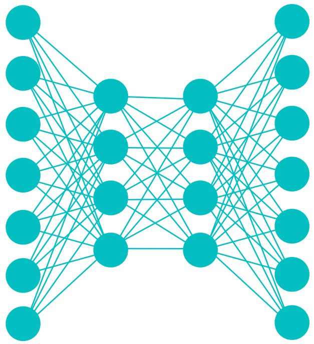
Lecture Notes for

Neural Networks and Machine Learning

Transformers and Vision Transformers

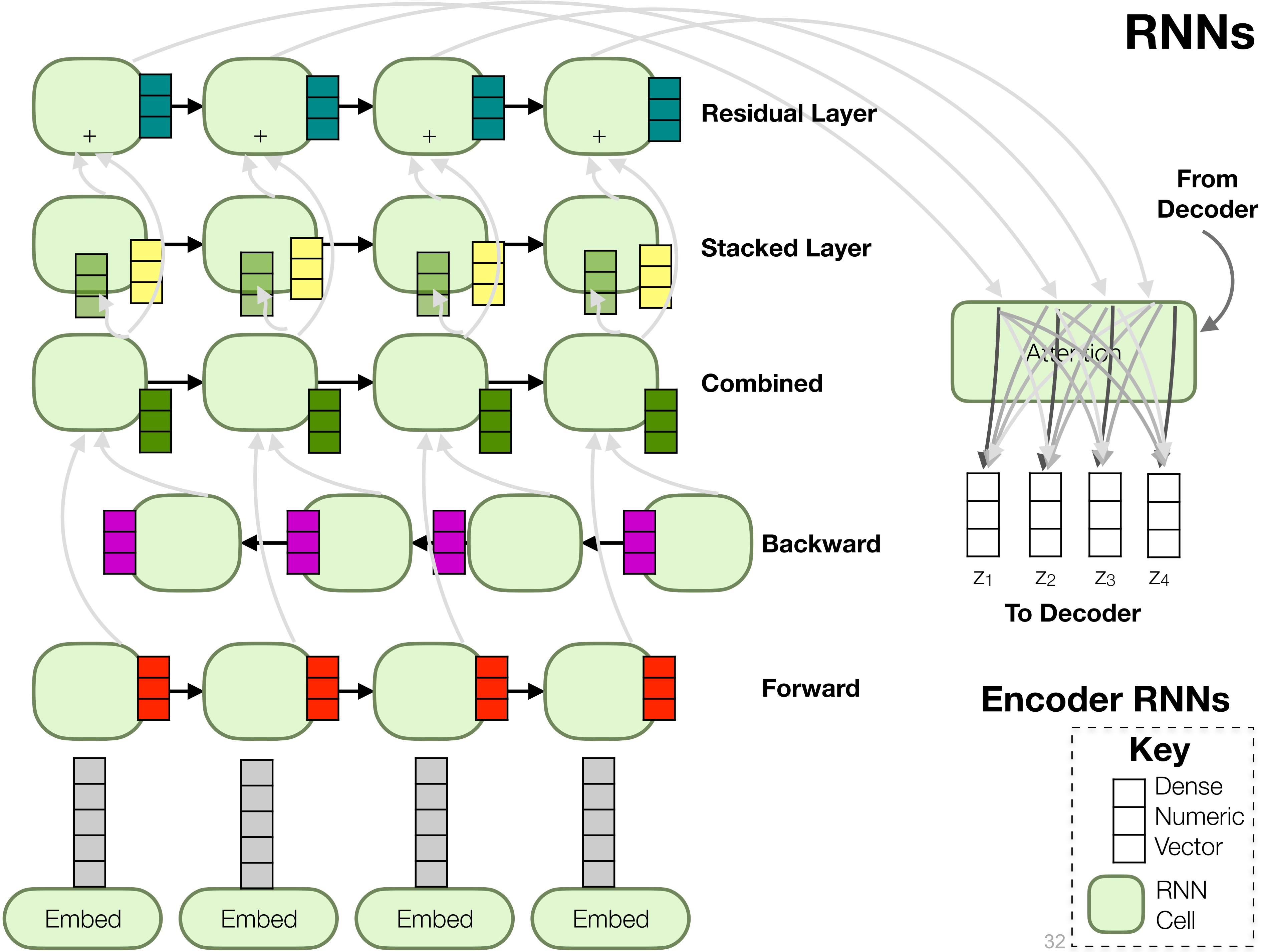


Logistics and Agenda

- Logistics
  - Paper presentations
- Agenda
  - Transformers
- Next Time:
  - Vision Transformers
  - Paper Presentation
  - Self-supervised learning and other consistency losses

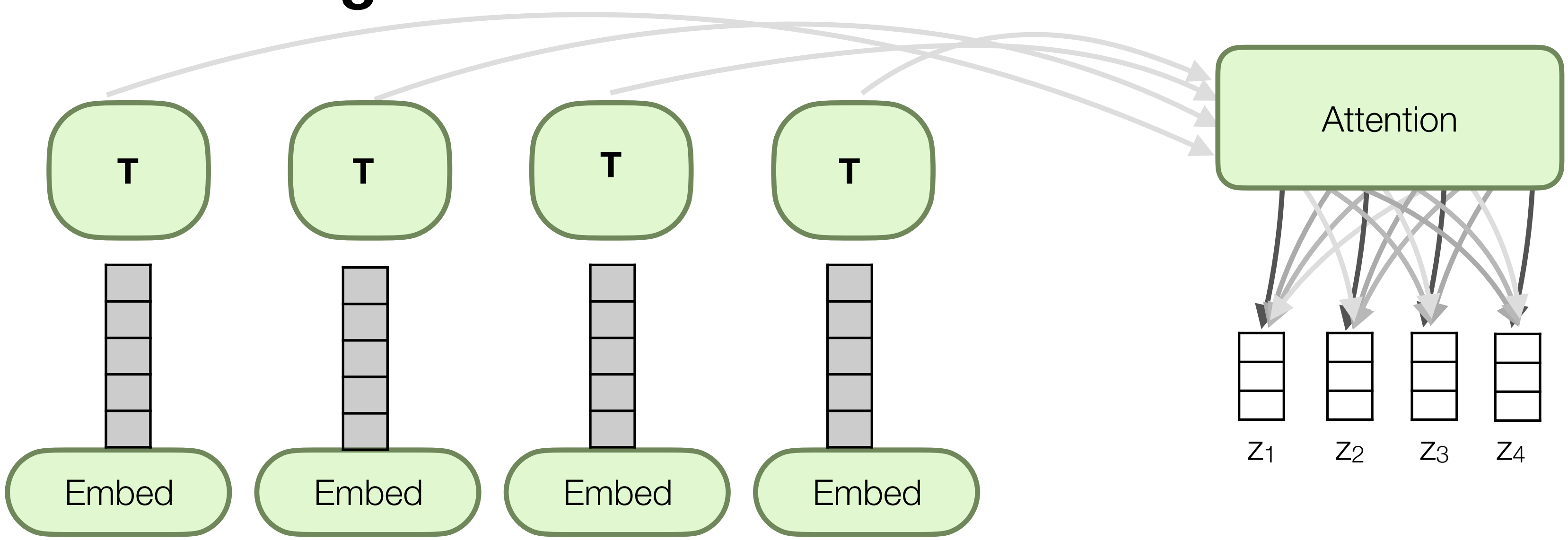
Transformers

RNNs


Residual Layer
Stacked Layer
Combined
Backward
Forward
Embed
Embed
Embed
Embed
From Decoder
Attention
z_1
z_2
z_3
z_4
To Decoder
Encoder RNNs
Key
Dense Numeric Vector
RNN Cell

Transformers Intuition

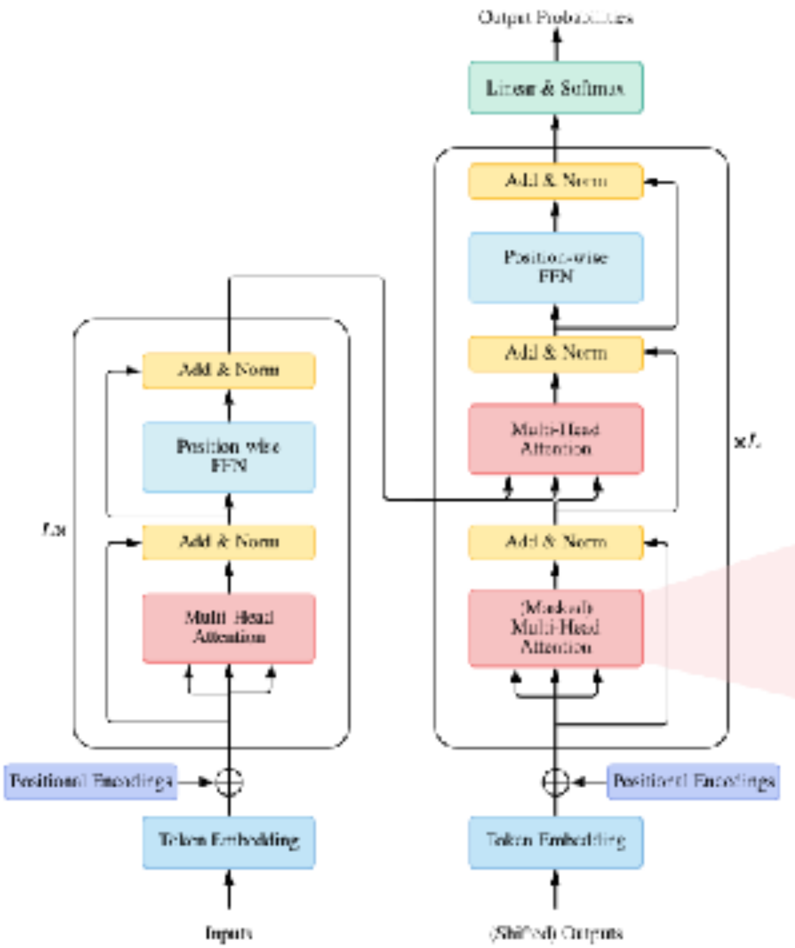
- Recurrent networks track state using an "updatable" state vector, but this takes lots of processing to across sequence
- Attention mechanism (in RNNs) already takes a weighted sum of state vectors to generate new token in a decoder
- … so why not just use attention on a transformation of the embedding vectors? **Do away with the recurrent state vector all together?**



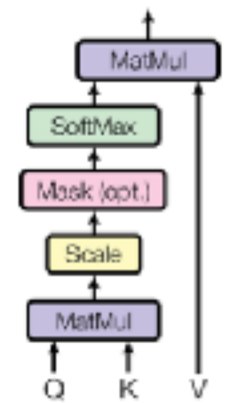
Attention is All You Need

- **Continued Motivation**:
  - RNNs are not inherently parallelized or efficient at remembering based on state vector
  - CNNs are not resilient to long-term word relationships, limited by filter size
- **Transformer Solution:**
  - Build attention into model from the **beginning**
  - Compare all words to each other through **self-headed** attention
  - Define a notion of “**position**”in the sequence
  - ***Should be resilient to long term relationships and be highly parallelized for GPU computing!!***

Transformer



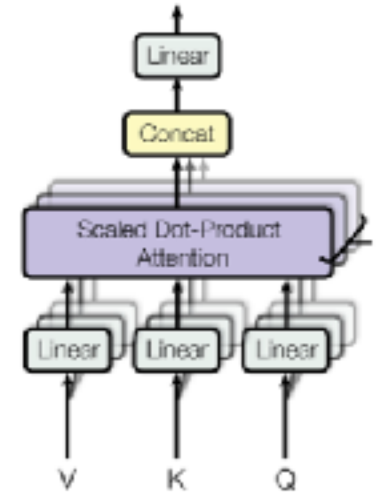
Scaled Dot-Product Attention



for each word

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Multi-Head Attention

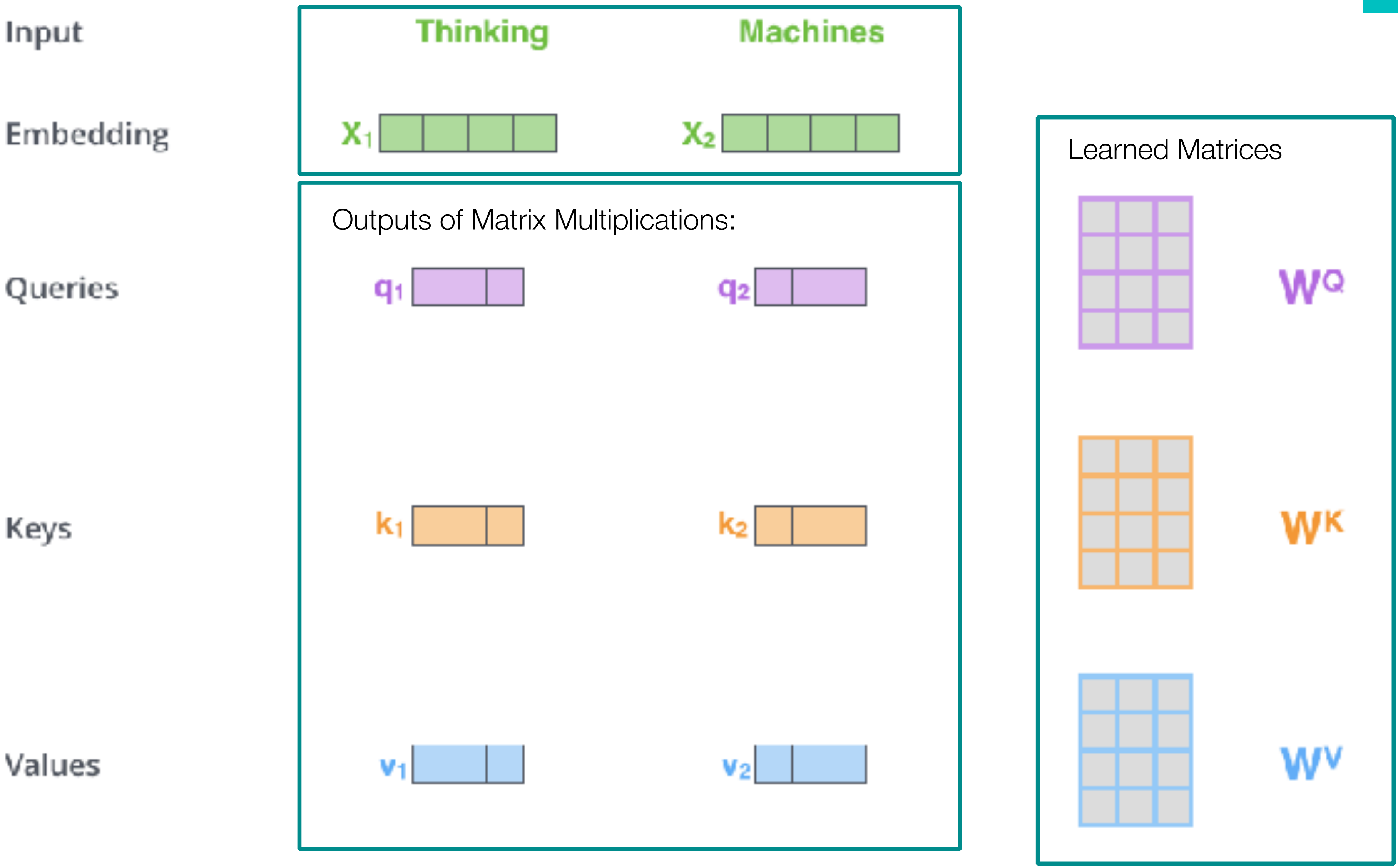


more than one
Q,K,V use in document

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

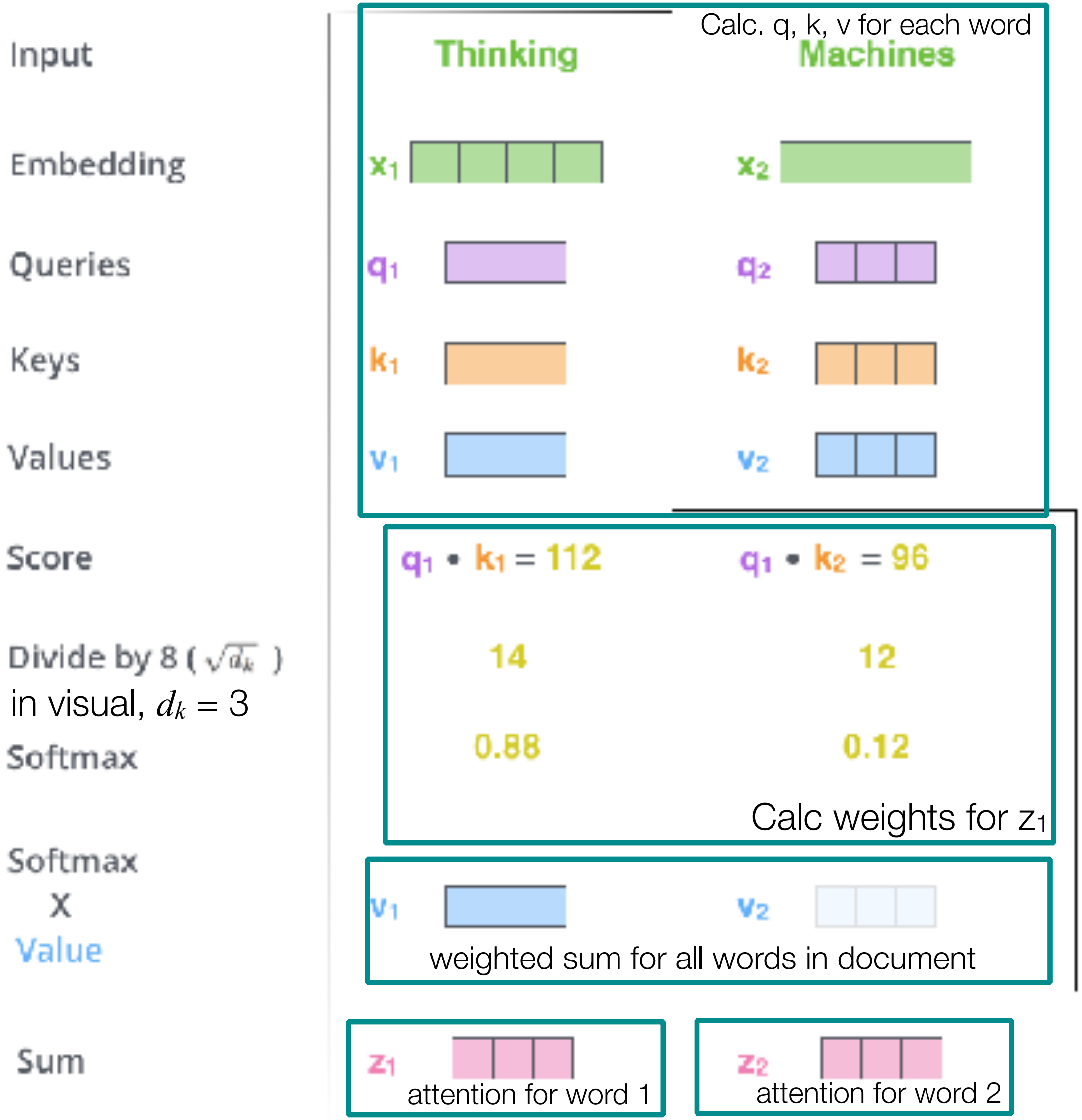
https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

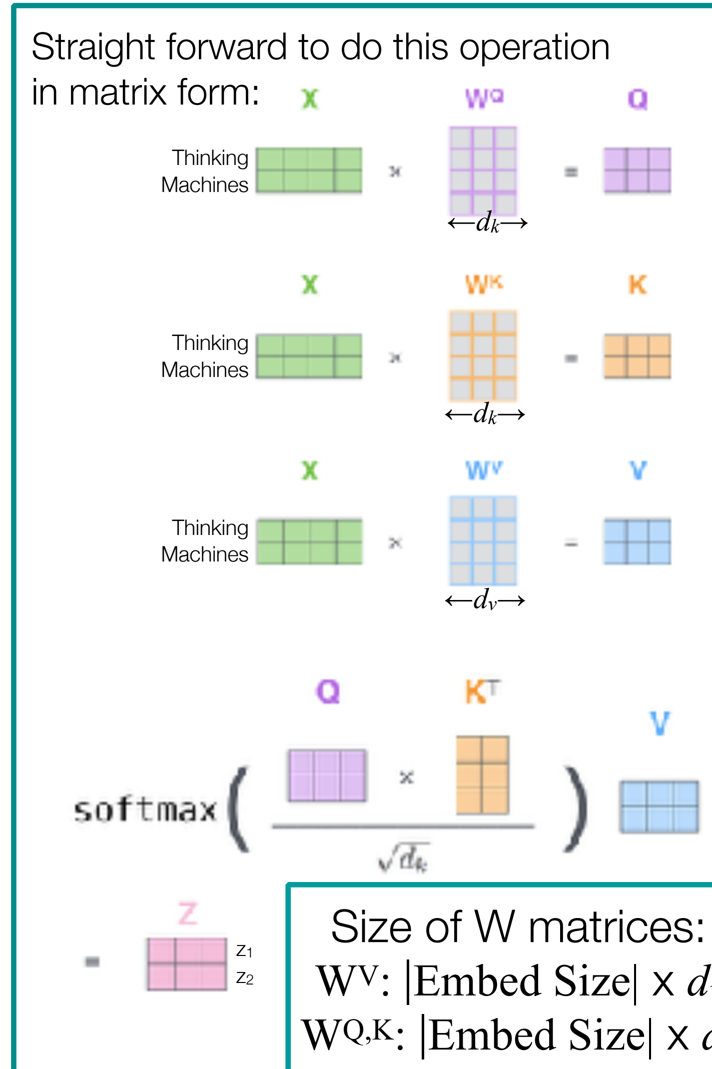
Transformer: in more detail



Excellent Blog on Transformers: http://jalammar.github.io/illustrated-transformer/

Transformer: in more detail



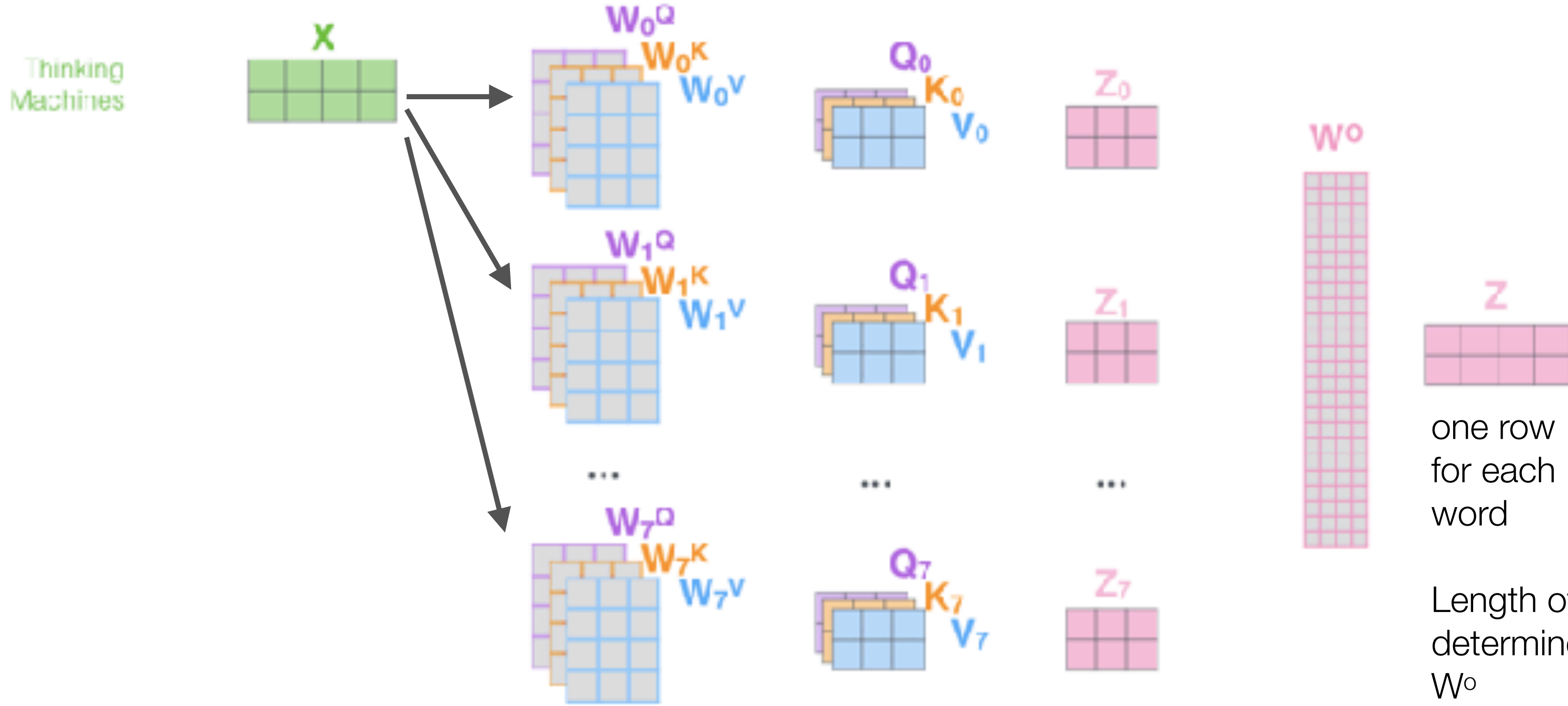


Size of W matrices:
W^V: |Embed Size| x d_v
$W^{Q,K}$: |Embed Size| x d_k

Size of Q,K,V:
|Seq Len| x d_v

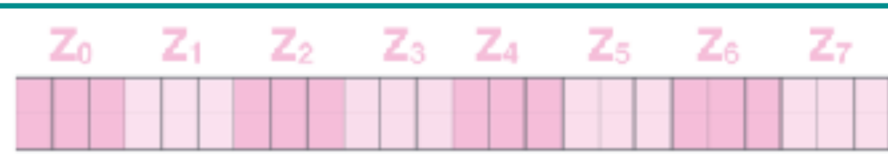
Excellent Blog on Transformers: http://jalammar.github.io/illustrated-transformer/

Transformer: Multi-headed Attention



one row
for each
word

Length of row
determined by
W^O

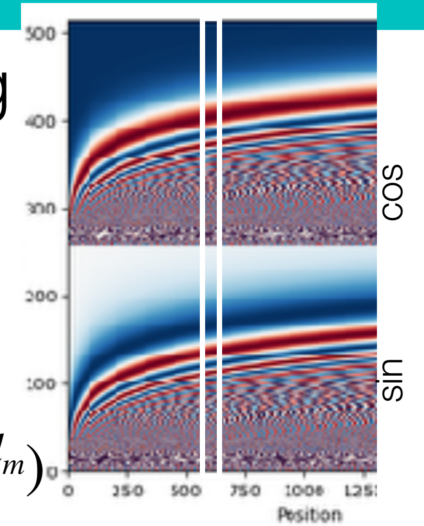


x W^O = Z

Excellent Blog on Transformers: http://jalammar.github.io/illustrated-transformer/

Transformer: Positional Encoding

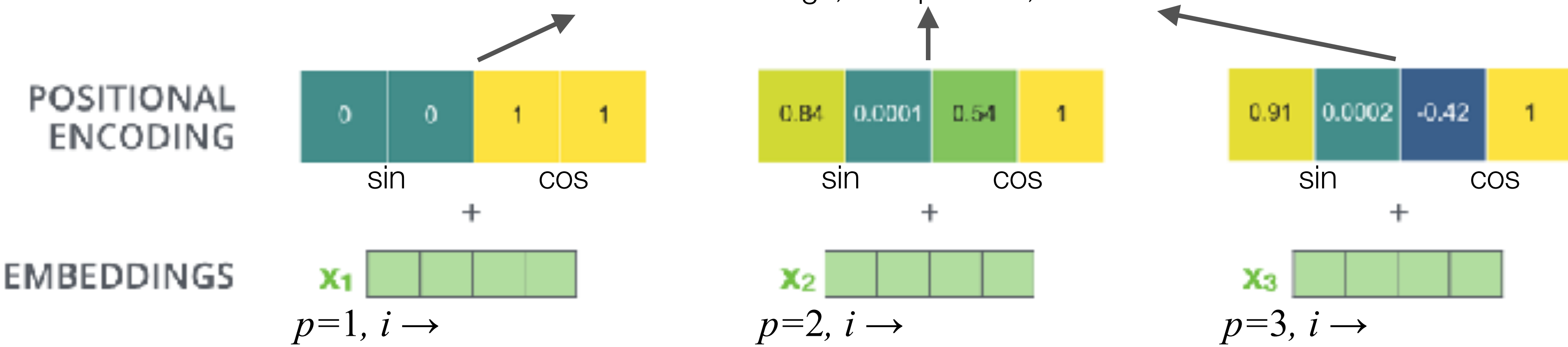
- Objective: add notion of position to embedding
- Attempt in paper: add sin/cos to embedding
- But could be anything that encodes position



p: in sequence
d_m: 0.5*max dim of embed
i = position in embed

$$PE_{(p,i\in 0\ldots d_m-1)} = \sin(p/10000^{i/d_m})$$
$$PE_{(p,i\in d_m\ldots 2d_m)} = \cos(p/10000^{(i-d_m)/d_m})$$

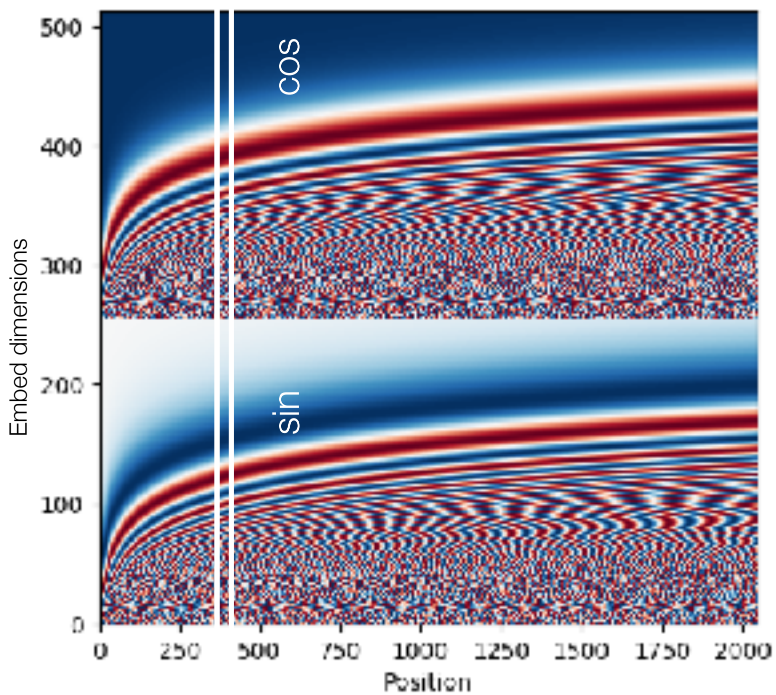
Now use the new embeddings, with position, into transformer architecture

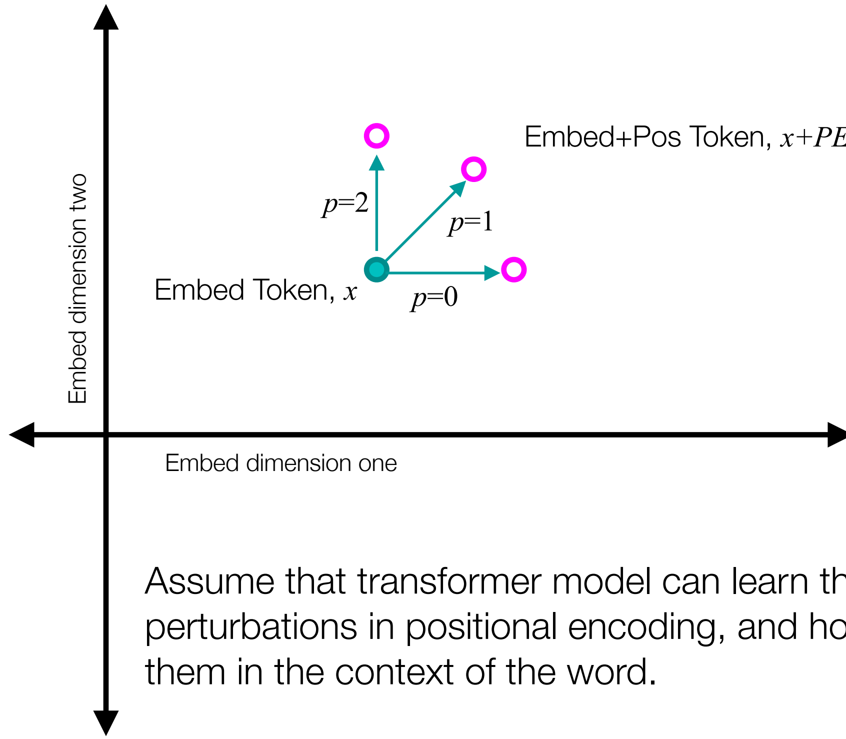


Hypothesis: Now the word proximity is encoded in the embedding matrix, with other pertinent information. Well, it does help… so it could be true that this is a good way to do it.

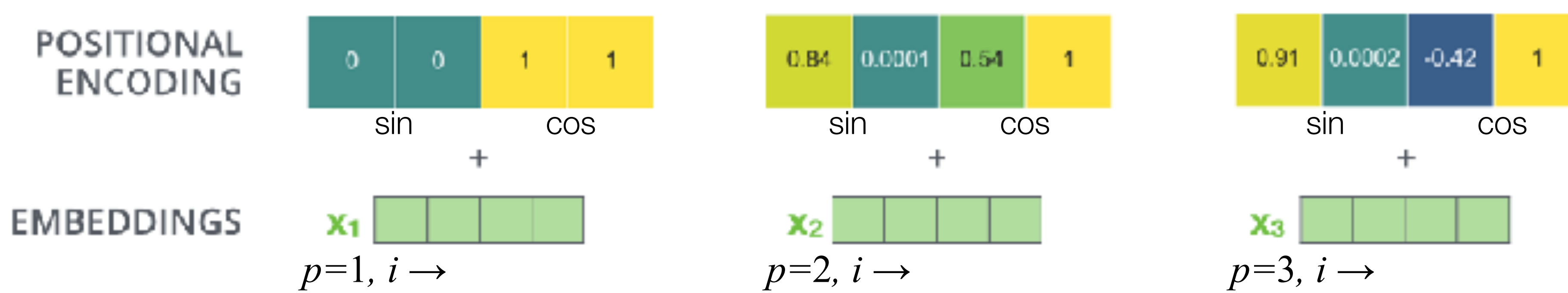
Excellent Blog on Transformers: http://jalammar.github.io/illustrated-transformer/

Positional Intuition, Geometrically

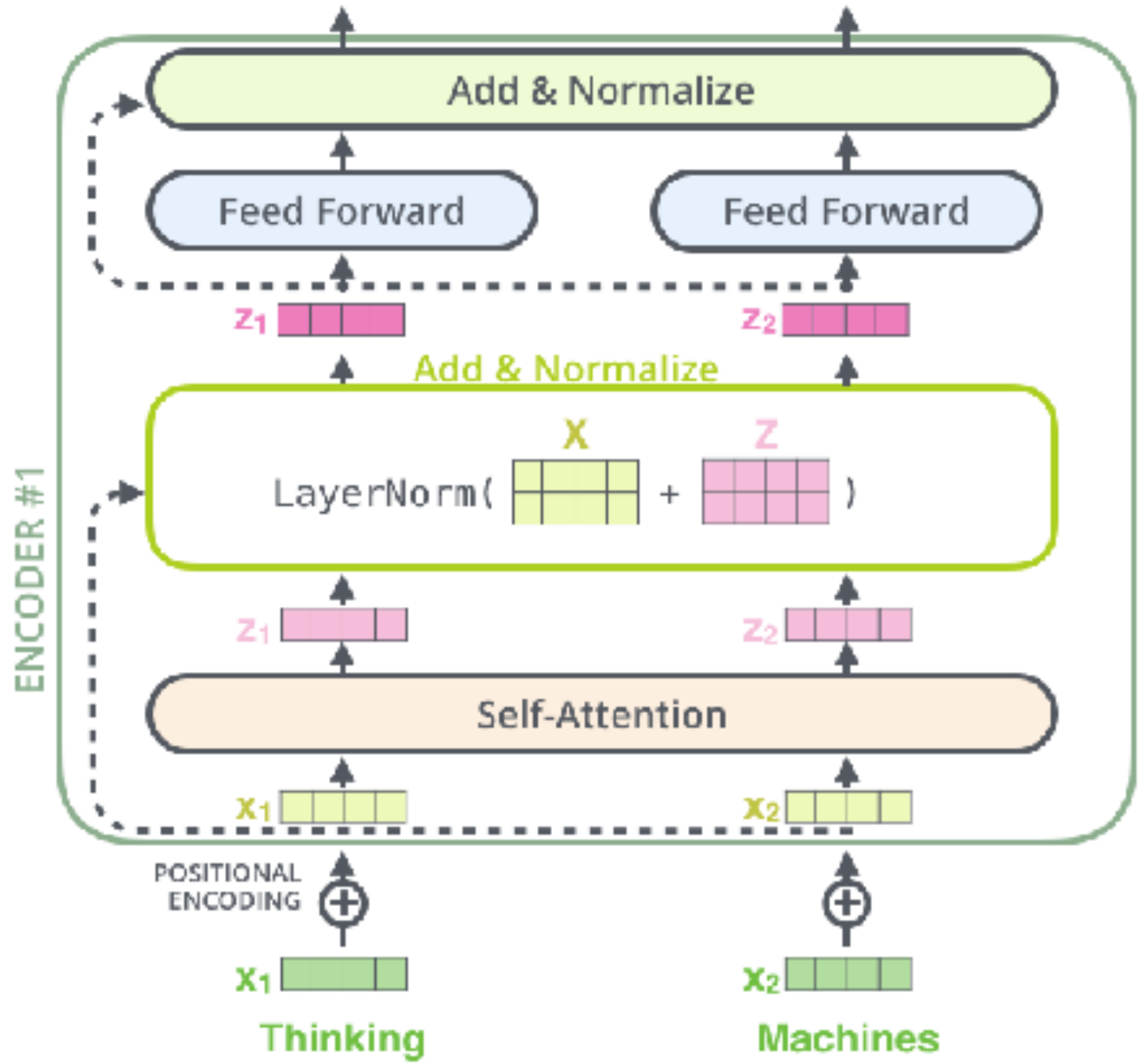




Assume that transformer model can learn the small perturbations in positional encoding, and how to use them in the context of the word.

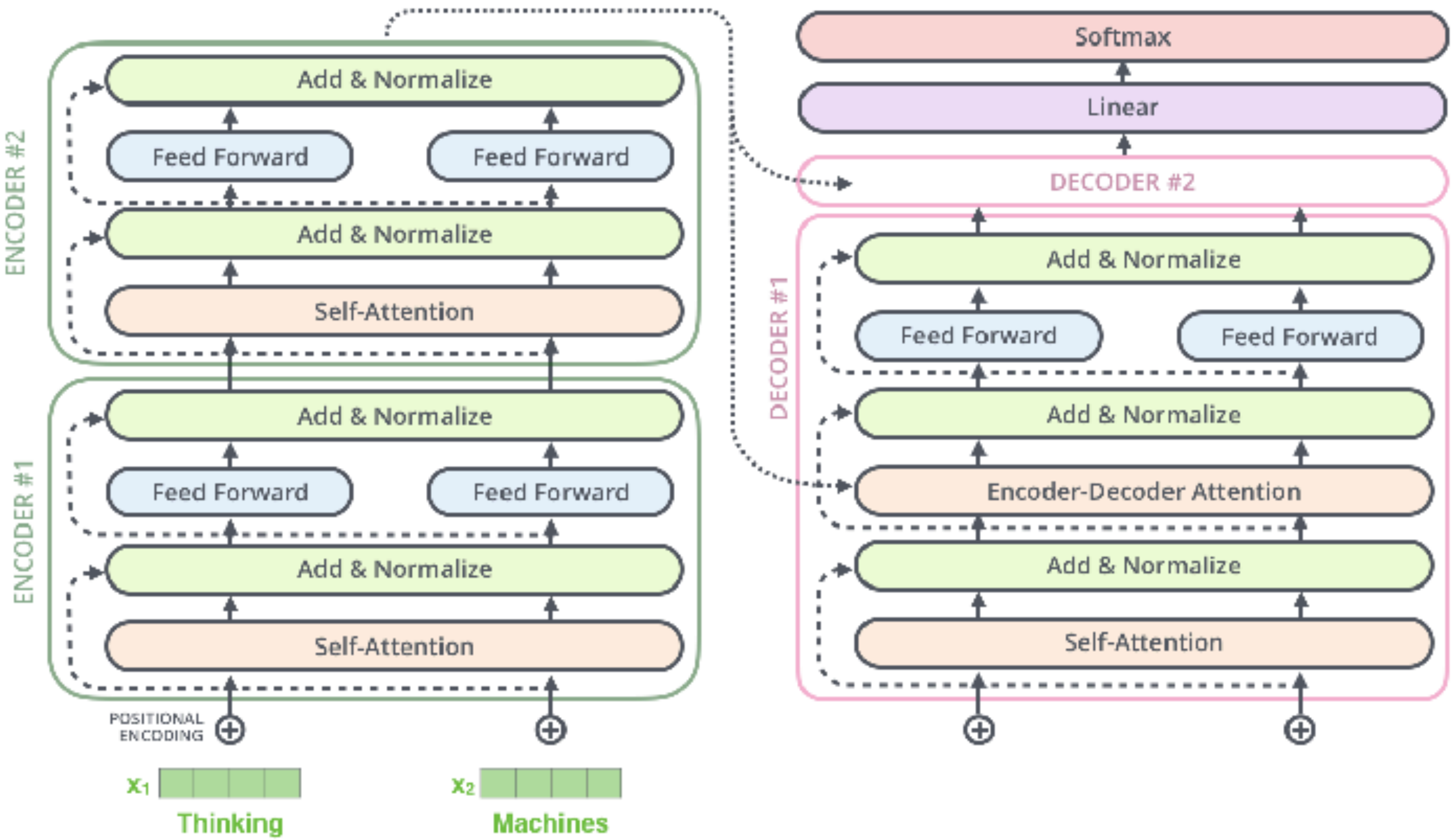


Transformer: Residual Connections

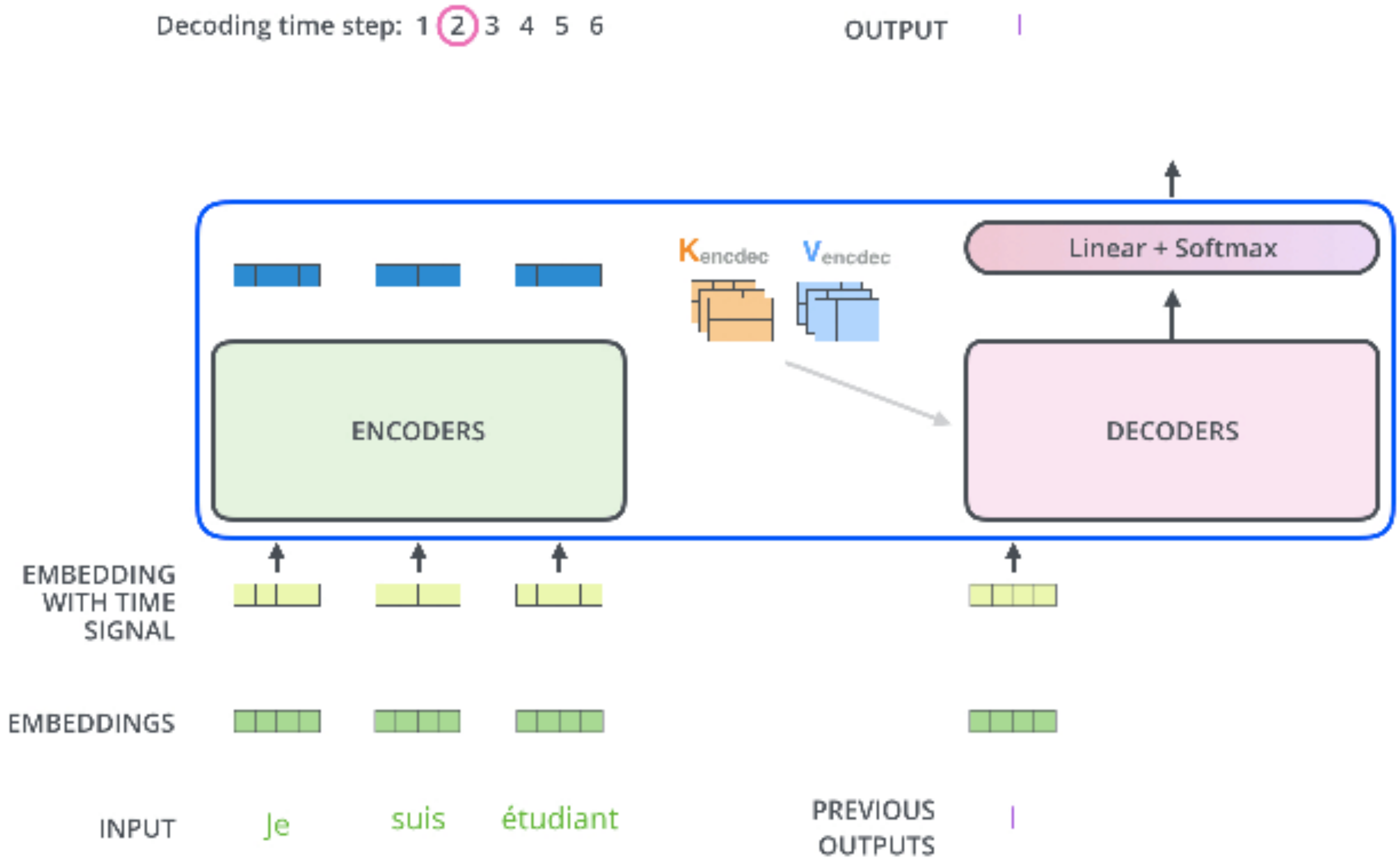


LN: prevents vanishing gradients from softmax in attention

Transformer: Putting it all together

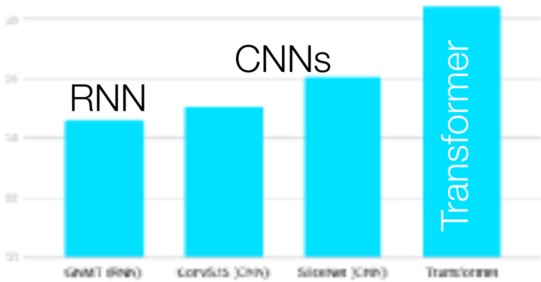


Transformer: Putting it all together

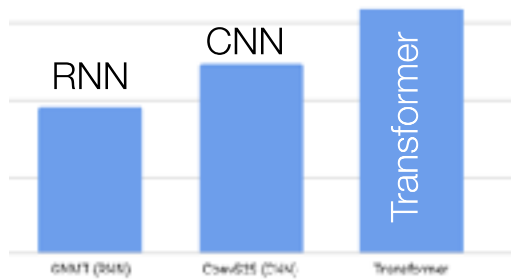


Results

Language Translation (German)



Language Translation (French)



https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Implementations:

- Many open source Keras or Tensorflow Implementations Exist
    - https://www.tensorflow.org/text/tutorials/transformer
- Many people like PyTorch for this
- HuggingFace has asme great implementations for free
    - https://huggingface.co/docs/transformers/index

Text Classification GLUE

Task	Dataset Variant	Best Model
Text Classification	GLUE	deberta-v3-small
Sentiment Analysis	SST-2 Binary classification	T5-11B
Semantic Textual Similarity	STS Benchmark	StructBERTRoBERTa ensemble
Natural Language Inference	MultiNLI	T5-11B
Natural Language Inference	RTE	PaLM 540B

General Language Understanding Evaluation (**GLUE**) benchmark is a collection of nine natural language understanding tasks, including single-sentence tasks CoLA and SST-2, similarity and paraphrasing tasks …

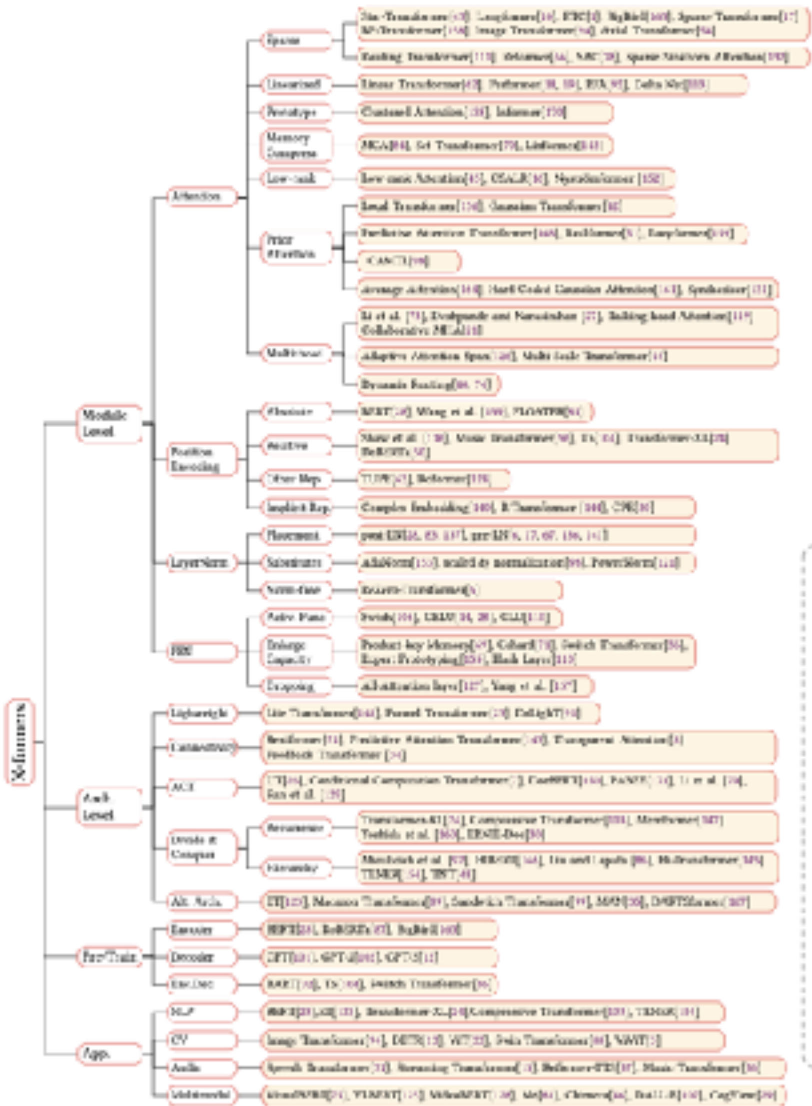
IMDb Classifications

Task	Dataset Variant	Best Model
Text Classification	IMDb	ERNIE-Doc-Large
Sentiment Analysis	IMDb	XLNet
Sentiment Analysis	User and product information	MA-BERT
SQL Parsing	IMDb	Seq2Seq with copying
Node Clustering	IMDb	MAGNN
Graph Similarity	IMDb	SimGNN
Link Prediction	IMDb	Event2vec

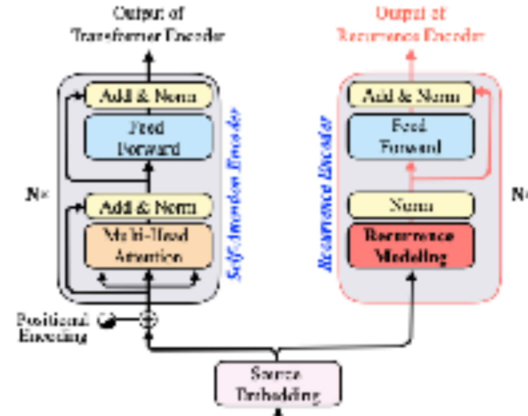
MELVA Results (my lab)

- Measuring English Language Vocabulary Acquisition
- Or results from my lab:
  - Can students use science terms in a sentence?
  - Collect and transcribe student verbal responses regarding a scientific term
- Combine transcribed sentence and “good example”
- Collected about 6000 sentences
- Put through a language model (recurrent or transformer)
- Transfer learn based upon LM output
  - Without transformer LM: ~78%
  - With transformer LM: ~84%

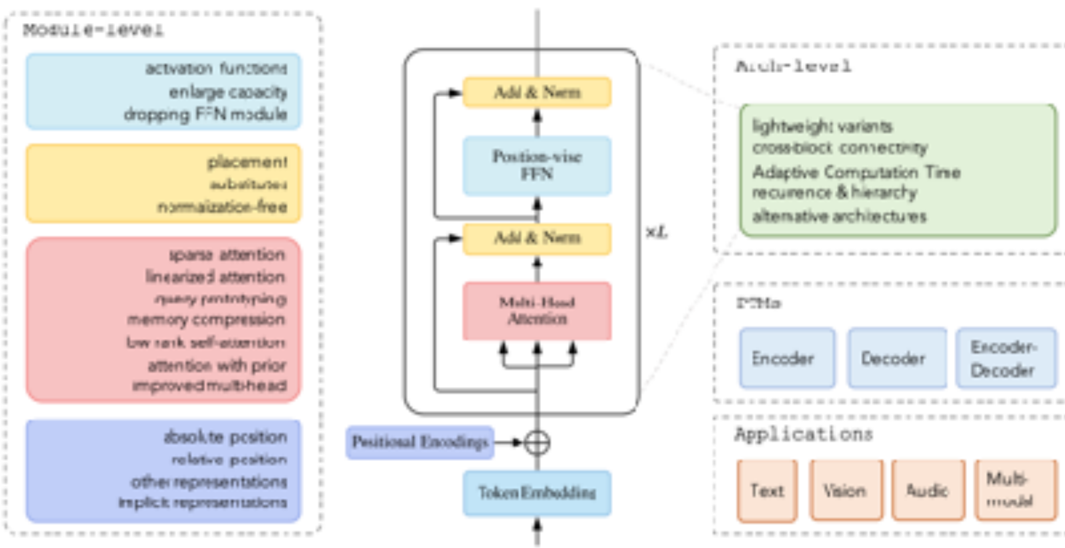
Lots of Transformer Variants



Architecture Tuning Matters Some
Pre-Training Matters
Sparse Attention Matters
Positional Encoding Doesn't
Recurrence Might… ?
X-formers are NOT just for Text



Hao et al. 2019



Lin et al "Survey of X-formers, 2021, https://arxiv.org/pdf/2106.04554.pdf

Lecture Notes for

Neural Networks and Machine Learning

Transformers

Next Time:
SSL, Multi-Modal and Multi-Task

Reading: Keras F-API