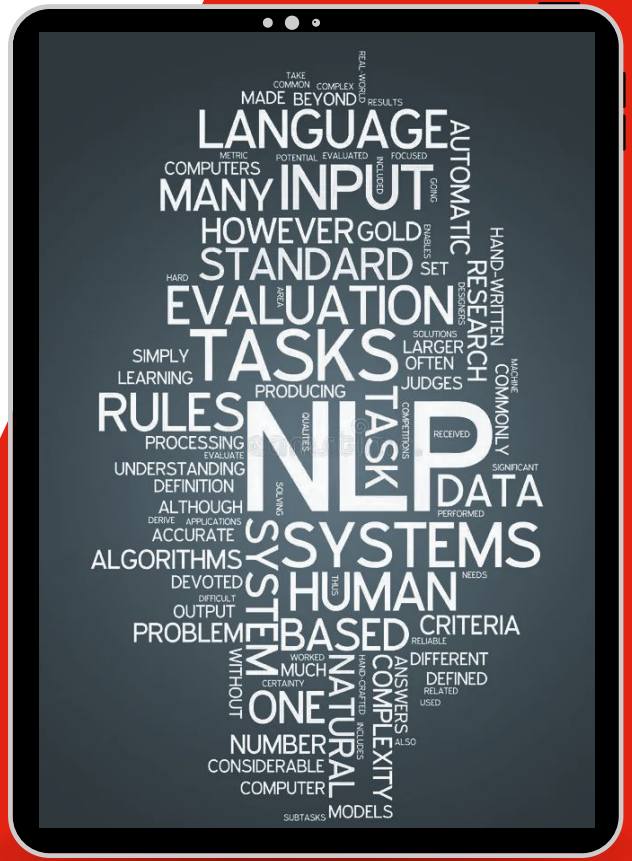
Transformer and BERT

Muhajir Akbar Hasibuan

Class Meeting 10

2023



Muhajir Akbar Hasibuan

Data Scientist

muhajirakbarhsb@gmail.com
+62 85297023234
Jakarta, Indonesia
linkedin.com/in/muhajirakbarhsb
github.com/muhajirakbarhsb
medium.com/@muhajirakbarhsb

Experience

Telkom Indonesia *Sept 2020 - present*

- Develop internal and external use cases
- Provide data understanding in making a model
- Provide Preparation and data engineering according to the use case implemented
- Provide data validation so that the analysis results are as expected
- Building modeling for the development or improvement of internal and external programs
- Provide descriptive and diagnostic insight into data processing
- Recommend and define new growth hacking strategy for digital marketing team

Project

- Pioneered a Robust Big Data Solution for MyIndihome, Revolutionizing Customer Experiences
- Orchestrated a High-Performing ML Team, Elevating myIndihome TV's Personalized Content Impact by 25%
- Envisioned and Executed a Cutting-Edge Big Data Solution for Elevated Customer Engagement on myIndihomeTV
- Fueled Business Insights via Dynamic Data Profiling, Performance Dashboards, and Insights for Langit Musik, RBT, and Upoint
- Engineered Innovative Big Data Solutions that Propelled Growth for PadiUmkm (E-Commerce)
- Powered Success for GameQoo through Strategic Big Data Solutions
- Architected and Established MLOps Framework, Elevating Telkom Indonesia's Digital Business Products
- Crafted Visionary Video Analytics Solutions for Telkom Indonesia's Revolutionary Digital Business IoT Product

MEMBER OF DATA SCIENTIST TASK FORCE | NOVEMBER 2021 - PRESENT
MEMBER OF AI TASK FORCE | NOVEMBER 2022- PRESENT

A Pool of data scientists and AI Engineer Expert in Telkom Indonesia. It was established to leverage data-driven culture for decision-making within the organization. (Applied Research, Standardization, Consultation)

Achievement

BEST Talent of the Year at Telkom Indonesia, Digital Business and Technology Division - Digital Technology and Platform -2022

Codex by Telkom Indonesia *Apr 2020- Sept 2020*

- Building Data Pipeline for Langit Musik Recommender System
- Business Analytic for Langit Musik
- Build Recommender System for Langit Musik

Universitas Syiah Kuala *2015-2020*

- Bachelor of Science, Statistics

Skills

Hard Skills

- Data Analytics
- Statistics
- Machine Learning
- Deep Learning
- MlOps
- Business Intelligence
- Data Engineering
- Cloud Computing
- Time Series & Demand Forecasting
- Natural Language Processing
- Fraud & Anomaly Detection
- Computer Vision

Tools Skills

- Python
- Pyspark
- Sql
- Apache Airflow
- GCP
- Docker
- MLflow
- Prometheus
- Evidently
- Grafana
- Redash, Metabase, Superset, Looker Studio
- Pytorch

Class Meeting 1: Introduction to NLP (1 session)
Introduction to NLP
Basics of text data and its characteristics
The importance of NLP in today's world
Overview of the course structure and objectives

Class Meeting 2: Text Preprocessing (1 session)
Understanding the text preprocessing pipeline
Tokenization, stemming, and lemmatization
Stop words removal
Hands-on exercises with Python for text preprocessing

Class Meeting 3: Text Representation (1 session)
Introduction to text representation techniques
Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF)
n-gram
Word embeddings (Word2Vec, GloVe)**just an introduction
Practical exercises on text representation

Class Meeting 4: Introduction to ML, DL Supervised and Unsupervised (1 session)
Introduction to deep learning (ML)
Introduction to deep learning (DL)
DL vs. traditional machine learning (ML)
Machine Learning and their applications
Neural networks and their applications
Basic ML DL concepts and terminology

Class Meeting 5: Sentiment Analysis (1 session)
What is sentiment analysis?
Data collection and labeling for sentiment analysis
Building a sentiment analysis model
Practical sentiment analysis examples

Class Meeting 6: Text Classification (1 session)
Introduction to text classification
Binary and multi-class classification
Building a text classification model
Real-world text classification examples

Class Meeting 6 Introduction to Topic Modeling(1 session)
Introduction to topic Modelingg
Topic Modeling Implementation
LDA(Latent dirichlet allocation)
Hands-on exercises with Topic Modeling

mid term

Class Meeting 8: **Sesi Khusus deep Learning**

Class Meeting 9: Introduction to Word Embeddings (1 session)
Fundamentals of word embeddings
Word2Vec, GloVe, and FastText
Word embedding applications
Hands-on exercises with word embedding

Class Meeting 10: Attention Mechanisms (1 session)
Introduction to attention mechanisms
Self-attention and multi-head attention
Transformers architecture
Practical examples of attention in NLP

Class Meeting 11: Transformer Models (1 session)
In-depth study of the Transformer model
Pre-trained transformer models (BERT, GPT-2)
Fine-tuning transformers for NLP tasks
Transformer-based applications

Class Meeting 12: Advanced NLP Topics (1 session)
Advanced NLP topics such as BERT, XLNet, and RoBERTa
Transfer learning in NLP
Ethics in NLP
Recent developments and trends in NLP

Class Meeting 13: Advanced NLP Topics and Deployment Process in Industry (1 session)
Introduction to LLM
Introduction how industry utilize NLP to generate revenue
Introduction MLOps for NLP(bonuses from practitioners)

Class Meeting 14-15: Project Work (2 sessions)
Dedicated sessions for students to work on NLP projects with guidance and assistance.

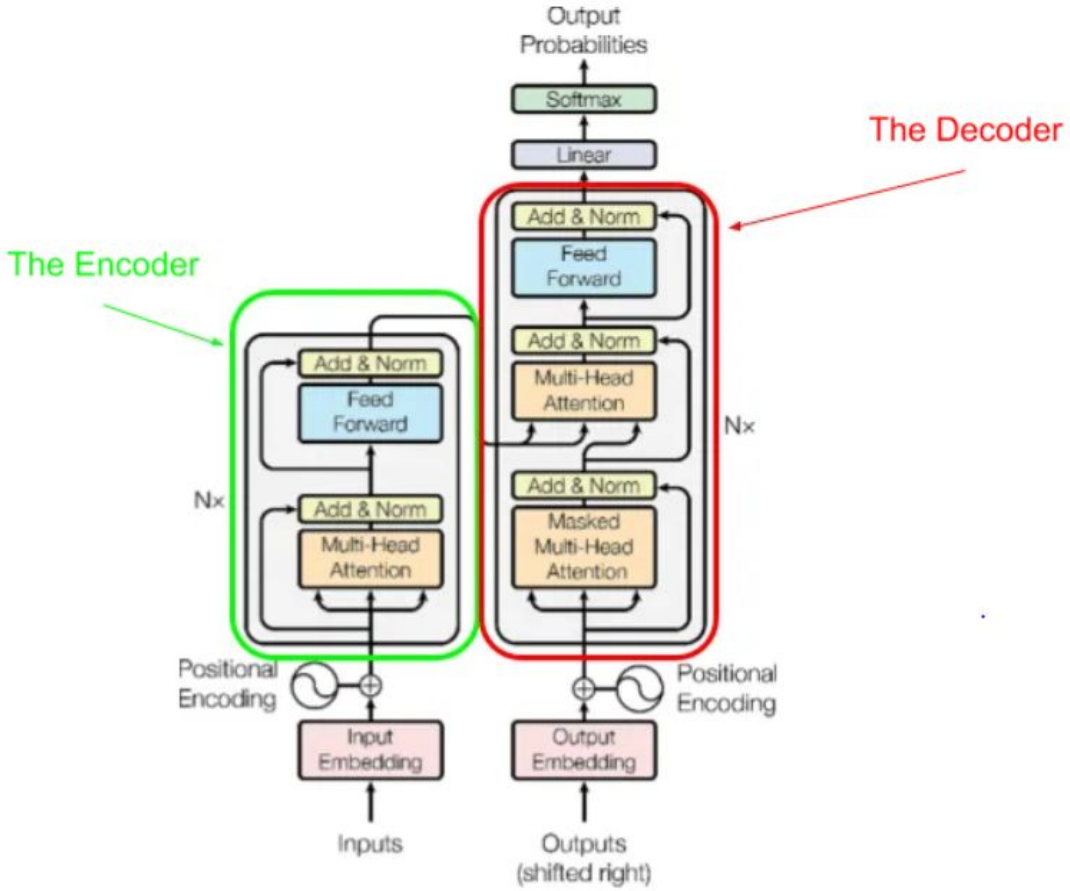
Class Meeting 16: Project Presentations and Conclusion (1 session)
Students present their NLP projects
Recap of key takeaways from the course
Discuss further resources for NLP enthusiasts
Course conclusion and feedback

Agenda

- Introduction to Transformer
- BERT

Transformer Architecture

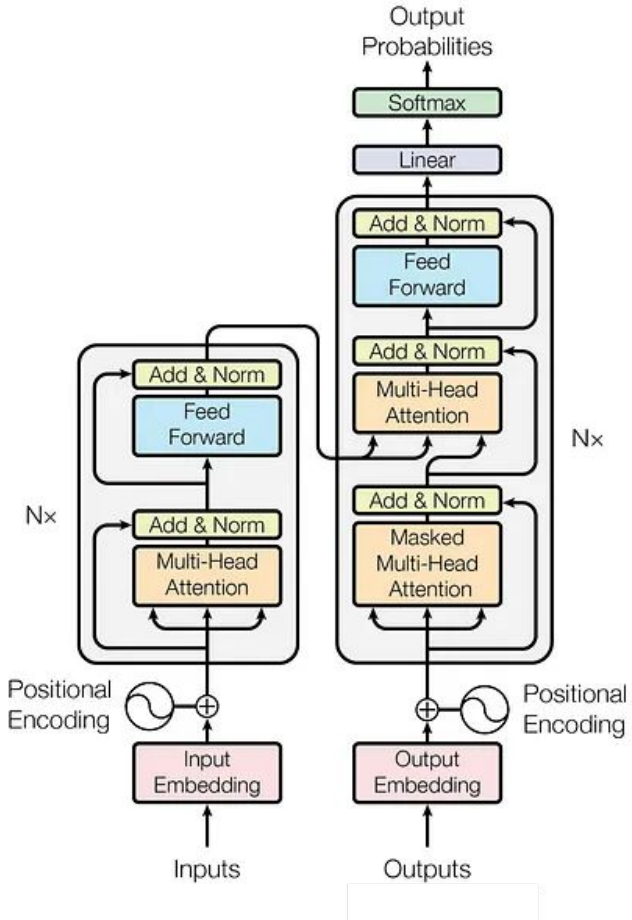
Transformer Architecture



Transformer Architecture

BERT

Encoder



GPT

Decoder

Transformer

Transformers are a type of deep learning architecture that have **revolutionized the field of natural language processing (NLP) in recent years**. They are widely used for tasks such as language translation, text classification, sentiment analysis, and more

The transformer architecture is composed of an encoder and a decoder, each of which is made up of multiple layers of self-attention and feedforward neural networks. **The self-attention mechanism is the heart of the transformer, allowing the model to weigh the importance of different words in a sentence based on their affinity with each other**. This is similar to how a **human might read a sentence, focusing on the most relevant parts of the text rather than reading it linearly from beginning to end**.

Transformer(Encoder - Decoder)

The transformer encoder-decoder architecture is used for tasks like language translation, **where the model must take in a sentence in one language and output a sentence in another language**. **The encoder takes in the input sentence and produces a fixed-size vector representation of it**, which is then fed into the decoder to generate the output sentence. The decoder uses both self-attention and cross-attention, **where the attention mechanism is applied to the output of the encoder and the input of the decoder**.

One of the most popular transformer encoder-decoder models is the T5 (Text-to-Text Transfer Transformer), which was introduced by Google in 2019. The T5 can be fine-tuned for a wide range of NLP tasks, including language translation, question answering, summarization, and more.

Transformer(Encoder - Decoder)

The transformer encoder-decoder architecture is used for tasks like language translation, **where the model must take in a sentence in one language and output a sentence in another language**. **The encoder takes in the input sentence and produces a fixed-size vector representation of it**, which is then fed into the decoder to generate the output sentence. The decoder uses both self-attention and cross-attention, **where the attention mechanism is applied to the output of the encoder and the input of the decoder**.

One of the most popular transformer encoder-decoder models is the T5 (Text-to-Text Transfer Transformer), which was introduced by Google in 2019. The T5 can be fine-tuned for a wide range of NLP tasks, including language translation, question answering, summarization, and more.

Transformer(Encoder)

The transformer encoder architecture is used for tasks like text classification, where the model must classify a piece of text into one of several predefined categories, such as sentiment analysis, topic classification, or spam detection. **The encoder takes in a sequence of tokens and produces a fixed-size vector representation of the entire sequence, which can then be used for classification**.

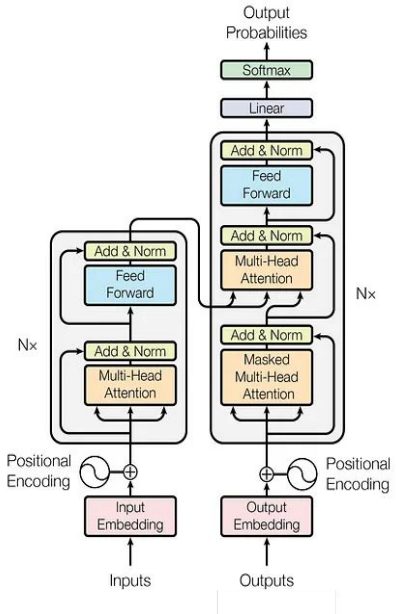
One of the most popular transformer encoder models is **BERT (Bidirectional Encoder Representations from Transformers), which was introduced by Google in 2018. BERT is pre-trained on large amounts of text data and can be fine-tuned for a wide range of NLP tasks**.

Transformer(Encoder)

The transformer encoder architecture is used for tasks like text classification, where the model must classify a piece of text into one of several predefined categories, such as sentiment analysis, topic classification, or spam detection. **The encoder takes in a sequence of tokens and produces a fixed-size vector representation of the entire sequence, which can then be used for classification**.

BERT

Encoder

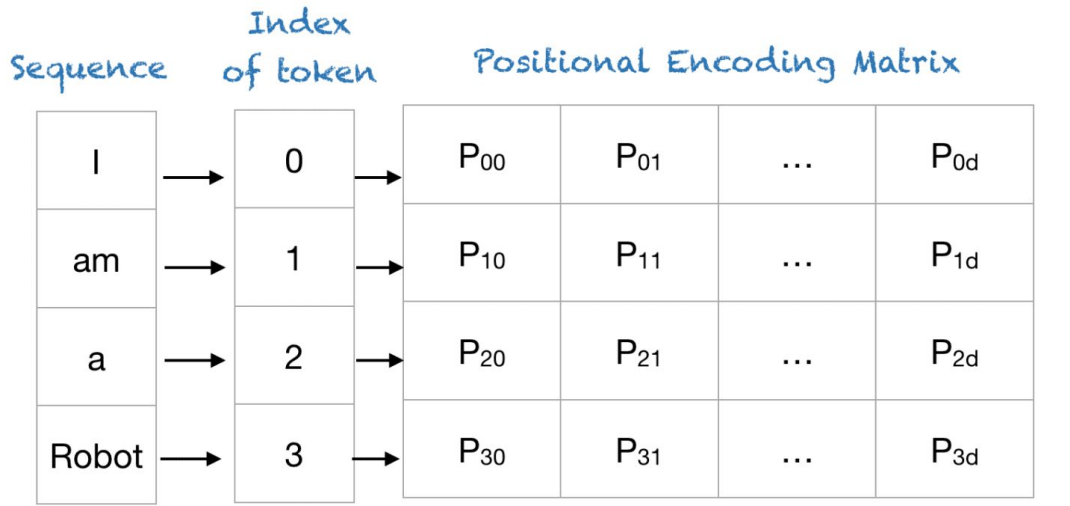


GPT

Decoder

Positional Encoding

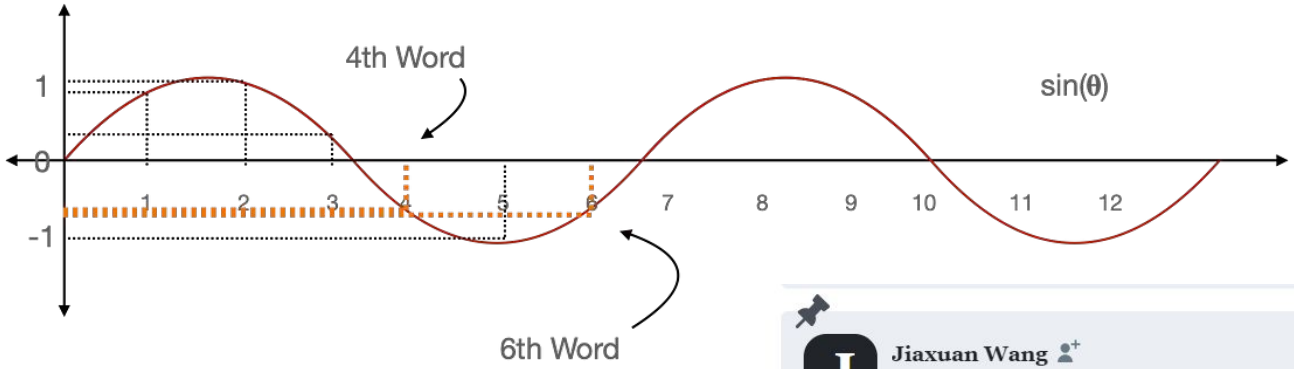
Positional encoding describes the location or position of an entity in a **sequence so that each position is assigned a unique representation**. There are many reasons why a single number, such as the index value, is not used to represent an item's position in transformer models. For long sequences, the indices can grow large in magnitude. If you normalize the index value to lie between 0 and 1, it can create problems for variable length sequences as they would be normalized differently.

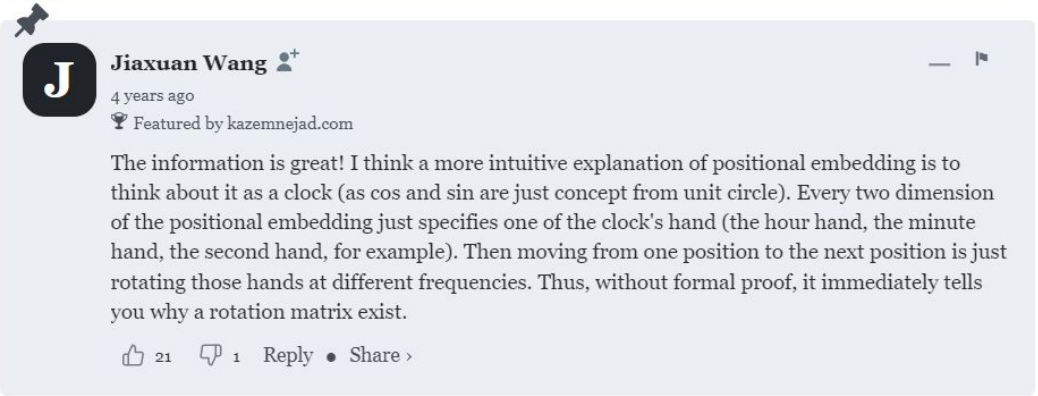


Positional Encoding Matrix for the sequence 'I am a robot'

Sinusoidal Trigonometric Sine Function

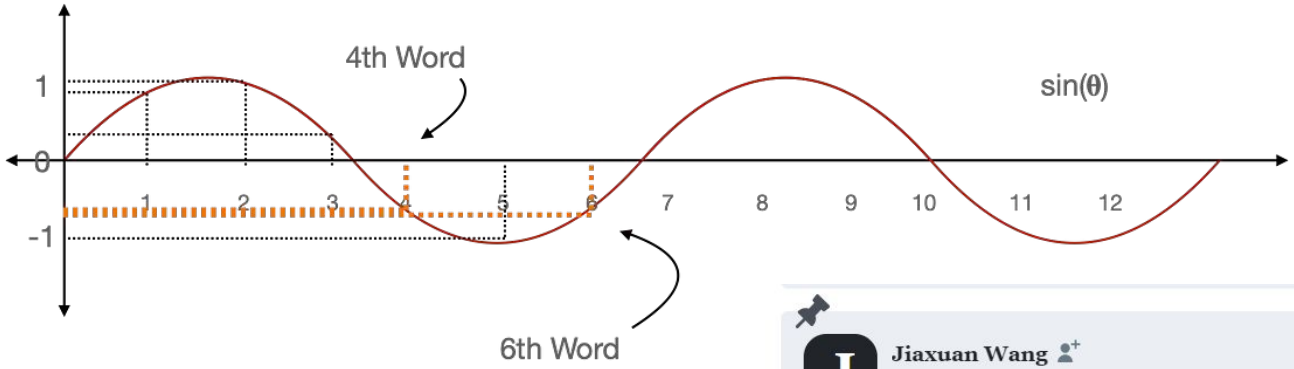
This is a quick recap of sine functions; you can work equivalently with cosine functions. The function's range is [-1,+1]. The frequency of this waveform is the number of cycles completed in one second. The wavelength is the distance over which the waveform repeats itself. The wavelength and frequency for different waveforms are shown below:

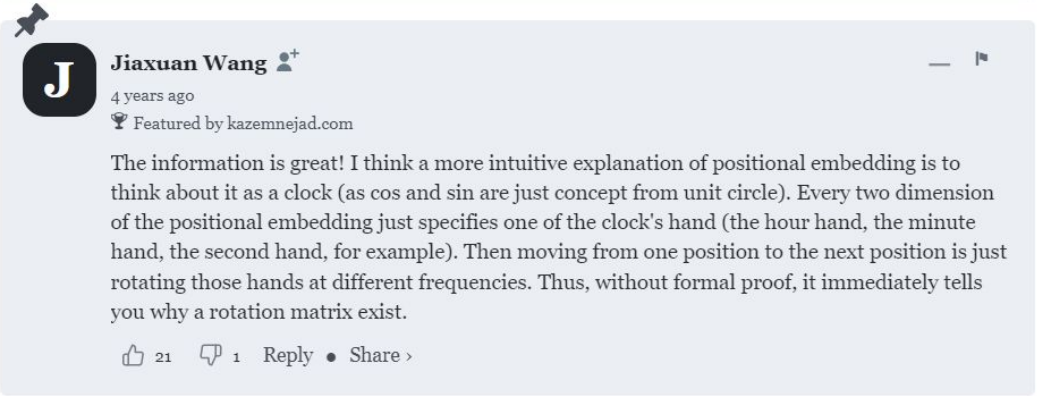




Sinusoidal Trigonometric Sine Function

This is a quick recap of sine functions; you can work equivalently with cosine functions. The function's range is [-1,+1]. The frequency of this waveform is the number of cycles completed in one second. The wavelength is the distance over which the waveform repeats itself. The wavelength and frequency for different waveforms are shown below:





Positional Encoding

For example, in two different sentences:

"I am learning Python." (referring to the programming language)
"The Python is a large snake." (referring to the snake)

The word "**Python**" **will have the same base word vector in both sentences**. However, the positional encodings for this word will be different in each sentence due to its different positions and surrounding context. W**hen these positional encodings are added to the base word vector**, the resulting vectors for **"Python" will be different in each sentence, allowing the Transformer model to distinguish between the programming language and the snake**.

BERT

BERT

At the end of 2018 researchers at Google AI Language open-sourced a new technique for Natural Language Processing (NLP) called BERT (Bidirectional Encoder Representations from Transformers) — a major breakthrough which took the Deep Learning community by storm because of its incredible performance.

One of the biggest challenges in NLP is the **lack of enough training data**. Overall there is enormous amount of text data available, but if we want to create task-specific datasets, we need to split that pile into the very many diverse fields. And when we do this, **we end up with only a few thousand or a few hundred thousand human-labeled training examples**.

To help bridge this gap in data, researchers have developed various techniques for training general purpose language representation models using the enormous piles of unannotated text on the web (this is known as pre-training). **These general purpose pre-trained models can then be fine-tuned on smaller task-specific dataset**

BERT

What is language modeling really about? Which problem are language models trying to solve? Basically, their task is to “fill in the blank” based on context. For example, given

“The woman went to the store and bought a _____ of shoes.”

a language model might complete this sentence by saying that the word “cart” would fill the blank 20% of the time and the word “pair” 80% of the time.

Instead of predicting the next word in a sequence, BERT makes use of a novel technique called Masked LM (MLM): i**t randomly masks words in the sentence and then it tries to predict them**. **Masking means that the model looks in both directions and it uses the full context of the sentence**, both left and right surroundings, in order to predict the masked word. **Unlike the previous language models, it takes both the previous and next tokens into account at the same time. The existing combined left-to-right and right-to-left LSTM based models were missing this “same-time part”. (It might be more accurate to say that BERT is non-directional though.)**

BERT

What is language modeling really about? Which problem are language models trying to solve? Basically, their task is to “fill in the blank” based on context. For example, given

“The woman went to the store and bought a _____ of shoes.”

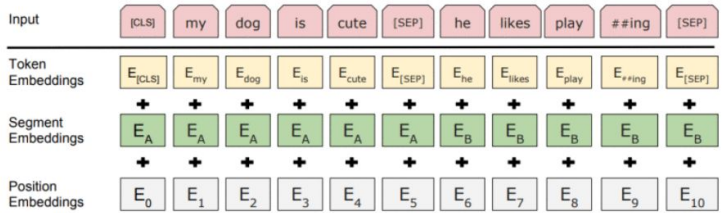
a language model might complete this sentence by saying that the word “cart” would fill the blank 20% of the time and the word “pair” 80% of the time.

Instead of predicting the next word in a sequence, BERT makes use of a novel technique called Masked LM (MLM): i**t randomly masks words in the sentence and then it tries to predict them**. **Masking means that the model looks in both directions and it uses the full context of the sentence**, both left and right surroundings, in order to predict the masked word. **Unlike the previous language models, it takes both the previous and next tokens into account at the same time. The existing combined left-to-right and right-to-left LSTM based models were missing this “same-time part”. (It might be more accurate to say that BERT is non-directional though.)**

BERT

BERT relies on a Transformer (the attention mechanism that learns contextual relationships between words in a text). A basic Transformer consists of an encoder to read the text input and a decoder to produce a prediction for the task. S**ince BERT's goal is to generate a language representation model, it only needs the encoder part**. The input to the encoder for BERT is a sequence of tokens, which are first converted into vectors and then processed in the neural network.

1. **Token embeddings:** A [CLS] token is added to the input word tokens at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence.
2. **Segment embeddings:** A marker indicating Sentence A or Sentence B is added to each token. This allows the encoder to distinguish between sentences.
3. **Positional embeddings:** A positional embedding is added to each token to indicate its position in the sentence.



BERT

1. Masked LM (MLM)

The idea here is "simple": Randomly mask out 15% of the words in the input — replacing them with a [MASK] token — run the entire sequence through the BERT attention based encoder and then predict only the masked words, based on the context provided by the other non-masked words in the sequence. However, there is a problem with this naive masking approach — the model only tries to predict when the [MASK] token is present in the input, while we want the model to try to predict the correct tokens regardless of what token is present in the input. To deal with this issue, out of the 15% of the tokens selected for masking:

- 80% of the tokens are actually replaced with the token [MASK].
- 10% of the time tokens are replaced with a random token.
- 10% of the time tokens are left unchanged.

While training the BERT loss function considers only the prediction of the masked tokens and ignores the prediction of the non-masked ones. This results in a model that converges much more slowly than left-to-right or right-to-left models.

2. Next Sentence Prediction (NSP)

In order to understand *relationship* between two sentences, BERT training process also uses next sentence prediction. A pre-trained model with this kind of understanding is relevant for tasks like question answering. During training the model gets as input pairs of sentences and it learns to predict if the second sentence is the next sentence in the original text as well.

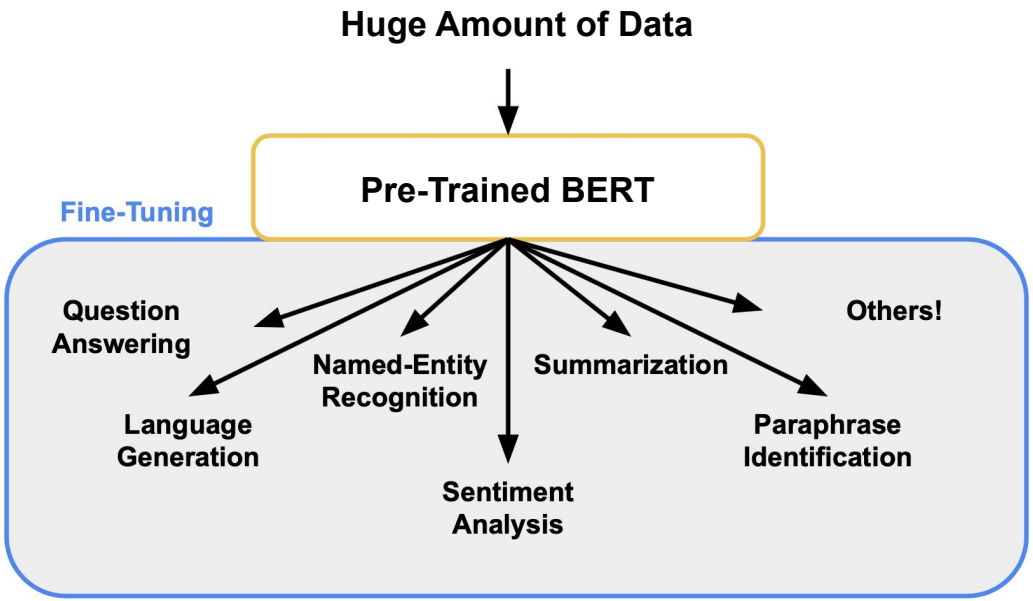
As we have seen earlier, BERT separates sentences with a special [SEP] token. During training the model is fed with two input sentences at a time such that:

- 50% of the time the second sentence comes after the first one.
- 50% of the time it is a a random sentence from the full corpus.

BERT

There are two steps in BERT: p**re-training and fine-tuning**. D**uring pre-training, the model is trained on unlabeled data over different pre-training tasks**. **For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks**. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters.

BERT Pretrained Models



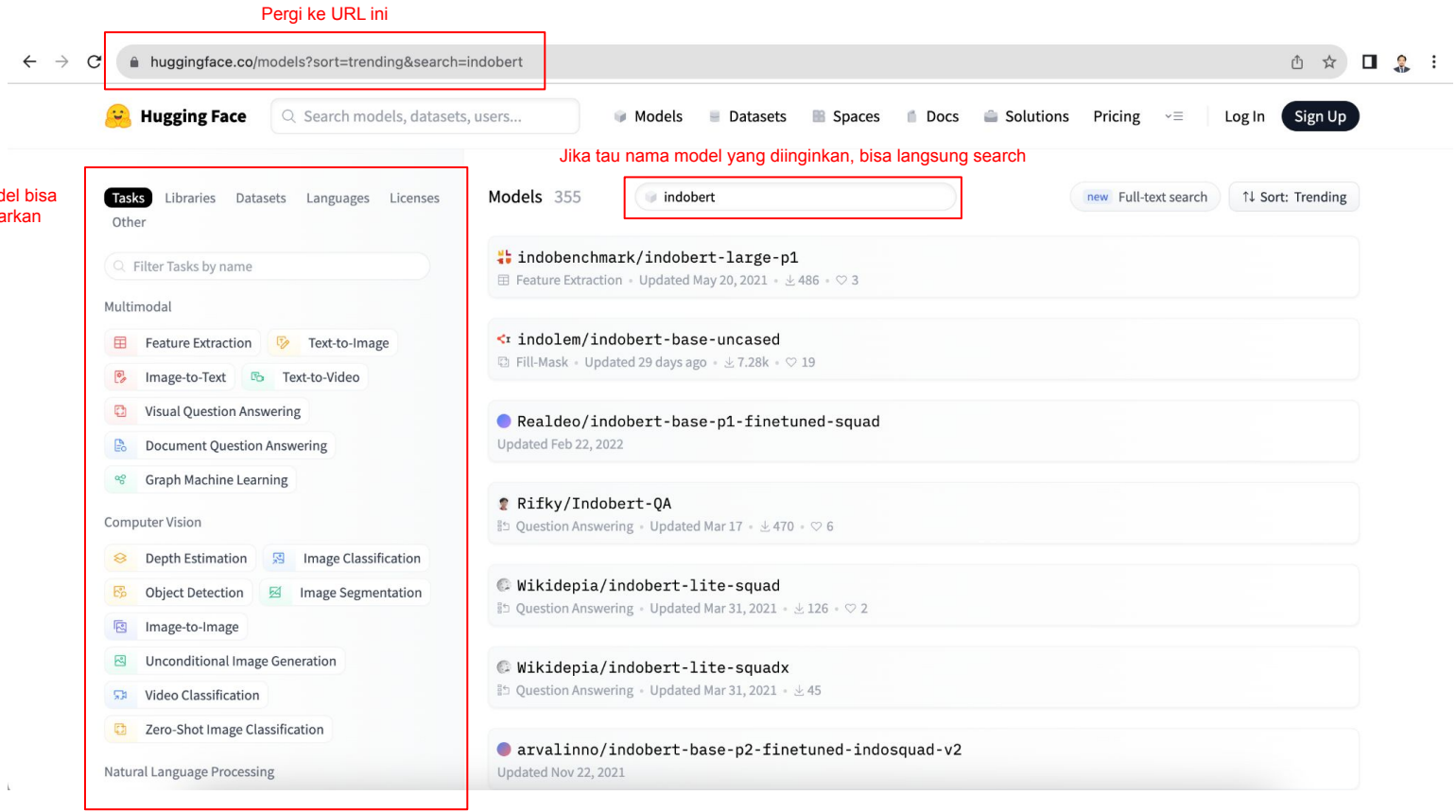
Why do We Need Pretrained BERT?

- Butuh data yang sangat banyak
- Butuh computational resource yang besar untuk dilatih

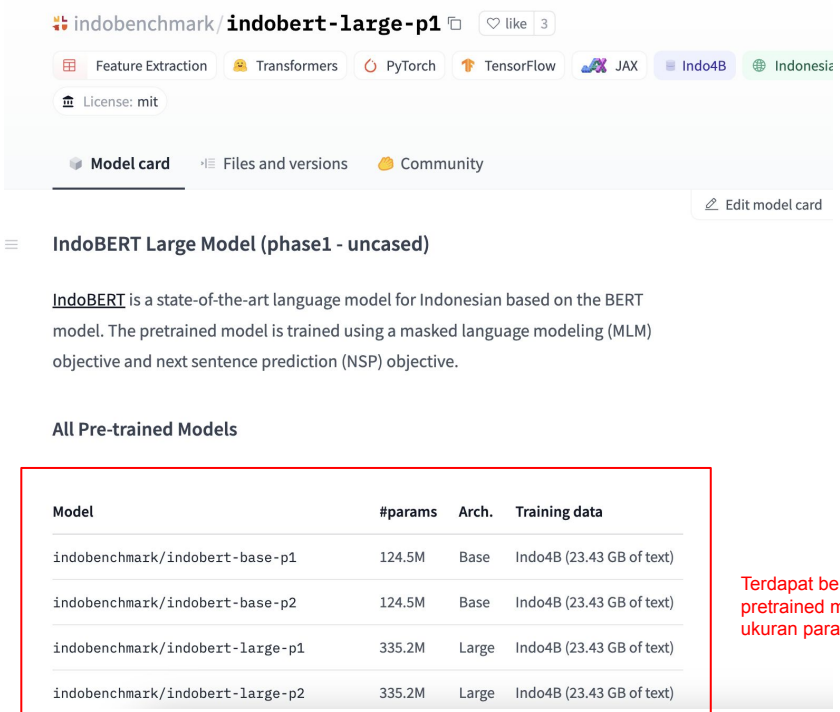
How do We Utilize Pretrained BERT?

- Pretrained BERT biasanya dilatih untuk task general seperti language modeling
- Pretrained model dilatih di data yang sangat banyak sehingga memiliki tingkat akurasi yang baik
- Pretrained model ini nantinya bisa kita gunakan dengan melakukan ***fine tune*** sesuai dengan task yang kita inginkan (Question Answering, Sentiment Analysis, dsb.)

Where to Find BERT Pretrained?



Where to Find BERT Pretrained?



Terdapat beberapa pilihan pretrained model berdasarkan ukuran parameter

Data Serving - Predict from HF API

```python
import pandas as pd
import numpy as np
import re
from transformers import pipeline
import torch

def predict_sentiment(text):
    result = nlp(text)
    return result[0]['label']



# Load pre-trained model and tokenizer
pretrained_name = "w11wo/indonesian-roberta-base-sentiment-classifier"
nlp = pipeline("sentiment-analysis", model=pretrained_name, tokenizer=pretrained_name)


# Apply sentiment prediction to dataframe
df['predicted_sentiment'] = df['description'].apply(predict_sentiment)

# Display updated dataframe
print(df)
```

Python Time

https://github.com/muhajirakbarhsb/NLP_class_2023/blob/main/transformer_embeddings.ipynb

https://github.com/muhajirakbarhsb/NLP_class_2023/blob/main/Week_Class_10.ipynb