
# Introduction to Transformer-Based Language Model

**Presenters : Bishnu Sarker, Sayane Shome**

**Date: 17-18 July, 2023**


# Learning Objectives of the session

Understanding the fundamental concepts behind transformers including ProtTrans transformers.

# What is a Language Modeling?

1. Goal: compute the probability of a sentence or sequence of words:

   $$P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$$

2. Related task: probability of an upcoming word:

   $$P(w_5 | w_1,w_2,w_3,w_4)$$

3. A model that computes either of these:

   $$P(W) \text{ or } P(w_n | w_1,w_2 \ldots w_{n-1})$$ is called a **language model**.



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# The Chain Rule applied to compute joint probability of words in a sentence

$$P(w_1 w_2 \ldots\ w_n) = \prod_i P(w_i \mid w_1 w_2 \ldots\ w_{i-1})$$

P("its water is so transparent") = P(its) × P(water|its) × P(is|its water)
× P(so|its water is) × P(transparent|its water is so)



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# Markov Assumption

$$P(w_1 w_2 \ldots\ w_n) \approx \prod_i P(w_i \mid w_{i-k} \ldots\ w_{i-1})$$

In other words, we approximate each component in the product

$$P(w_i \mid w_1 w_2 \ldots\ w_{i-1}) \approx P(w_i \mid w_{i-k} \ldots\ w_{i-1})$$



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# Simplest case: Unigram model

$$P(w_1 w_2 \ldots\ w_n) \approx \prod_i P(w_i)$$

**Some automatically generated sentences from a Unigram model**

```
fifth, an, of, futures, the, an, incorporated, a,
a, the, inflation, most, dollars, quarter, in, is,
mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the
```

# Bigram model

- **Condition on the previous word:**

$$P(w_i \mid w_1 w_2 \ldots \ w_{i-1}) \approx P(w_i \mid w_{i-1})$$

texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached

this, would, be, a, record, november

# N-gram models

- We can extend to trigrams, 4-grams, 5-grams.
- In general this is an insufficient model of language because language has **long-distance dependencies**:

  *"The computer which I had just put into the machine room on the fifth floor crashed."*
- But we can often get away with N-gram models.



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# Neural Language Models (LMs)

- **Language Modeling**: Calculating the probability of the next word in a sequence given some history.
  - **We've seen N-gram based LMs**
  - **But neural network LMs far outperform n-gram language models**
- State-of-the-art neural LMs are based on more powerful neural network technology like Transformers
- But **simple feedforward LMs** can do almost as well!



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# Simple feedforward Neural Language Models

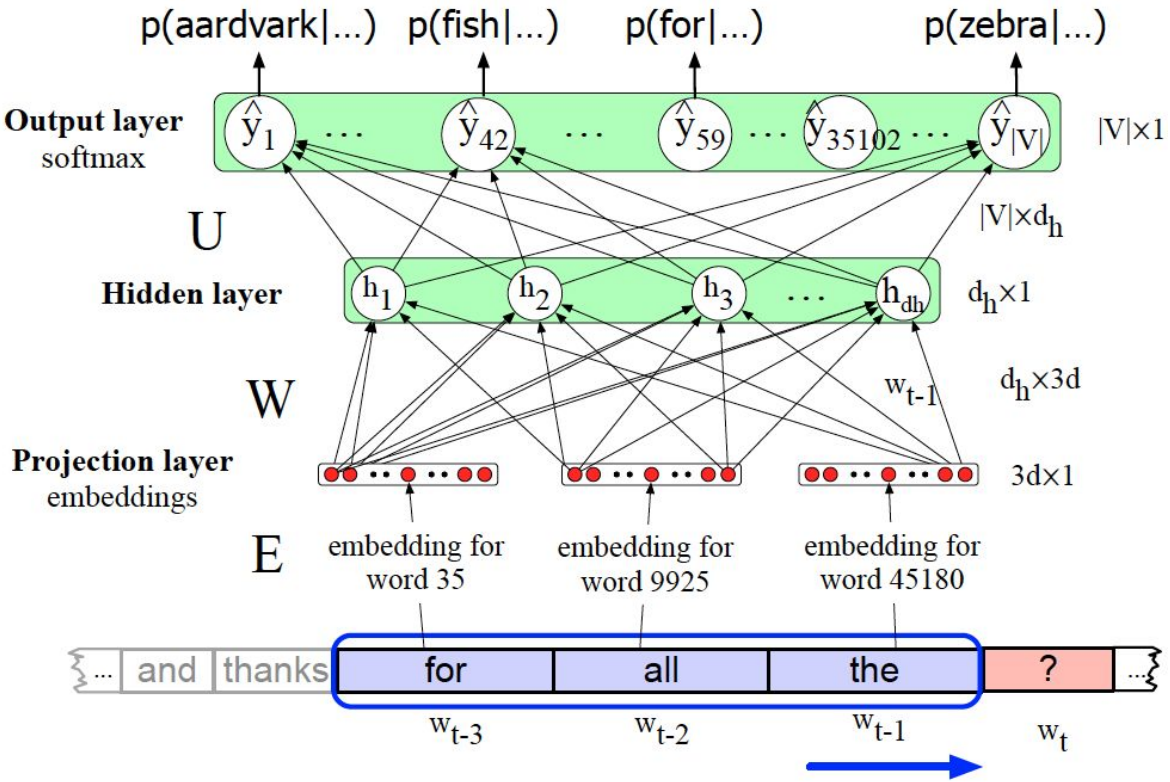
- **Task**: predict next word $w_t$, given prior words $w_{t-1}$, $w_{t-2}$, $w_{t-3}$, …
- **Problem**: Now we're dealing with sequences of arbitrary length.
- **Solution**: Sliding windows (of fixed length)

$$P(w_t|w_1^{t-1}) \approx P(w_t|w_{t-N+1}^{t-1})$$



Jurafsky, D. and Martin, J.H. (2023) Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/

# Neural Language Model



# Why Neural LMs work better than N-gram LMs

- **Training data:**
  - We've seen: I have to make sure that the cat gets fed.
  - Never seen: dog gets fed
- **Test data:**
  - I forgot to make sure that the dog gets ___
  - N-gram LM can't predict "fed"!
  - Neural LM can use similarity of "cat" and "dog" embeddings to generalize and predict “fed” after dog

# Transformers-based Large Language Model

## Attention Is All You Need


**Ashish Vaswani*** Google Brain avaswani@google.com

**Noam Shazeer*** Google Brain noam@google.com

**Niki Parmar*** Google Research nikip@google.com

**Jakob Uszkoreit*** Google Research usz@google.com

**Llion Jones*** Google Research llion@google.com

**Aidan N. Gomez*** † University of Toronto aidan@cs.toronto.edu

**Łukasz Kaiser*** Google Brain lukaszkaiser@google.com

**Illia Polosukhin*** ‡ illia.polosukhin@gmail.com



IEEE TRANS PATTERN ANALYSIS & MACHINE INTELLIGENCE, VOL. 14, NO. 8, AUGUST 2021


## Transformers: State-of-the-Art Natural Language Processing


Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, Alexander M. Rush

Hugging Face, Brooklyn, USA / {first-name}@huggingface.co



**Abstract**

Recent progress in natural language processing has been driven by advances in both model architecture and model pretraining. Transformer architectures have facilitated building higher-capacity models and pretraining has made it possible to effectively utilize this capacity for a wide variety of tasks. *Transformers* is an open-source library with the goal of opening up these advances to the wider machine learning community. The library consists of carefully engineered state-of-the art Transformer architectures under a unified API. Backing this library is a curated collection of pretrained models made by and available for the community. *Transformers* is designed to be extensible by researchers, simple for practitioners, and fast and robust in industrial deployments. The library is available at https://github.com/huggingface/transformers.


(Liu et al., 2019b; Wang et al., 2018, 2019), machine translation (Lample and Conneau, 2019a), coreference resolution (Joshi et al., 2019), commonsense inference (Bosselut et al., 2019), and summarization (Lewis et al., 2019) among others.

This advance leads to a wide range of practical challenges that must be addressed in order for these models to be widely utilized. The ubiquitous use of the Transformer calls for systems to train, analyze, scale, and augment the model on a variety of platforms. The architecture is used as a building block to design increasingly sophisticated extensions and precise experiments. The pervasive adoption of pretraining methods has led to the need to distribute, fine-tune, deploy, and compress the core pretrained models used by the community.

*Transformers* is a library dedicated to supporting Transformer-based architectures and facilitating the distribution of pretrained models. At the core of

## ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Learning


Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik and Burkhard Rost

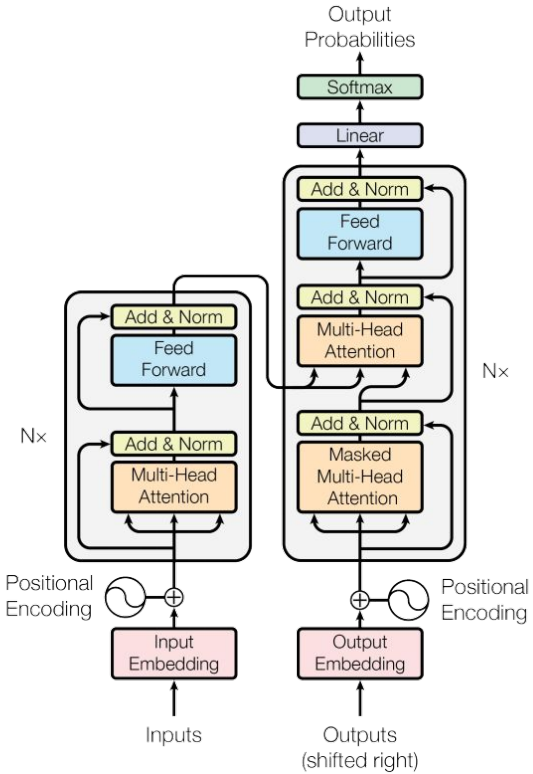


**Abstract—**Computational biology and bioinformatics provide vast data gold-mines from protein sequences, ideal for Language Models taken from NLP. These LMs reach for new prediction frontiers at low inference costs. Here, we trained two auto-regressive models (Transformer-XL, XLNet) and four auto-encoder models (BERT, Albert, Electra, T5) on data from UniRef and BFD containing up to 393 billion amino acids. The LMs were trained on the Summit supercomputer using 5616 GPUs and TPU Pod up-to 1024 cores. Dimensionality reduction revealed that the raw protein LM-*embeddings* from unlabeled data captured some biophysical features of protein sequences. We validated the advantage of using the *embeddings* as exclusive input for several subsequent tasks. The first was a per-residue prediction of protein secondary structure (3-state accuracy Q3=81%-87%); the second were per-protein predictions of protein sub-cellular localization (ten-state accuracy: Q10=81%) and membrane vs. water-soluble (2-state accuracy Q2=91%). For the per-residue predictions the transfer of the most informative embeddings (ProtT5) for the first time outperformed the state-of-the-art without using evolutionary information thereby bypassing expensive database searches. Taken together, the results implied that protein LMs learned some of the *grammar* of the *language of life*. To facilitate future work, we released our models at https://github.com/agemagician/ProtTrans.



**Index Terms—**Computational Biology, High Performance Computing, Machine Learning, Language Modeling, Deep Learning

# Transformers-based Large Language Model

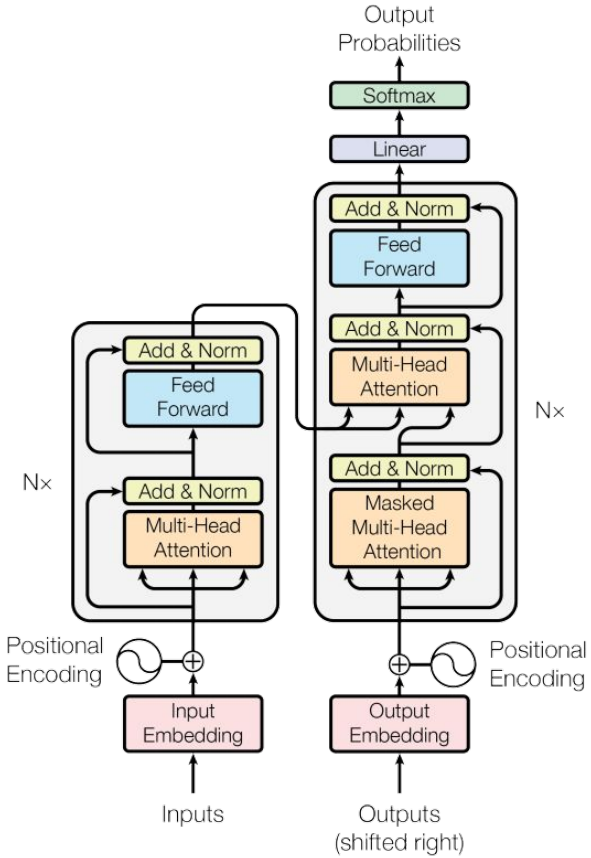



- The de-facto sequence model architecture includes multiple identical encoders and decoders.
- Each encoder consists of 1) **an attention layer**, 2) Feedforward layer.
- Through 8 heads, the attention layer attend different parts of the input.
- Each token is passed to individual feed forward neural network.
- The output from the encoder is passed through the top level encoders until fed to the decoders.
- The output from the top encoder is used as embeddings.



Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

# Transformers-based Large Language Model




Embedding Layer
Position Encoding

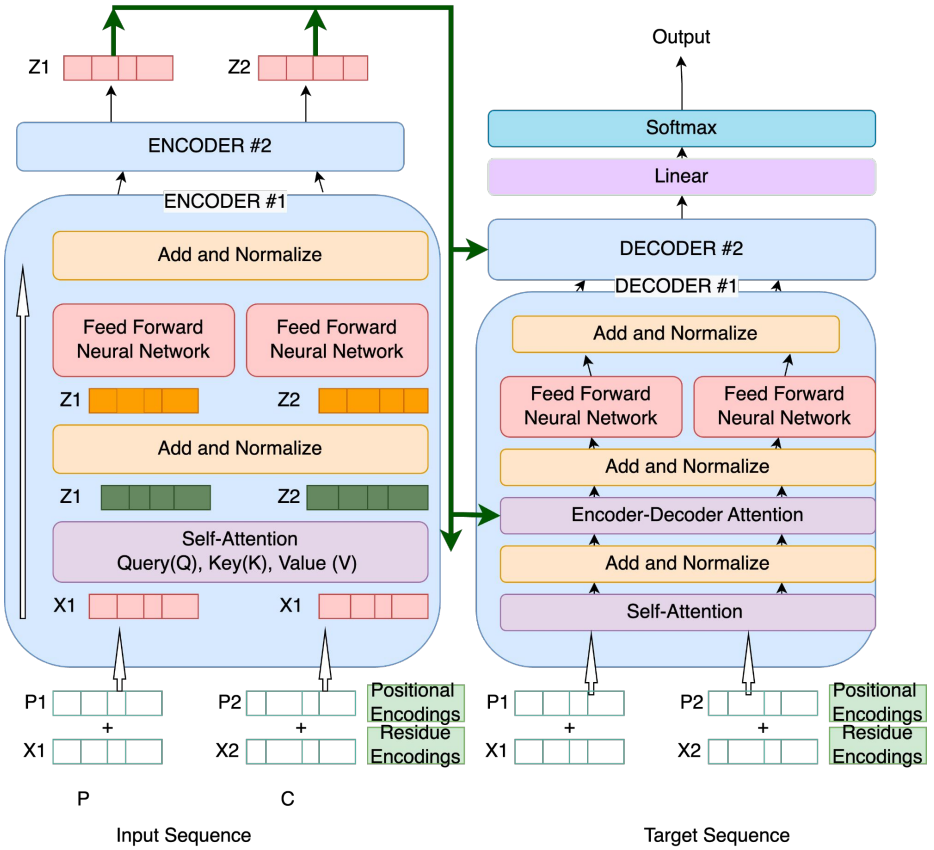
Multi-Head Attention Layer
Feed Forward Layer

Multi-Head Attention Layer
Encoder-Decoder Attention Layer
Feed Forward Neural Network

Linear Layer
Softmax Layer

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

# Transformers-based Large Language Model




Embedding Layer
Position Encoding

Multi-Head Attention Layer
Feed Forward Layer

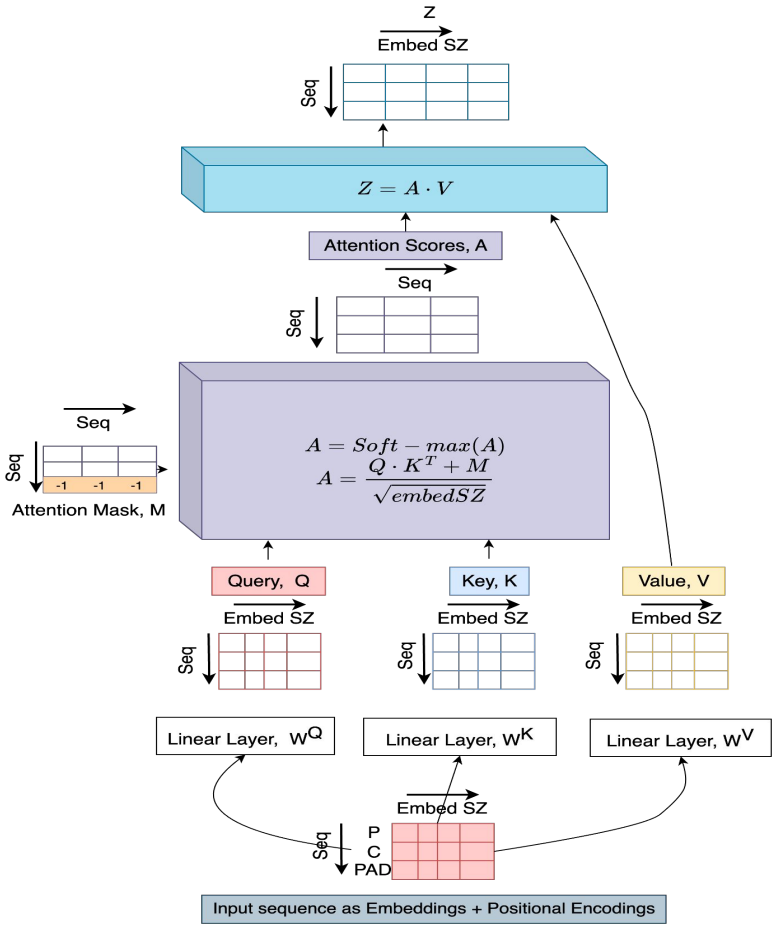
Multi-Head Attention Layer
Encoder-Decoder Attention Layer
Feed Forward Neural Network

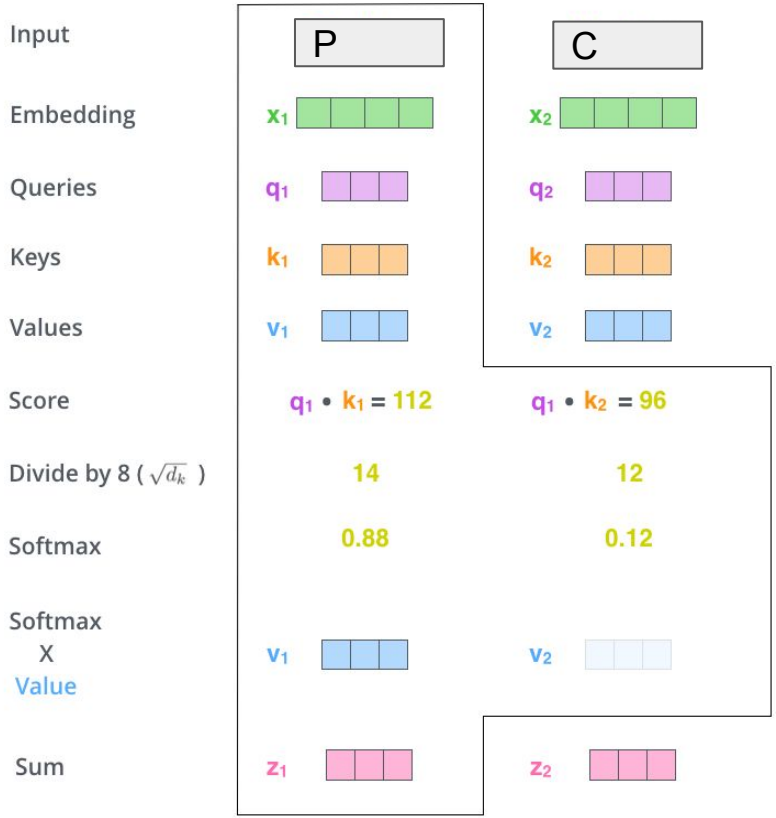
Linear Layer
Softmax Layer



Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

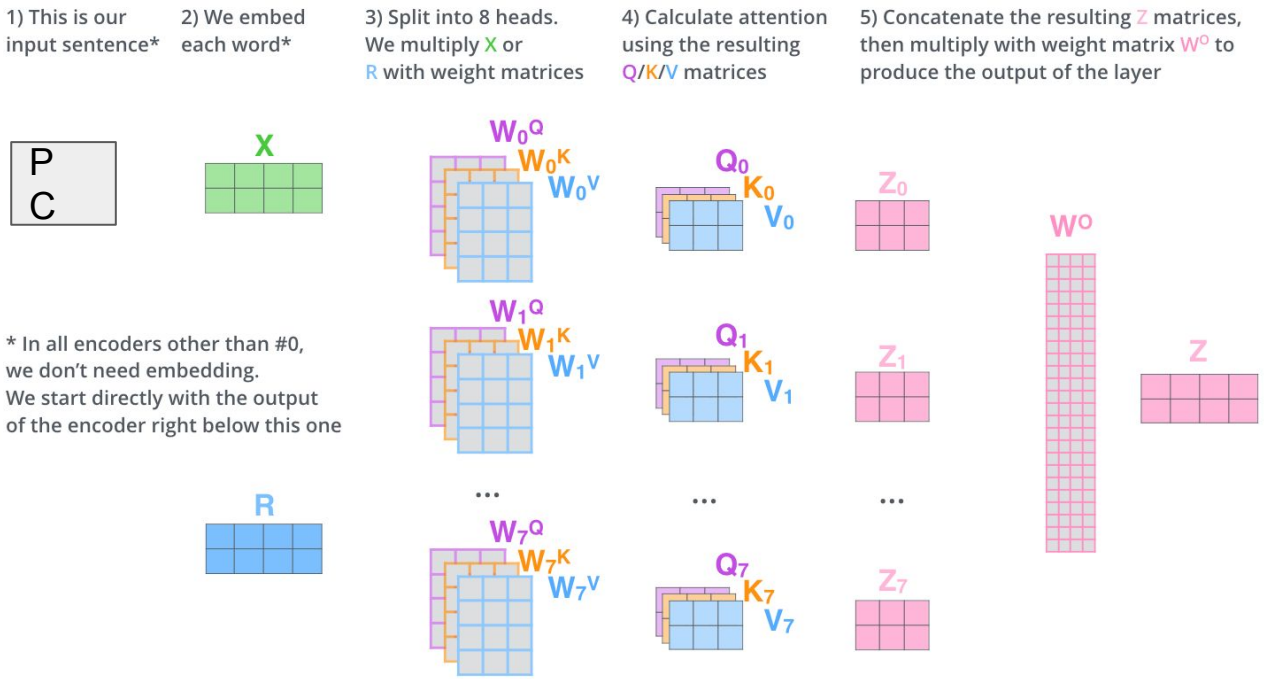
# Transformers Self Attention Layer




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

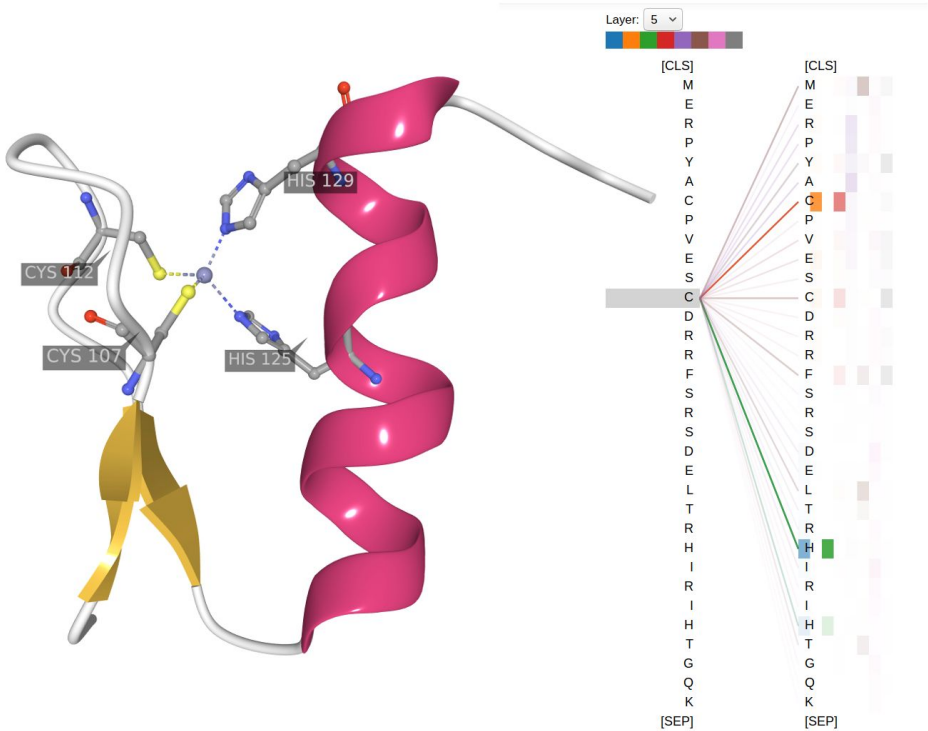
# Multi-Head Attention





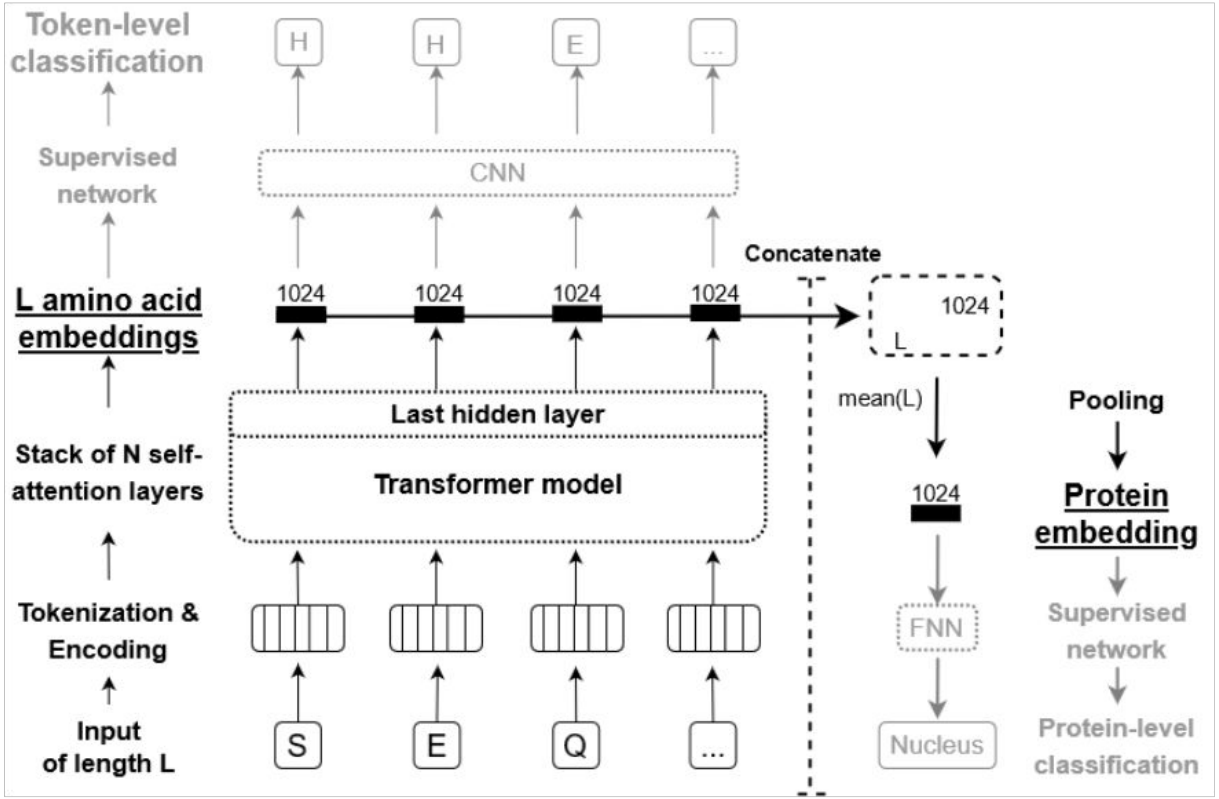

*Alammar, J (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/*

# Attention in Sequence Analysis




Elnaggar, Ahmed, et al. "Prottrans: Toward understanding the language of life through self-supervised learning." *IEEE transactions on pattern analysis and machine intelligence* 44.10 (2021): 7112-7127. https://github.com/agemagician/ProtTrans

# ProtTrans Architecture for Sequence Embedding




Elnaggar, Ahmed, et al. "Prottrans: Toward understanding the language of life through self-supervised learning." *IEEE transactions on pattern analysis and machine intelligence* 44.10 (2021): 7112-7127.

# Hands on Tutorial

*Google colab notebook*

Link : Colab-Notebook-Transformer

# Break !

We will reconvene in 15 mins. Meanwhile, we are available for Q/As

Next in line : **Hands-on case study of Protein Function Annotation**