Attention Mechanism dan Transformer

The team

Nara Surya (Presenter)

Ketua Divisi Smart Technology Tecart

Jana

Anggota Divisi Smart Technology Tecart

Gede Ocha

Anggota Divisi Smart Technology Tecart

Recap Workshop week 3

1. Apa itu Recurrent Neural Network
2. Permasalahan Permasalahan dalam RNN
3. Architecture RNN
4. Mengklasifikasi Sentiment

Materi Hari ini

1. Apa itu Attention Mechanism
2. Transformer
3. BERT
4. Mengklasifikasi Sentiment

Available :

On Recording

https://github.com/NaraSurya/Deep-Learning-Workshop-Tecart-2020

Attention Mechanism

Attention Mechanism

Attention Mechanism memberikan output vector yang berisi Attention-score pada sebuah element A terhadap element element lainnya (input). attention-score merupakan score hubungan/korelasi seberapa kuat hubungan element A dengan element lainnya , sehingga kita dapat menghasilkan vector representasi yang baru dengan menjumlahkan hasil kali pasangan element dengan attention score tersebut. vector representasi tersebut dapat kita gunakan dalam membantu memprediksi target.

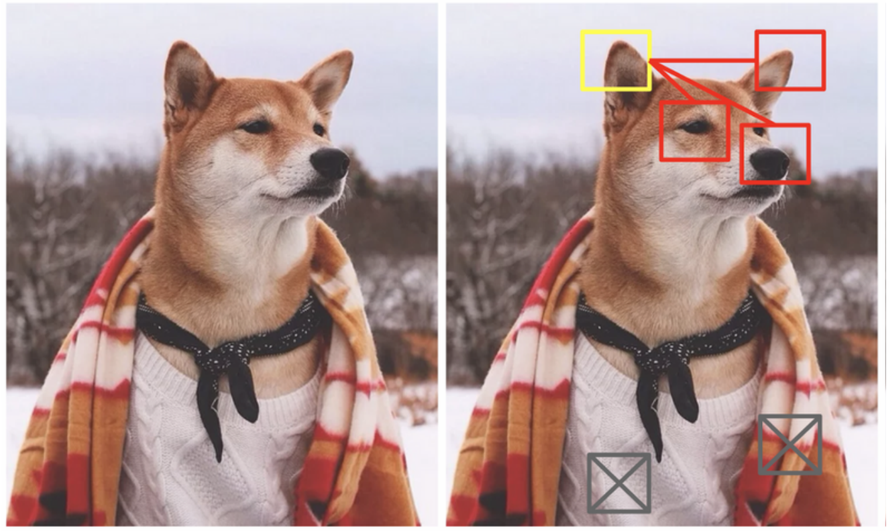
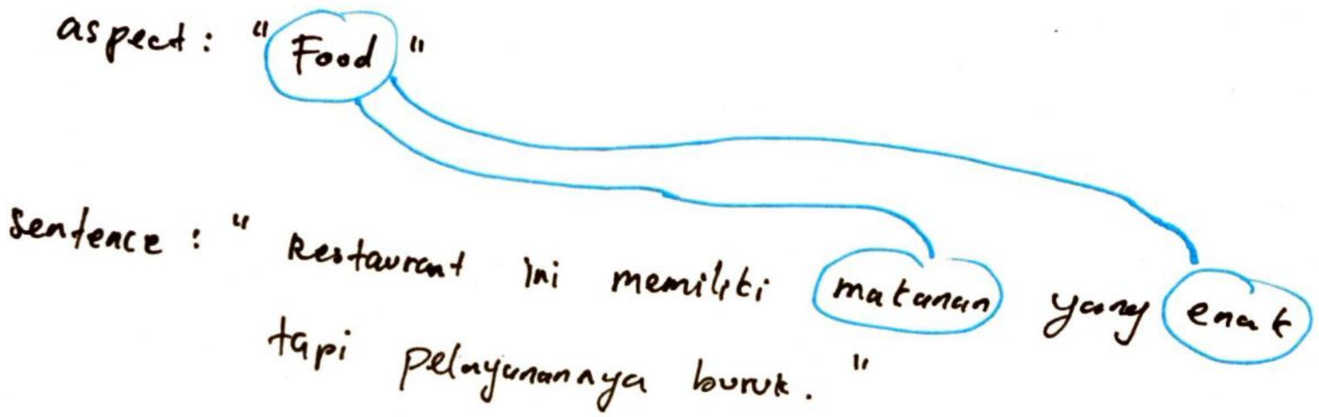
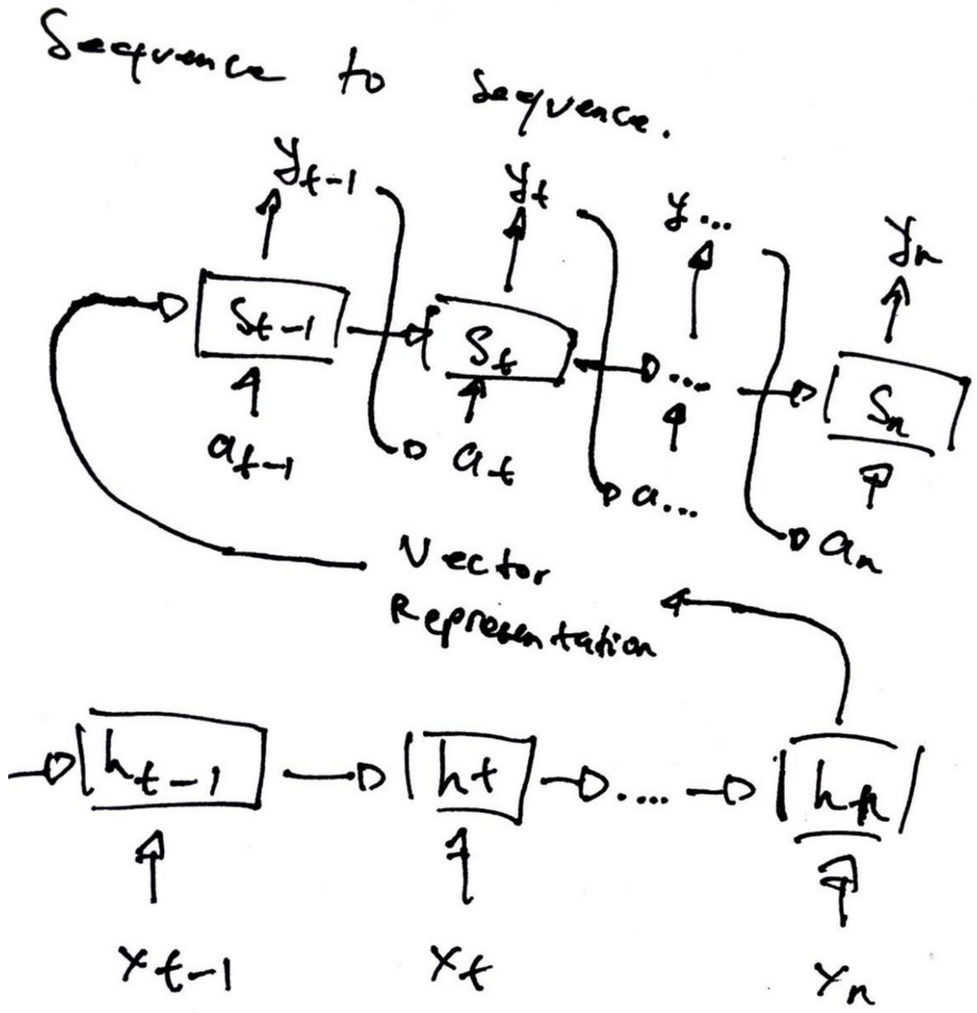


Image By Lil'Log



What's Wrong With Seq2Seq



1. hanya menggunakan vector representasi pada hidden state terakhir sebagai Decoder initial hidden state
2. Vector representation memiliki panjang yang tetap sehingga tidak capatible untuk mengingat kalimat yang panjang
3. Vector representation hanyalah summary dari inputan , tentu saja kita kehilangan detail detail penting yang ada pada inputan

Machine Translation and Attention

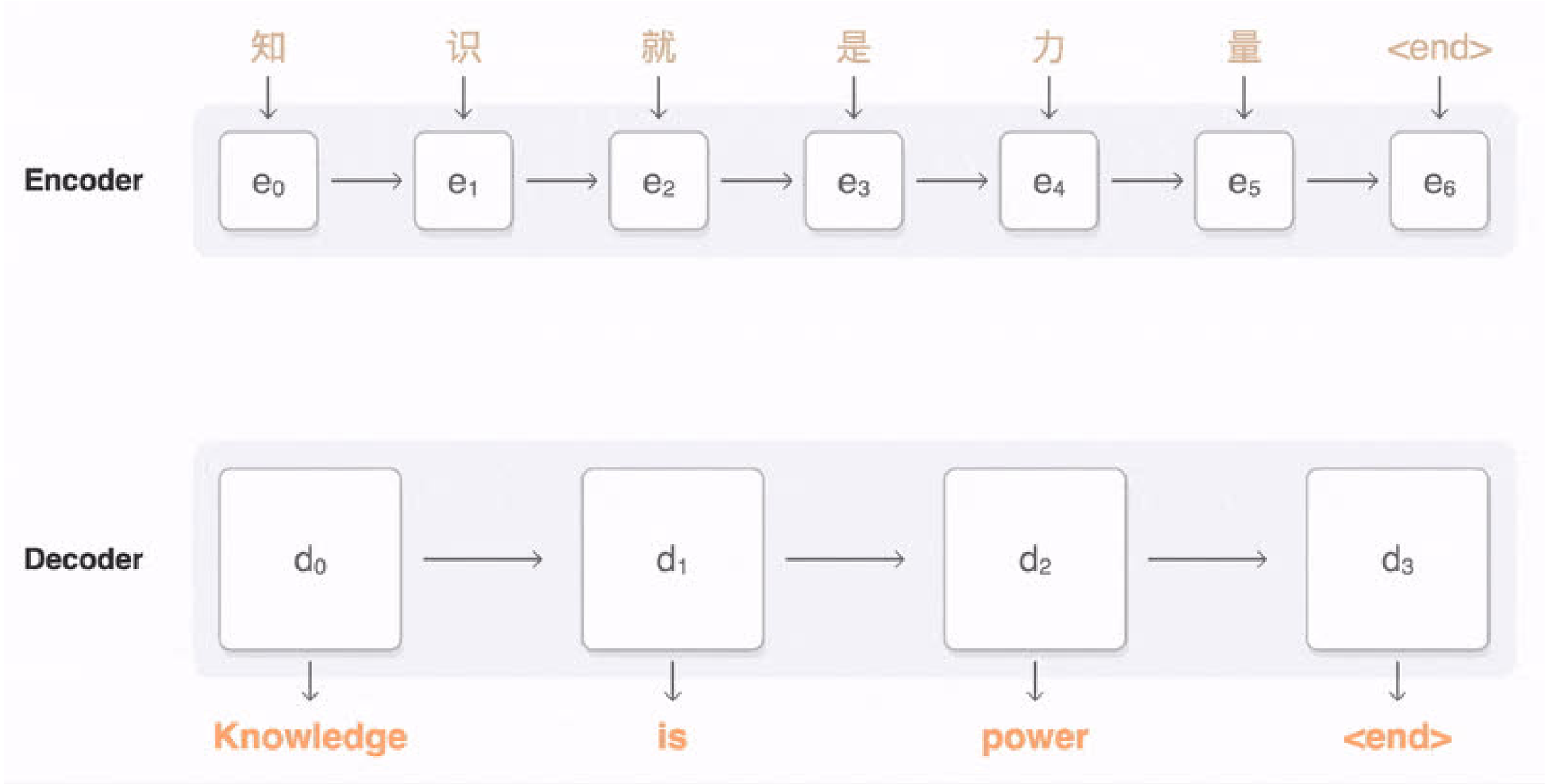
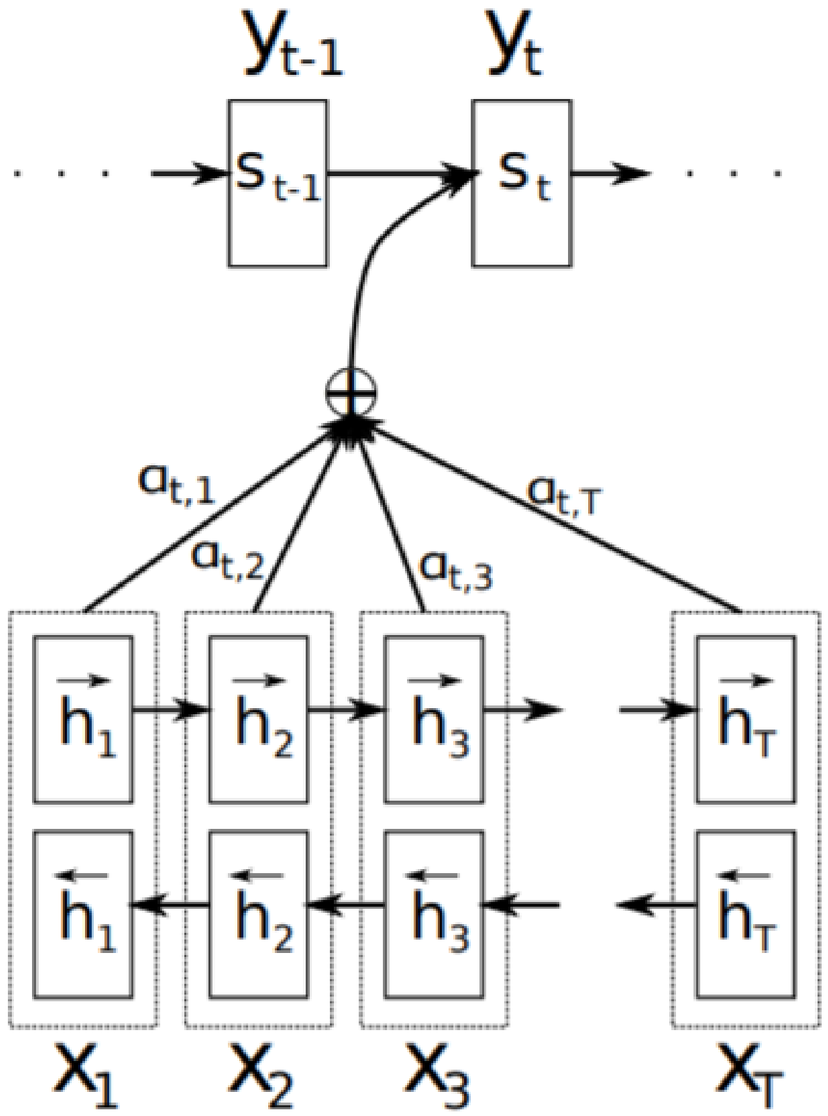


Image By Buomsoo Kim

Machine Translation and Attention



$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

$$\alpha_{t,i} = \text{align}(y_t, x_i) = \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))}$$

$$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$$

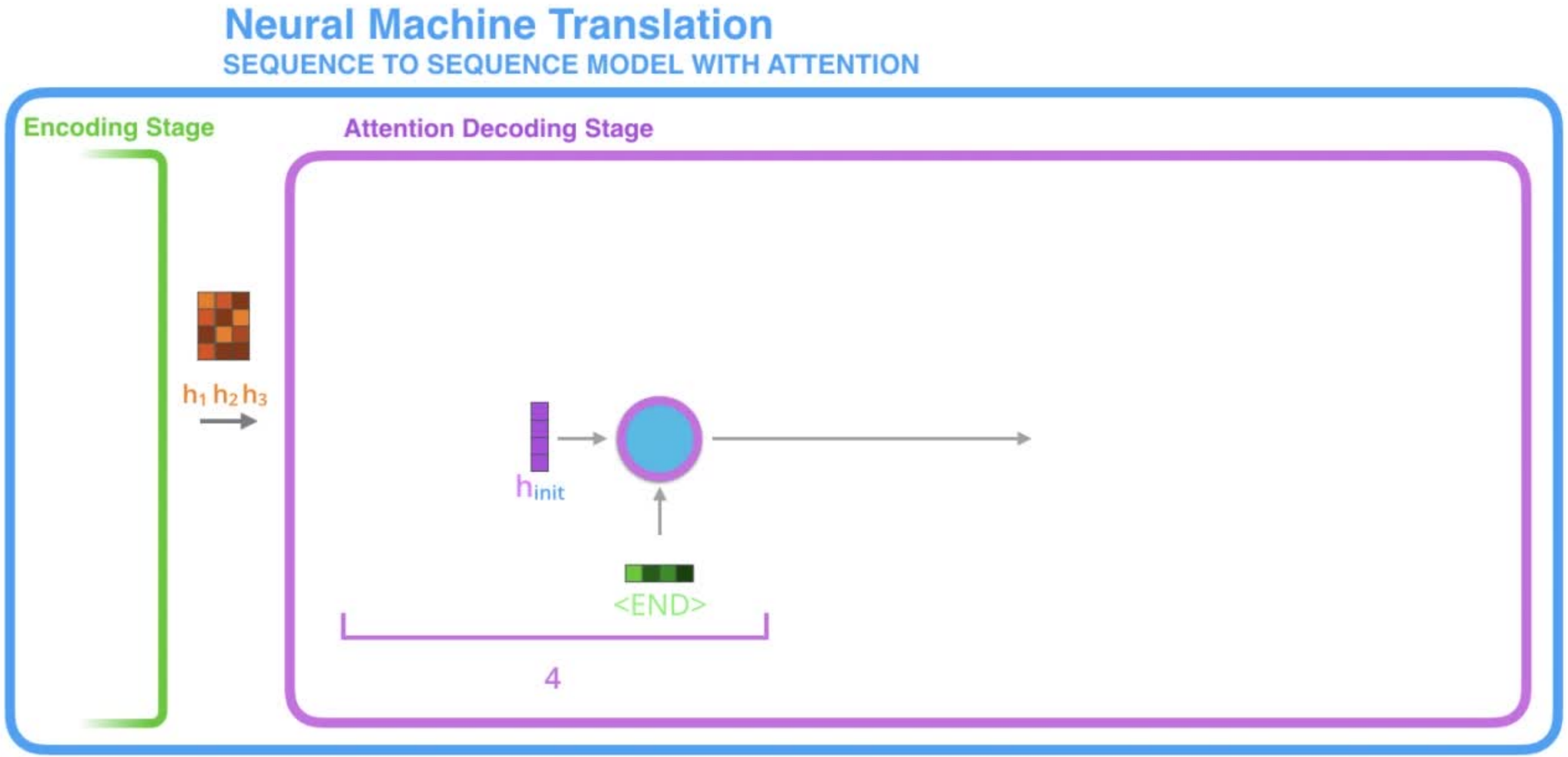
Image By Lil'Log

Reference : Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

Machine Translation and Attention

Video By Alammar, Jay (2018). Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention). Retrieved from hhttps://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

Machine Translation and Attention



Video By Alammar, Jay (2018). Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention). Retrieved from hhttps://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

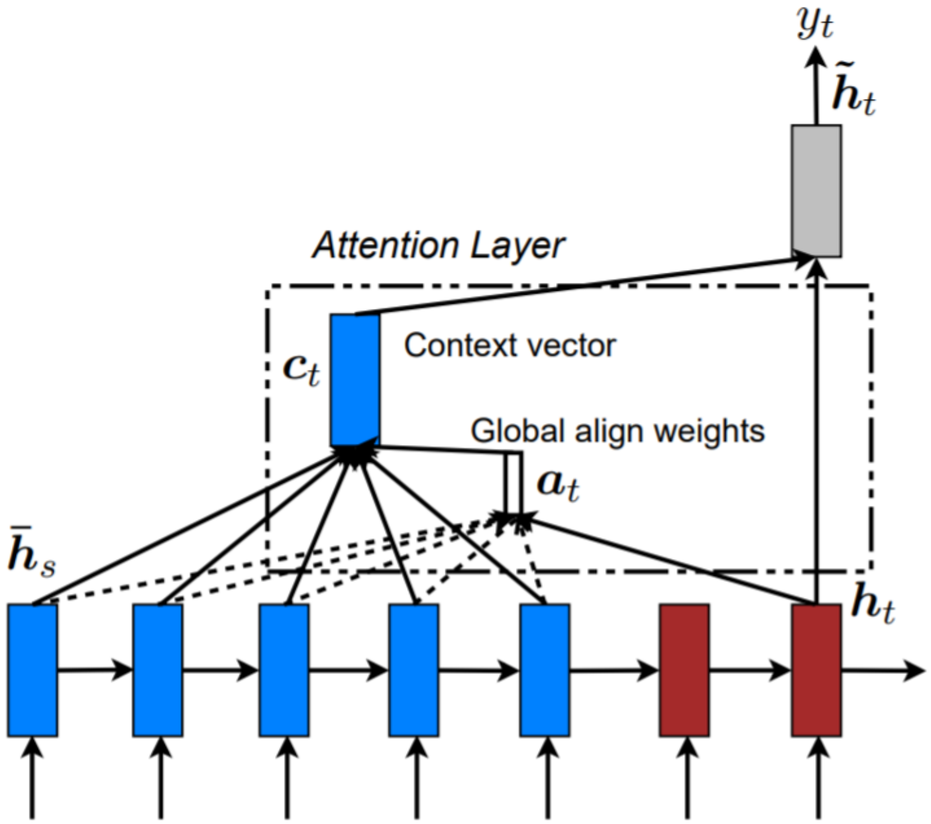
Attention Score

Name	Alignment score function	Citation
Content-base attention	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \text{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$	Graves2014
Additive(*)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \boldsymbol{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \mathbf{W}_a \boldsymbol{h}_i$ where $\mathbf{W}_a$ is a trainable weight matrix in the attention layer.	Luong2015
Dot-Product	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \frac{\boldsymbol{s}_t^\top \boldsymbol{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017

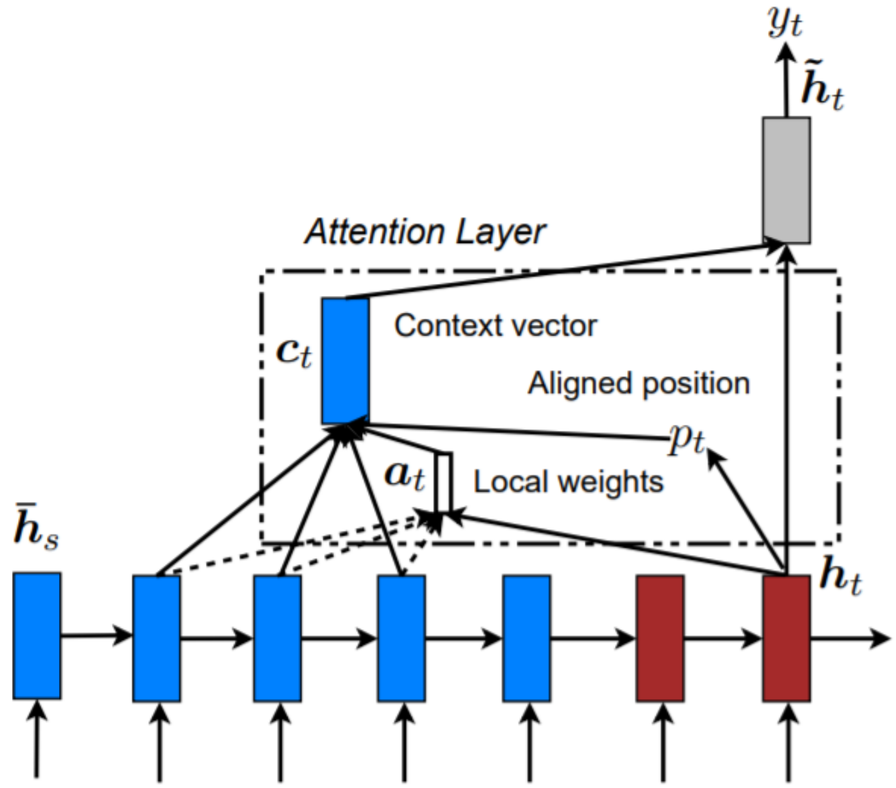
Image By Lil'Log

Global and Local Attention

Learn More : Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." arXiv preprint arXiv:1508.04025 (2015).

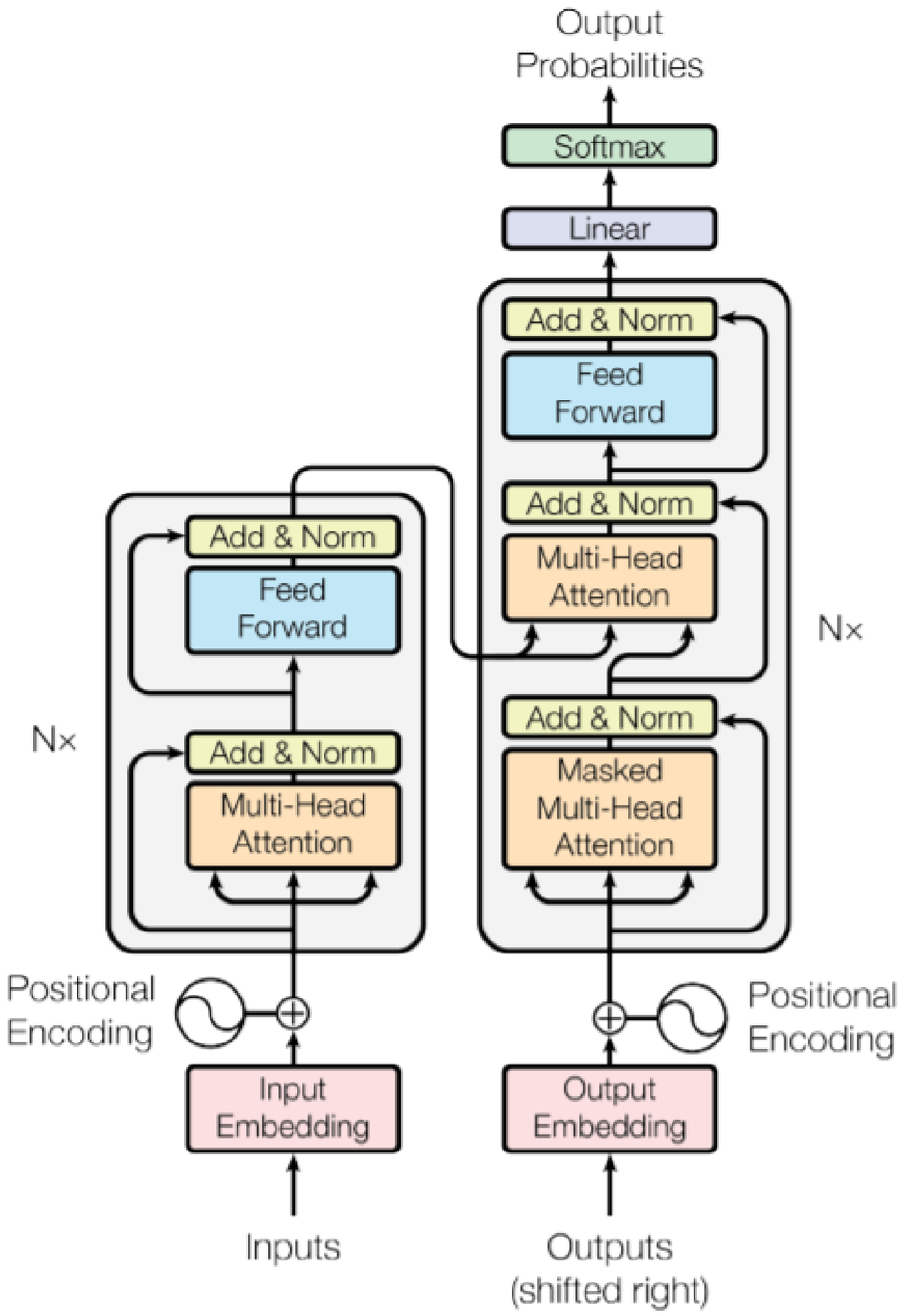


Global Attention



Local Attention

Transformer



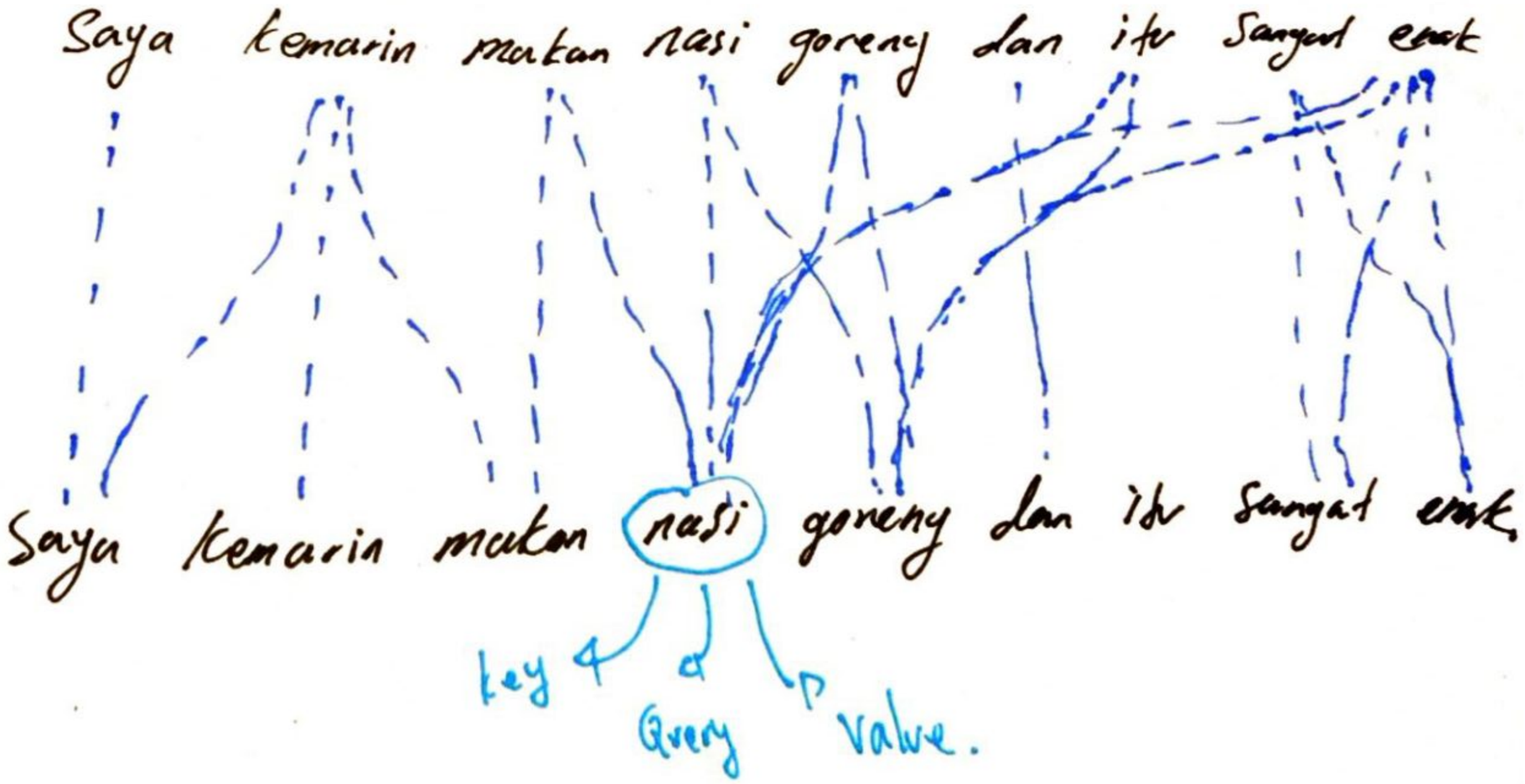
Gambaran Umum Transformer

Key Component :

1. Self Attention
2. Multi-Head Attention
3. Encoder
4. Decoder
5. Positional Encoding

Learn More : Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017

Self Attention



Self Attention

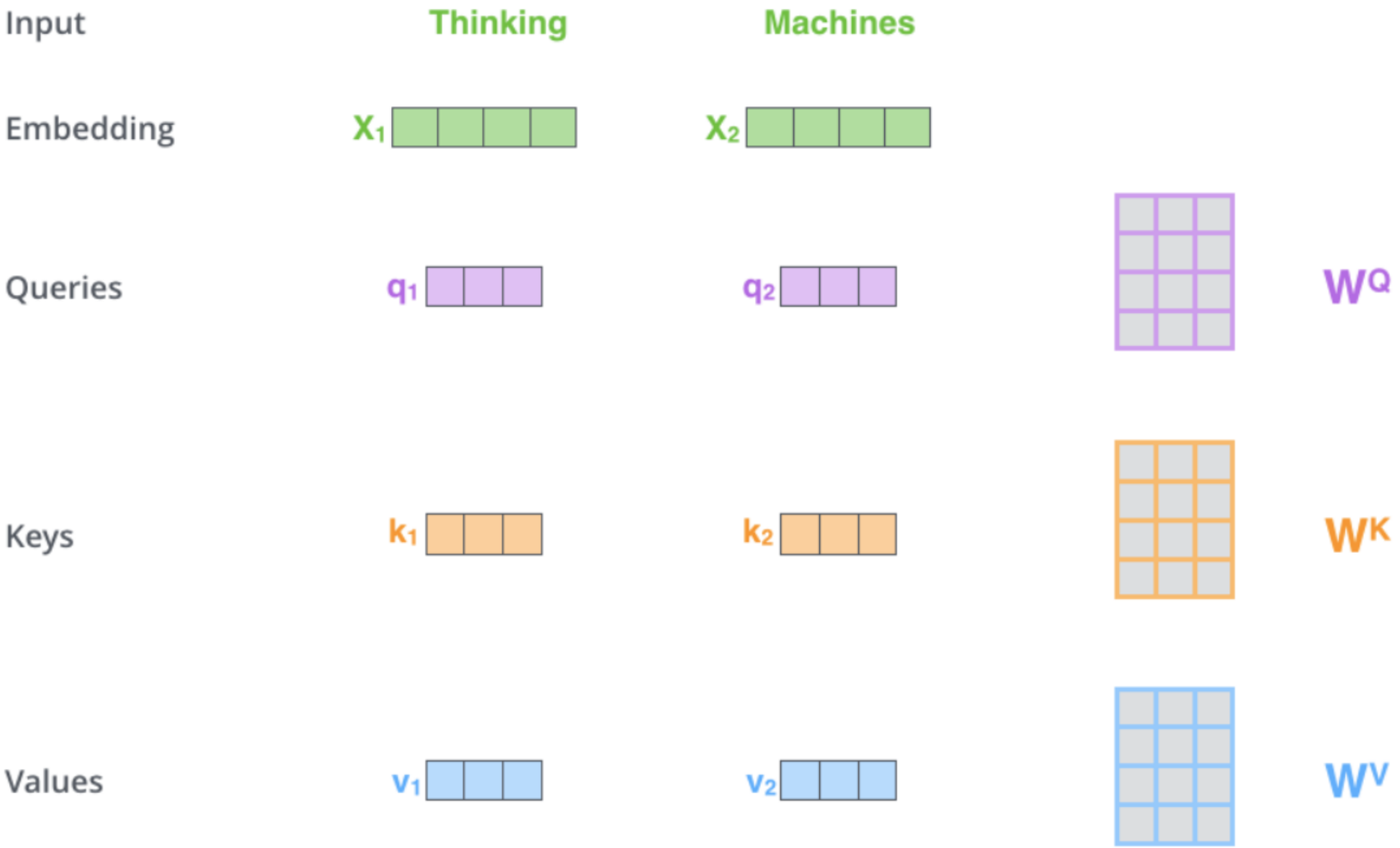


Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

Self Attention

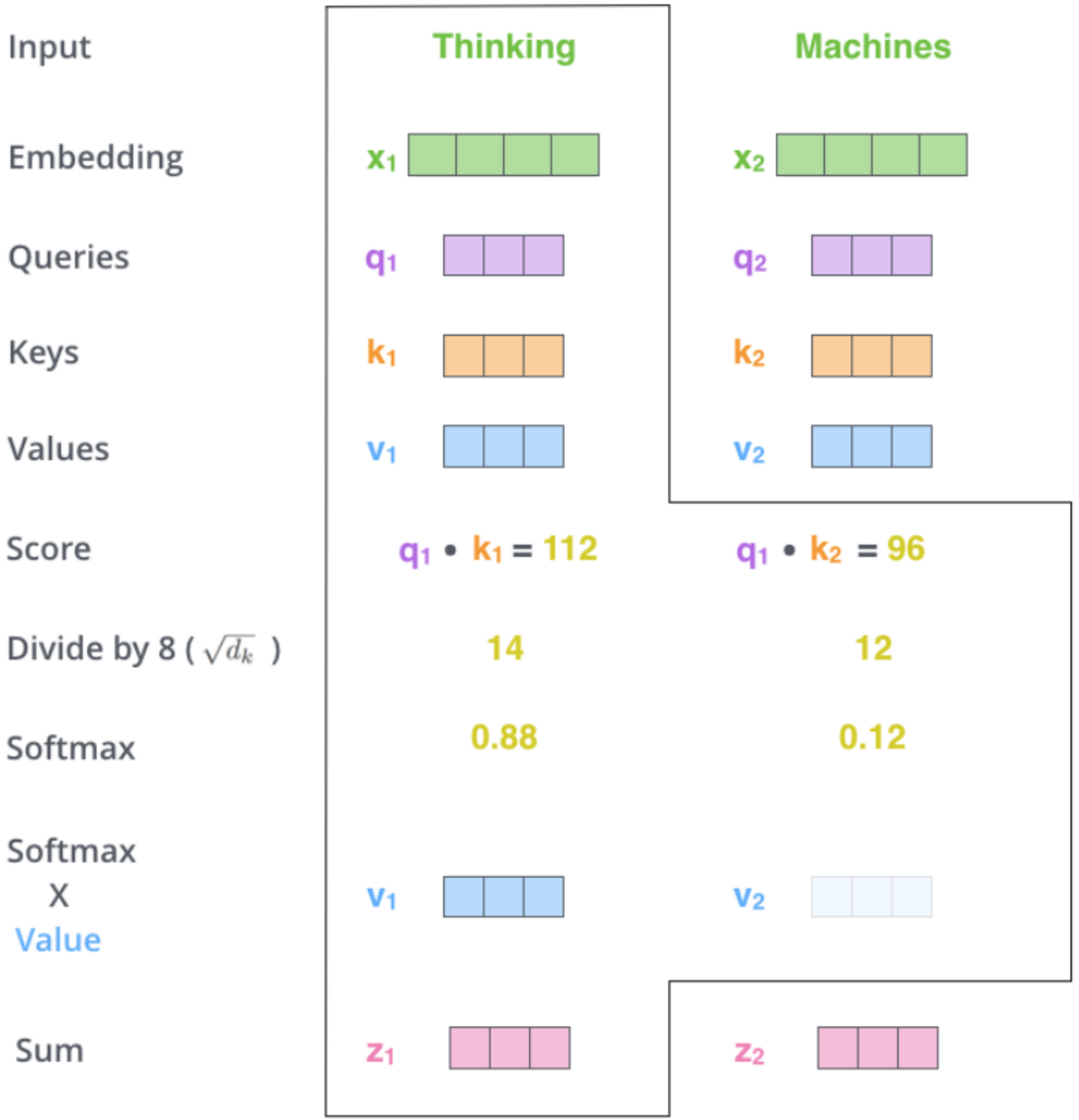


Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

Multi-Head Attention

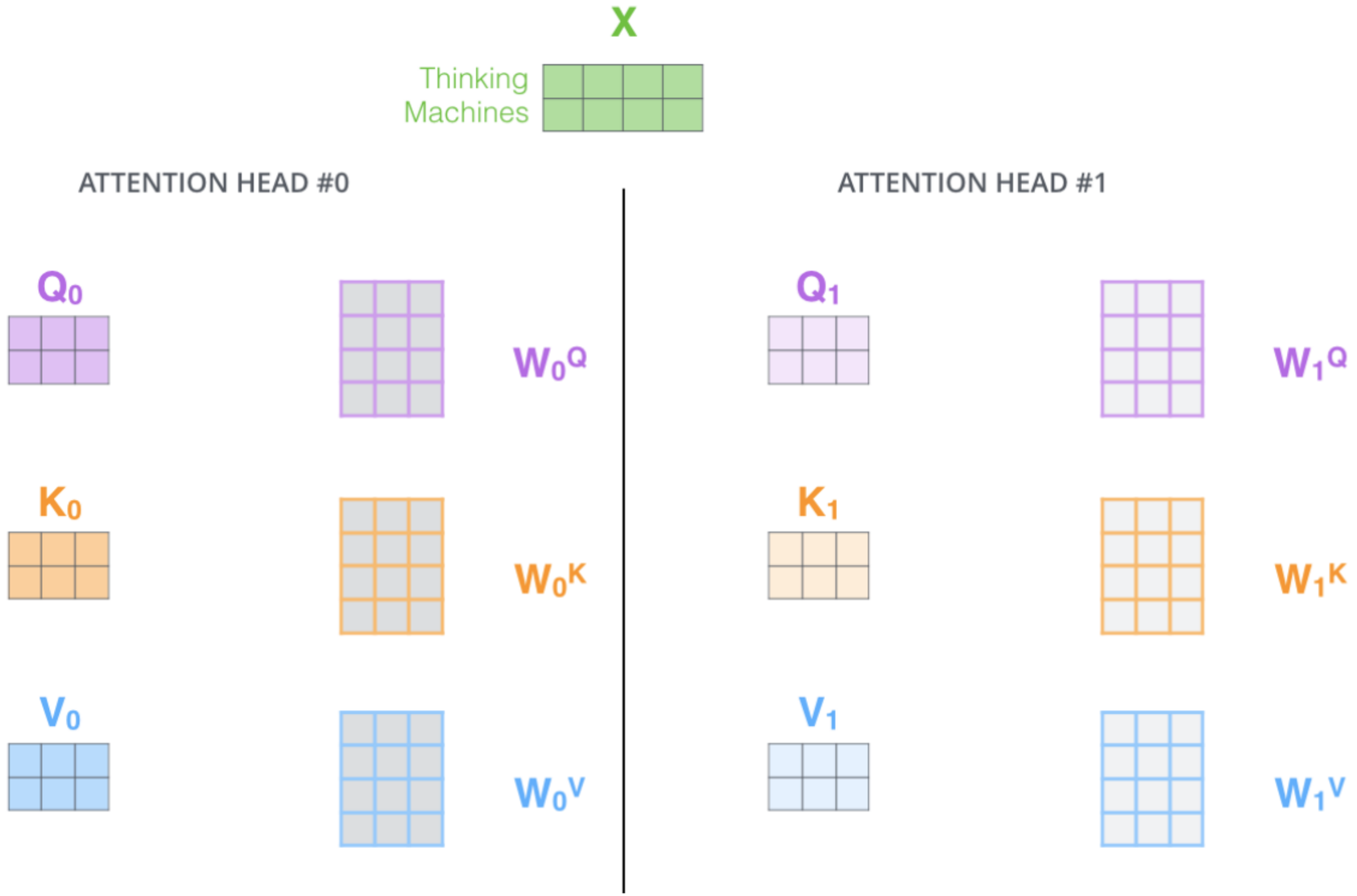


Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

Multi-Head Attention

1) This is our input sentence*

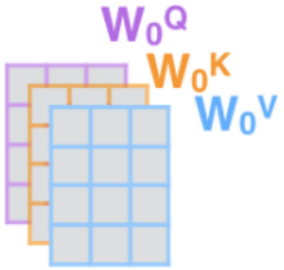
2) We embed each word*

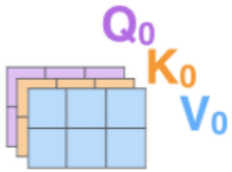
3) Split into 8 heads. We multiply X or R with weight matrices

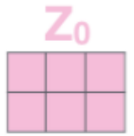
4) Calculate attention using the resulting Q/K/V matrices

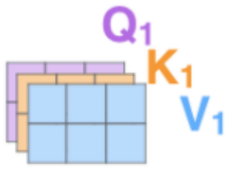
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

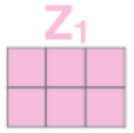
Thinking Machines



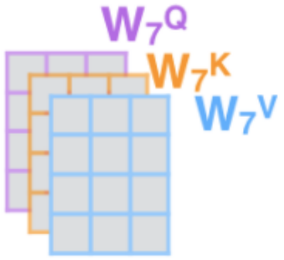


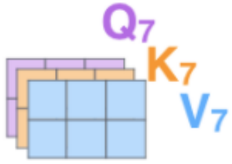


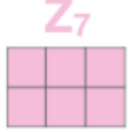


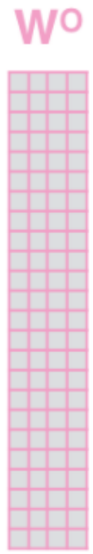


* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one









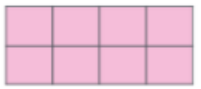
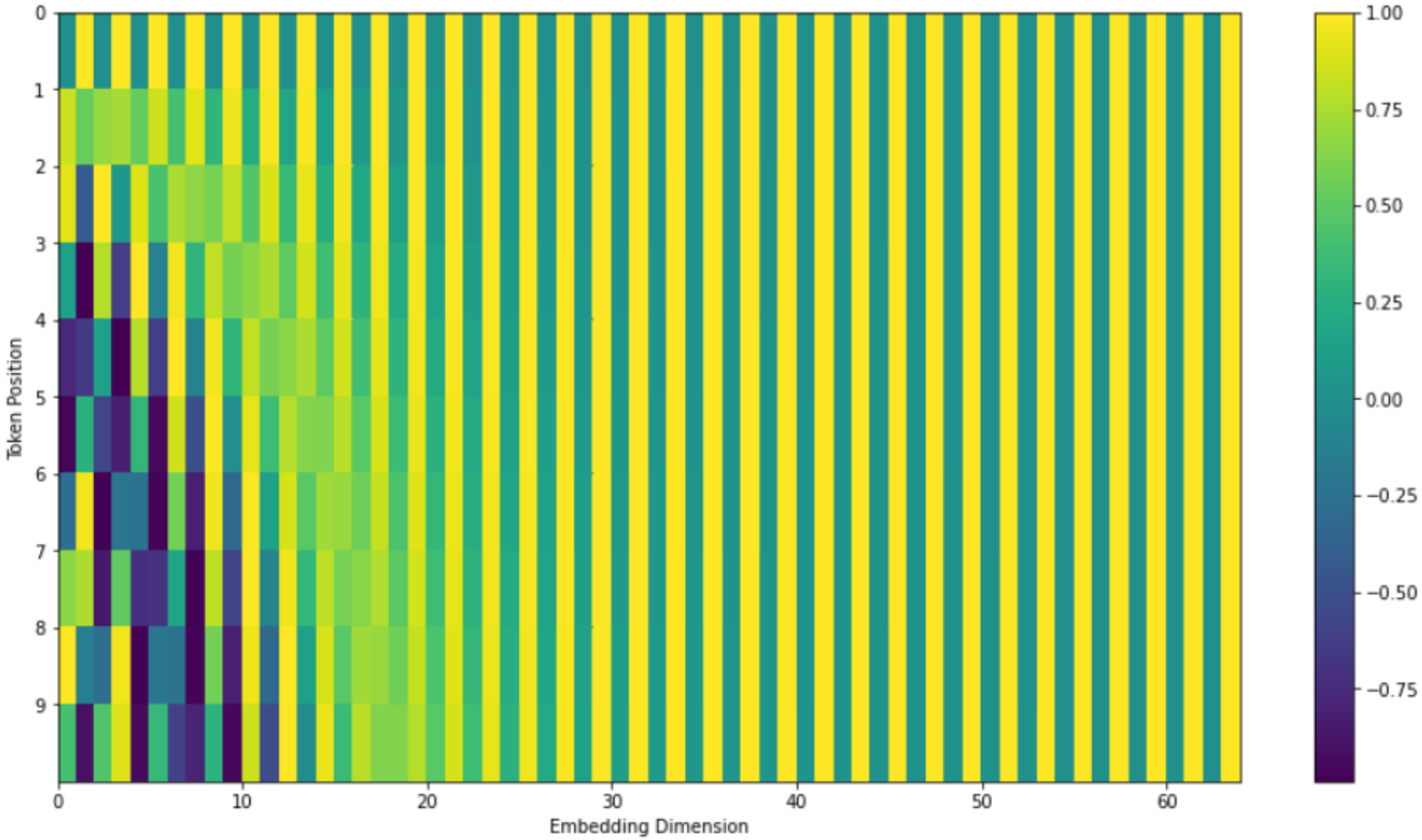


Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

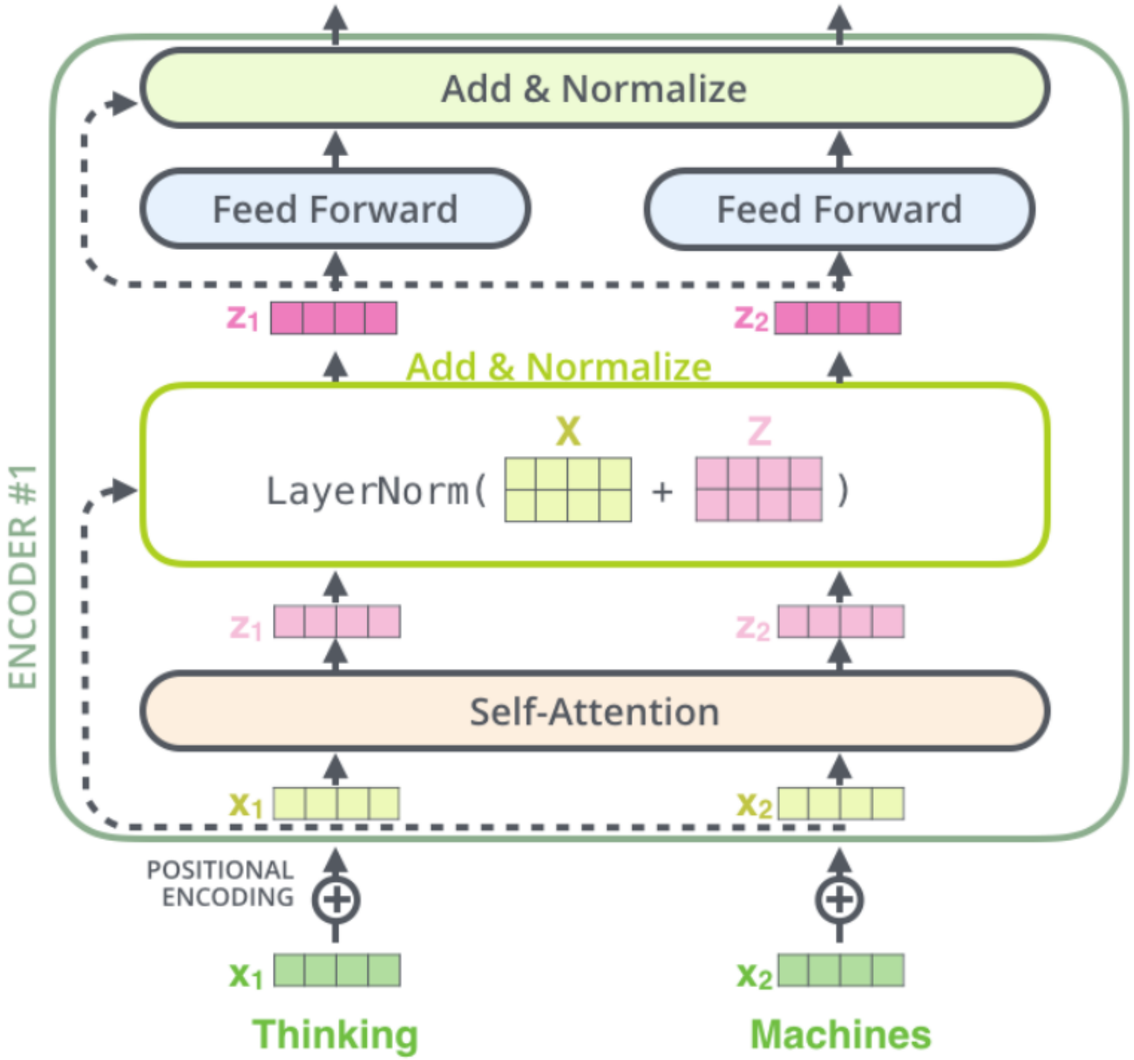
Positional Encoding



$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

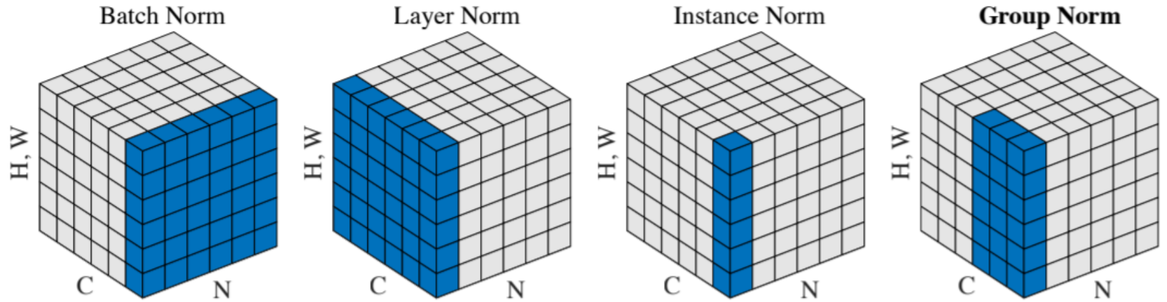
Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

Encoder



Learn More : Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016)..
Wu, Yuxin, and Kaiming He. "Group normalization." Proceedings of the European conference on computer vision (ECCV). 2018.

Residual Connection dan Layer Normalization

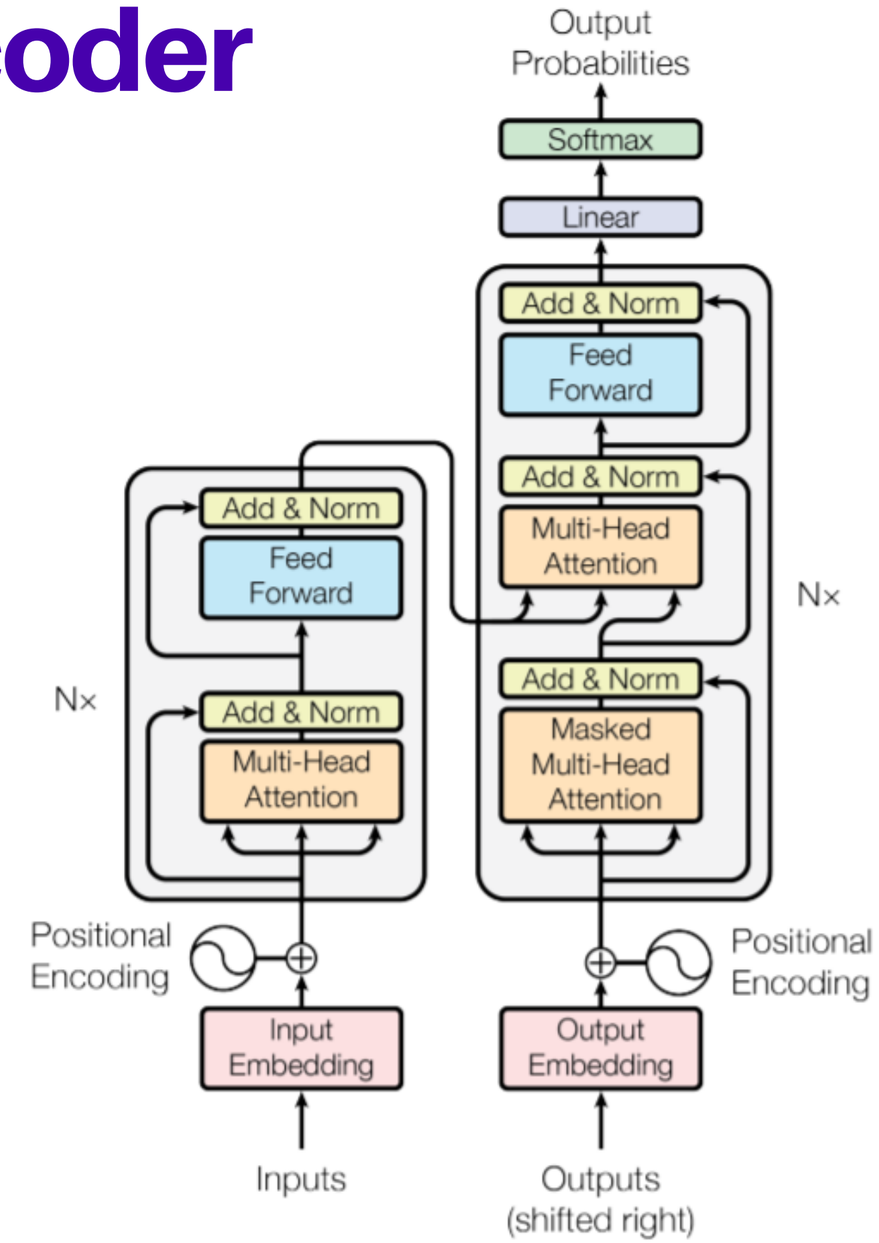


Position-wise Feed-Forward Networks

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Image By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

Decoder



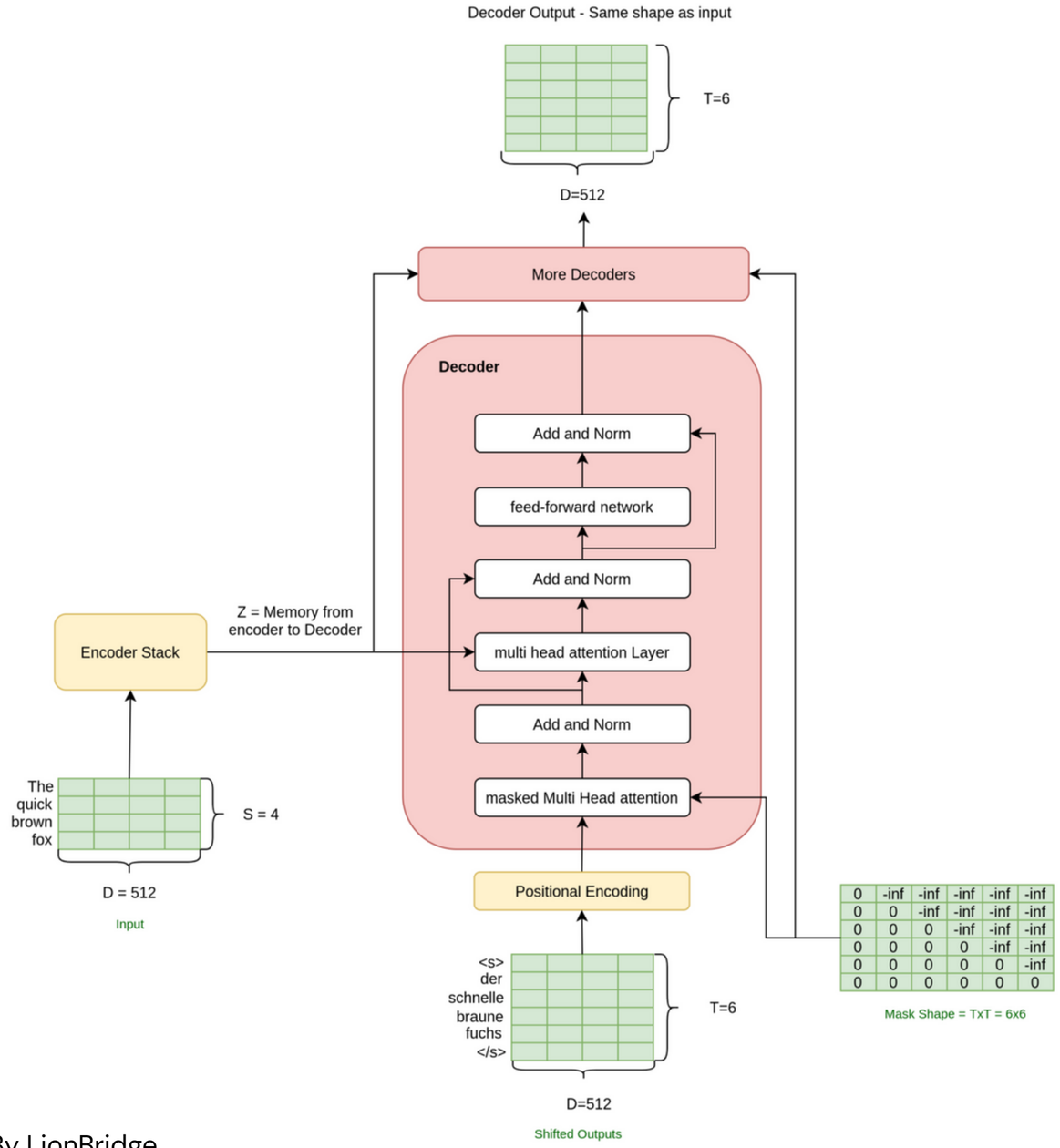
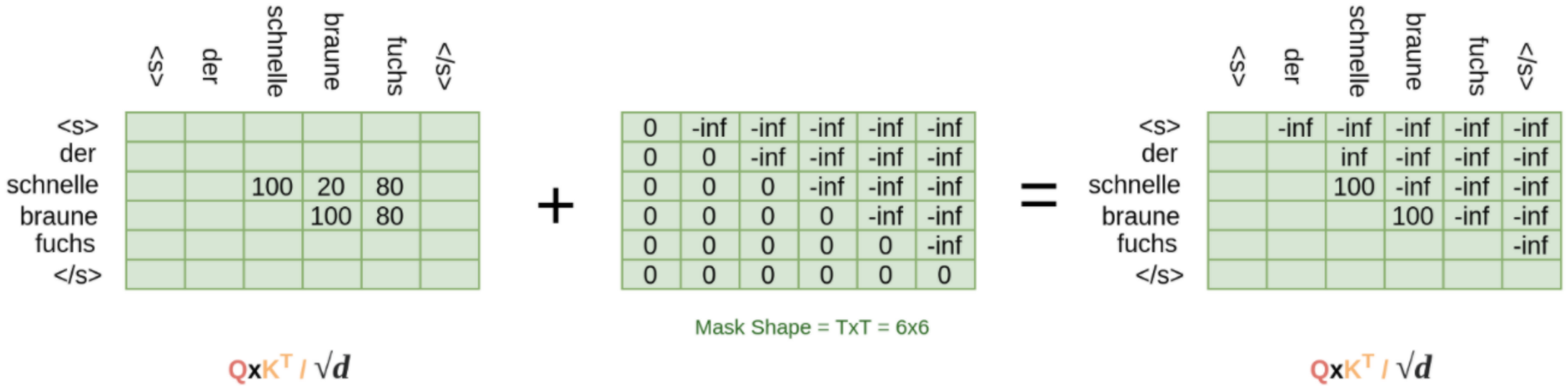


Image By LionBridge

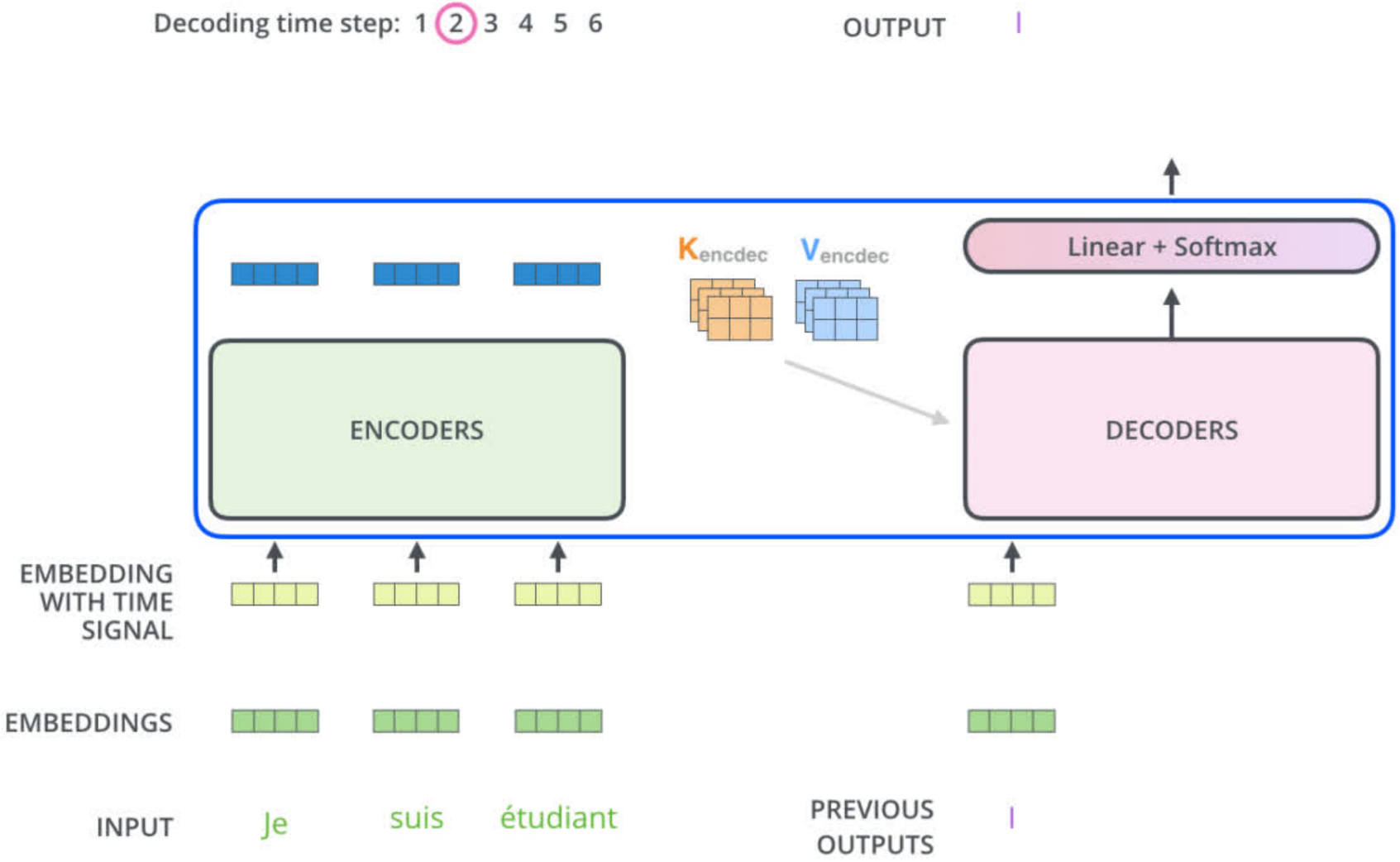
Masked Multi-Head Attention



The mask operation applied to the matrix.

Image By LionBridge

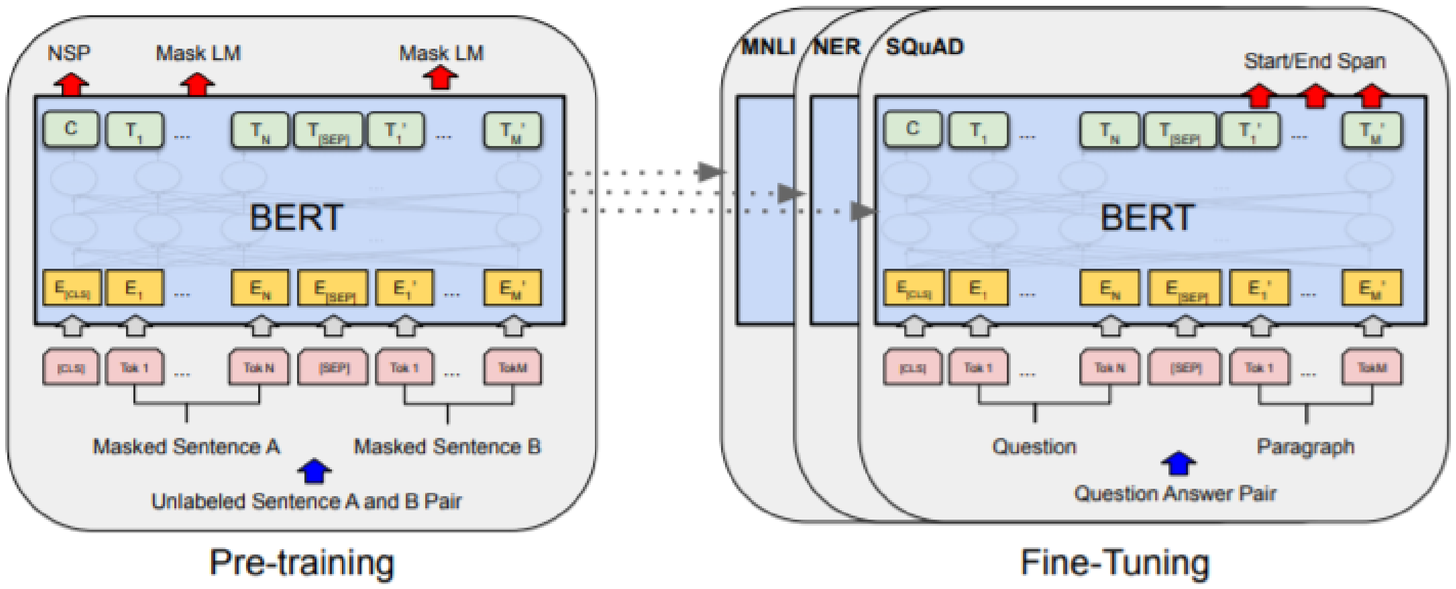
Proses Decoding

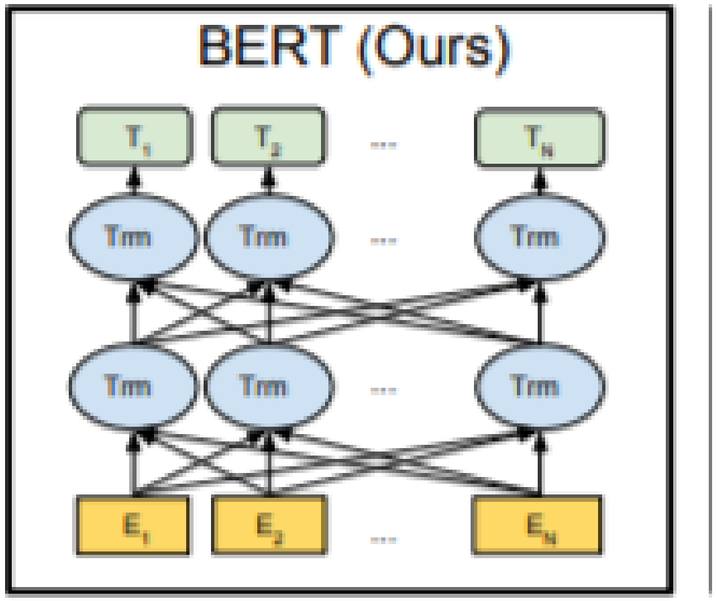


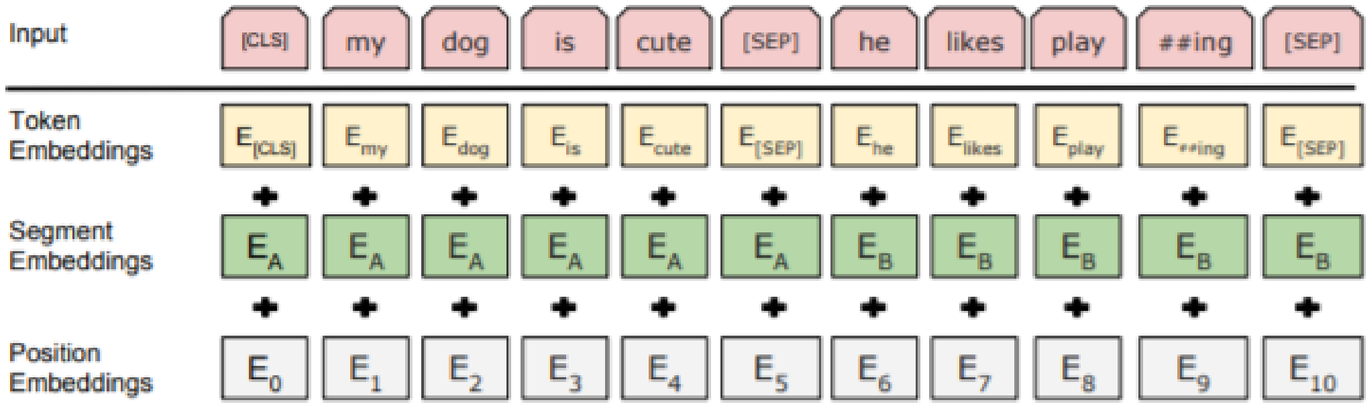
Video By Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

BERT

BERT Model

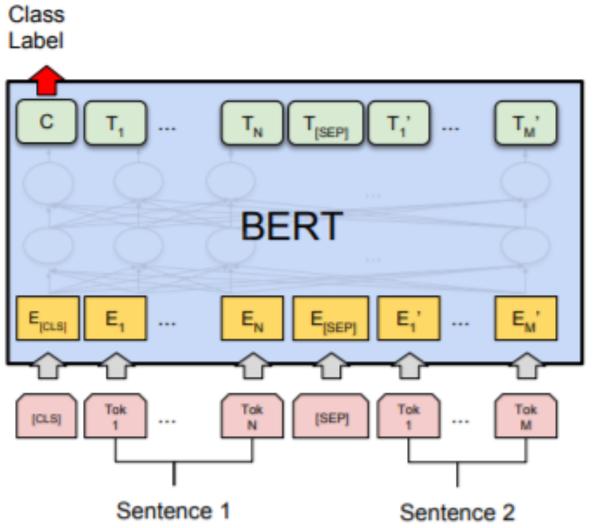




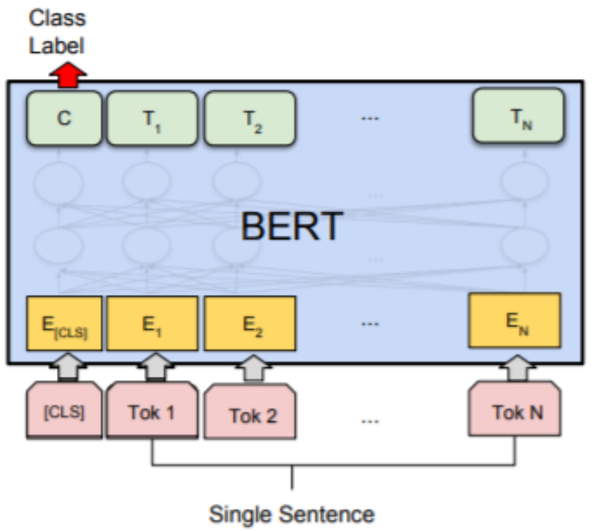


Learn More : Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

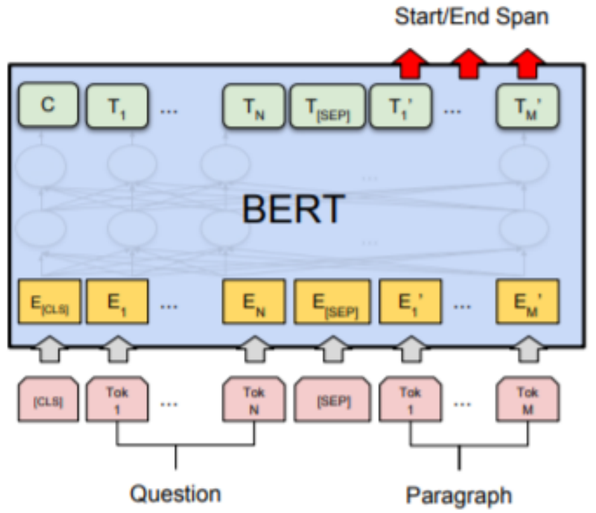
Using BERT



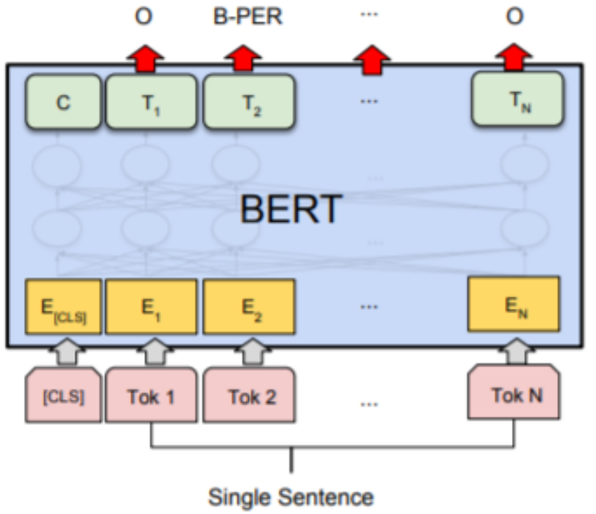
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1



(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Demo

Contact

Twitter

@NaraSurya13

Email Address

ibagungnsd13@gmail.com

Telegram

@Gusagung

Thank You