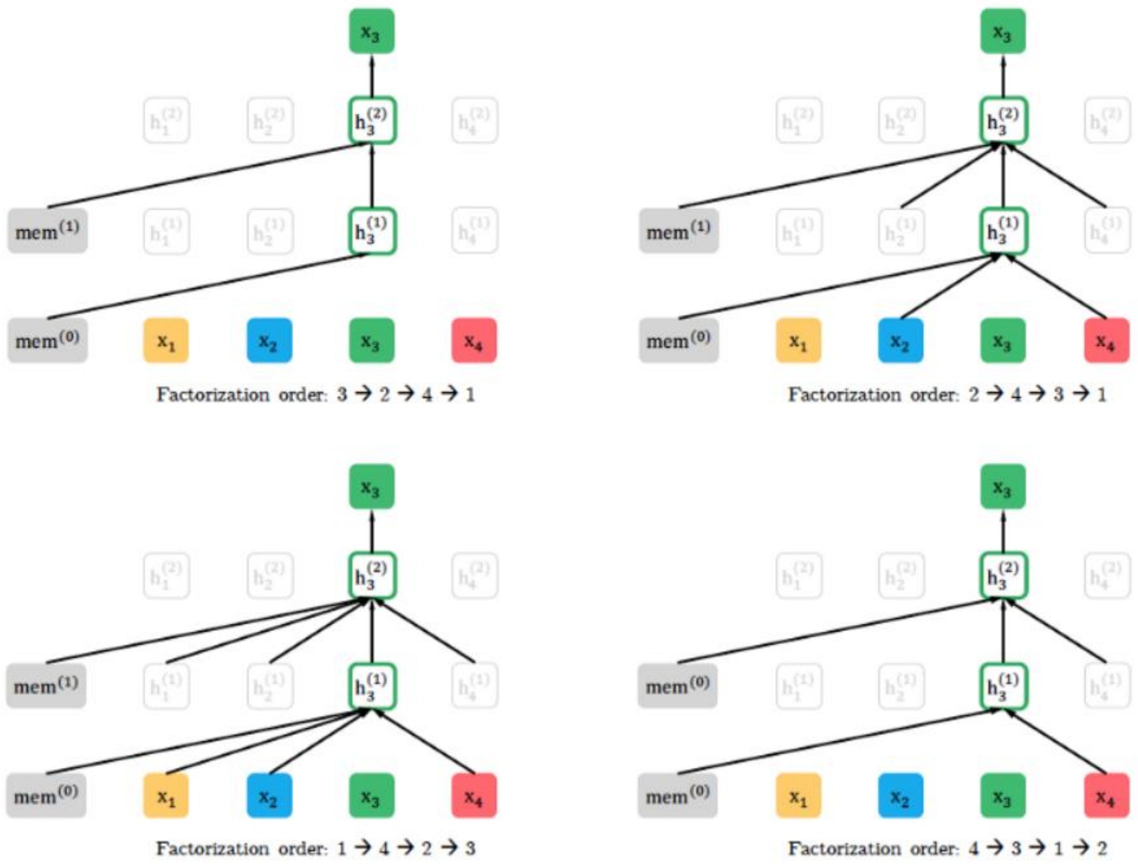


Autoencoding (AE) Language Model

While AR can estimate the probability of either a forward or backward product in the form of conditional probability distribution, BERT cannot model the joint probability using the product rule due to its independence assumption for the masked tokens.

HOW DOES XLNET DIFFER FROM CONVENTİONAL AR and AE (BERT)?

The authors of XLNet propose to retain the benefits of AR language model while having it learn from bidirectional context as AE models (e.g., BERT) during the pretraining phase. The interdependency between tokens will be preserved, unlike in BERT. The proposed new objective is called “Permutation Language Modeling.”



x_3
$h_1^{(2)}$
$h_2^{(2)}$
$h_3^{(2)}$
$h_4^{(2)}$
mem$^{(1)}$
$h_1^{(1)}$
$h_2^{(1)}$
$h_3^{(1)}$
$h_4^{(1)}$
mem$^{(0)}$
x_1
x_2
x_3
x_4
Factorization order: 3 → 2 → 4 → 1
Factorization order: 2 → 4 → 3 → 1
Factorization order: 1 → 4 → 2 → 3
Factorization order: 4 → 3 → 1 → 2

Illustration of the permutation language modeling objective from the paper. The "mem" node has the read-only activations from the previous tokens without back-propagating to them.

The basic idea behind this modeling is "Permutations". In this illustration above from the paper, we see an example for predicting the x3 token given the same input sequence x1 →x2 →x3 →x4 with 4 tokens. For a sentence with N tokens, there will be N! permutations. In this case, there are a total of 24 permutations, and the illustration demonstrates 4. In each permutation/factorization order, the (t-1) tokens that proceed the token of interest (at t-th position) will be feed forward into the hidden layers to predict the t-th token. In this example, we are predicting x3. The benefit of using permutation language modeling is to capture information from both sides by varying the factorization order. Note that the input sequence order is not randomly permuted since we need to preserve natural order during finetuning. Only the factorization order is permuted.

Here, the goal is to maximize the expected log-likelihood of a word sequence considering all the possible permutations of the factorization order. The following permutation language modeling objective formalizes the idea, where the first (t-1) tokens in the factorization order is used to predict the t-th token.

$$\max_{\theta} \quad \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\left[\sum_{t=1}^{T} \log p_\theta(x_{z_t} \mid \mathbf{x}_{\mathbf{z}_{<t}})\right]$$

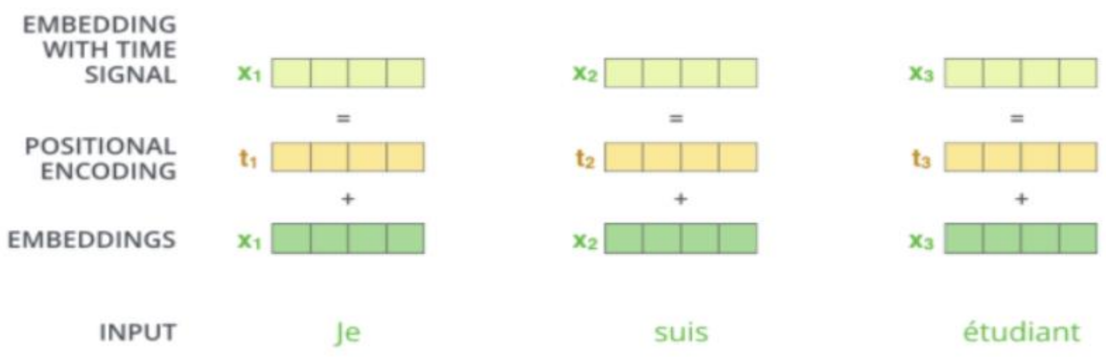
Permutation language modeling objective. Possible permutation: Z_T.

- Factorization orders: z~Z_T
- Likelihood function p_θ
- x_{z_t}: the t-th token in the factorization
- x_{z<t}: first (t-1) tokens before t-th token

However, as the paper has pointed out, naive implementation with standard Transformer parameterization won't work. The standard Transformer does not fulfill the following two requirements:

1. Predict token x_{z_t} at z_t position based on only the position z_t, not the content of x_{z_t}.
2. Predict token x_{z_t} with all the contents of the tokens before x_{z_t} encoded.

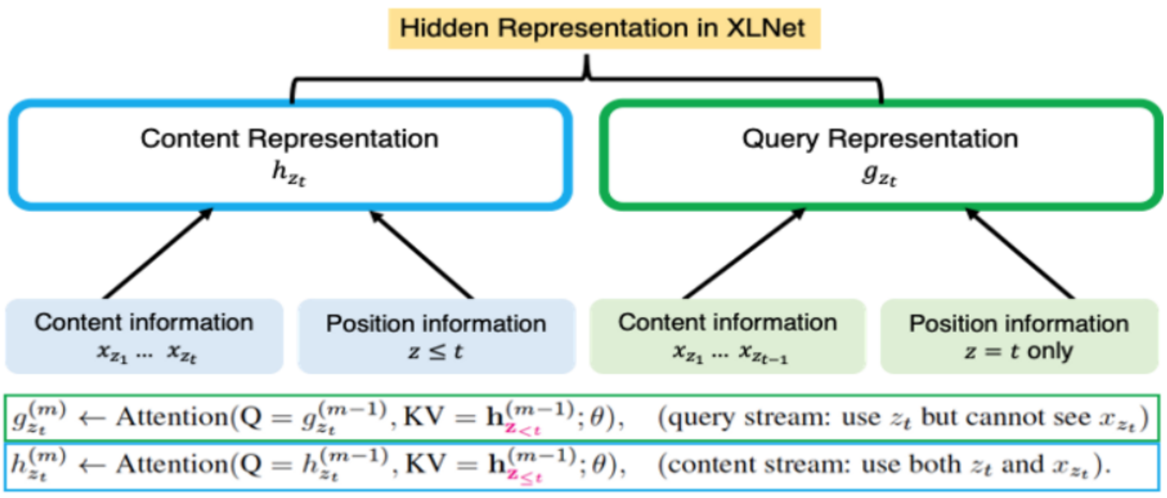
One key property of the Transformer is that it includes position encoding into the token embedding. Hence, the position information is inseparable from the token embedding. In the case of permutation language modeling, this Transformer property poses a problem since the position information remains the same even when the sequence is shuffled, producing identical model prediction for different target positions.



Different from BERT and other transformers that combines position embedding and content embedding for prediction, XLNet predicts the next-token distribution by taking into account the target position z_t as input. This brings us to the Two-Stream Self-Attention architecture XLNet proposes.

XLNET USES TWO-STREAM SELF-ATTENTİIN ARCHİTECTURE to be TARGET-AWARE

Two-Stream Self-Attention architecture is employed to address the problems traditional Transformer poses. Just as what the name suggests, the architecture consists of two different types of self-attention. The first one is content stream representation, same as the standard self-attention in Transformer that considers both content (x_{z_t})and position information (z_t). The other one is query representation, it essentially replaces the [MASK] from BERT, learned by query stream attention to predict x_{z_t} only with position information but not its content. Only the position information of the target token and context information before the token is available.


Hidden Representation in XLNet
Content Representation h_{z_t}
Query Representation g_{z_t}
Content information $x_{z_1} \dots x_{z_t}$
Position information $z \leq t$
Content information $x_{z_1} \dots x_{z_{t-1}}$
Position information $z = t$ only

$$g_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = g_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{<t}}^{(m-1)}; \theta), \quad \text{(query stream: use } z_t \text{ but cannot see } x_{z_t}\text{)}$$

$$h_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = h_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{\leq t}}^{(m-1)}; \theta), \quad \text{(content stream: use both } z_t \text{ and } x_{z_t}\text{)}.$$

XLNet with 2 sets of hidden representation. The content representation uses content stream as standard Transformer(-XL). The query representation helps to compute target-position-aware next-token distribution.

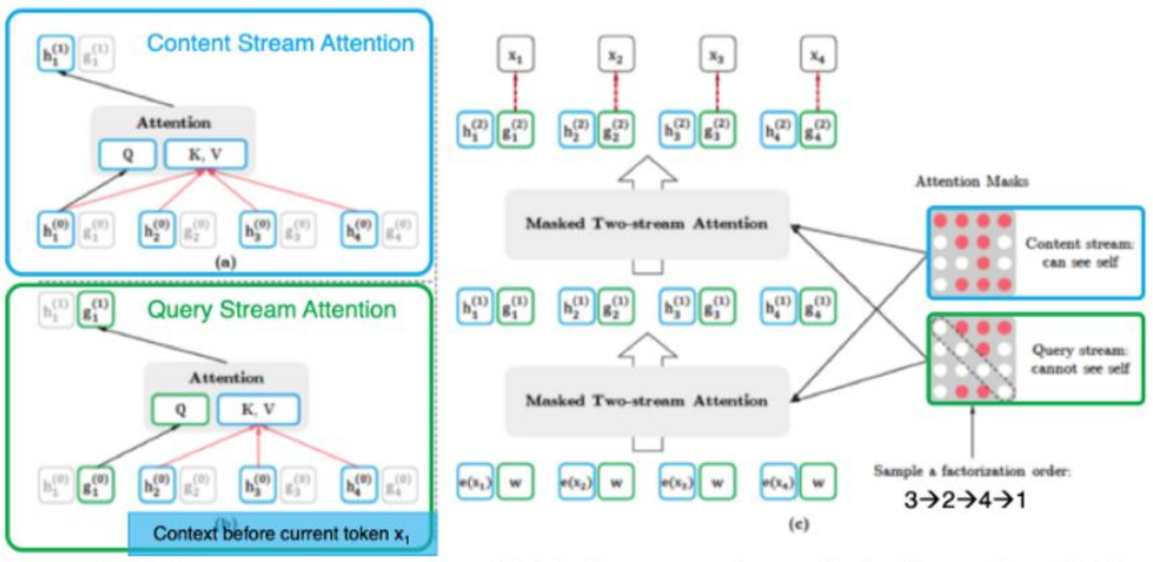
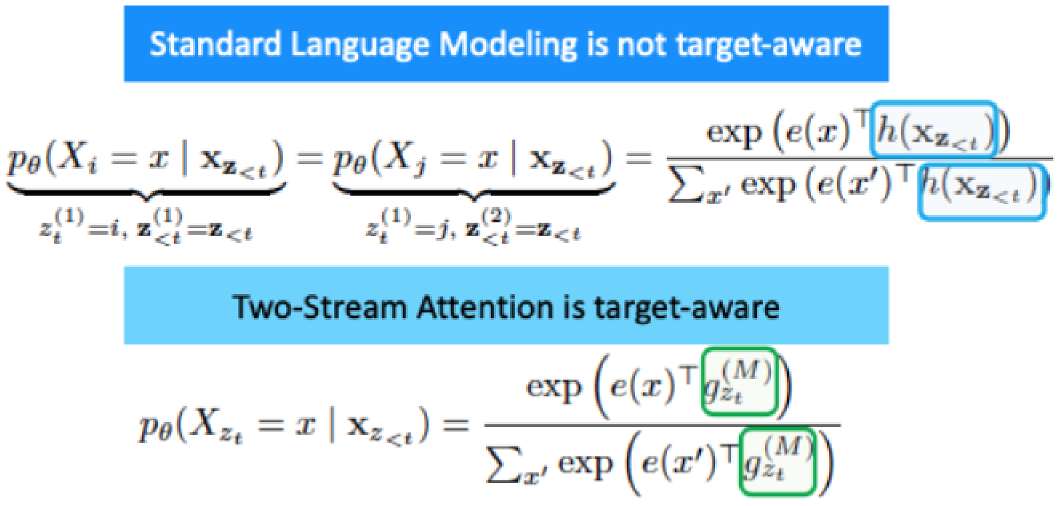

Content Stream Attention
Attention
Q
K, V
(a)
Query Stream Attention
Attention
Q
K, V
Context before current token x_1
Masked Two-stream Attention
Masked Two-stream Attention
Attention Masks
Content stream: can see self
Query stream: cannot see self
Sample a factorization order:
3→2→4→1
(c)

Figure 1: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content x_{z_t}. (c): Overview of the permutation language modeling training with two-stream attention.

The end result of Two-Stream attention is target-aware prediction distribution. The main difference between XLNet and BERT is that XLNet is not based on data corruption as BERT does, so it can avoid BERT's limitations arising from masking, described earlier in AE model.



Standard Language Modeling is not target-aware

$$\underbrace{p_\theta(X_i = x \mid \mathbf{x}_{\mathbf{z}_{<t}})}_{z_t^{(1)}=i,\ \mathbf{z}_{<t}^{(1)}=\mathbf{z}_{<t}} = \underbrace{p_\theta(X_j = x \mid \mathbf{x}_{\mathbf{z}_{<t}})}_{z_t^{(1)}=j,\ \mathbf{z}_{<t}^{(2)}=\mathbf{z}_{<t}} = \frac{\exp\left(e(x)^\top \boxed{h(\mathbf{x}_{\mathbf{z}_{<t}})}\right)}{\sum_{x'} \exp\left(e(x')^\top \boxed{h(\mathbf{x}_{\mathbf{z}_{<t}})}\right)}$$

Two-Stream Attention is target-aware

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

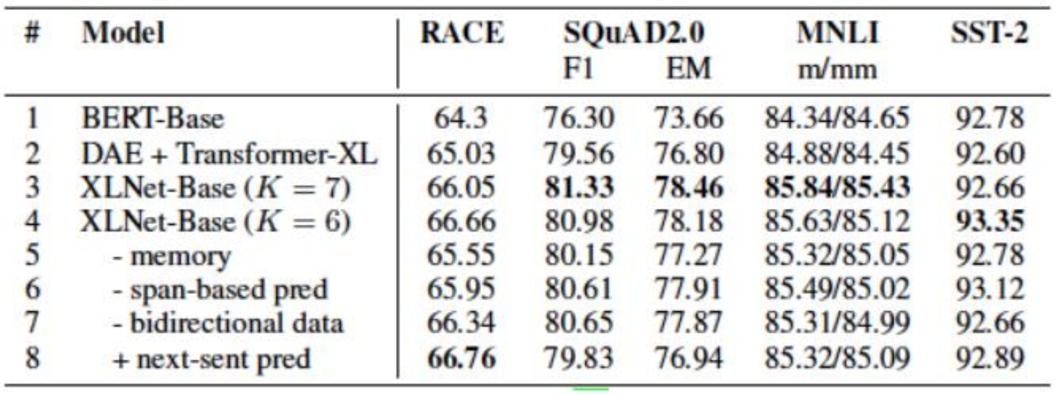
Comparison of standard transformer and target-aware two-stream self-attention.

XLNET RESULTS

XLNet combines bidirectional capability of BERT and the autoregressive technology of Transformer-XL to achieve substantial improvement; it beats BERT in more than a dozen tasks. Empirically speaking, XLNet outperforms BERT in:

- GLUE language understanding tasks
- Reading comprehension tasks (SQuAD and RACE)
- Text classification tasks (Yelp and IMDB)
- ClubWeb09-B document ranking task

To see which design choice for XLNet affects the performance more, the authors carried out an interesting ablation study on Wikipedia and the BooksCorpus. The results are shown in the following table.



#	Model	RACE	SQuAD2.0		MNLI	SST-2
			F1	EM	m/mm	
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

Ablation studies shows superior performance of XLNet compared to BERT.

The pretrained models are evaluated on downstream tasks to justify the design choices in XLNet. In particular, Transformer-XL backbone and the permutation LM play a heavy role in improving XLNet's performance over that of BERT.

- RACE (ReAding Comprehension from Examinations) dataset is a challenging benchmark for long text understanding.

- SQuAD (Stanford Question Answering Dataset) is a large-scale reading comprehension dataset with paragraphs and corresponding questions.

- GLUE (General Language Understanding Evaluation) dataset consists of 9 natural language understanding tasks. GLUE/SST-2 (Stanford Sentiment Treebank) consists of sentences from movie reviews and sentiments. GLUE/MNLI (Multi-Genre Natural Language Inference Corpus) is for entailment analysis.

Beside NLP, XLNet can likely unleash its power in computer vision tasks and reinforcement learning problems.