# Transformers

**Akshitha Kumbam**

Acknowledgement - Yilun Kuang

10/16/2023

# Outline

- Encoder-Only / Decoder-Only
- Huggingface Transformer - Provides pretrained models, tools and APIs for using transformer based architectures.

# Encoder-Only | BERT

**Bidirectional Encoder Representations from Transformers (BERT) - Google.**

Model Architecture

- Transformer Encoder

What is Special About It

- Mask Language Modeling & Next Sentence Prediction
- Downstream task adaptation



Source:
https: //lena-voita.github.io/nlp_course/transfer_learning.html#bert

BERT’s primary goal is to create contextualized word embeddings, meaning it captures the meaning of a word based on its context in the sentence.
BERT’s embeddings are dynamic and depend on the **words surrounding the target word**.

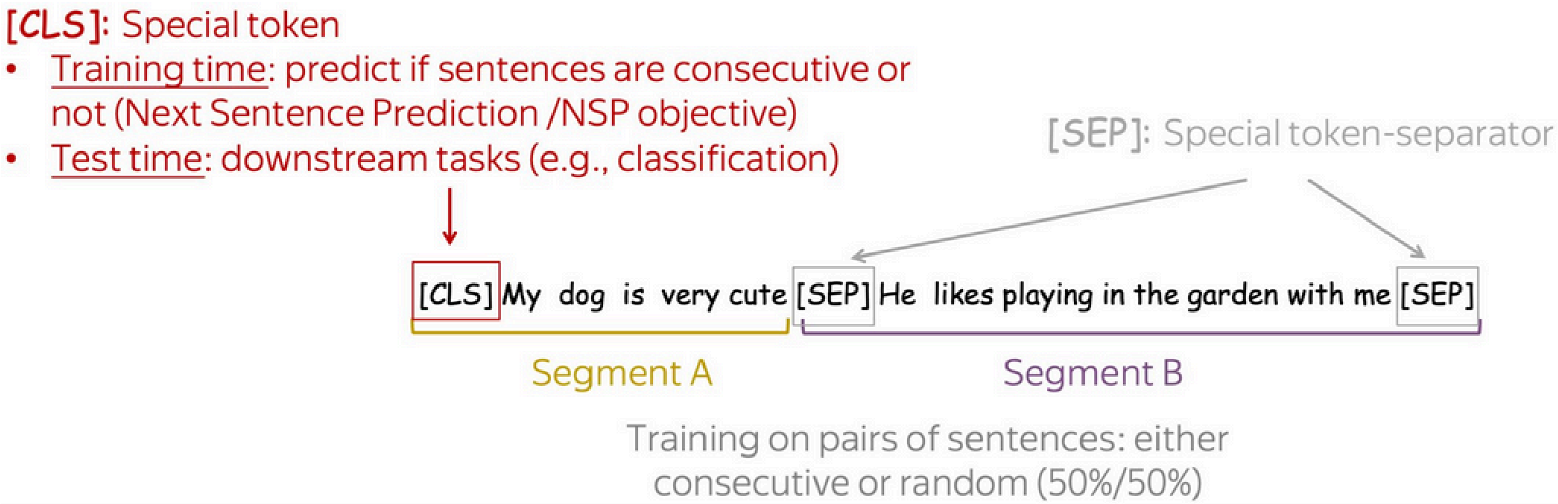
## How BERT Achieves This Goal:

To achieve contextualized embeddings, BERT is trained using two key tasks:

1. Next Sentence Prediction (NSP):
    a. What is NSP? The model is given two sentences, and it must predict whether the second sentence logically follows the first sentence or if it is a random sentence.
    b. Why? This task helps BERT understand relationships between sentences, which is useful for downstream tasks like question answering, text classification, or sentence similarity.
2. Masked Language Model (MLM):
    a. What is MLM? In this task, random words in the input sentence are masked (replaced with [MASK]), and the model is trained to predict those masked words based on the context provided by the other words in the sentence.
    b. Why? By doing this, BERT learns how words relate to each other in context. It helps the model understand how different words fit together, which is essential for producing rich, context-aware word representations.

# BERT - Pretrain | Inputs

**Training Input:** 1) pairs of sentence; 2) [CLS] token; 3) [SEP] token





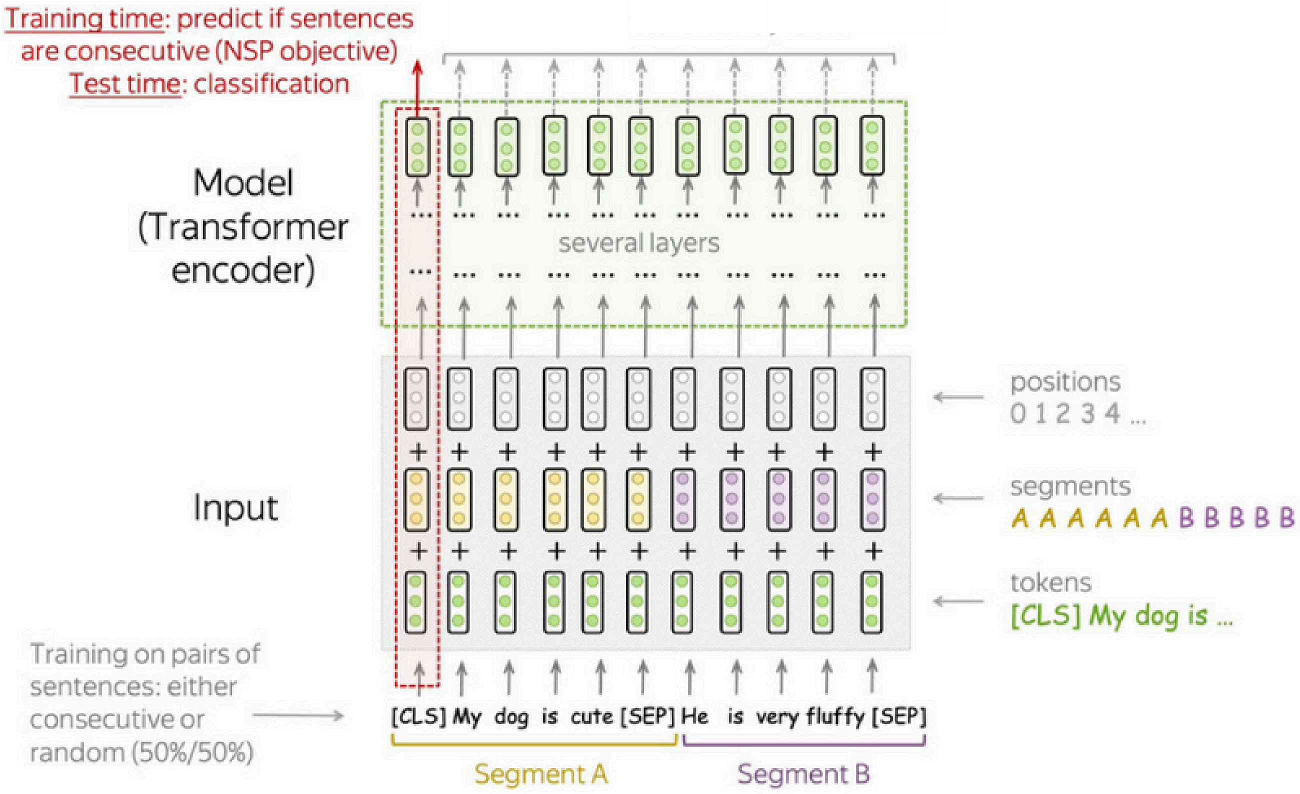

Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

## Embedding

1) Token

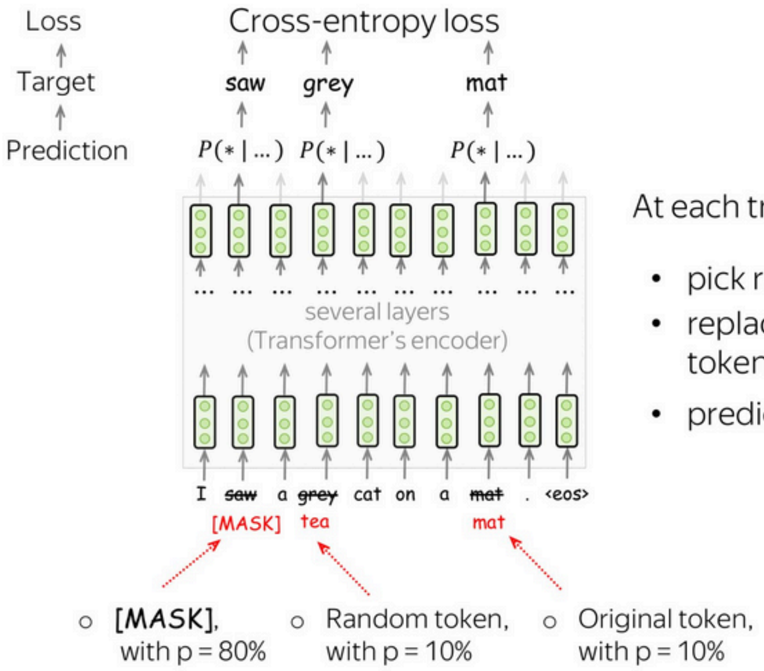
2) Segment

3) Position




NYU

# BERT - Pretrain | Inputs




At each training step:

- pick randomly 15% of tokens
- replace each of the chosen tokens with something
- predict original chosen tokens

NYU



Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

# BERT - Pretrain | Objective

**Next Sentence Prediction (NSP)**

**Input:** [CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]
**Label:** isNext

**Input:** [CLS] the man went to [MASK] store [SEP] penguin [MASK] are flight ##less birds [SEP]

**Label:** notNext

NYU

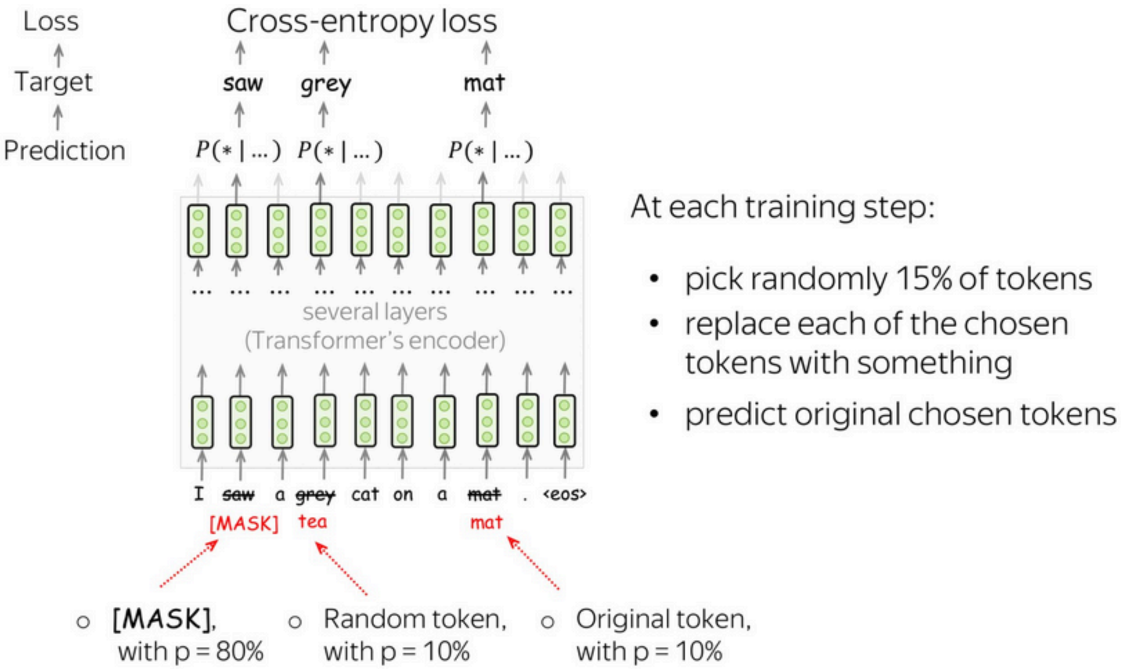


Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

# BERT - Pretrain | Objective

**Masked Language Modeling (MLM)**

Output layer of MLM - Output Linear Layer: Generates logits for vocabulary.




NYU



Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

**During training:**

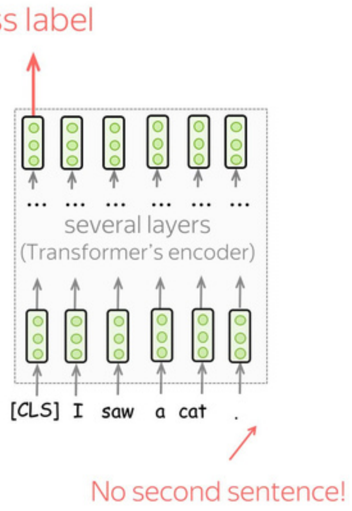
Masked tokens: Predicted tokens are compared to original tokens.
Loss calculation: Cross-entropy loss calculated for each masked token.
Backpropagation: Gradients computed to update model weights.

**Benefits:**

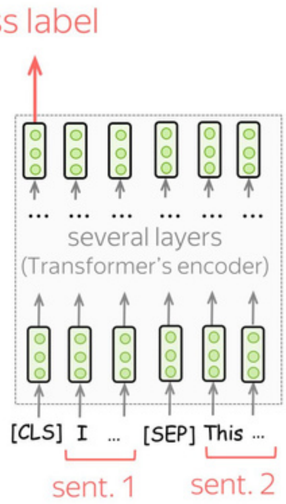
1. Contextual understanding: BERT learns to represent tokens in context.
2. Improved language modeling: Better handling of out-of-vocabulary tokens.
3. Transfer learning: Fine-tuning benefits from pre-trained MLM.

# BERT - Finetune | Tasks

### Single sentence classification




### Sentence Pair Classification

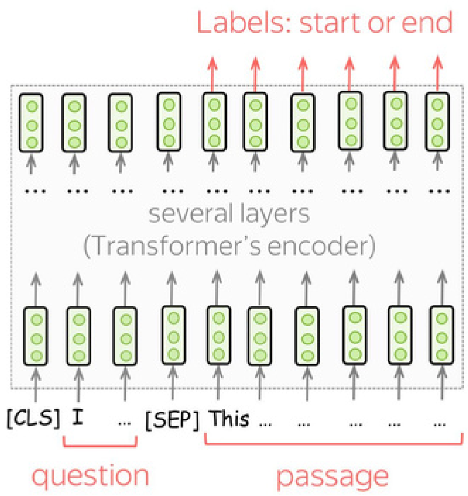




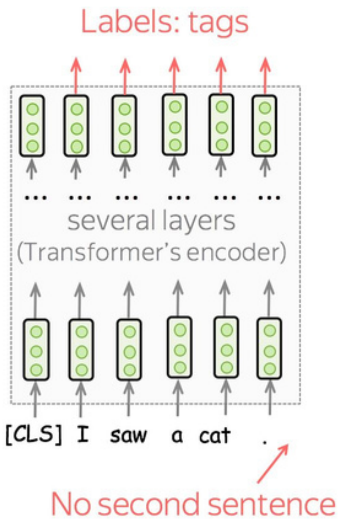

Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

# BERT - Finetune | Tasks

## Question Answering




## Single sentence tagging




NYU



Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#bert

# Decoder-Only | GPT

**Generative Pretrained Transformer (GPT) - OpenAI**

Model Architecture

- Transformer **Decoder**

What is Special About It

1. Autoregressive Language Modeling - Predict next token given previous tokens.
2. Downstream task adaptation.



Source: https://lena-voita.github.io/nlp_course/transfer_learning.html#gpt

Mathematically:
$P(token_n \mid token_1, token_2, \ldots, token_{n-1})$

Key features:

- **Unsupervised Learning**: Trained on large corpus without labeled data.
- **Generative Capabilities**: Generates coherent text.
- **Transfer Learning**: Fine-tunes for downstream tasks.

**Downstream Task Adaptation**

GPT's adaptation:

- **Language Translation**: Generate translations.
- **Text Summarization**: Summarize long texts.
- **Chatbots**: Engage in conversations.

# GPT - Pretrain | Inputs

**Training Input:** 1) sentences; 2) [PAD] / [EOS] token

Sentences: Sequential text data.
[PAD] token: Padding token for fixed-length sequences.
[EOS] token: End-of-sequence token.

**Example:**

My dog is very cute. He likes playing in the garden with me [EOS] [PAD] ... [PAD]

**Embedding:**

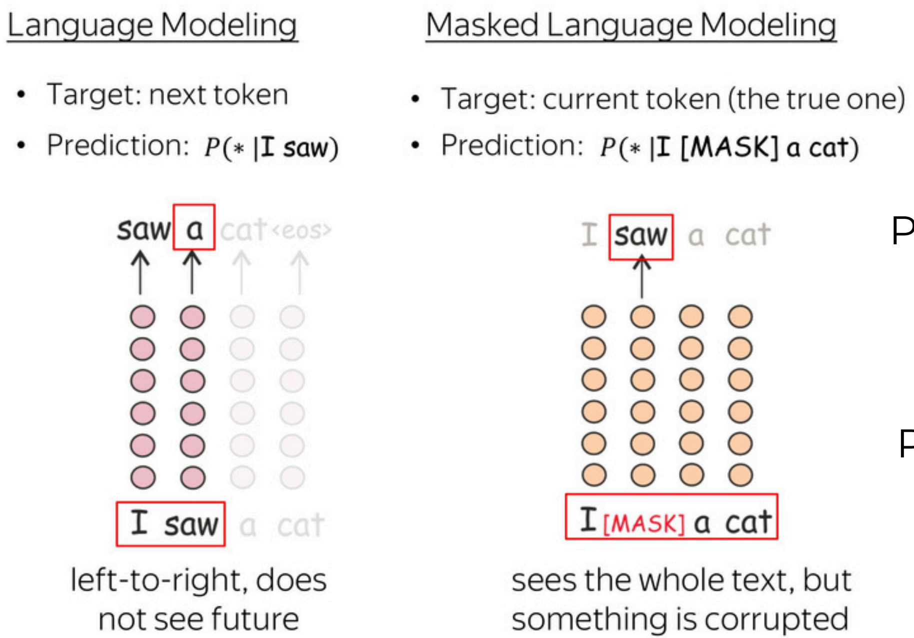
Token embeddings + positional embeddings



Source: https: //lena-voita.github.io/nlp_course/transfer_learning.html#gpt

# GPT - Pretrain | Objective

## Autoregressive Language Modeling




P(token_n | token_1, token_2, ..., token_{n-1})

P(token_i | token_1, ..., token_{i-1}, [MASK], token_{i+1}, ...)

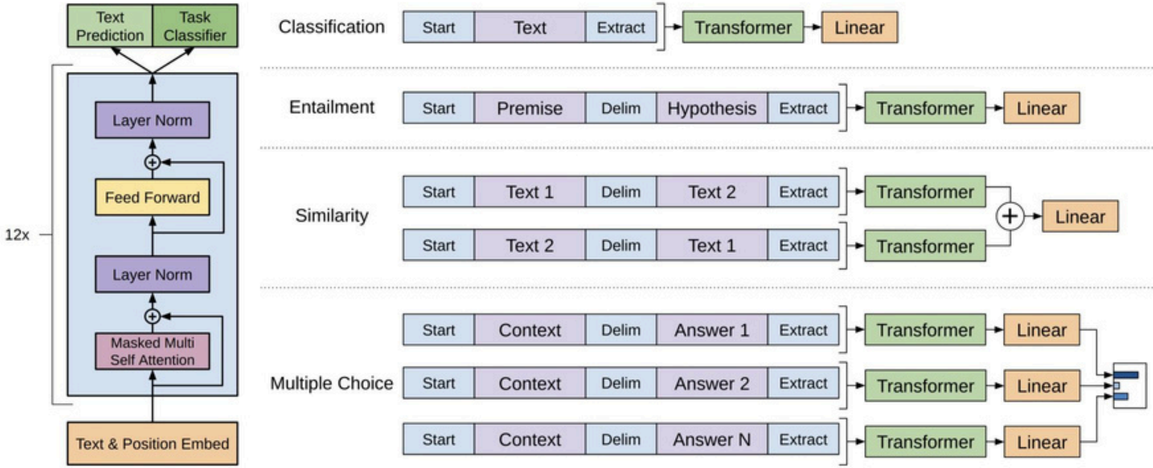


Source: https: //lena-voita.github.io/nlp_course/transfer_learning.html#gpt

# GPT - Finetune | NLU Tasks

## GPT for Natural Language Understanding (NLU) Tasks

- You can train an additional linear head on top of the final transformer block activation vector.




NYU



Source: Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

# Encoder-Decoder | T5

**Text-to-Text Transfer Transformer (T5)** Model Architecture

- Transformer Encoder & Decoder

What is Special About It

- pretraining on multi-task mixture of unsupervised and supervised tasks (converted to Text-to-Text format)

NYU



Source:
https: //arxiv.org/pdf/ 1910.10683.pdf

# Resources

- Encoder-Only (BERT)
  - https://github.com/JonasGeiping/cramming
- Decoder-Only (GPT)
  - https://github.com/karpathy/nanoGPT
- Encoder-Decoder (T5)
  - https://github.com/PiotrNawrot/nanoT5

NYU

# Summary

- Encoder-Only (BERT)
- Decoder-Only (GPT)
- Encoder-Decoder (T5)

# Acknowledgement

This presentation is adapted from Elena (Lena) Voita's NLP Course | For You (https: //lena-voita.github.io/nlp_course.html)