Reformer: The Efficient Transformer

Nikita Kitaev, Lukasz Kaiser, Anselm Levskaya

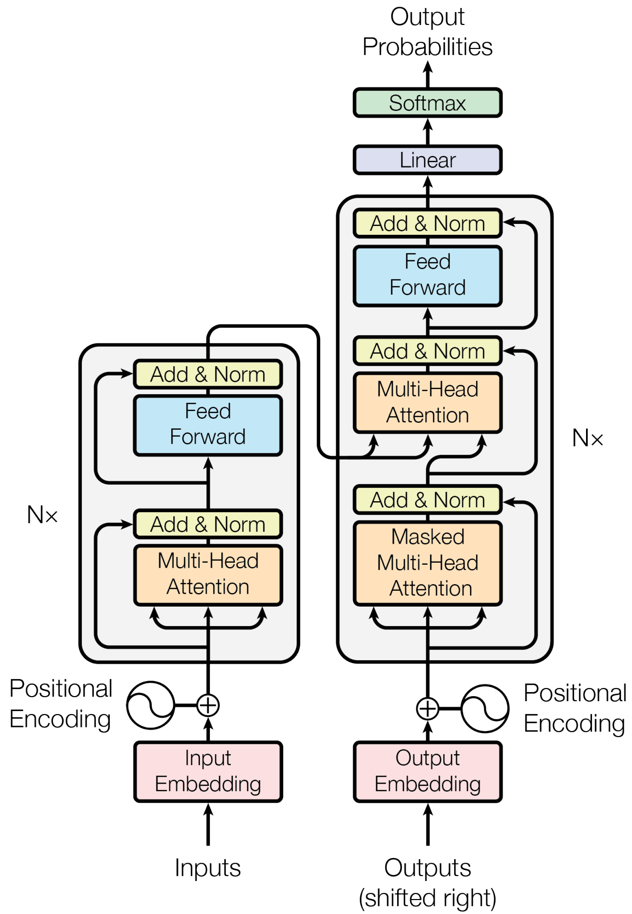
HSE
Pavel Yurlov

5 March 2020

Outline

Introduction

The Transformer (Vaswani et al., 2017)



Transformer's Pros and Cons

- (+) state-of-the-art
- (−) requires too many resources

The Reformer

Modifications:

(1) reversible layers

(2) splitting activations inside feed-forward layers

(3) approximate attention computation

LSH-based attention

Multi-head self-attention in Transformer

(1) $\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$

(2) $\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$,
where $\text{head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

(3) The main problem is the QK^T term:
$[batch\_size, length, length]$

Solution

Attention scores calculation:

softmax

$\Downarrow$

we only need the largest elements for an approximation

$\Downarrow$

we have to find the nearest neighbours

$\Downarrow$

we should use locality-sensitive hashing (Andoni et al., 2015)

Locality-sensitive hashing

Goals:

(1) Nearby vectors get the same hash with high probability.

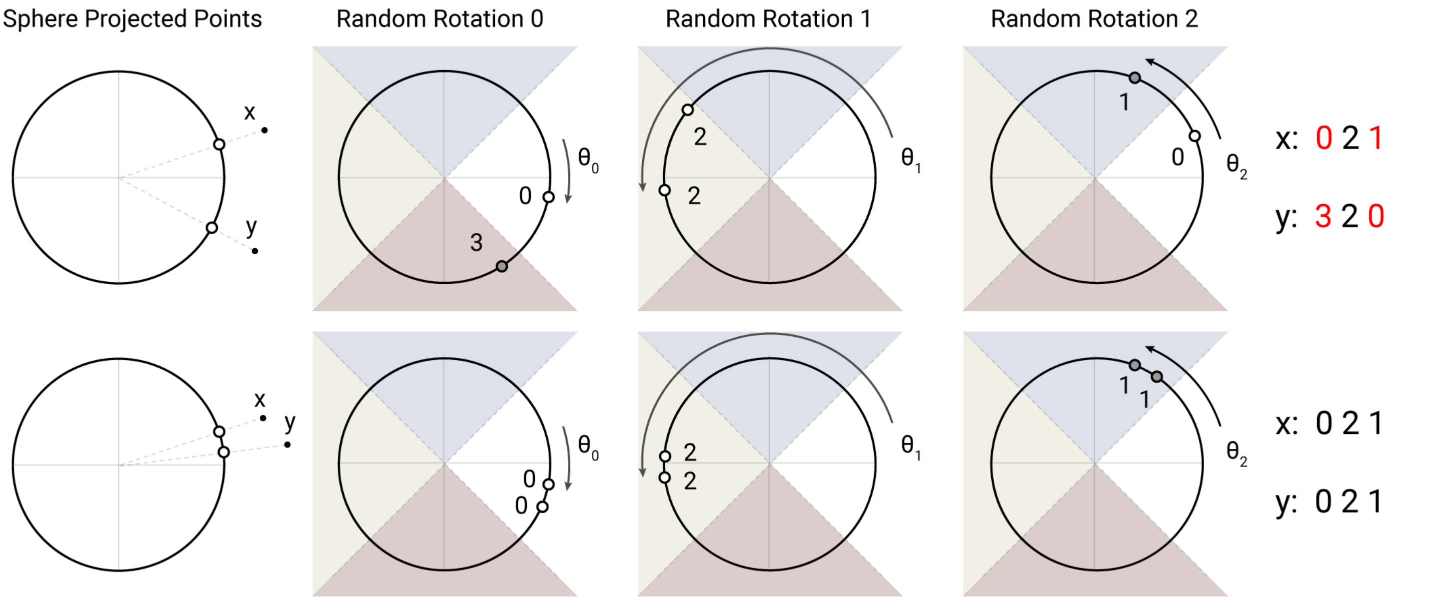
(2) Hash-buckets are of similar size with high probability.

Algorithm (b hash-buckets):

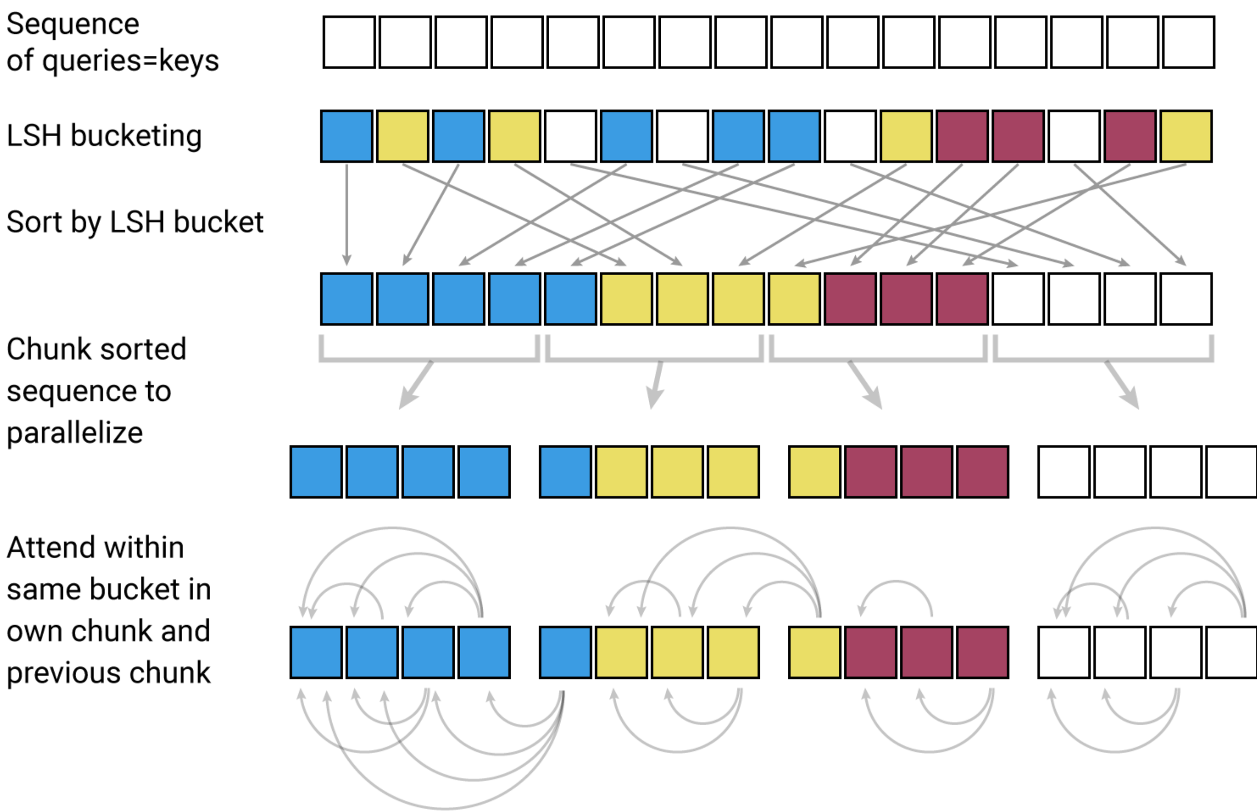
(1) Fix a random matrix R of size $[d_k, b/2]$.

(2) $h(x) = \text{argmax}([xR; -xR])$

Locality-sensitive hashing



Attention with LSH: the algorithm



Accuracy comparison

Table 2: Accuracies on the duplication task of a 1-layer Transformer model with full attention and with locality-sensitive hashing attention using different number of parallel hashes.

Train \ Eval	Full Attention	LSH-8	LSH-4	LSH-2	LSH-1
Full Attention	100%	94.8%	92.5%	76.9%	52.5%
LSH-4	0.8%	100%	99.9%	99.4%	91.9%
LSH-2	0.8%	100%	99.9%	98.1%	86.8%
LSH-1	0.8%	99.9%	99.6%	94.8%	77.9%

Reversible Transformer

Further modifications

We need further space optimisation:

(1) Reversible blocks in order not to store many layers of activations

(2) Processing feed-forward activations in chunks

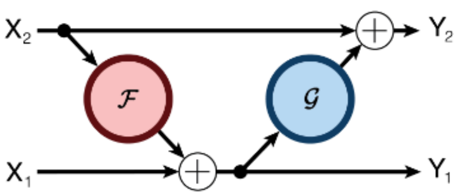
RevNet block (Gomez et al., 2017)

$$(x_1, x_2) \to (y_1, y_2)$$

Forward:

$$y_1 = x_1 + \mathcal{F}(x_2)$$

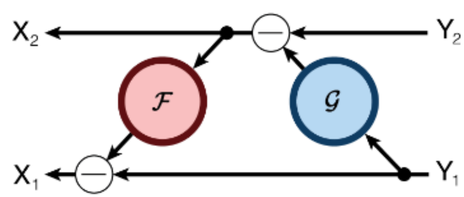
$$y_2 = x_2 + \mathcal{G}(y_1)$$



Backward:

$$x_2 = y_2 - \mathcal{G}(y_1)$$

$$x_1 = y_1 - \mathcal{F}(x_2)$$



RevNet in Transformer

Forward:

$$Y_1 = X_1 + \mathrm{Attention}(X_2)$$
$$\mathrm{Y}_2 = X_2 + \mathrm{FeedForward}(Y_1)$$

Backward:

$$X_2 = Y_2 - \mathrm{FeedForward}(Y_1)$$
$$\mathrm{X}_1 = Y_1 - \mathrm{Attention}(X_2)$$

Chunking activations

Since computations in feed-forward layers are independent across positions in a sequence, they can be split into c chunks:

$$Y_2 = [Y_2^{(1)}; \dots ; Y_2^{(c)}] =$$

$$= [X_2^{(1)} + \mathrm{FeedForward}(Y_1^{(1)}); \dots ; X_2^{(c)} + \mathrm{FeedForward}(Y_1^{(c)})]$$

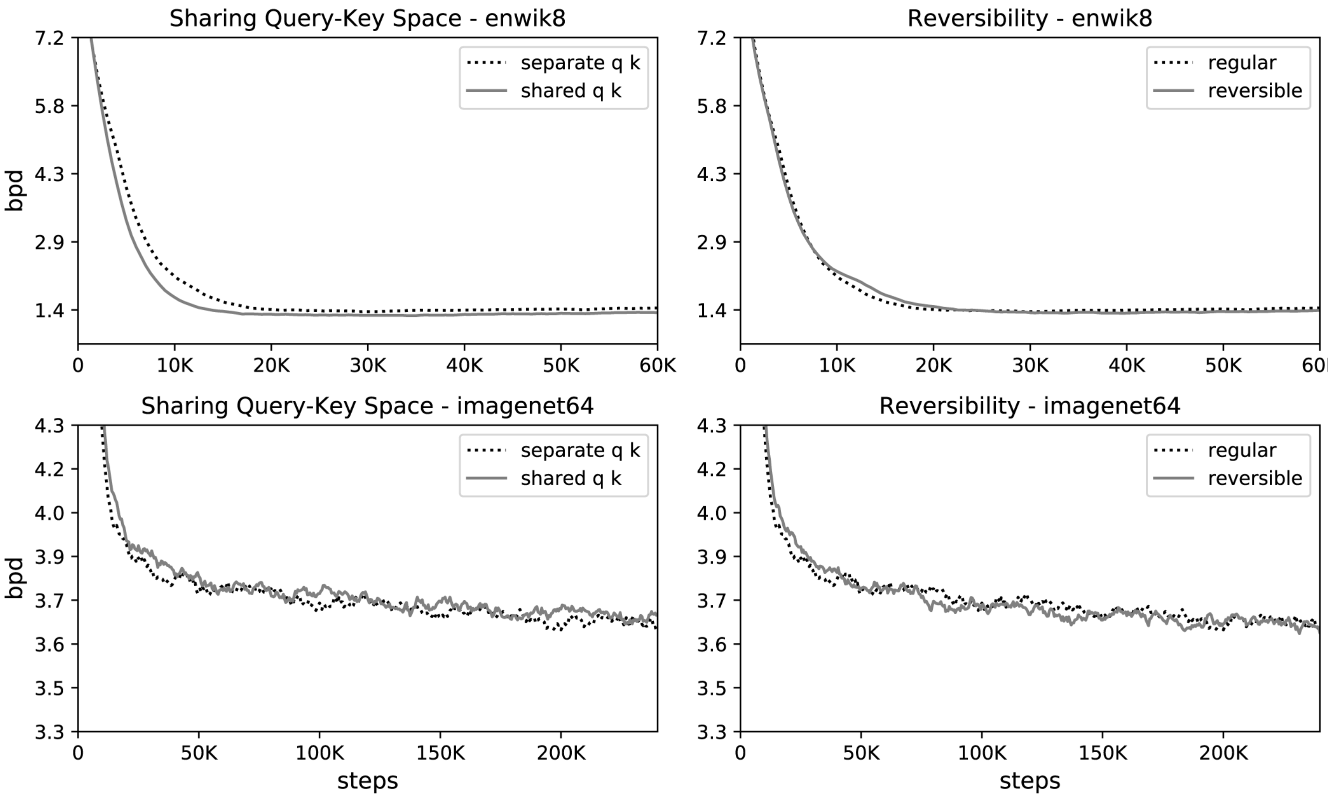
Complexity comparison

Table 3: Memory and time complexity of Transformer variants. We write d_{model} and d_{ff} for model depth and assume $d_{ff} \geq d_{model}$; b stands for batch size, l for length, n_l for the number of layers. We assume $n_c = l/32$ so $4l/n_c = 128$ and we write $c = 128^2$.

Model Type	Memory Complexity	Time Complexity
Transformer	$\max(bld_{ff}, bn_h l^2)n_l$	$(bld_{ff} + bn_h l^2)n_l$
Reversible Transformer	$\max(bld_{ff}, bn_h l^2)$	$(bn_h ld_{ff} + bn_h l^2)n_l$
Chunked Reversible Transformer	$\max(bld_{model}, bn_h l^2)$	$(bn_h ld_{ff} + bn_h l^2)n_l$
LSH Transformer	$\max(bld_{ff}, bn_h ln_r c)n_l$	$(bld_{ff} + bn_h n_r lc)n_l$
Reformer	$\max(bld_{model}, bn_h ln_r c)$	$(bld_{ff} + bn_h n_r lc)n_l$

Results

Sharing QK and reversible layers

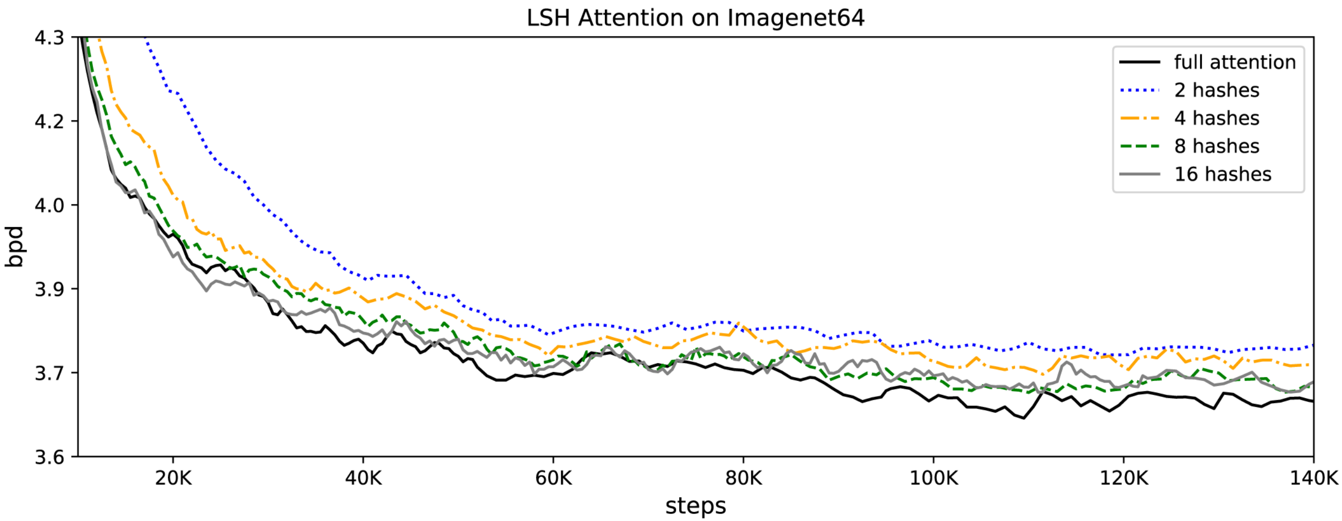


Reversible layers in machine translation

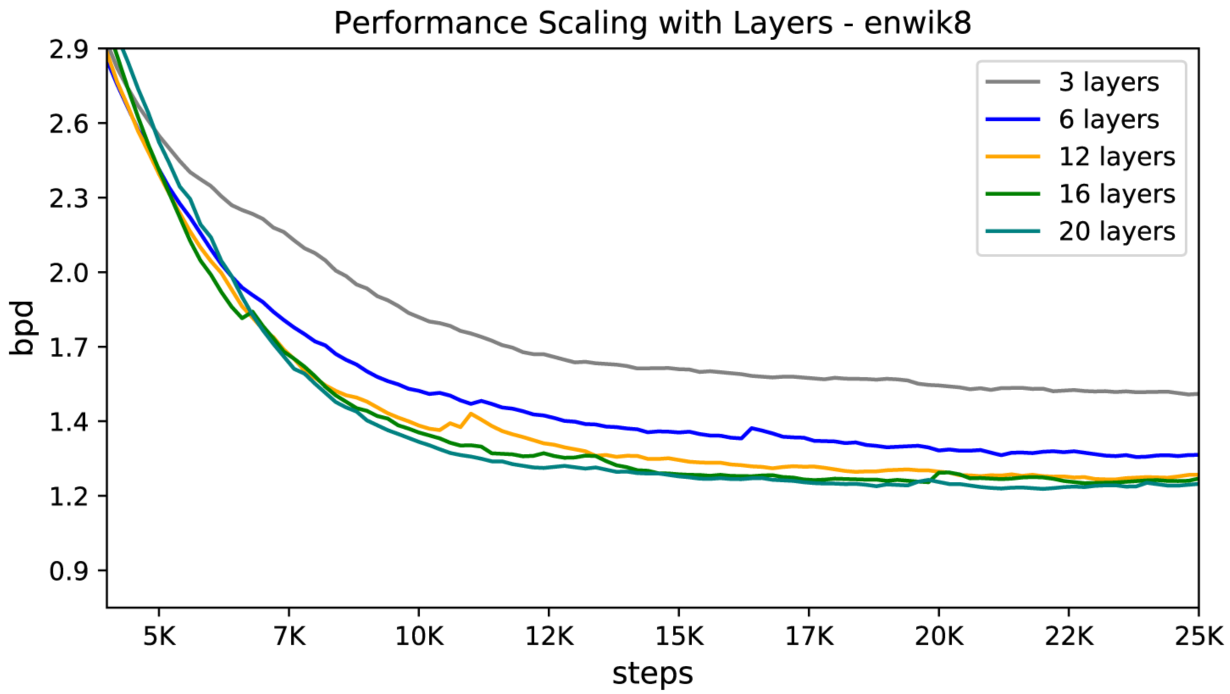
Model	BLEU	*sacreBLEU* *Uncased*[3]	*sacreBLEU* *Cased*[4]
Vaswani et al. (2017), base model	27.3		
Vaswani et al. (2017), big	28.4		
Ott et al. (2018), big	29.3		
Reversible Transformer (base, 100K steps)	27.6	*27.4*	*26.9*
Reversible Transformer (base, 500K steps, no weight sharing)	28.0	*27.9*	*27.4*
Reversible Transformer (big, 300K steps, no weight sharing)	29.1	*28.9*	*28.4*

Scores on newstest2014 for WMT English-German

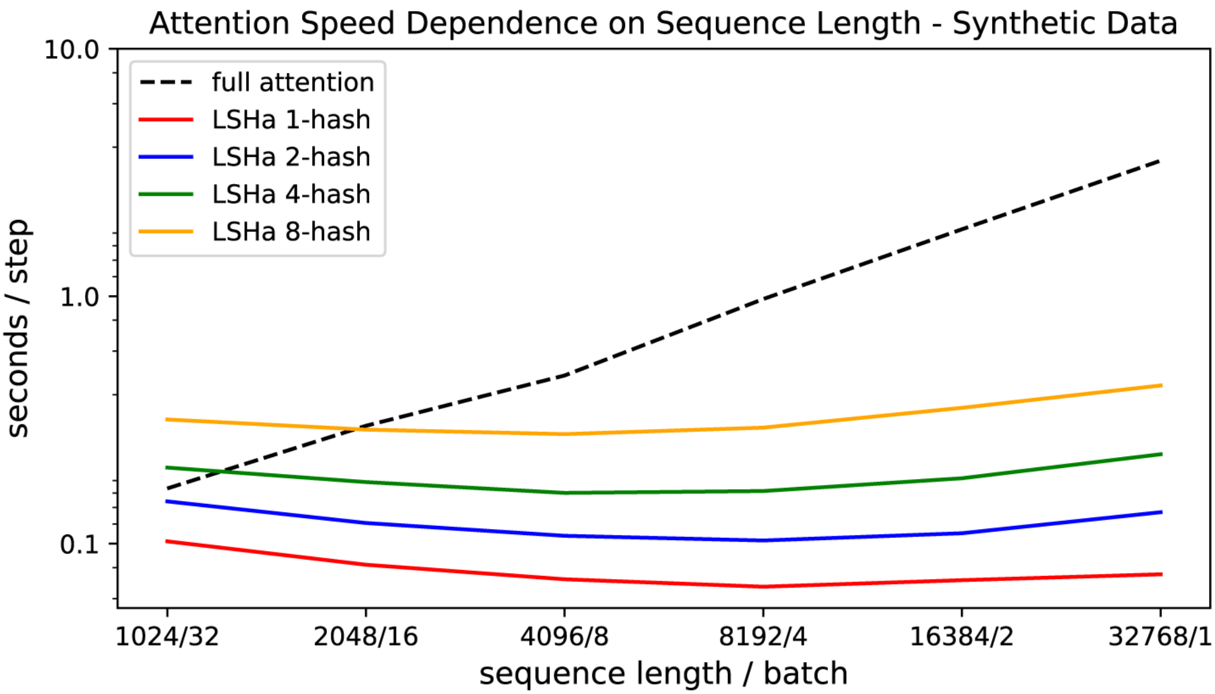
LSH attention 1



LSH attention 2



LSH attention 3



Conclusions

The Reformer:

(1) Transformer with LSH-based attention and memory optimisations
(2) Performace on par with Transformer models
(3) More efficient

Questions

(1) Что такое locality-sensitive hashing? Запишите формулу схемы хеширования, используемую в статье, поясните обозначения.
(2) Опишите алгоритм вычисления внимания в статье.
(3) Зачем нужны обратимые слои? Запишите формулу прямого и обратного прохода по ним.

References

- Kitaev et al., *Reformer: The Efficient Transformer*: https://openreview.net/forum?id=rkgNKkHtvB
- Vaswani et al., *Attention Is All You Need*: https://arxiv.org/abs/1706.03762
- Andoni et al., *Practical and Optimal LSH for Angular Distance*: https://arxiv.org/abs/1509.02897
- Gomez et al., *The Reversible Residual Network: Backpropagation Without Storing Activations*: https://arxiv.org/abs/1707.04585