Fine-tuning a transformer pipeline for text classification

Sep 16th, 2025

French NACE dataset

- Origin dataset:
  - 846652 rows labelled with French NACE version with 5 digits

nace	text
8220Z	MISSIONS PONCTUELLES A L AIDE D UNE PLATEFORME
8553Z	INSPECTEUR AUTOMOBILE
5520Z	LA LOCATION TOURISTIQUE DE LOGEMENTS INSOLITES...
4791A	COMMERCE DE TOUT ARTICLES ET PRODUITS MARCHAND...
9499Z	REGROUPEMENT RETRAITE

- Simplification for this tutorial :
  - Convert 5 digits to 1 digits (NACE section: 21 category from A to U)
  - Very small sample for train (18k) and test (2k)

Text as input is sophisticated

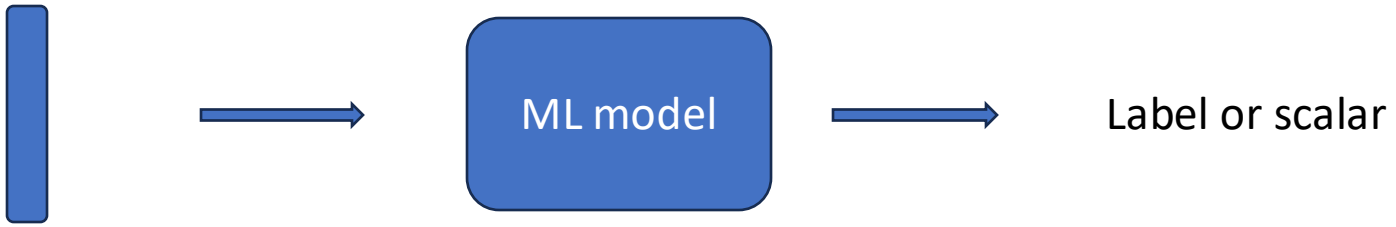
- ML model expect vectors, not strings

- Conversion from word to vectors
  - One hot encoding, ngrams, … etc
  - Word embeddings (word2vec, fasttext)

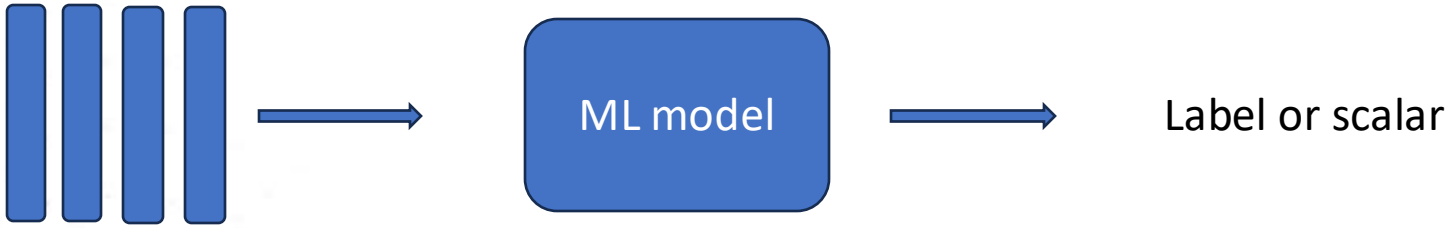
- Limited to deal with words that have different meanings (context)
  - Example:
    - *Bank* vs *river bank* vs *blood bank*?

ML model inputs

- Input is a vector



- Input is a 2D vector (image)



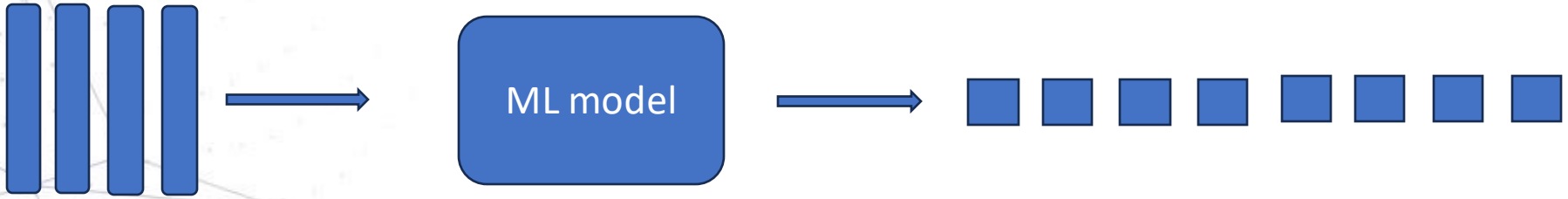
- A sentence is a list of words with unknown length (Sequence) <- Our case

ML model outputs

- Each vector has a label: POS tagging, BERT

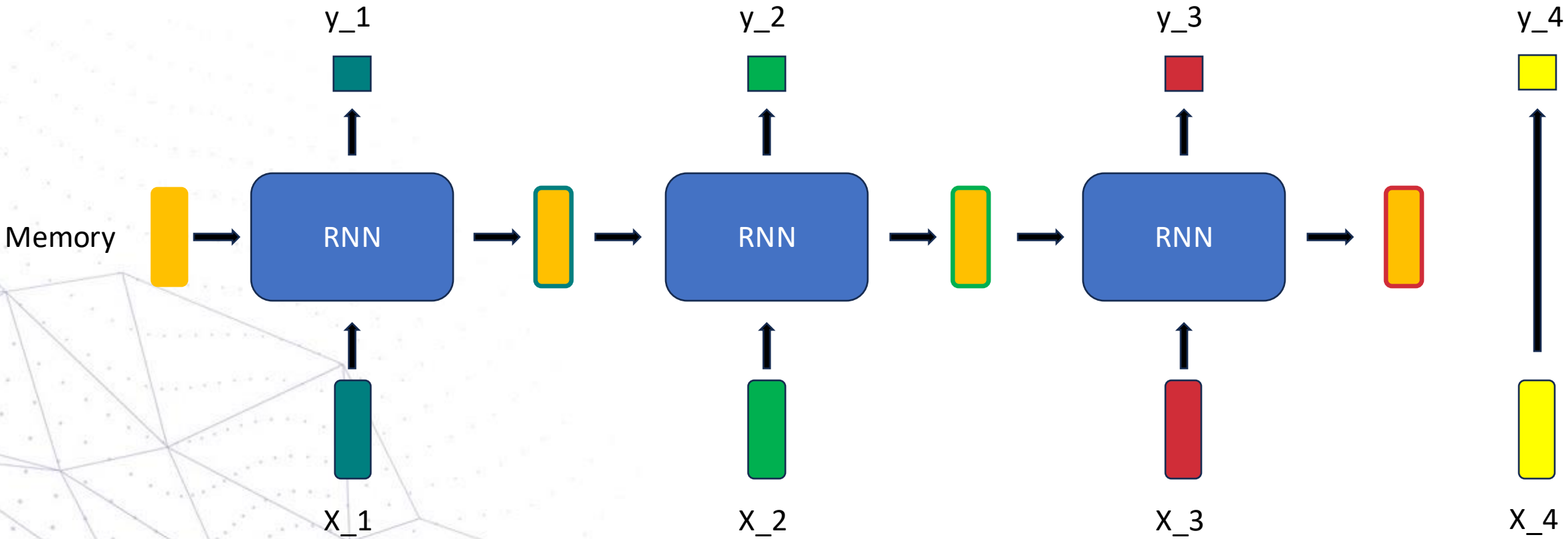
- Whole sequence has a label: Text classification <- Our case

- Model decide length of the labels: Text generation (GPT), text translation



Sequence model

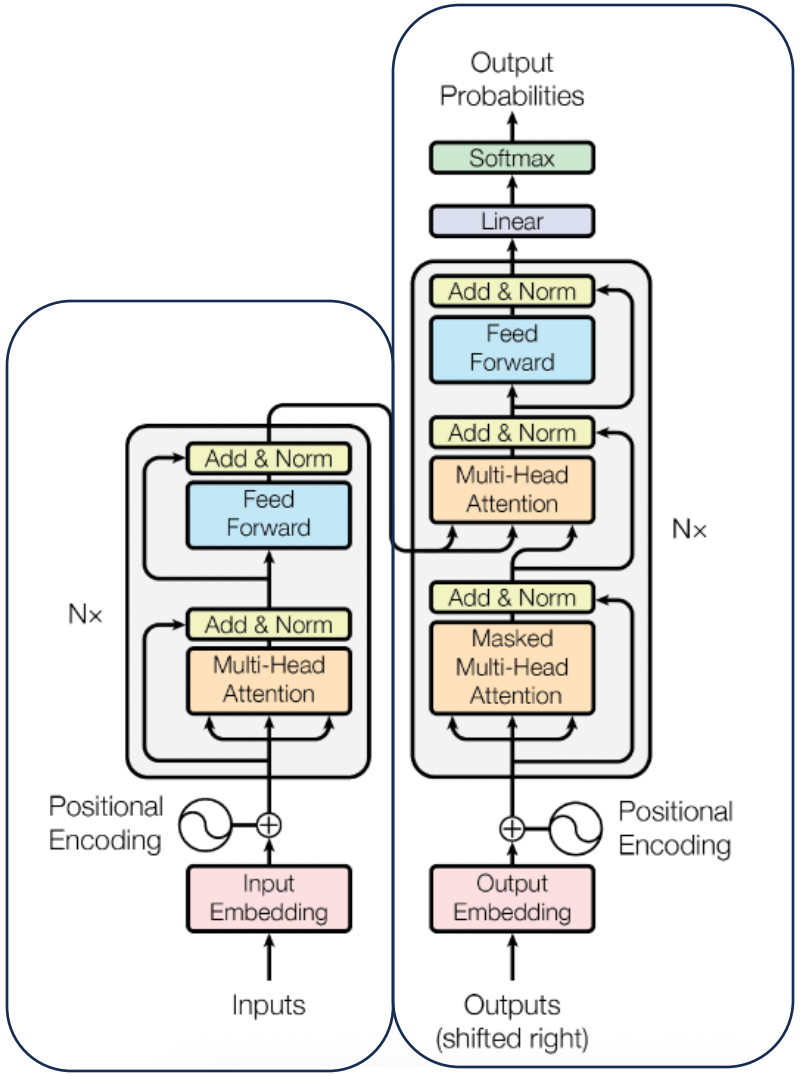
- Recurrent Neural Network (RNN): Given input sequence (X_1, X_2, X_3, …)
  - Model is reused for each X and store information to the memory



Transformer and self attention

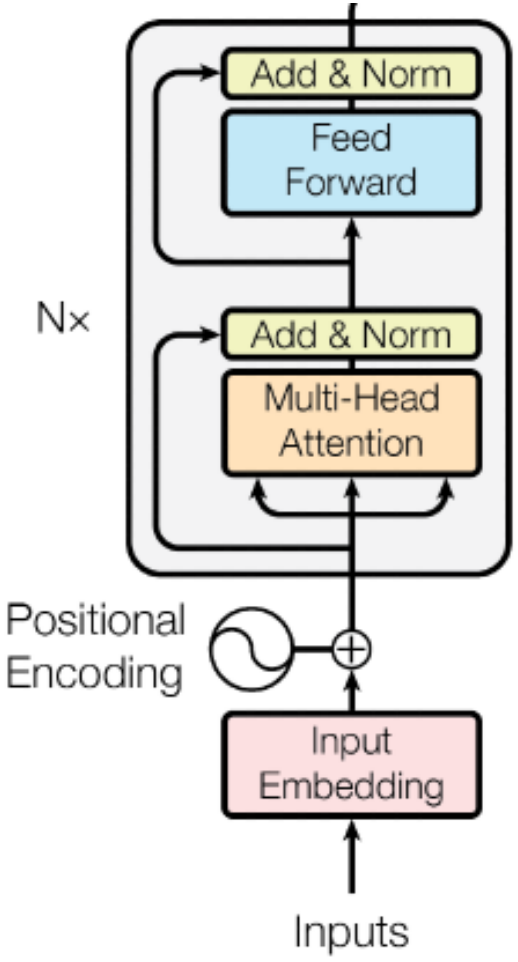
- Attention is all you need
- Encoder – Decoder architecture

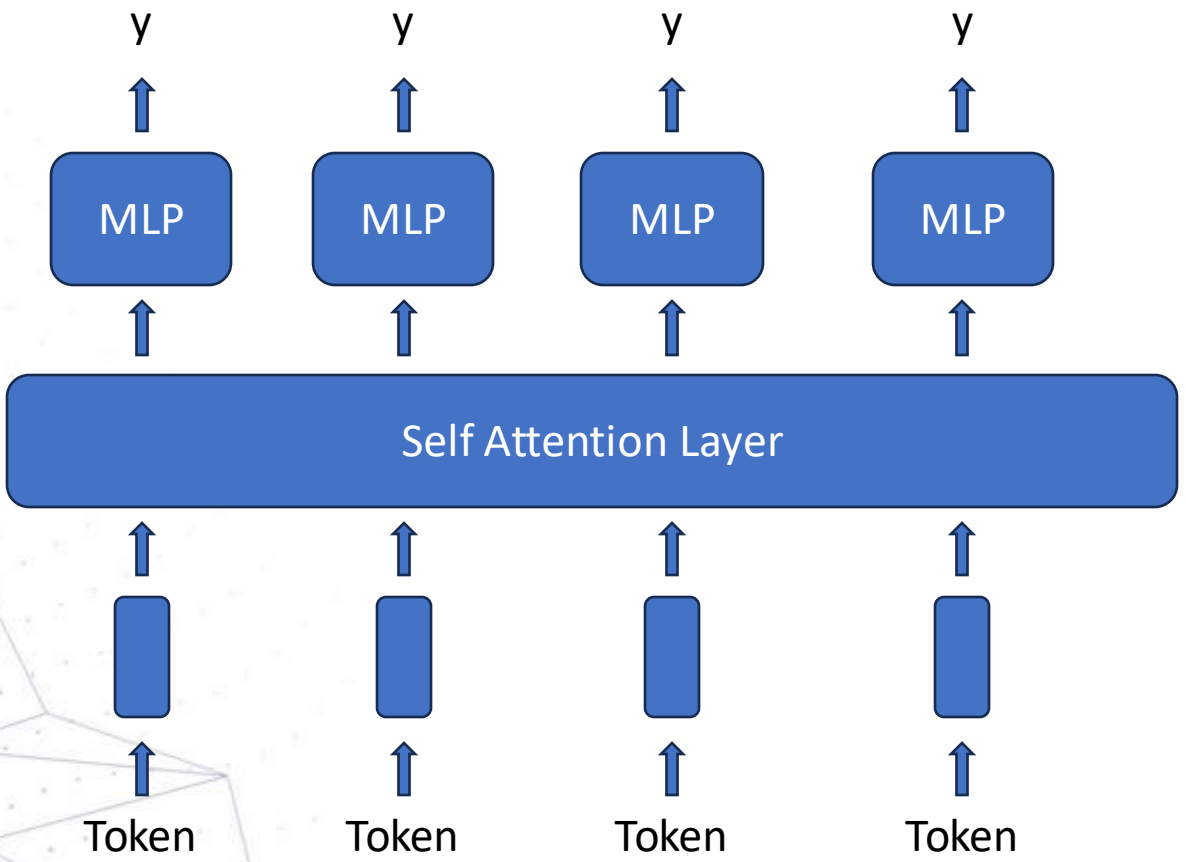
Encoder:
Bidirectional Encoder Representations from Transformers (BERT)



Decoder:
GPT

Self attention layer



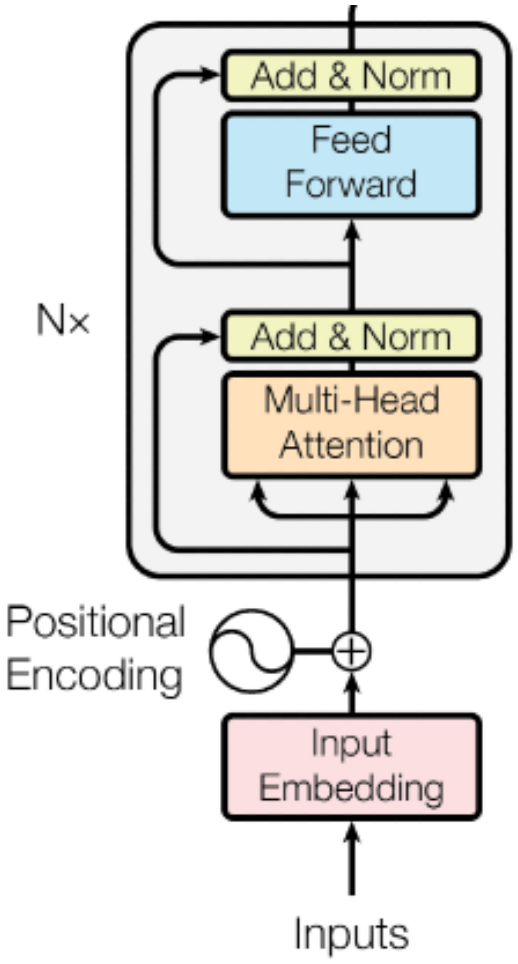


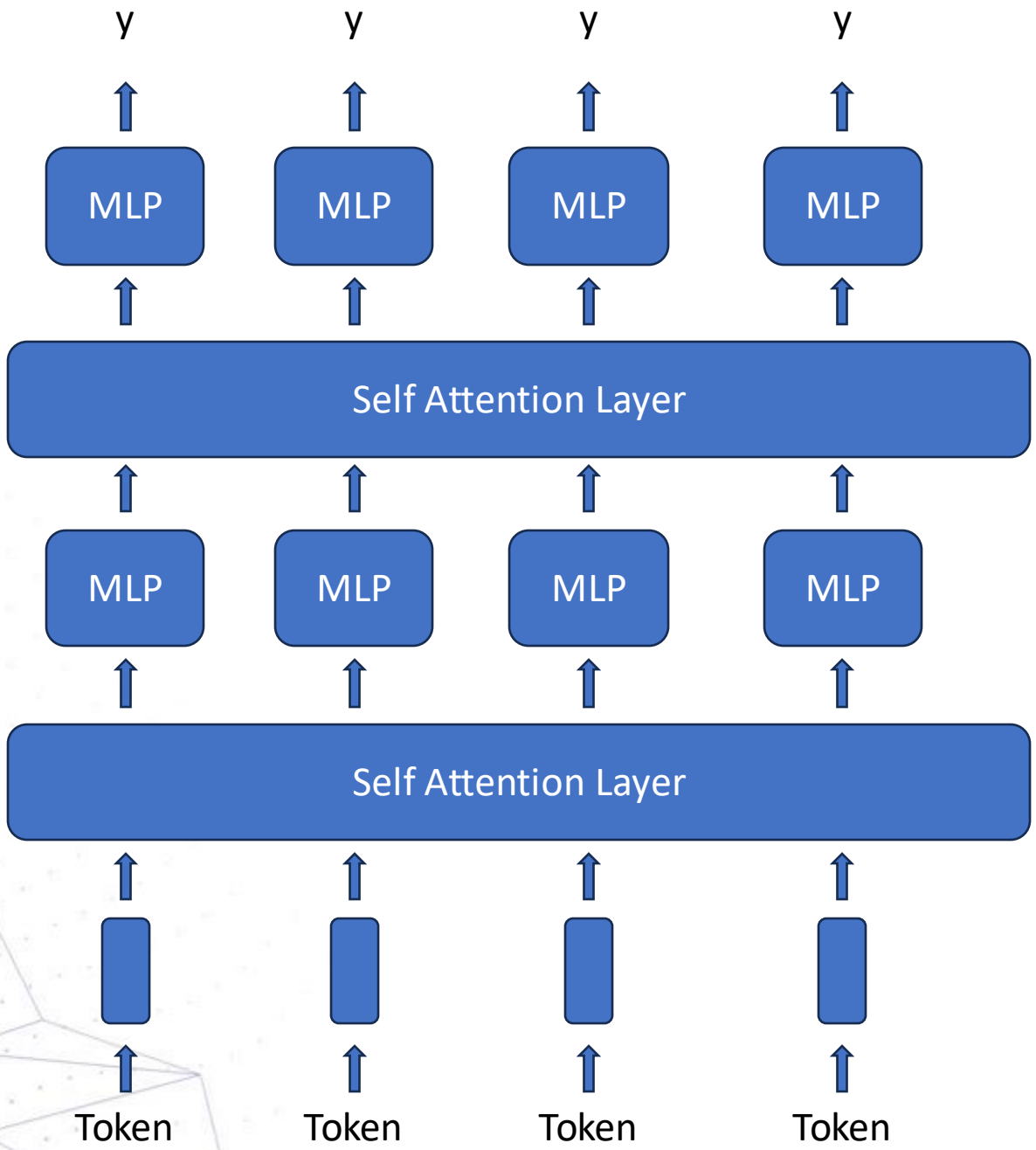
Same MultiLayer Perceptron

Composition of linear algebra operations

Token embedding

BERT



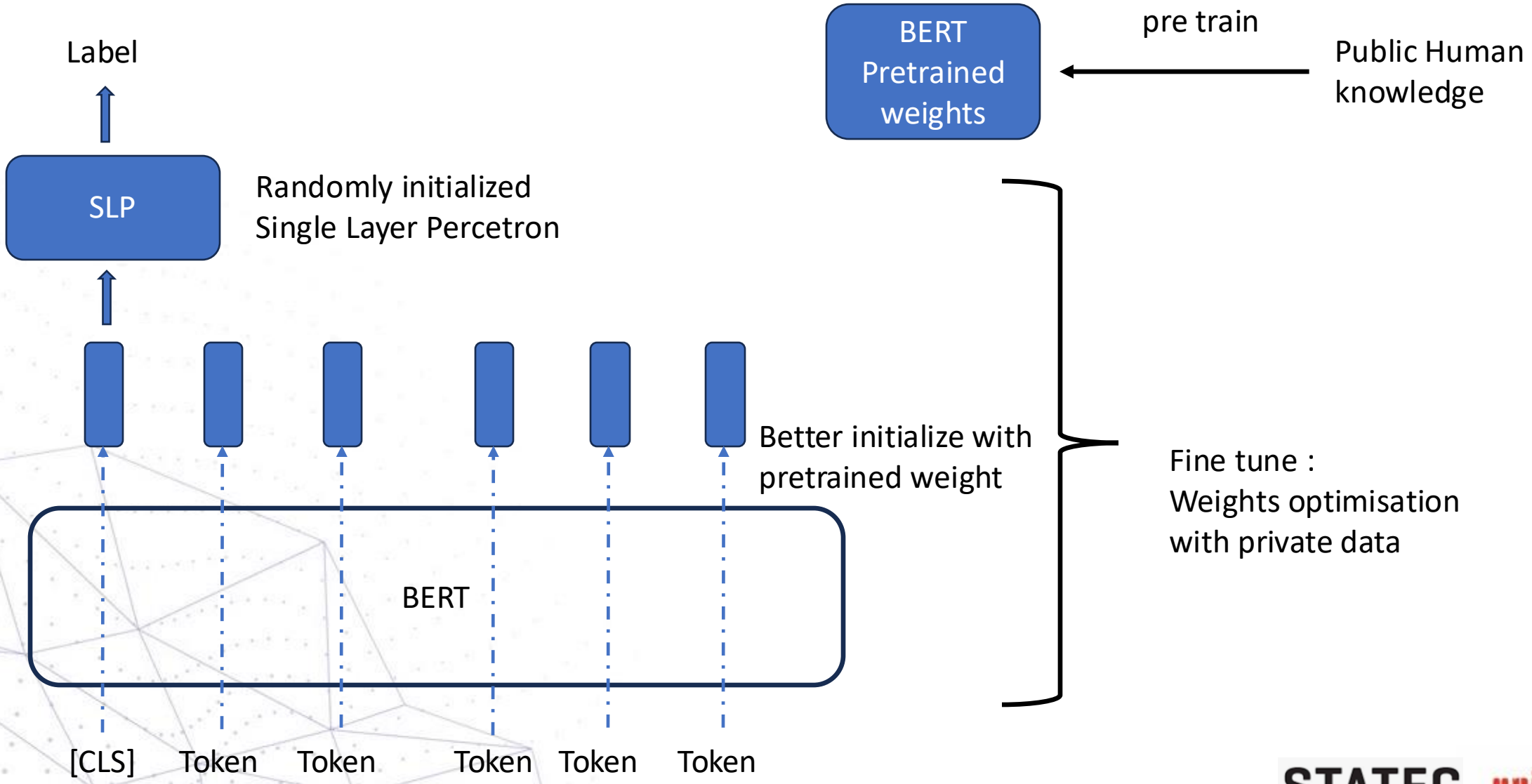


Same MultiLayer Perceptron

Composition of linear algebra operations

Token embedding

Text classification as downstream task of BERT



transformers: High level python package

Generic class:

- AutoModel
- AutoTokenizer
- Trainer

Task specific model class:

- **AutoModelForSequenceClassification** -> Text classification tasks with encoder
- AutoModelForCausalLM -> Text generation with decoder

Task pipeline: pipeline function

Tutorial: Overview of steps

1. Convert csv dataset into hugging face datasets format
2. Label encoding
3. Fetch pre trained model from hugging face
4. Fine Tune the pre trained model
5. Run fine tuned model on test csv dataset and evaluate

References

Attention is all you need

BERT

Hugging face transformers

- BERT model
- Auto Classes doc

ML course from NTU