Details of the T5 Model

(Exploring the limits of **T**ransfer Learning with a Unified **T**ext-**T**o-**T**ext **T**ransformer)

By: Hiva Mohammadzadeh

Structure of the model

- Pre-trained deep learning model that uses text-to-text transformer.
- An encoder-decoder only model - Reads the entire sequence at once allowing the model to learn the context of a word based on all of its surrounding words.
- Consists of 12 Transformer Encoder Decoder blocks where each transformer block has two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network.
- We will use T5 with 128 tokens and an embedding dimension of 768.
- Each transformer block has 768 hidden units (Embeddings), 3072 feed-forward filter size, and a residual connection.

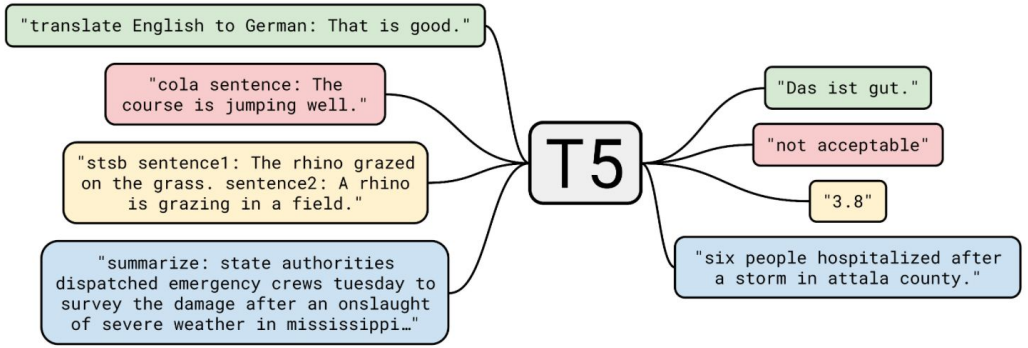
Training Scheme

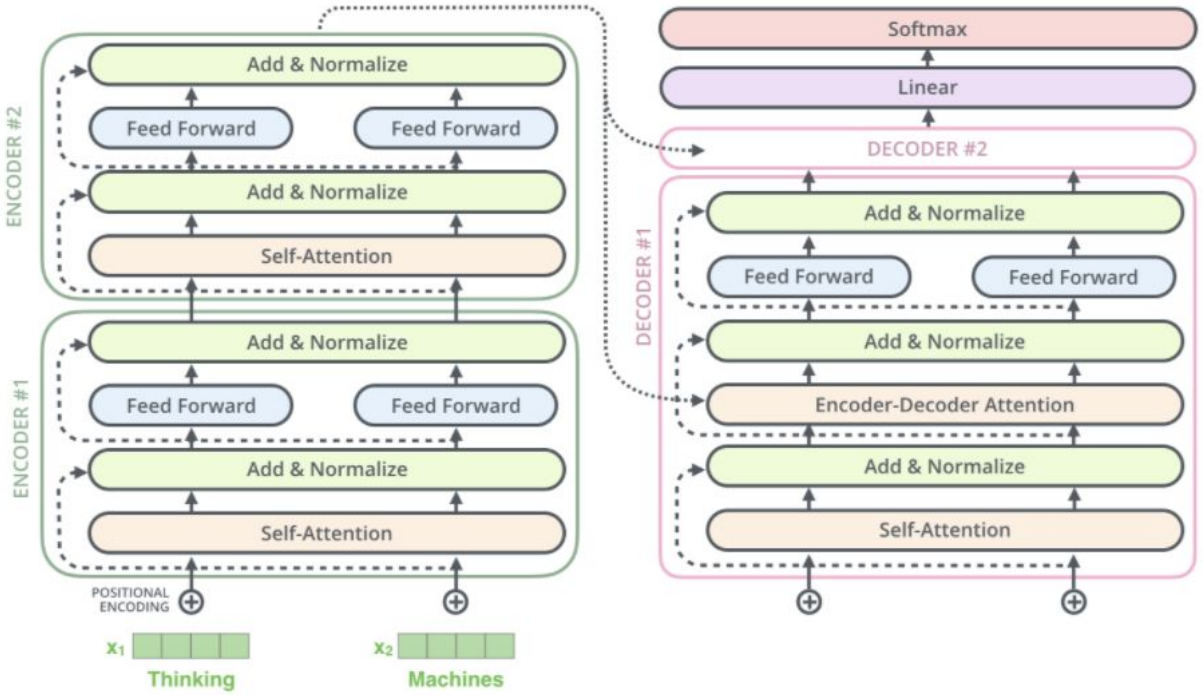
- Pre-Training:
  a. Pre-trained on C4 data. n the original text, some words are masked out with a unique sentinel token. Words are masked out independently uniformly at random. The model is trained to predict basically sentinel tokens to delineate the masked out text.
  b. This allows the model to be good at filling in the blanks missing from the input.
- Fine-Tuning:
  a. Fine-Tune the model on specific downstream tasks by converting all the tasks into a text-to-text format.
  b. Then they use "Prefix Conditioning" where each task is specified using a text prefix that is prepended to the input text before feeding the text into the model.
  c. During training, the T5 model learns to generate output text that corresponds to the specified task, conditioned on the input text and the task prefix.

T5 Architecture

Two key ingredients:

- Input Encoder
- Task specific Decoder

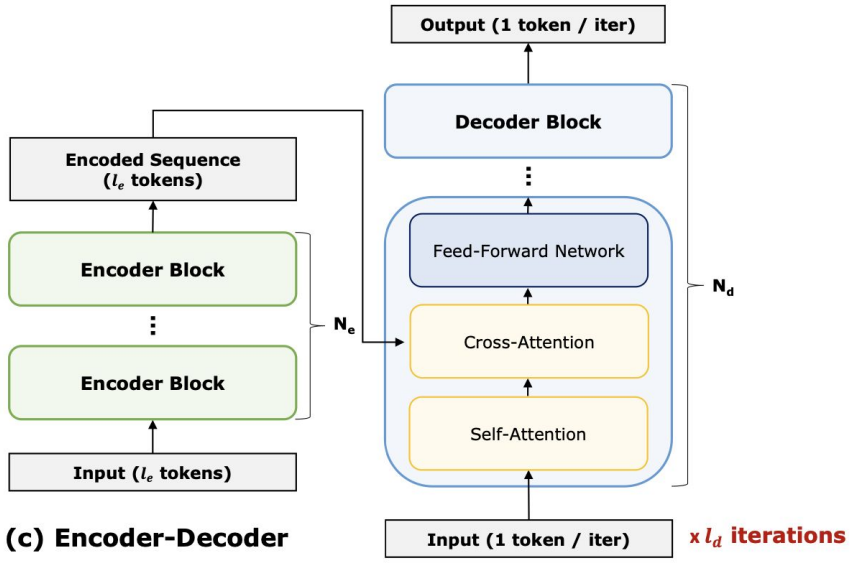




Transformer Encoder-Decoder

Two components: Attention and FFN

- Attention is used to capture the relationships between all the tokens in the input sequence.
- The Encoder block also has cross-attention to allow the decoder to also attend to the encoder inputs.
- Feed Forward Network (FNN) module is used in every transformer block to process the output of the normalization layer in a way to better fit it to the next attention layer. Same as GPT2 and BERT.



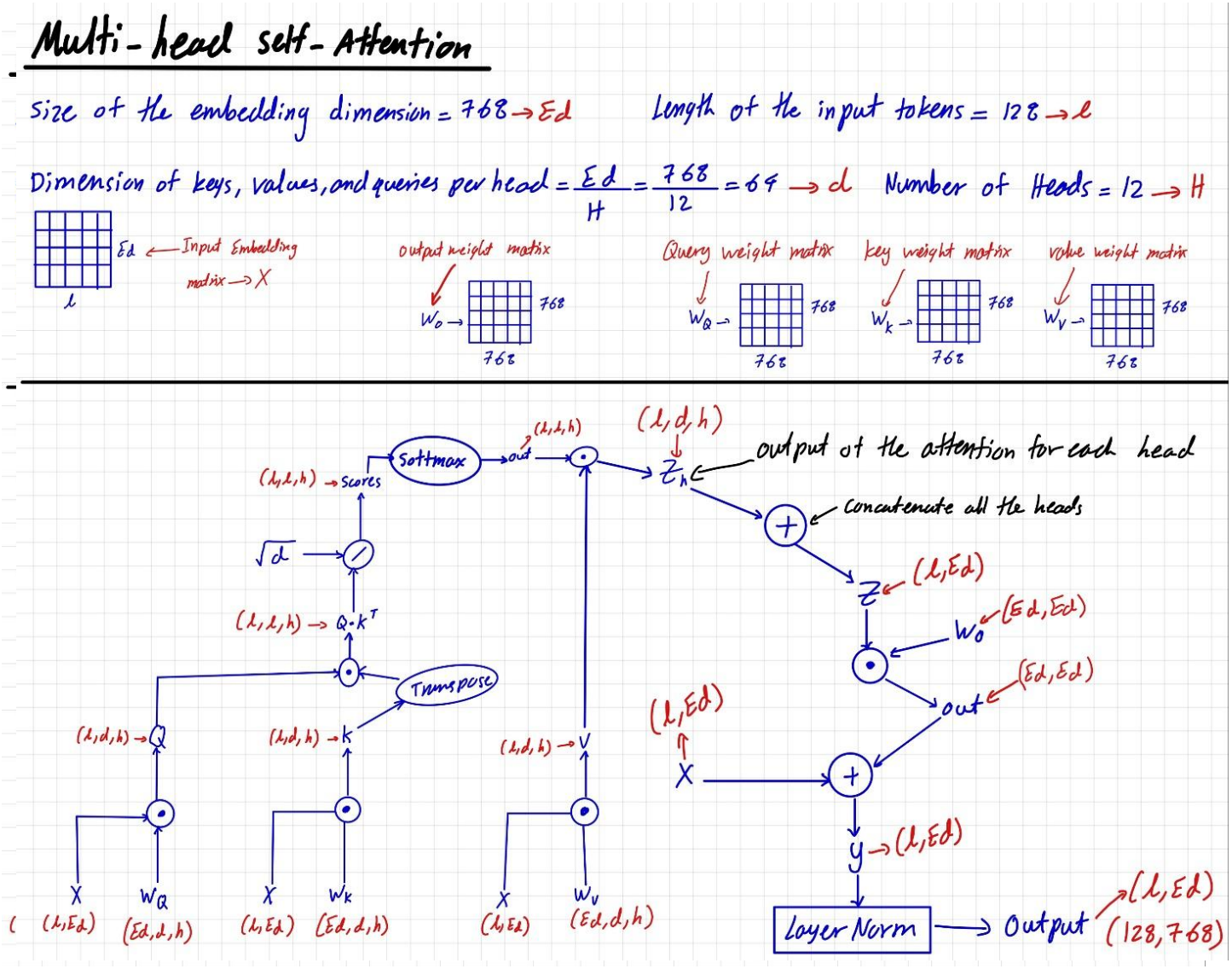
Dimensions of Encoder Weight Matrices

- The input embedding matrix has a dimension of (batch_size, sequence_length, embedding_size) = (batch_size, 128, 768). But in order to do the attention parallel for different heads, we use (batch_size, 128, 768/12 heads) = (batch_size, 128, 64, 12)
- The Wk, Wv, and Wq weight matrices used in the **self-attention mechanism Encoder** have dimensions of:
  - Query weight matrix: (batch_size, hidden_size, hidden_size) = (batch_size, 768, 768)
  - Key weight matrix: (batch_size, hidden_size, hidden_size) = (batch_size, 768, 768)
  - Value weight matrix: (batch_size, hidden_size, hidden_size) = (batch_size, 768, 768)
- The K, V, Q matrices also have dimensions of:
  - Query matrix: (batch_size, sequence_length, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
  - Key matrix: (batch_size, sequence_length, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
  - Value matrix: (batch_size, sequence_length, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
- The weight matrices used in the **feed-forward neural network** have dimensions of:
  - First dense layer: (hidden_size, 4*hidden_size) = (768, 3072).
  - Second dense layer: (4*hidden_size, hidden_size) = (3072, 768).

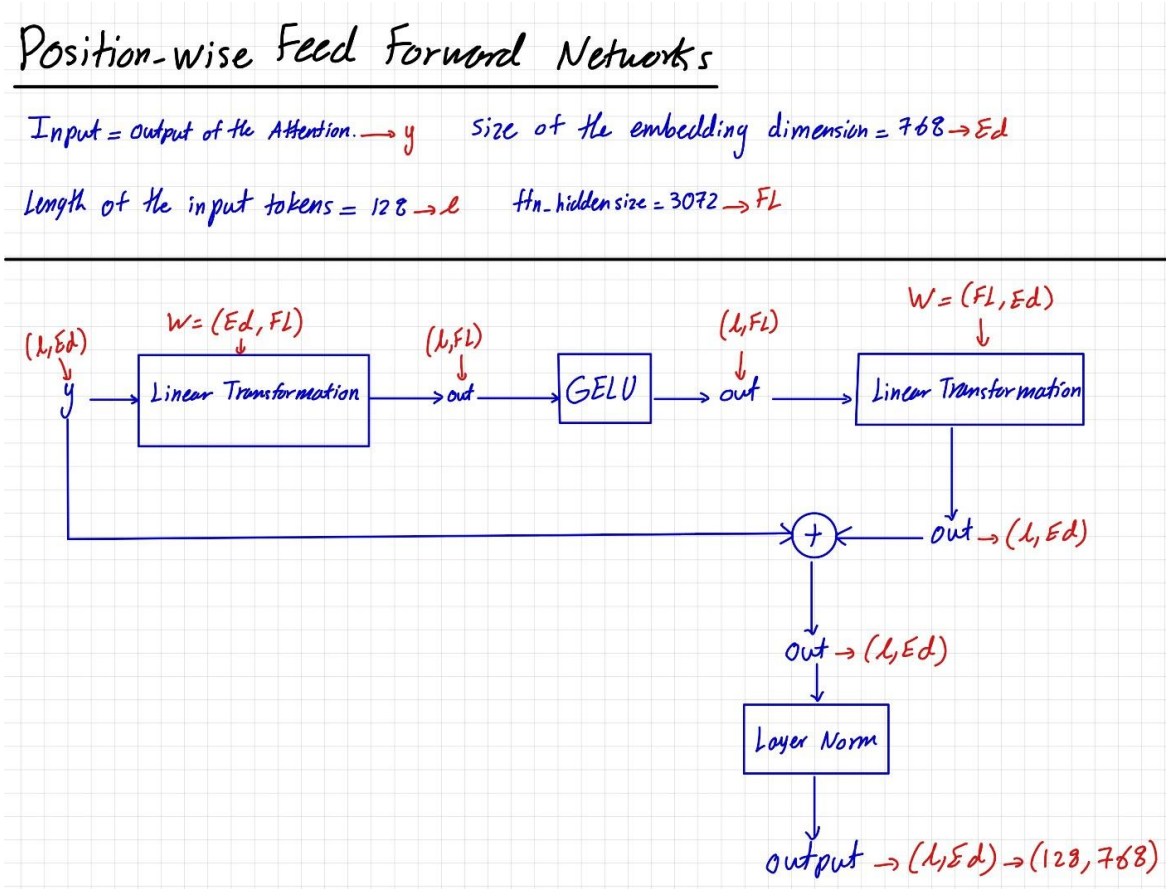
Dimensions of Decoder Weight Matrices

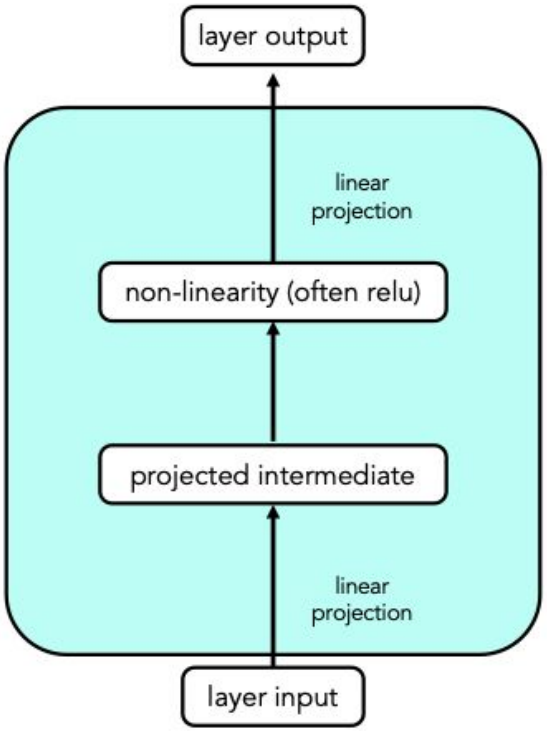
- The input embedding matrix has a dimension of (batch_size, sequence_length, embedding_size) = (batch_size, 128, 768) = (batch_size, 128, 64, 12)
- The Wk, Wv, and Wq weight matrices used in the **masked self-attention mechanism** have dimensions of:
  - Query weight matrix: (batch_size, hidden_size , hidden_size) = (batch_size, 768, 768)
  - Key weight matrix: (batch_size, hidden_size, hidden_size) = (batch_size, 768, 768)
  - Value weight matrix: (batch_size, hidden_size, hidden_size) = (batch_size, 768, 768)
- The K, V, Q matrices also have dimensions of:
  - Query matrix: (batch_size, 1 token, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
  - Key matrix: (batch_size, sequence_length, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
  - Value matrix: (batch_size, sequence_length, hidden_size) = (batch_size, 128, 768) or (batch_size, 128, 64, 12)
- The weight matrices used in the **feed-forward neural network** have dimensions of:
  - First dense layer: (hidden_size, 4*hidden_size) = (768, 3072).
  - Second dense layer: (4*hidden_size, hidden_size) = (3072, 768).

Computation Diagram of Attention of Encoder

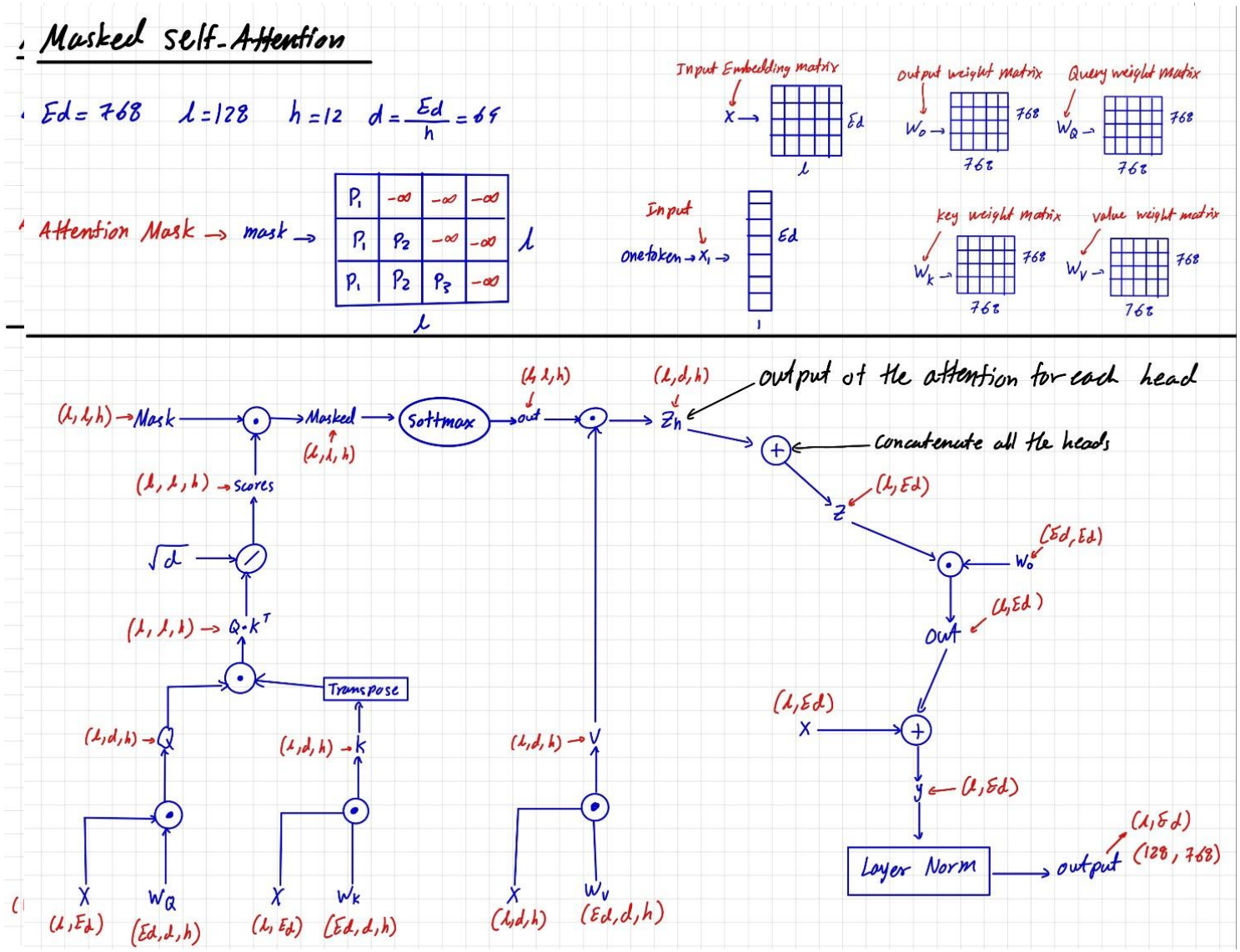


Computation Diagram of Feed Forward Layer of encoder

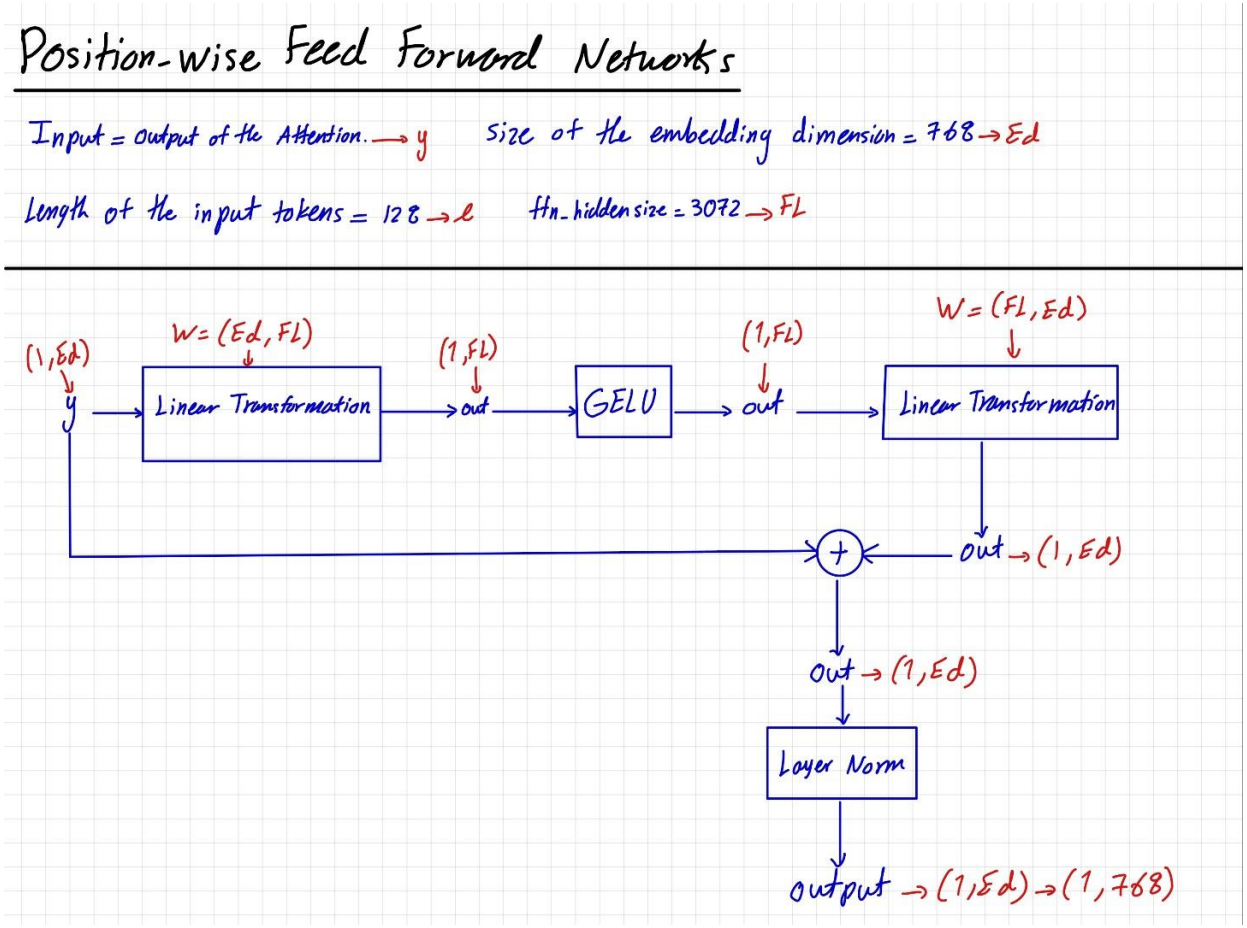


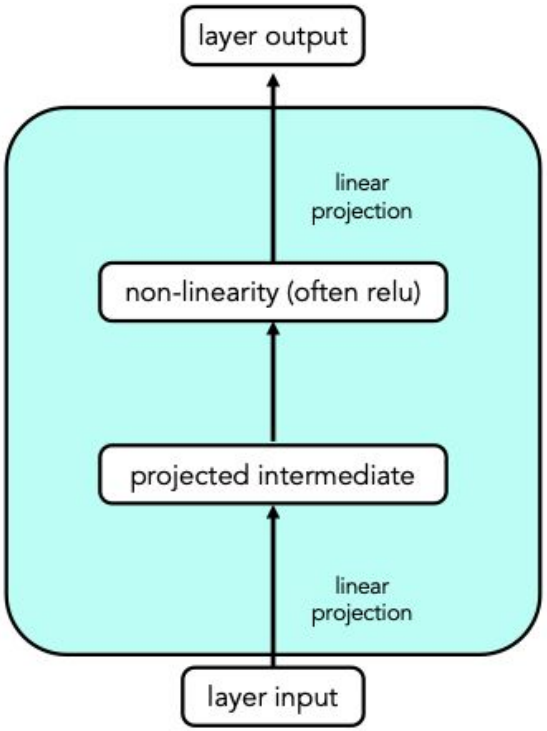


Computation Diagram of Attention of Decoder



Computation Diagram of Feed Forward Layer of Encoder





Computation Diagram of Cross Attention