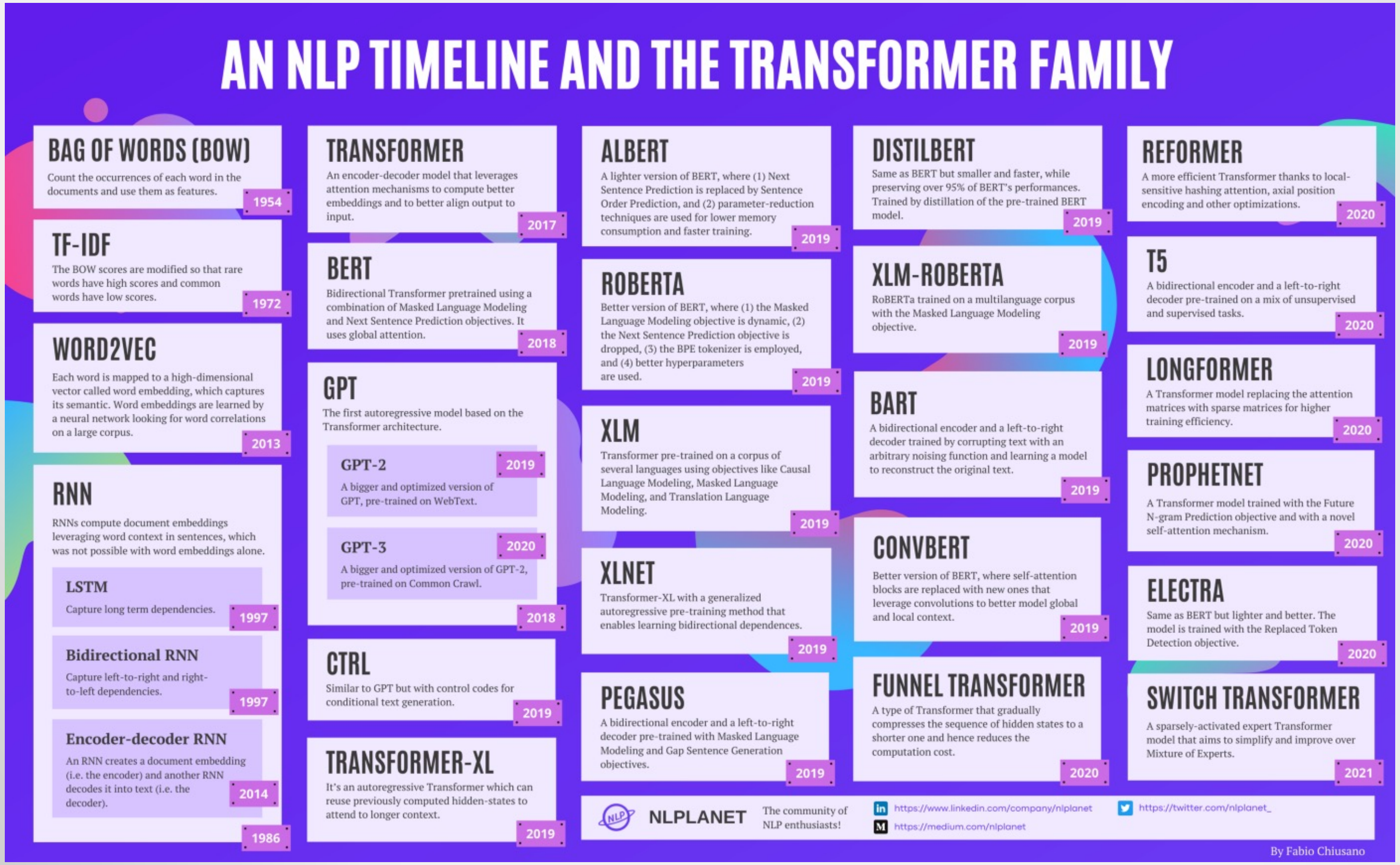
EURO²

Mastering Transformers: From Building Blocks to Real-World Applications

NLP Applications by Prof.Dr. Tuğba Taşkaya Temizel

12 Sept 2023

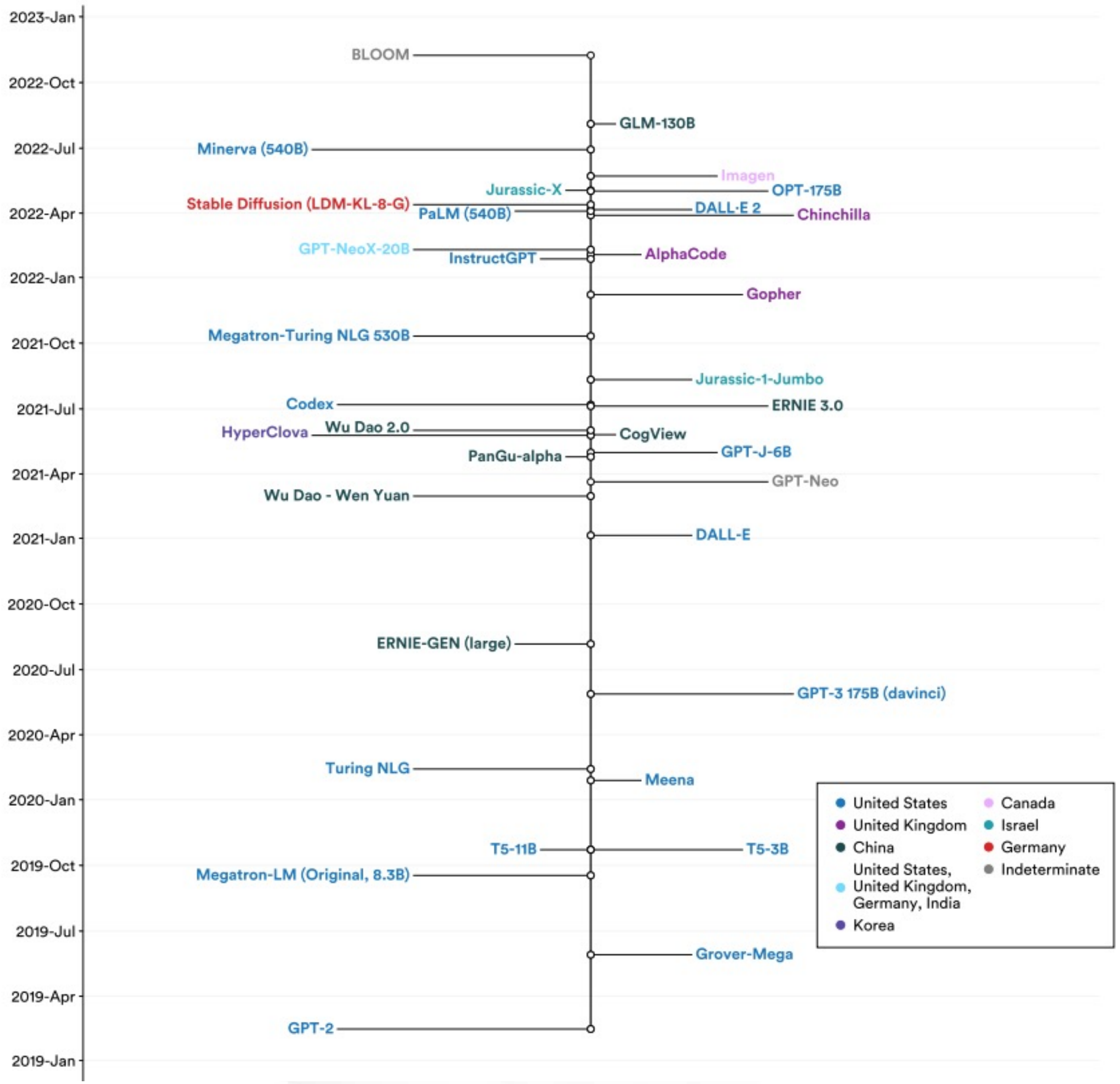
A Brief History in Transformers

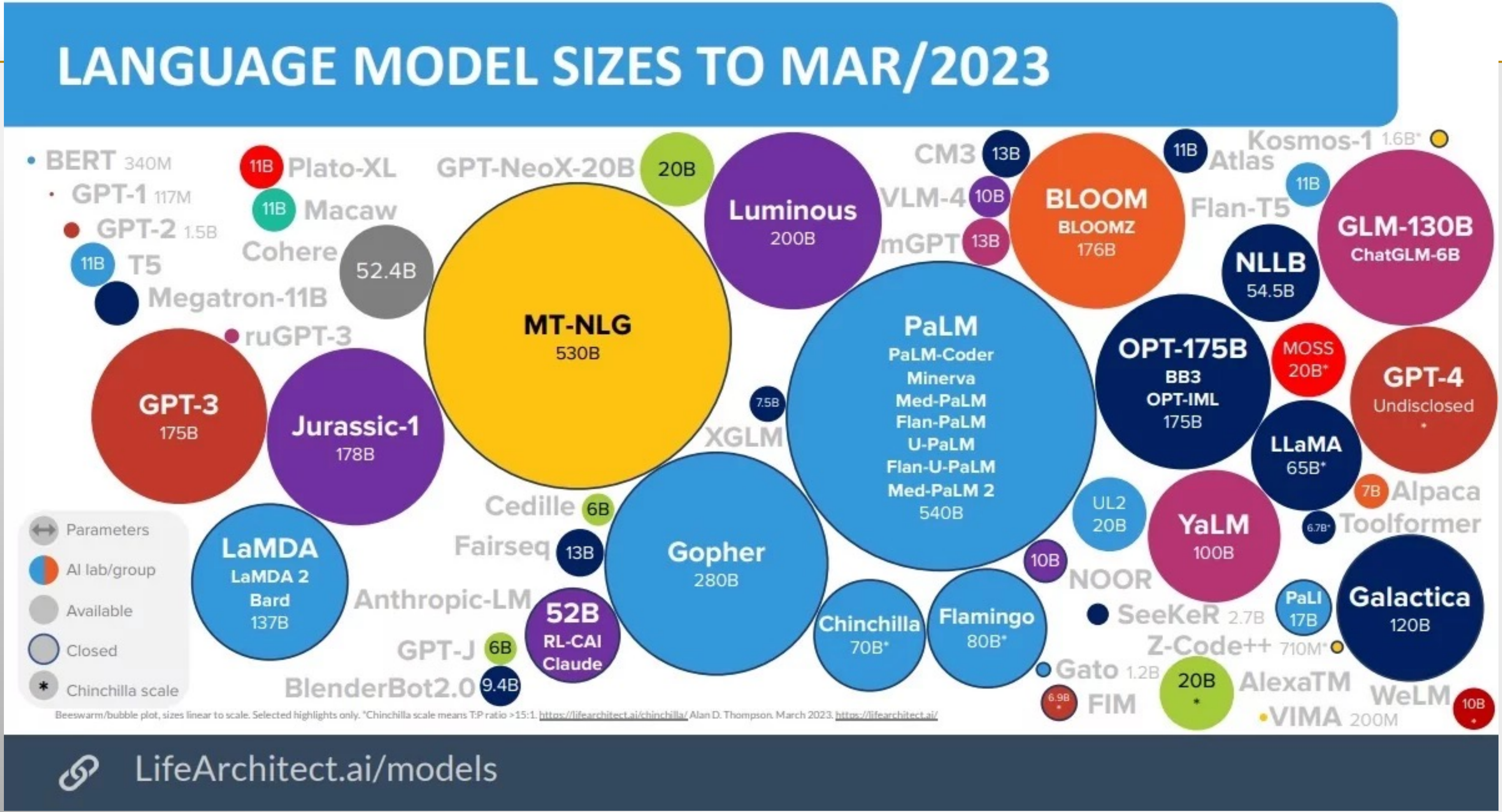


https://medium.com/nlplanet/a-brief-timeline-of-nlp-from-bag-of-words-to-the-transformer-family-7caad8bbba56

Timeline and National Affiliation of Select Large Language and Multimodal Model Releases

Source: AI Index, 2022 | Chart: 2023 AI Index Report





A fast growing area:
LLaMa, LLaMa2
II's Falcon 140B
GPT4
Claude
Falcon
Cohere
...

https://pureinsights.com/blog/2023/what-are-large-language-models-llms-search-and-ai-perspectives/
https://sungkim11.medium.com/list-of-open-sourced-fine-tuned-large-language-models-llm-8d95a2e0dc76
https://datasciencedojo.com/blog/best-large-language-models/

Language Models

- **Language modeling:** predicting upcoming words from prior word context.
- **Neural net-based language models** turn out to have many advantages over the n-gram language models:
  - They can handle much longer histories, and they can generalize over contexts of similar words.
  - For a training set of a given size, a neural language model has much higher predictive accuracy than an n-gram language model.
  - They underlie many of the models for tasks like machine translation, dialog, and language generation.

NLP Applications

- Language models, once created, can be adapted and used for numerous NLP tasks.
- Motivation: Unlabeled text corpora are abundant, but labeled data for learning these specific tasks is scarce.
- Large gains on these tasks can be realized by generative pre-training of a language model on a diverse corpus of unlabeled text, followed by discriminative fine-tuning on each specific task.
- It falls under the category of semi-supervised learning for natural language.
  - Semi-supervised learning combines many unlabeled data with small labeled data for training.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

BERT (Bidirectional Encoder Representations from Transformers)

- BERT’s model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in Vaswani et al. (2017)
  - Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems, pages 6000–6010
- Language model where k% of the input words were masked, and then were predicted

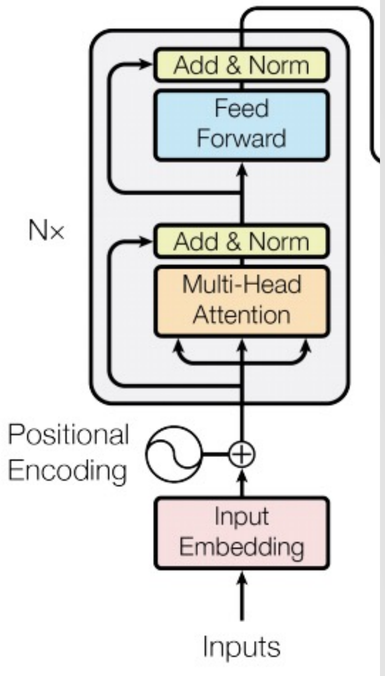
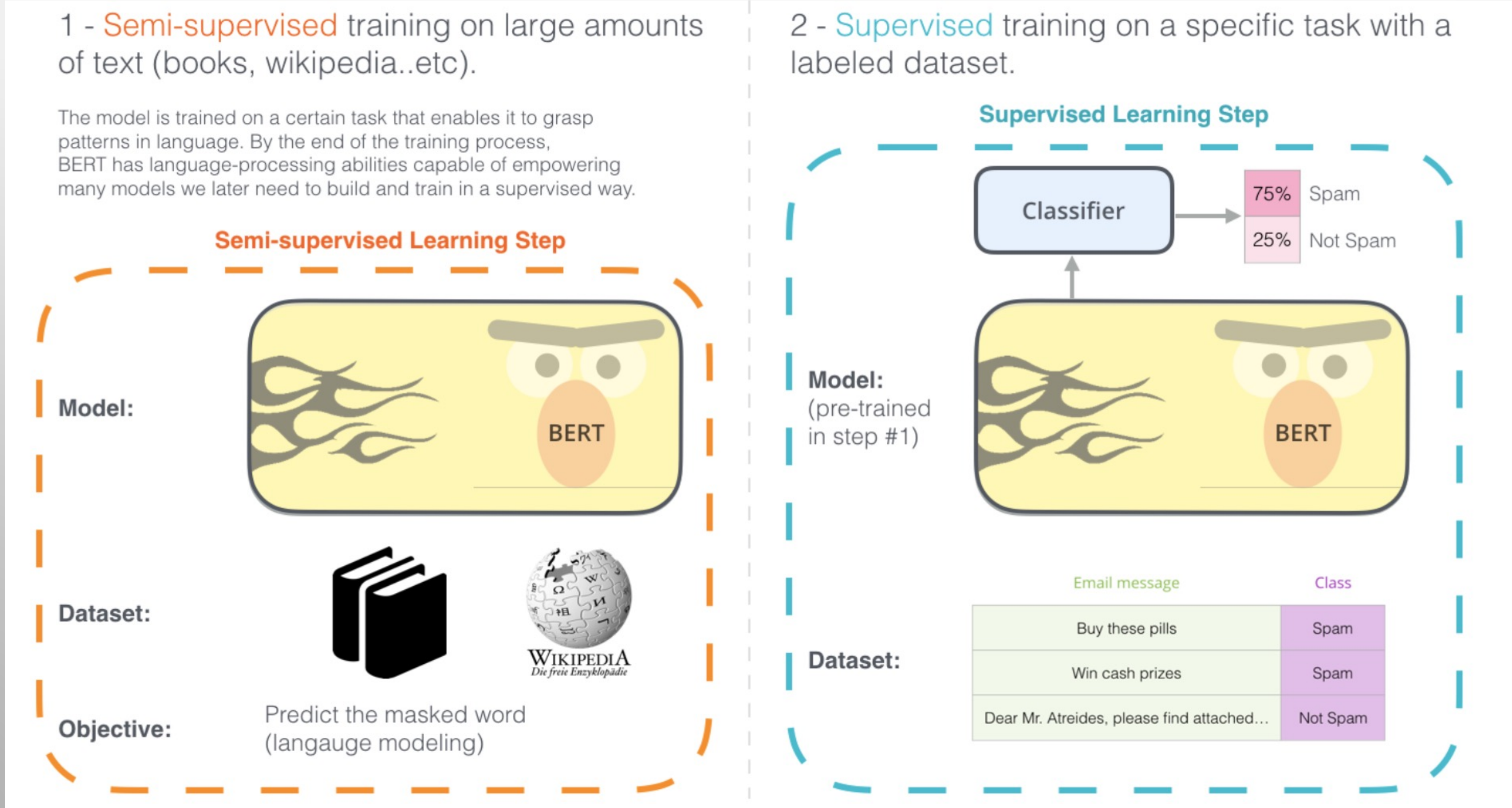


Figure 1: The Transform

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

BERT (Bidirectional Encoder Representations from Transformers)



The two steps of how BERT is developed. You can download the model pre-trained in step 1 (trained on un-annotated data), and only worry about fine-tuning it for step 2. [Source for book icon].

Widely used for classification tasks and sentiment prediction

https://jalammar.github.io/illustrated-bert/

Named Entity Recognition

- Named entity recognition: if we would like to detect named entities (person, organization, location) in sentences, we might produce a label for every single word denoting whether or not that word is part of a named entity.
  - To use sequence labeling for a span-recognition problem, we'll use a technique called IOB encoding (Ramshaw and Marcus, 1995).
  - In its simplest form, we label any token that begins a span of interest with the label B, tokens that occur inside a span are tagged with an I, and any tokens outside of any span of interest are labeled O. Consider the following example:

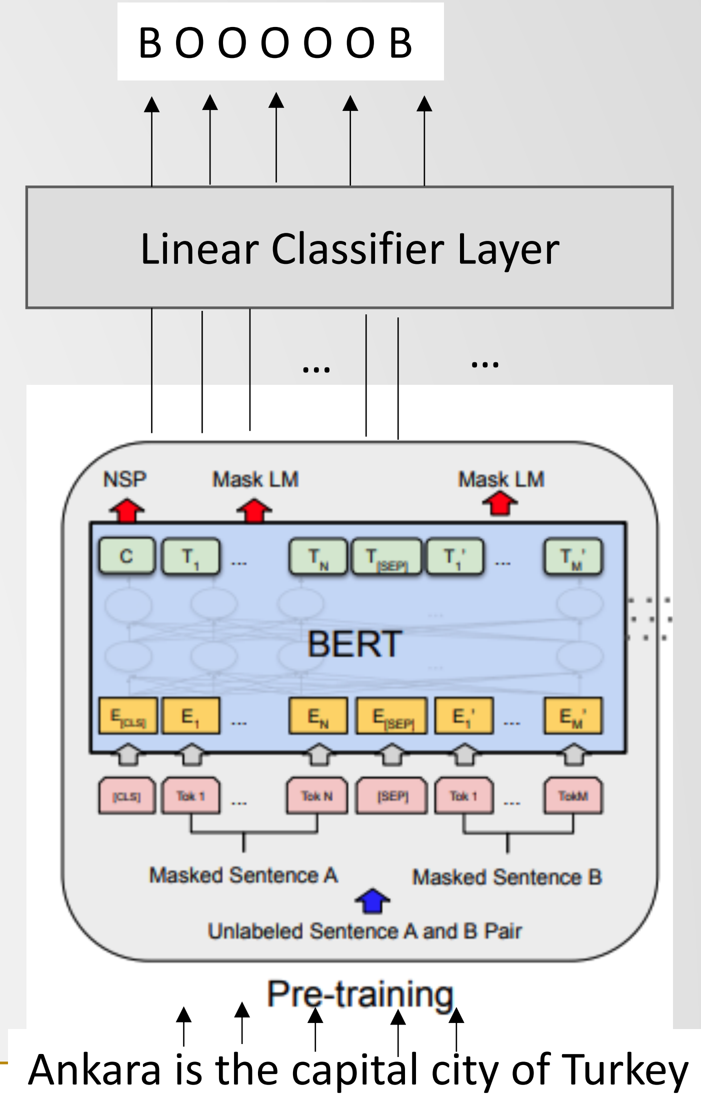
(9.13)	*United*	*cancelled*	*the*	*flight*	*from*	*Denver*	*to*	*San*	*Francisco.*
	B	O	O	O	O	B	O	B	I

Here, the spans of interest are United, Denver and San Francisco.
They could be extended as to include LOC, ORG etc. labels.

https://web.stanford.edu/~jurafsky/slp3/old_oct19/

Named Entity Recognition

- Named entity recognition:
- Input: The sentence is fed into the BERT or any alternative language model.
  - The model's last layer can be fine tuned for named entity recognition.
- Output will produce a class label for each input token.
  - Here: one of the B, O or I labels.

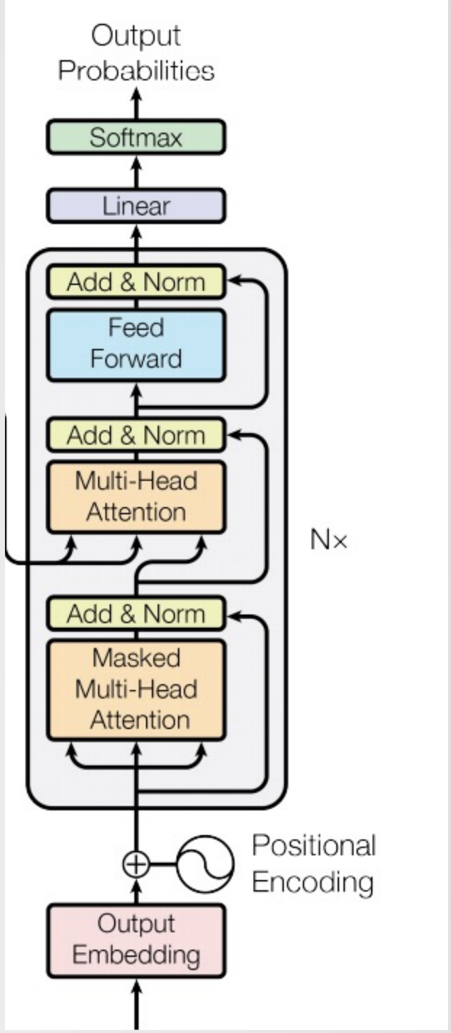


https://www.practicalai.io/understanding-transformer-model-architectures/

Generative Pre-Training (GPT-1)

Some Model Details

- They aim to predict the subsequent token as a part of a language modelling task.
  - Note that the original transformer was used for machine translation.
- Trained 12-layer decoder-only transformer with masked self-attention heads (768 dimensional states and 12 attention heads).
  - Total number of parameters is 117M parameters.
- Generating text that is both coherent and contextually consistent is a significant challenge in many NLP tasks, such as text generation, machine translation, and content summarization.
  - Decoder-only transformer architectures have been designed to address this problem. E.g.: GPT (Generative Pre-trained Transformer) models

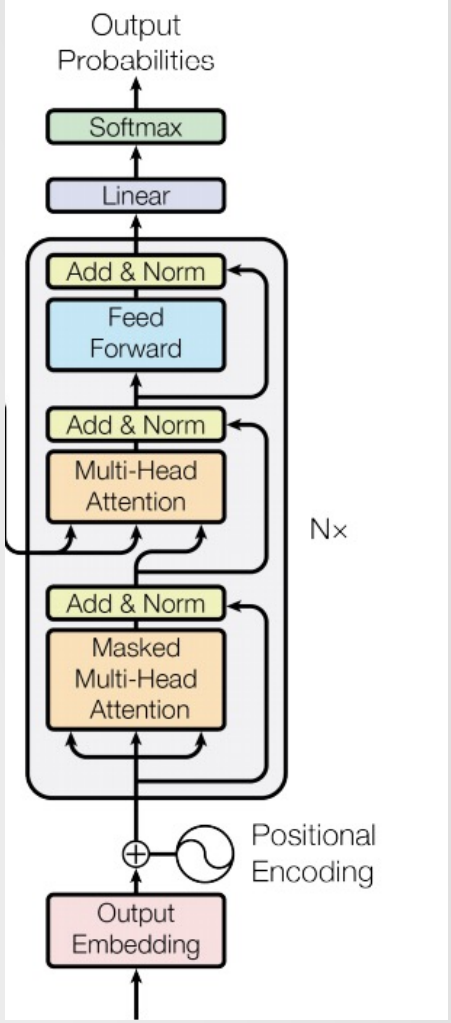


Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.
https://vitalflux.com/transformer-architecture-types-explained-with-examples/

Generative Pre-Training (GPT-1)

Some Model Details

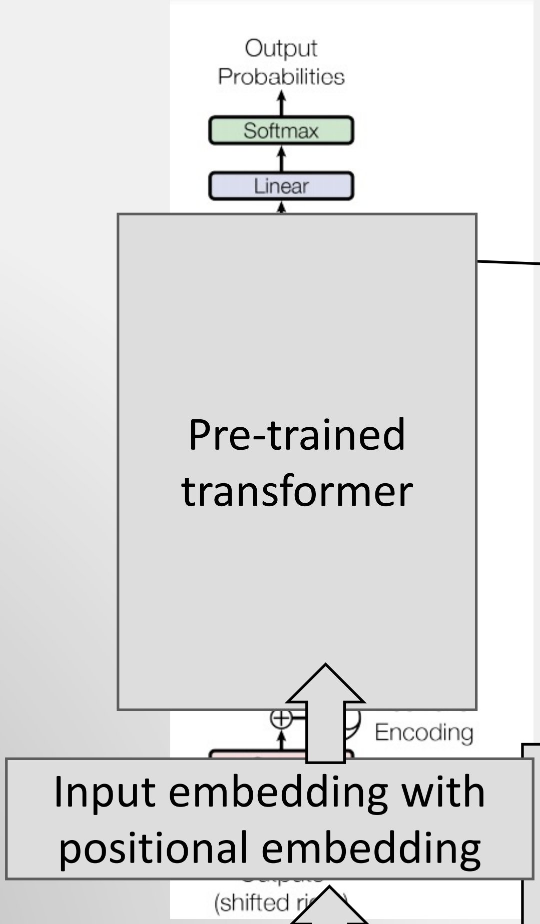
- For the position-wise feed-forward networks, they used 3072 dimensional inner states.
- They used the Adam optimization scheme with a max learning rate of 2.5e-4.
- They used a byte pair encoding (BPE) vocabulary with 40,000 merges [53] and residual, embedding, and attention dropouts with a rate of 0.1 for regularization.
- They used the Gaussian Error Linear Unit (GELU).
- Their language model did not have access to subsequent words to the right of current word.



Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

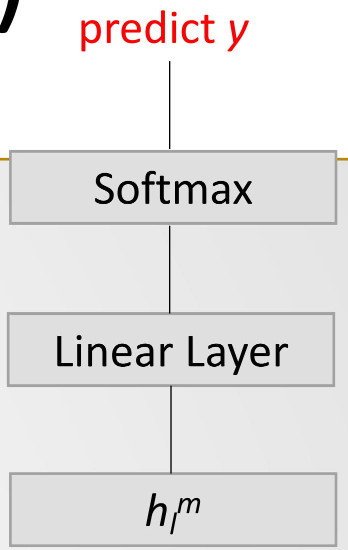
Generative Pre-Training (GPT-1)

Architecture



1. Feed the input to the pre-trained language model

2. Get the embeddings from the the final transformer block's activation h_l^m.



$$P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y)$$

The following objective is to be maximized

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m)$$

They assume a labeled dataset C, where each instance consists of a sequence of input tokens, $x_1, \ldots, x_m$, along with a label y.

e understanding with unsupervised learning. Technical report,

Generative Pre-Training (GPT-1)

Architecture: Auxiliary Training Objectives

- Adding auxiliary unsupervised training objectives is an alternative form of semi-supervised learning.
- They found that including language modeling as an auxiliary objective to the fine-tuning helped learning by
  - (a) improving generalization of the supervised model, and
  - (b) accelerating convergence.
- Specifically, they optimize the following objective (with weight λ):

$$\underbrace{L_3(\mathcal{C}) = L_2(\mathcal{C})}_{\text{Learning objective of the classification task.}} + \lambda * \underbrace{L_1(\mathcal{C})}_{\text{Learning objective of the language model task.}}$$

λ was set to 0.5

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Generative Pre-Training (GPT-1)

Sentiment Analysis with Huggingface

- Sentiment analysis with a zero-shot learning approach can also be implemented as follows using a textual entailment approach :

```
#!pip install transformers datasets
from transformers import pipeline
classifier = pipeline("zero-shot-classification")
candidate_labels = ["positive", "negative"]
text = "I can't understand transformer models as they are very complicated."
classifier(text, candidate_labels)
```

```
No model was supplied, defaulted to facebook/bart-large-mnli (https://huggingface.co/facebook/bart-large-mnli)
{'labels': ['negative', 'positive'],
 'scores': [0.8417665362358093, 0.15823349356651306],
 'sequence': "I can't understand transformer models as they are very complicated."}
```

An example for zero shot learning, the models can be selected based on https://huggingface.co/models

Textual Entailment

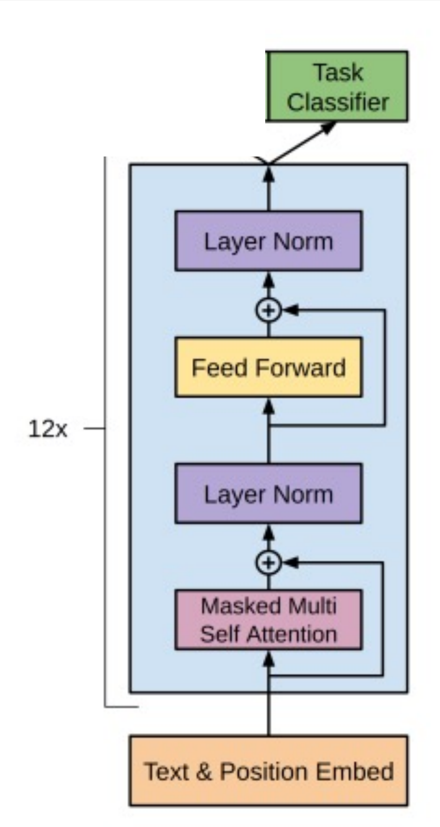
- Textual entailment recognition is the task of deciding, given two text fragments, whether the meaning of one text is entailed (can be inferred) from another text.
- It involves reading a pair of sentences and judging the relationship between them from one of entailment, contradiction or neutral.

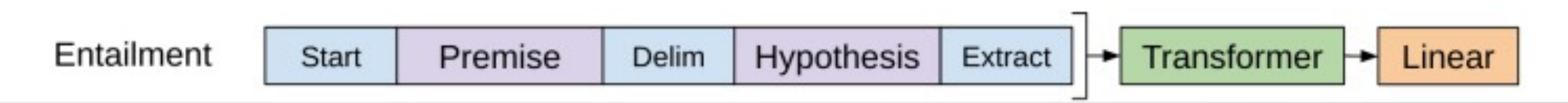
Textual entailment example from Stanford Natural Language Inference (SNLI) corpus:
SNLI corpus (version 1.0) is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels *entailment*, *contradiction*, and *neutral*.

Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling and laughing at the cats playing on the floor.
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a lonely road.
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a sport.
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fairy costume holds an umbrella.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.
https://nlp.stanford.edu/projects/snli/

Textual Entailment





The premise *p* and hypothesis *h* token sequences are concatenated, with a delimiter token ($) in between.
All transformations include adding randomly initialized start and end tokens(<s>,<e>).
E.g.
Input: <s>A man inspects the uniform of a figure in some East Asian country$The man is sleeping <e>).

Output: contradict, neutral, entailed (class variable)

 Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

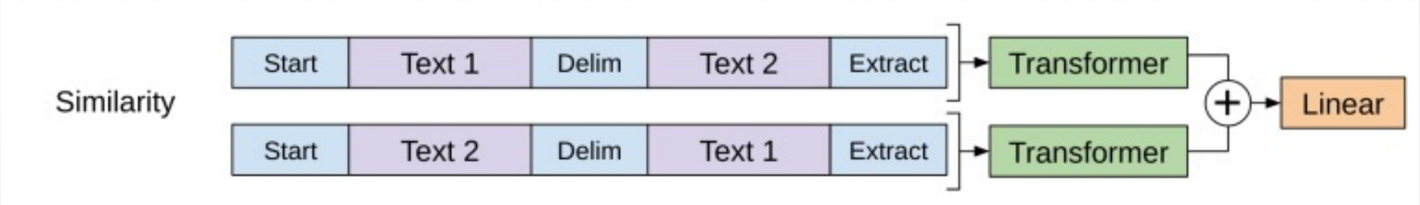
Text Similarity

- Semantic similarity (or paraphrase detection) tasks involve predicting whether two sentences are semantically equivalent or not.
- The challenges lie in recognizing the rephrasing of concepts, understanding negation, and handling syntactic ambiguity.

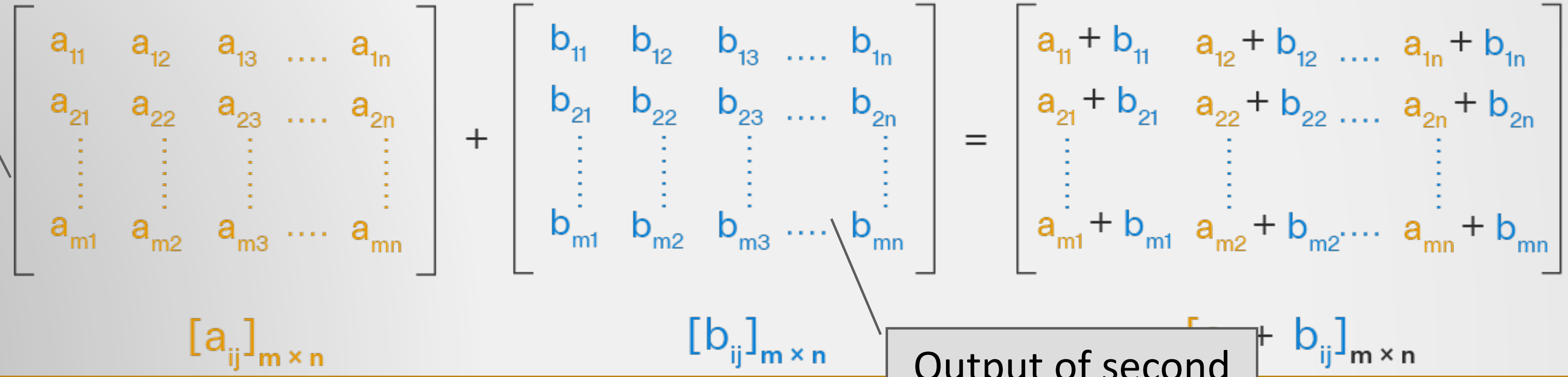
Sematically similar example:
The sun is shining brightly in the clear blue sky.
A radiant sun is illuminating the cloudless sky

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI
https://www.cuemath.com/algebra/addition-of-matrices/.

Text Similarity

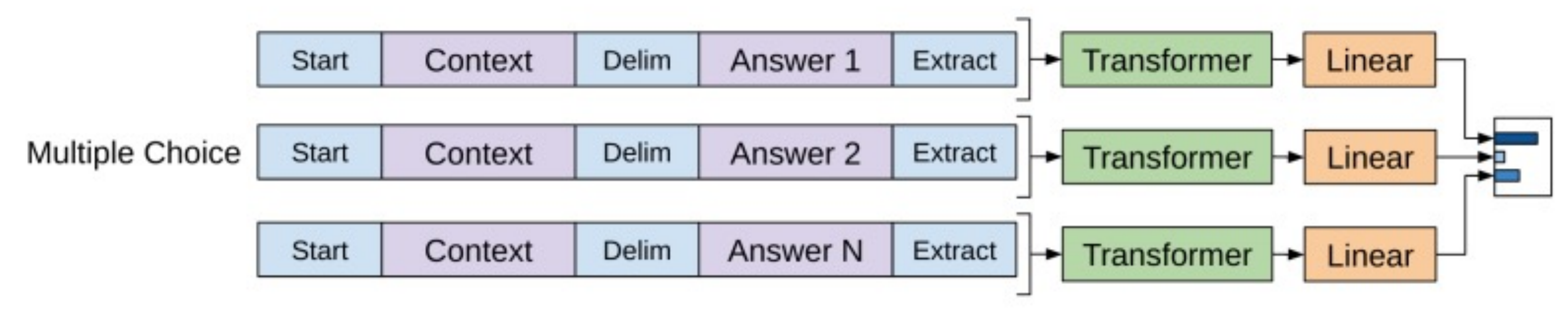


- There is no inherent ordering of the two sentences being compared.
  - Output: true/false
- They modify the input sequence to contain both possible sentence orderings (with a delimiter in between) and process each independently to produce two sequence representations added element-wise before being fed into the linear output layer.



Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving langua … d learning. Technical report, OpenAI
https://www.cuemath.com/algebra/addition-of-matrices/.

Question Answering

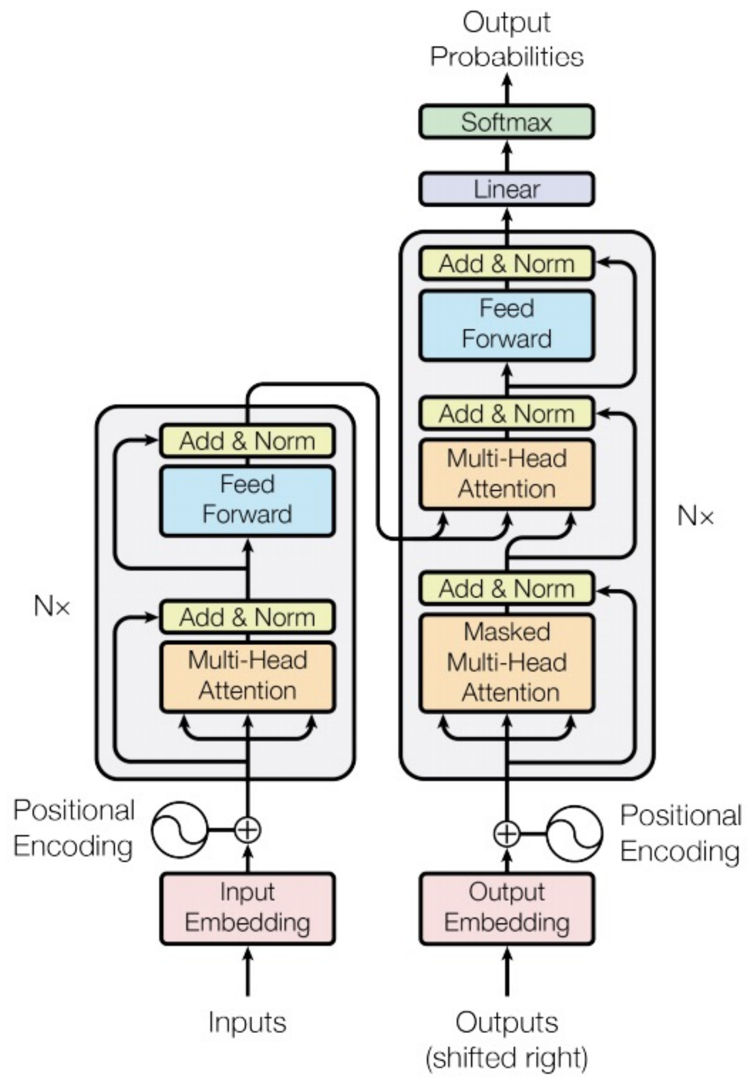


We are given a context document *z*, a question *q*,and a set of possible answers $\{a_k\}$.
We concatenate the document context and question with each possible answer, adding a delimiter token in between to get $[z;q;\$;a_k]$.
Each of these sequences is processed independently with their model and then normalized via a softmax layer to produce an output distribution over possible answers.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Text Summarization

- Encoder and decoder modules are both used.
- The input includes the original text.
  - Its compressed representation is fed into the decoder part.
- The output produces the summary.
  - Decoder helps due to its autoregressive nature.
  - One token is produced to construct the summary depending on the previously generated context.



Summary

- Encoder based models: Encoder part is responsible for understanding and extracting the relevant information from the input text. It generates a representation of the input text.
  - E.g. BERT, RoBERTa (good for label prediction)
- Decoder based models: Its masked multi-head attention component facilitates text generation by letting the model not seeing the subsequent tokens (autoregressive decoding lets generating token one at a time conditioning on the previously generated tokens).
  - E.g. GPT1, GPT2
- Encoder-decoder hybrids: Encoder-decoder models are typically used for natural language processing tasks that involve understanding input sequences and generating output sequences, often with different lengths and structures.
  - E.g. BART, T5. (machine translation, summarization tasks…)

https://magazine.sebastianraschka.com/p/understanding-encoder-and-decoder

Thanks!

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 951732. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, United Kingdom, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Switzerland, Turkey, Republic of North Macedonia, Iceland, Montenegro