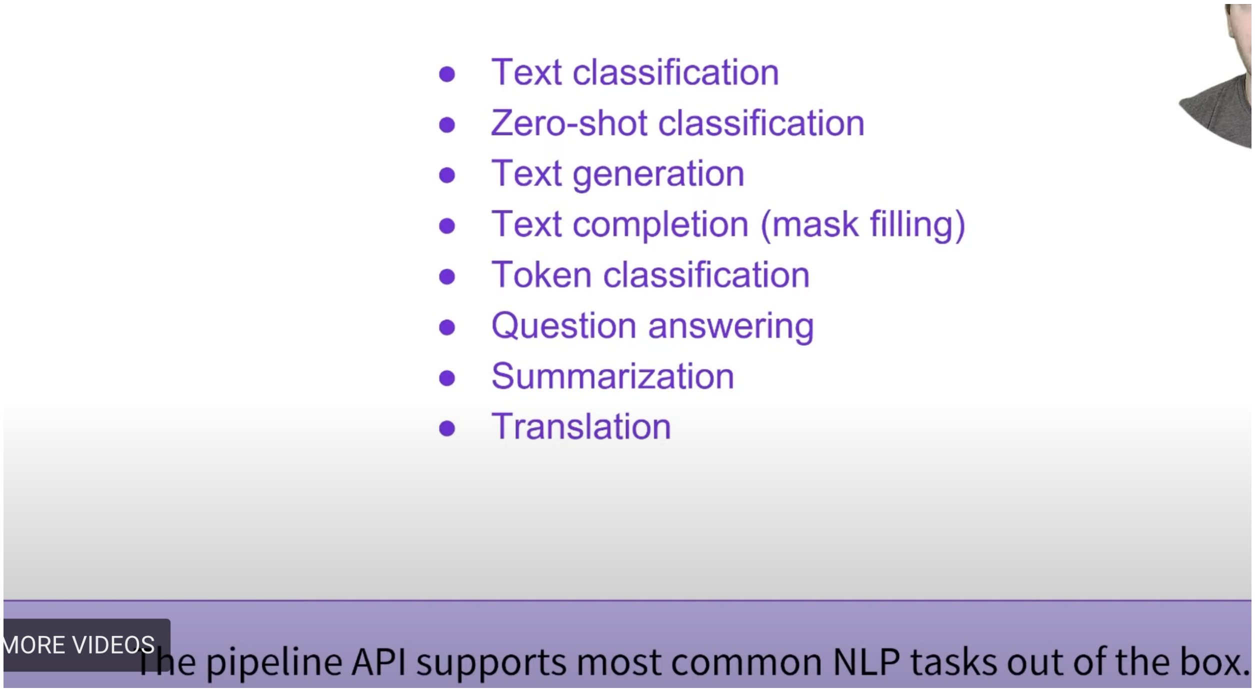
Hugging-Face-Transformers

1. Transformer, what can they do

Covers how the Hugging Face Transformer Library’s pipeline function gives you off-the shelf support to conduct the tasks described below.

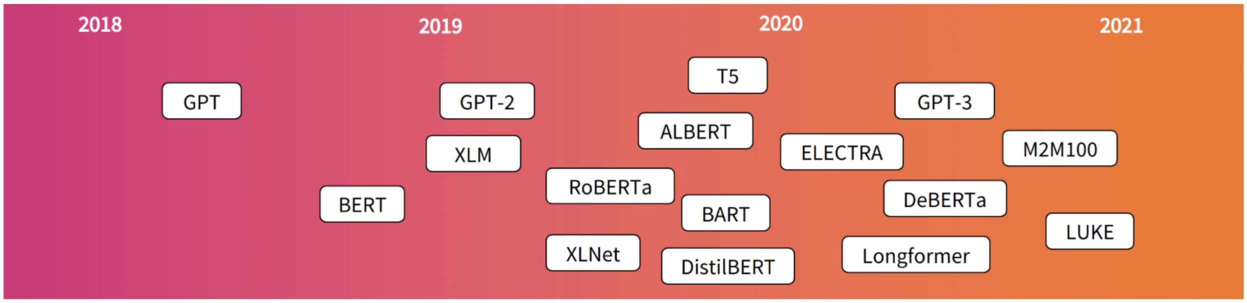


- Text classification
- Zero-shot classification
- Text generation
- Text completion (mask filling)
- Token classification
- Question answering
- Summarization
- Translation

The pipeline API supports most common NLP tasks out of the box.

How do Transformers Work?

Introduces a very breif history of Transformers:

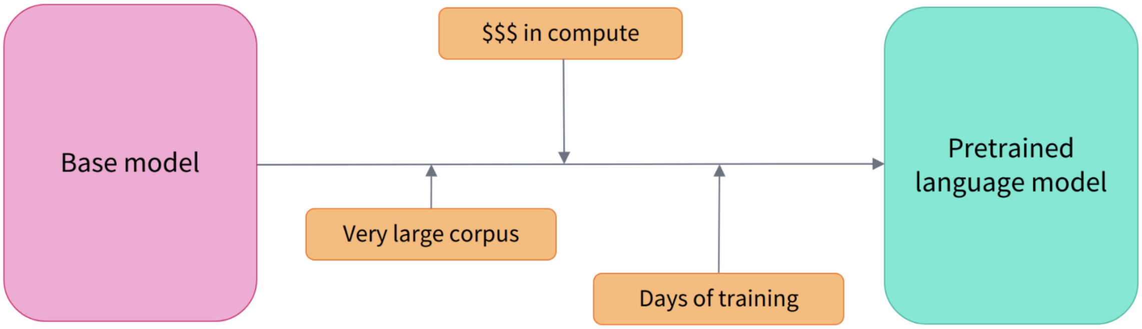


Language Models: A language model is usually trained on a raw text data in a self-suprevised fashion (no labels needed). The objective of the training is automatically computed from the inputs to the model.

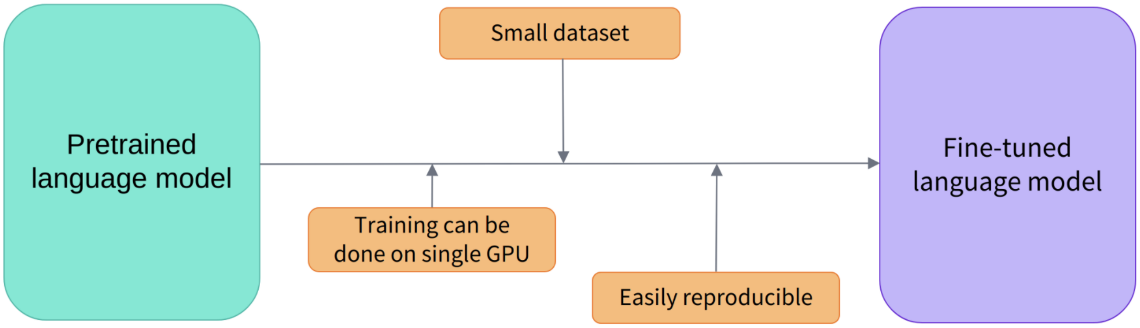
Transformers are usually very big models

Transfer Learning in Transformers

1. **Pre-Training** from scratch: The weights are randomly initialized, and the training starts without any prior knowledge.



2. **Fine-Tuning**: Where a pretrained model is trained on a relatively small dataset for a specific task (like text-classifcation)


Small dataset
Pretrained language model
Training can be done on single GPU
Easily reproducible
Fine-tuned language model

Transformers Encoders:

Dimension of the word-vectors is defined by the architecture of the transformer model (ex for BERT it is 768).

Encoders are usually bi-directional in reading the input sequence.

When to use Encoders:

- Bi-directional: context from the left, and the right
- Good at extracting meaningful information
- Sequence classification, question answering, masked language modeling
- NLU: Natural Language Understanding
- Example of encoders: BERT, RoBERTa, ALBERT

Masked Language Modeling: Predict a masked word in a sentence. Ex: My <> is Virajdatt

BERT was created for this task specifically and also next sentence prediction (NSP) objectives)

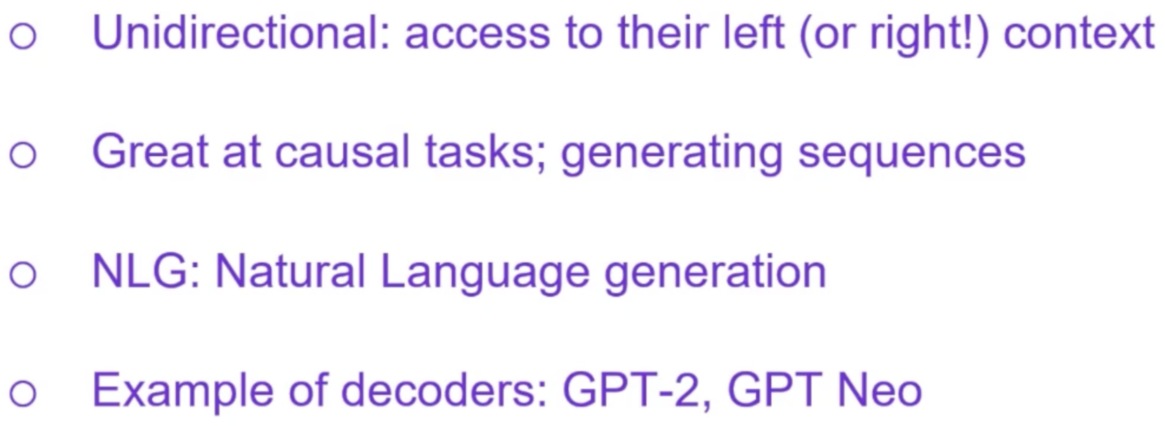
Example Models of Encoders

Transformers Decoders:

The decoder uses masked self attention.

Decoder have access to only one context, so they can look at the words on the right/left side.

When to use Decoders:



- Unidirectional: access to their left (or right!) context
- Great at causal tasks; generating sequences
- NLG: Natural Language generation
- Example of decoders: GPT-2, GPT Neo

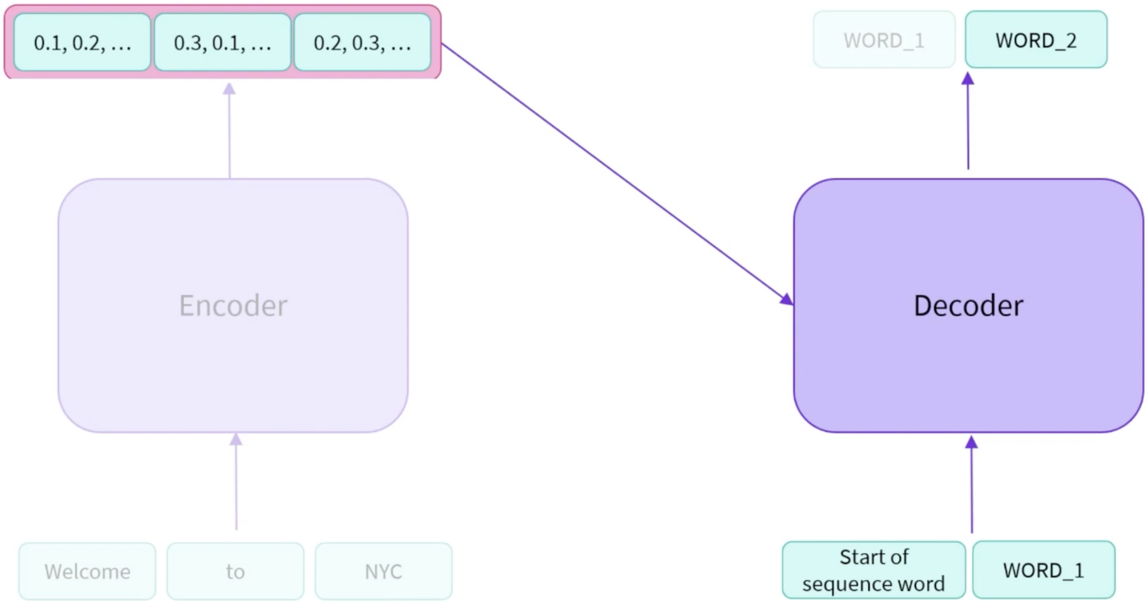
Casual Language Modeling

Example Models of Decoders:

1.
2.

Encoder-Decoder:

Encoder-Decoder don't share weights



When to use Encoder-Decoder::

- - Sequence to sequence tasks; many-to-many: translation, summarization

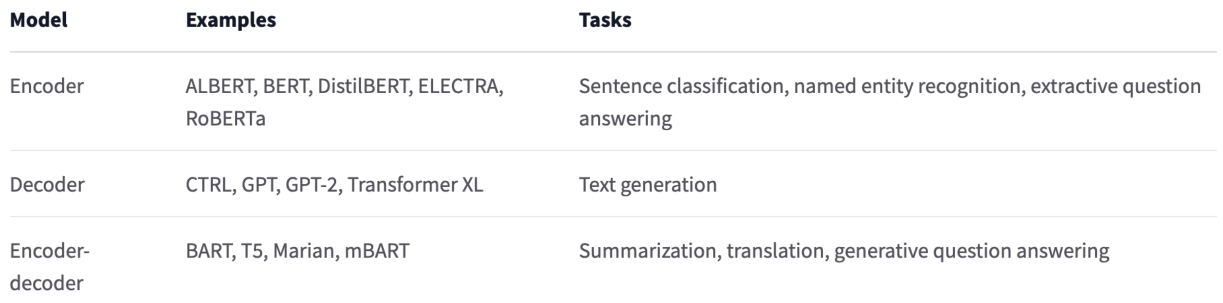
- - Weights are not necessarily shared across the encoder and decoder

- - Input distribution different from output distribution

Bias and Limiations

When you use these tools, you therefore need to keep in the back of your mind that the original model you are using could very easily generate sexist, racist, or homophobic content. Fine-tuning the model on your data won’t make this intrinsic bias disappear.Next Section

Summary:



Model	Examples	Tasks
Encoder	ALBERT, BERT, DistilBERT, ELECTRA, RoBERTa	Sentence classification, named entity recognition, extractive question answering
Decoder	CTRL, GPT, GPT-2, Transformer XL	Text generation
Encoder-decoder	BART, T5, Marian, mBART	Summarization, translation, generative question answering