

# Transfer Learning

# Building Tools with AI

- When building new applications, we want to...
  - use as much data as we can
  - train a large-as-needed model
- However, we don't have...
  - access to “big data”
  - time and resources to annotate thousands of examples
  - GPUs and energy for millions of Swiss Francs

# Building Tools with AI

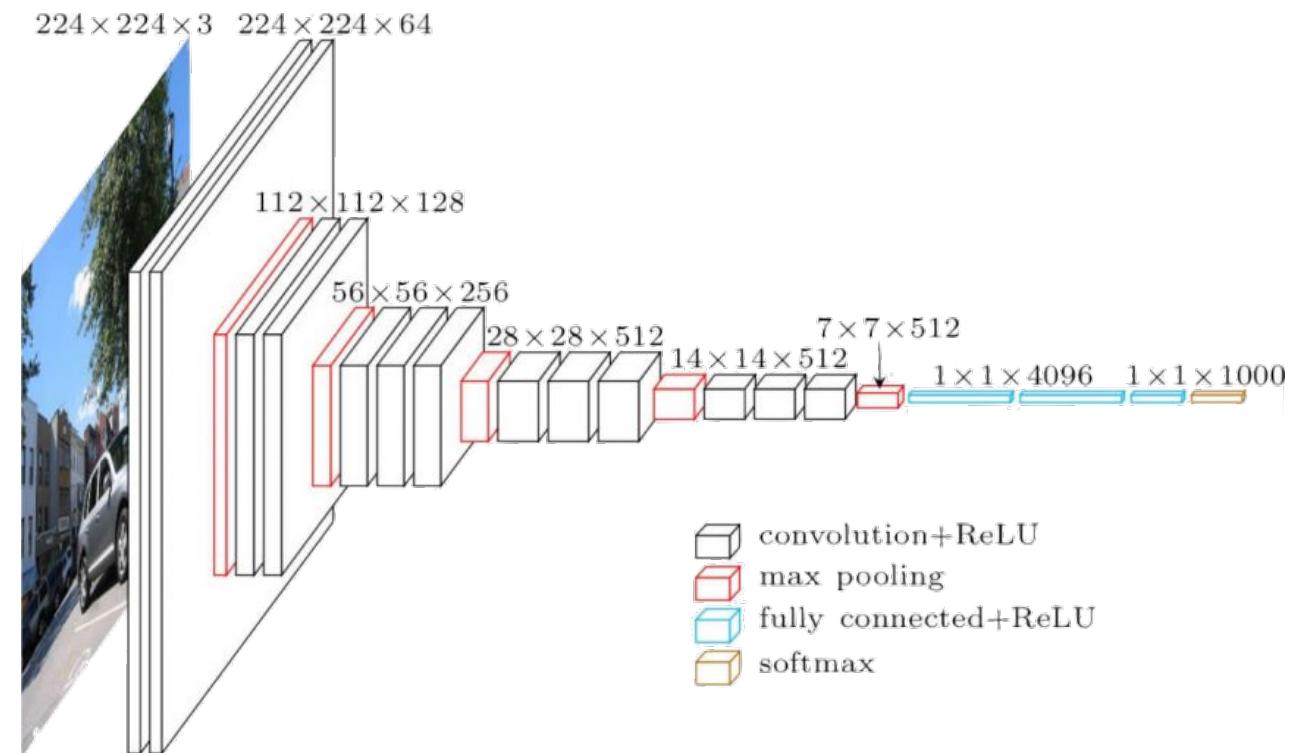
- Identify a large pre-trained model that fits your use case
  - These are often called “foundation models”
  - Many available open source (huggingface!)
- Choose a learning strategy
  - Fine-tuning?
  - Zero-/one-/few-shot?
  - RLHF?
- Label a few (hundred) examples
- Deploy!

# Terminology

- Pre-training
- Foundation Model
- Representation learning
- Transfer learning

# Image Processing

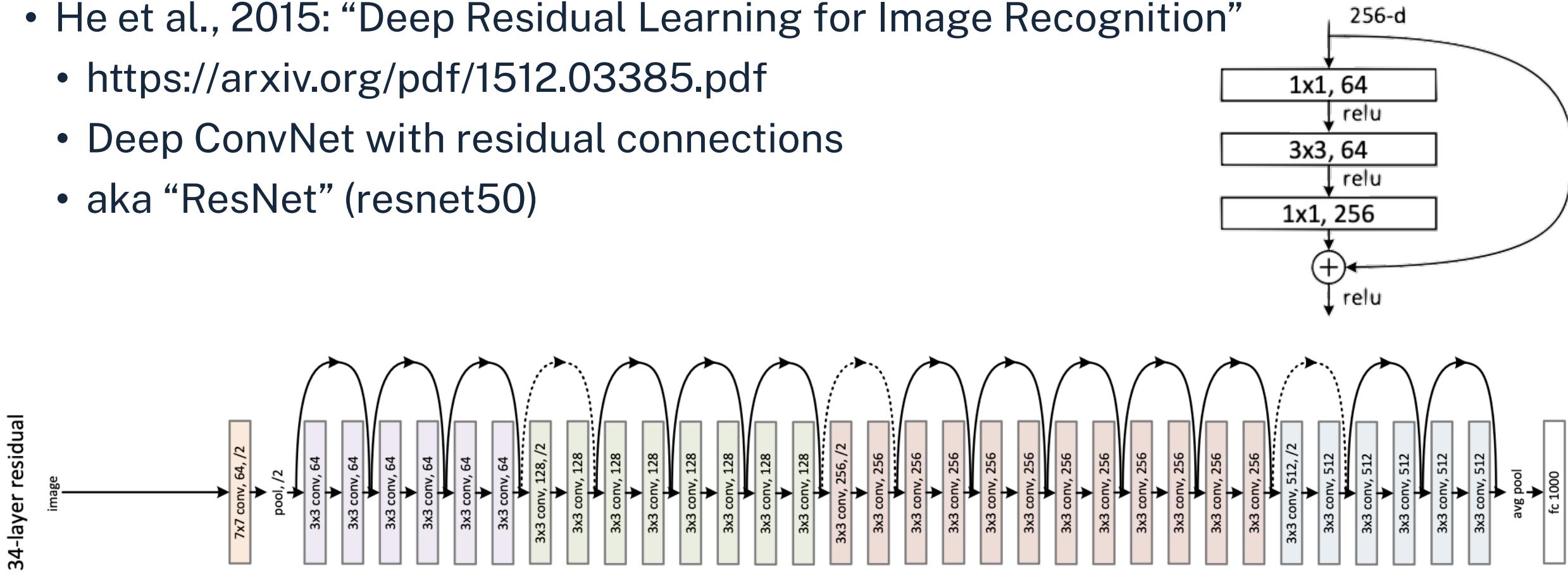
- Simonyan and Zisserman, 2015: “Very Deep Convolutional Networks for Large-Scale Image Recognition”
  - <https://arxiv.org/pdf/1409.1556.pdf>
  - Deep ConvNet trained on ImageNet (winner 2014!)
  - aka OxfordNet, VGG16



<https://neurohive.io/en/popular-networks/vgg16>

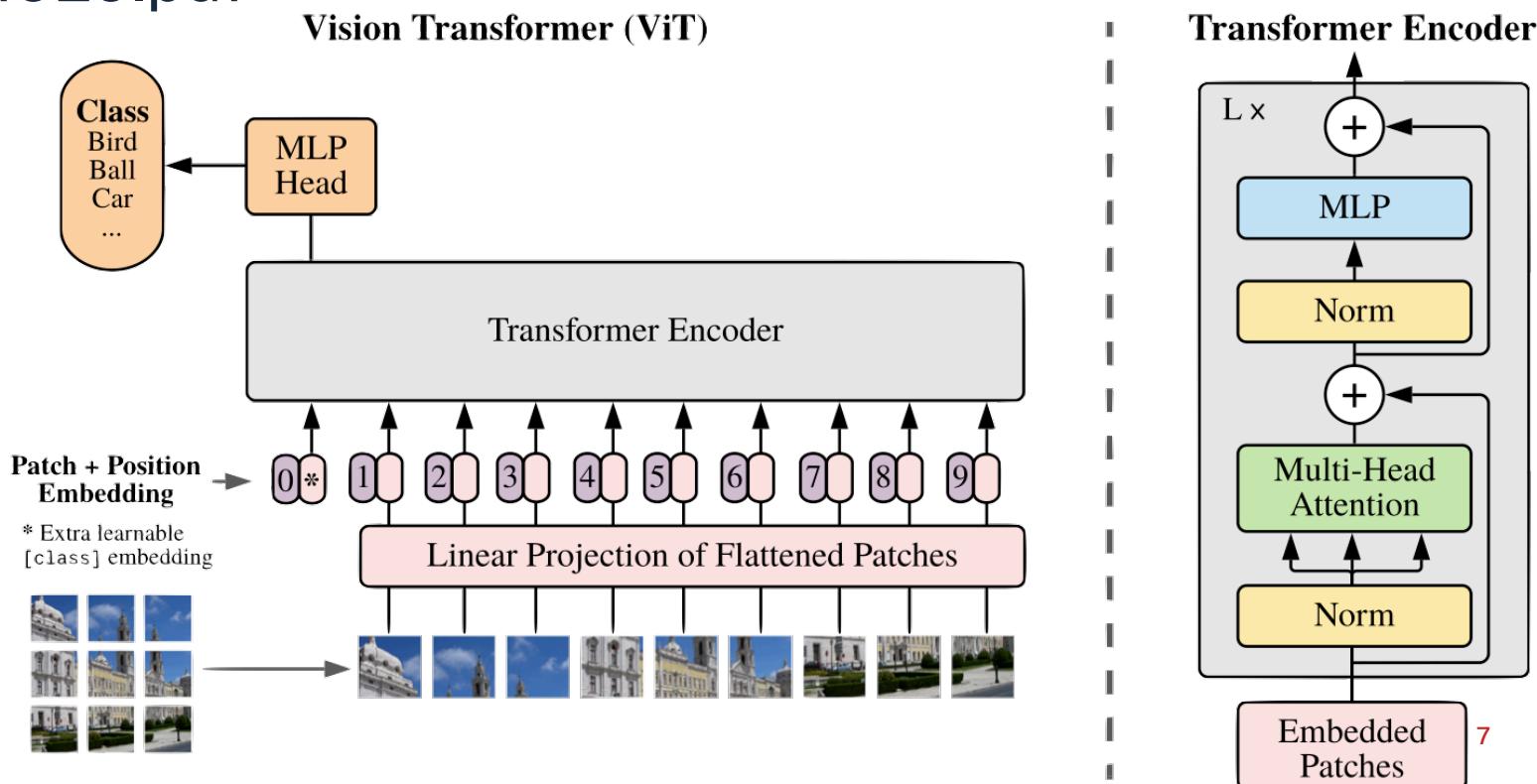
# Image Processing

- He et al., 2015: “Deep Residual Learning for Image Recognition”
    - <https://arxiv.org/pdf/1512.03385.pdf>
    - Deep ConvNet with residual connections
    - aka “ResNet” (resnet50)



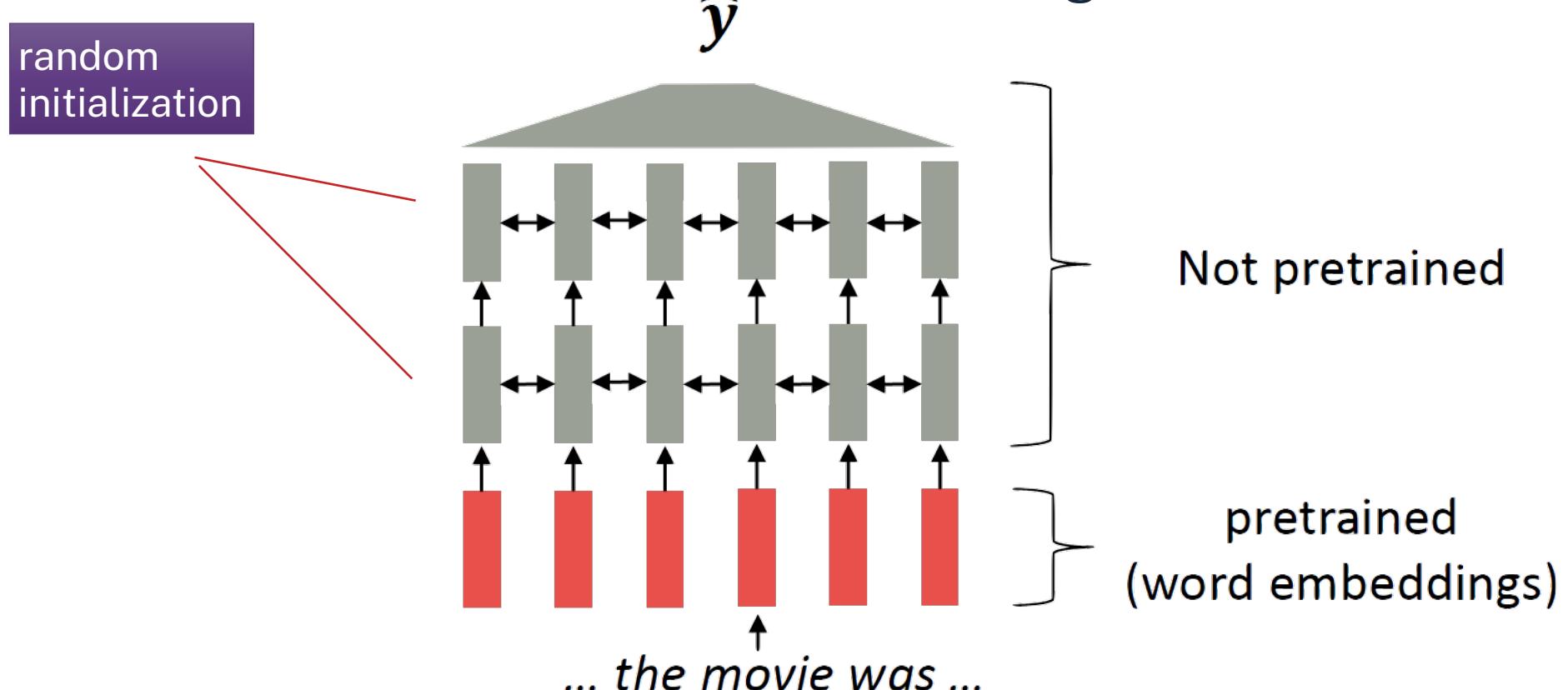
# Image Processing

- Dosovotskiy et al. 2021: “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”
  - <https://arxiv.org/pdf/2010.11929.pdf>
  - Transformer with patch+position embedding
  - aka “ViT”



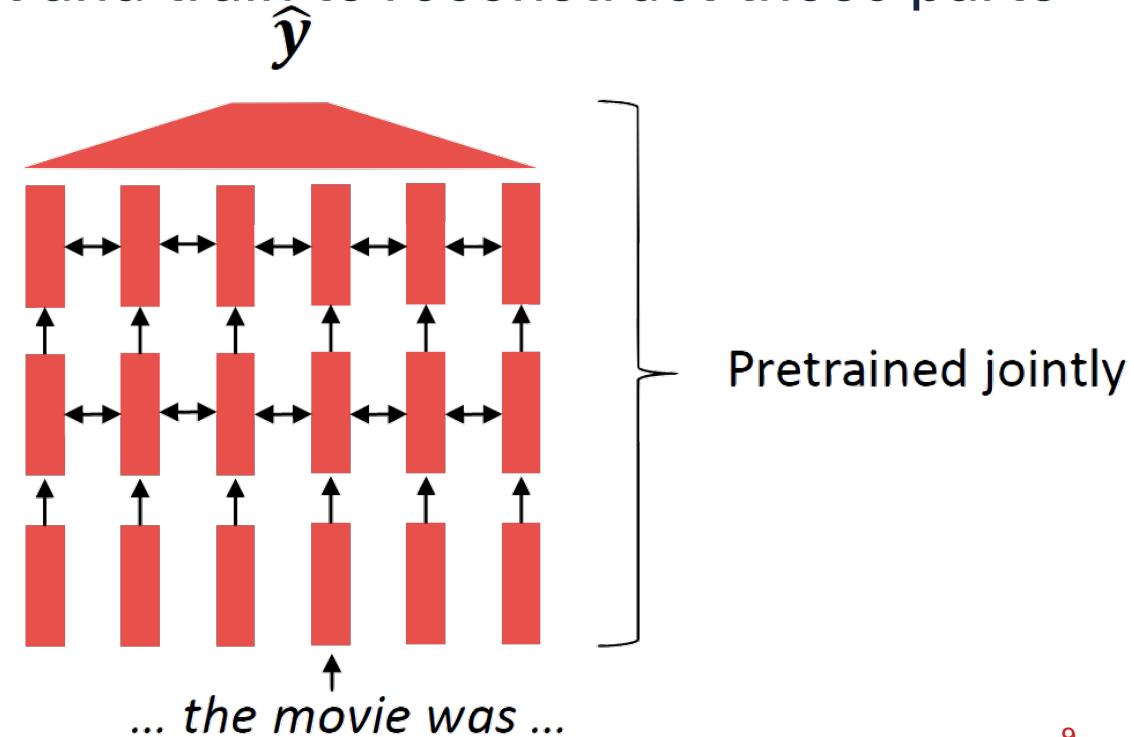
# Natural Language Processing

- Up to 2017: start with pre-trained word embeddings (no context!)
- Learn the context in LSTM/Transformer while training to the task



# Natural Language Processing

- Today:
  - (Almost) all parameters of the network initialized via pre-training
  - Pre-training methods hide parts of the input and train to reconstruct those parts
- With great success!
  - representations of language
  - parameter initializations for strong NLP models
  - probability distributions over language that we can sample from



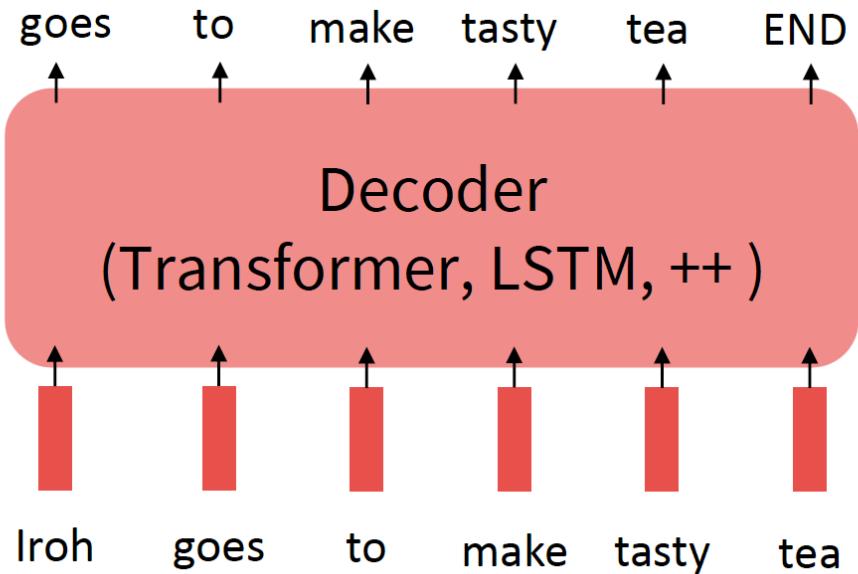
# Learning from reconstructing the input

- University of St. Gallen is located in \_\_\_\_\_, Switzerland.
- I put \_\_\_ fork down on the table.
- The man crossed the street, checking for bikes over \_\_\_ shoulder.
- I went to the zoo to see the monkeys, giraffes and \_\_\_\_\_.
- I wasted two hours of my precious lifetime; the movie was \_\_\_\_\_.

# The Pre-Training / Fine-tuning Paradigm

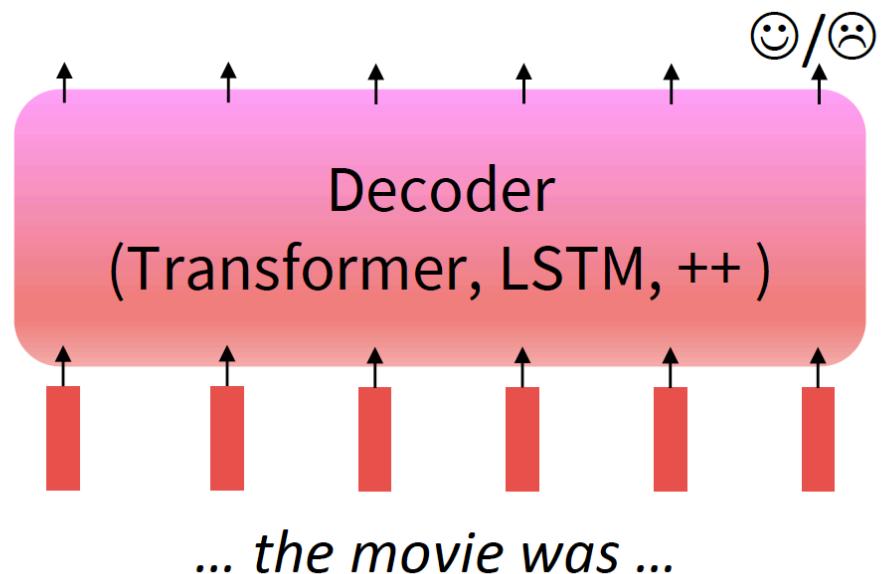
## Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



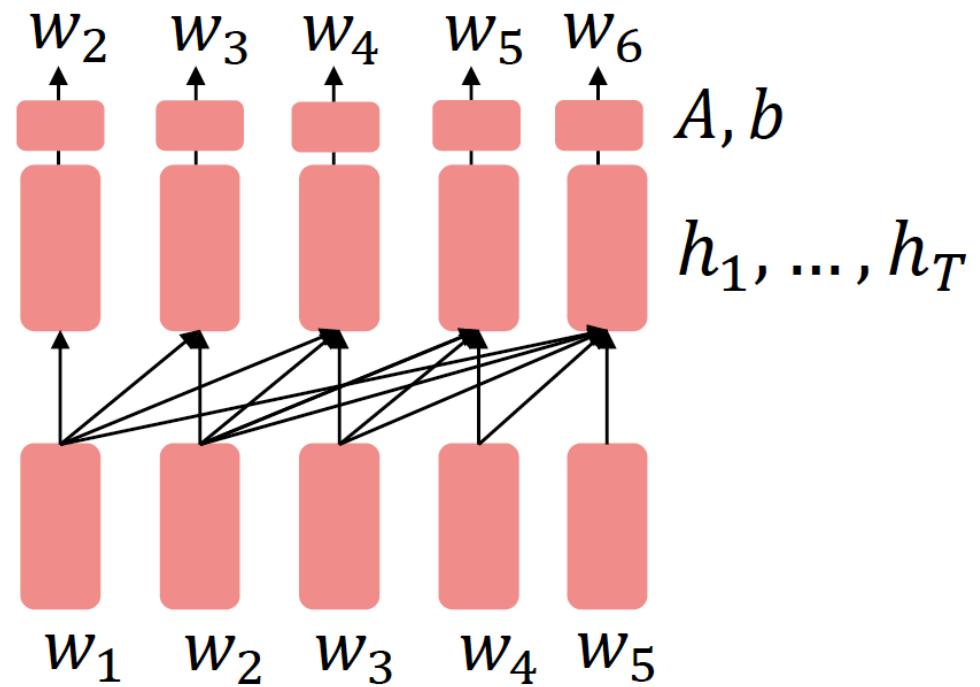
## Step 2: Finetune (on your task)

Not many labels; adapt to the task!



# Pre-training Decoders

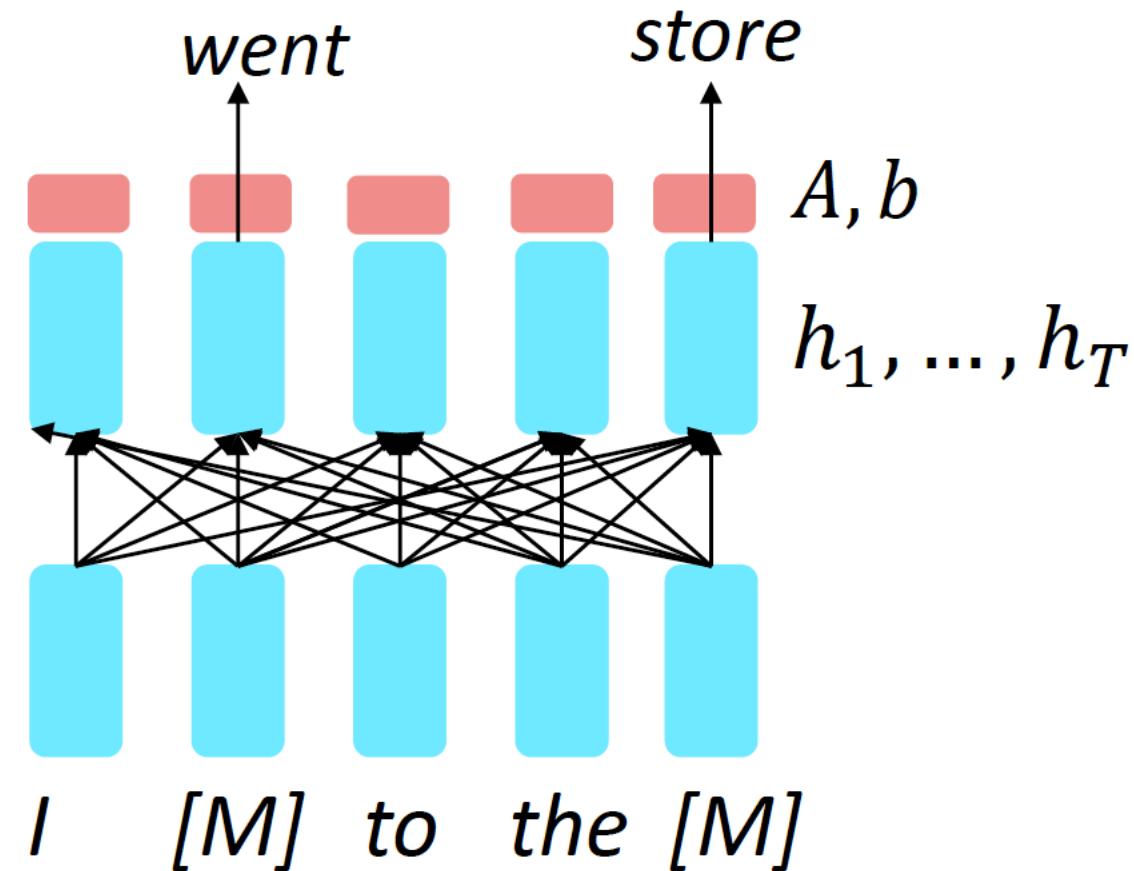
- Pre-train as language models
- ...fine-tune on target sequences
- Helpful where input and output share the vocabulary
- Mask/ignore future!
- Dialog, summarization, ...



[Note how the linear layer has been pretrained.]

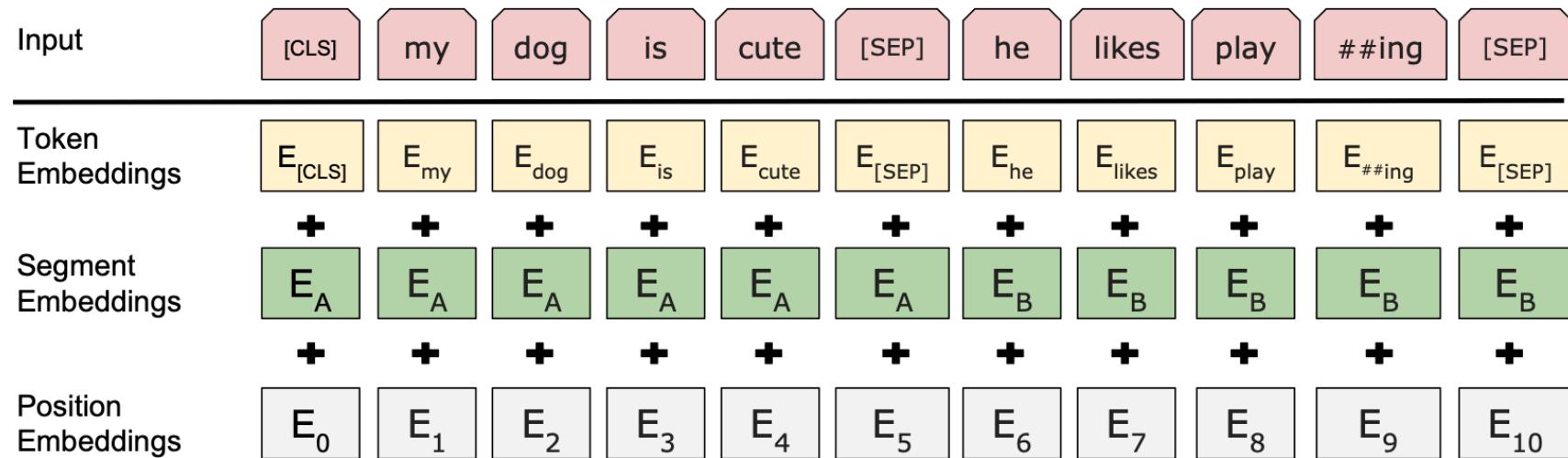
# Pre-training Encoders

- Full input accessible – can condition on the future!
- Replace fraction of words with special [MASK] symbol
- ..train networks to reconstruct, predict those words



# Ground-breaking Paper & Model

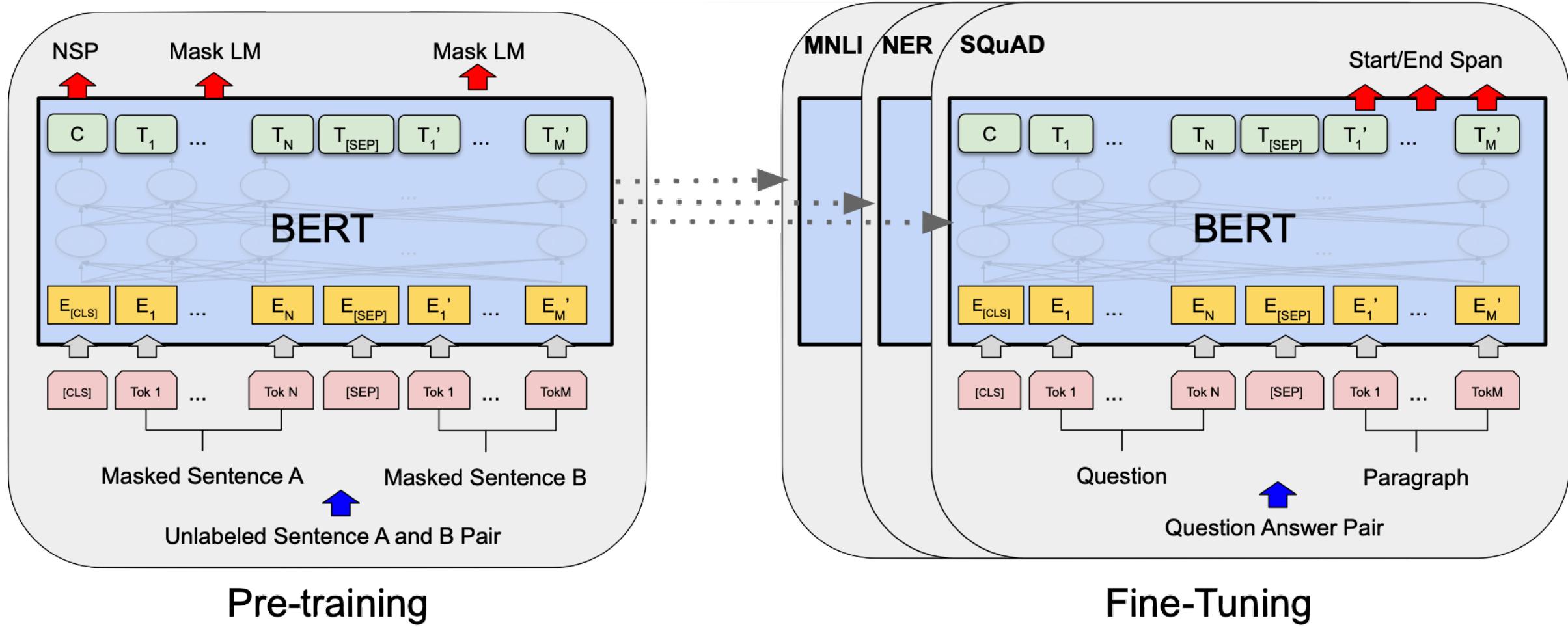
- Devlin et al., 2018: BERT: Bi-directional Encoder Representations from Transformer (<https://arxiv.org/abs/1810.04805>)
- During training: masked LM: randomly set some of the inputs to a placeholder (dropout)
- Fine-tune based on output corresponding to CLS



## BERT's success

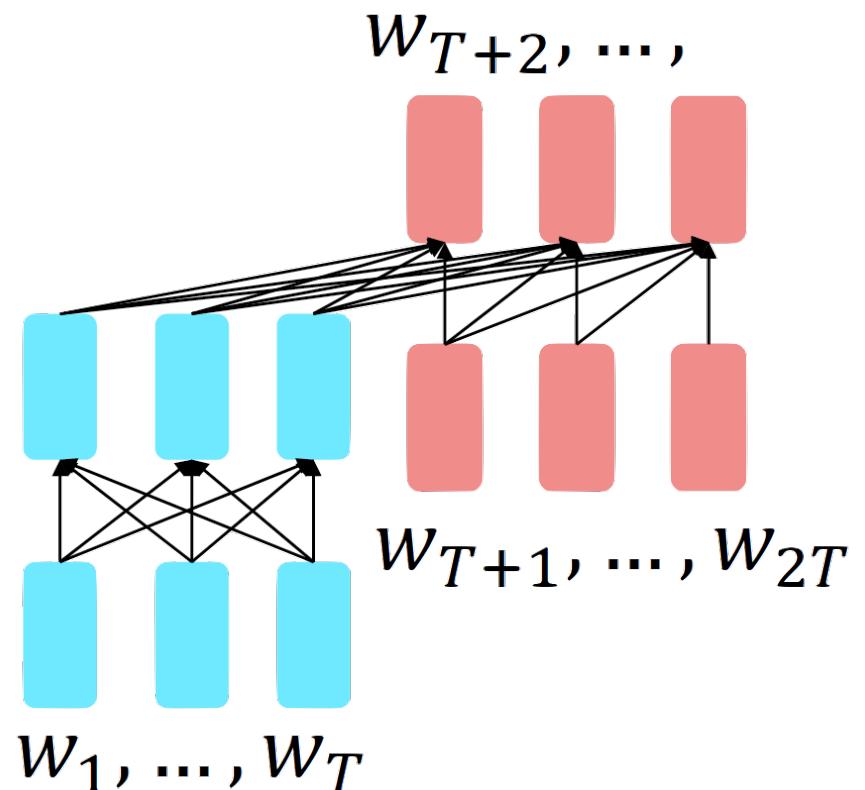
- Trained on huge corpus using masked LM
- In 2018, very same model achieved state-of-the-art values, for
  - Quora Question Pairs
  - CoLA (linguistic acceptability)
  - ...and many more.

# BERT Transfer Learning



# Pre-training Encoder-Decoders

- Raffel et al. 2018: “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer: Use sequence prefix to feed encoder”
  - <https://arxiv.org/pdf/1910.10683.pdf>
  - Use decoder part of the sequence to learn language modeling
  - Use span corruption instead of just masks



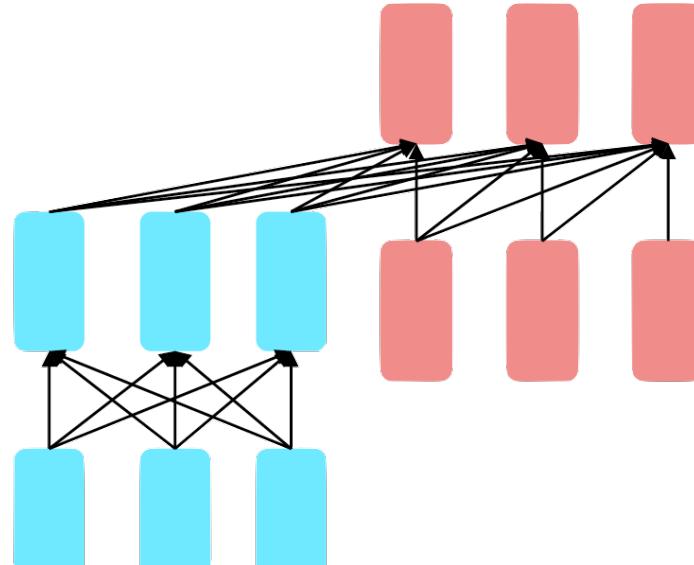
# Span corruption

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

Targets

<X> for inviting <Y> last <Z>



Inputs

Thank you <X> me to your party <Y> week.

"translate English to German: That is good."

"cola sentence: The course is jumping well."

"stsbs sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."

"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi..."

T5

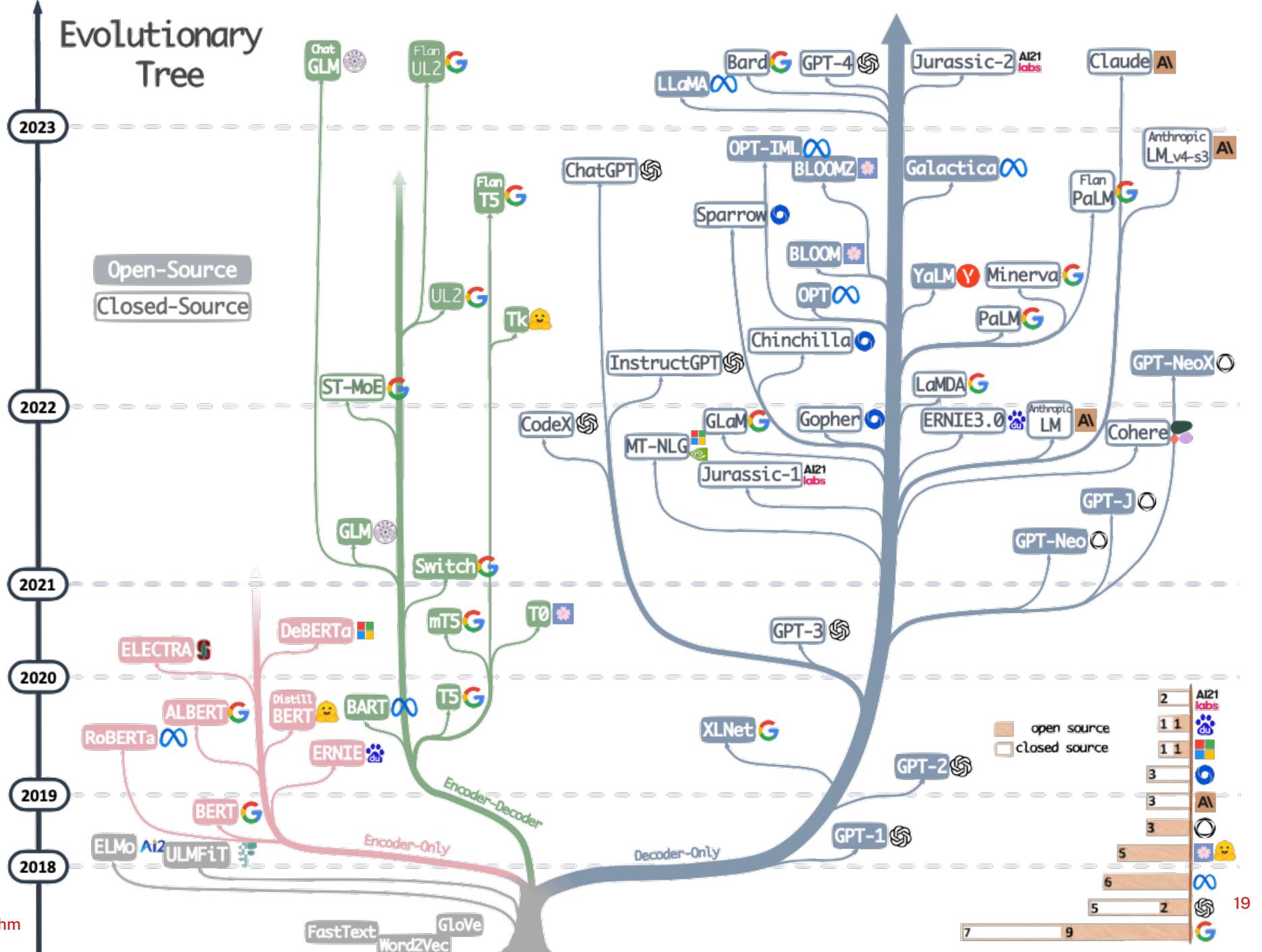
"Das ist gut."

"not acceptable"

"3.8"

"six people hospitalized after a storm in attala county."

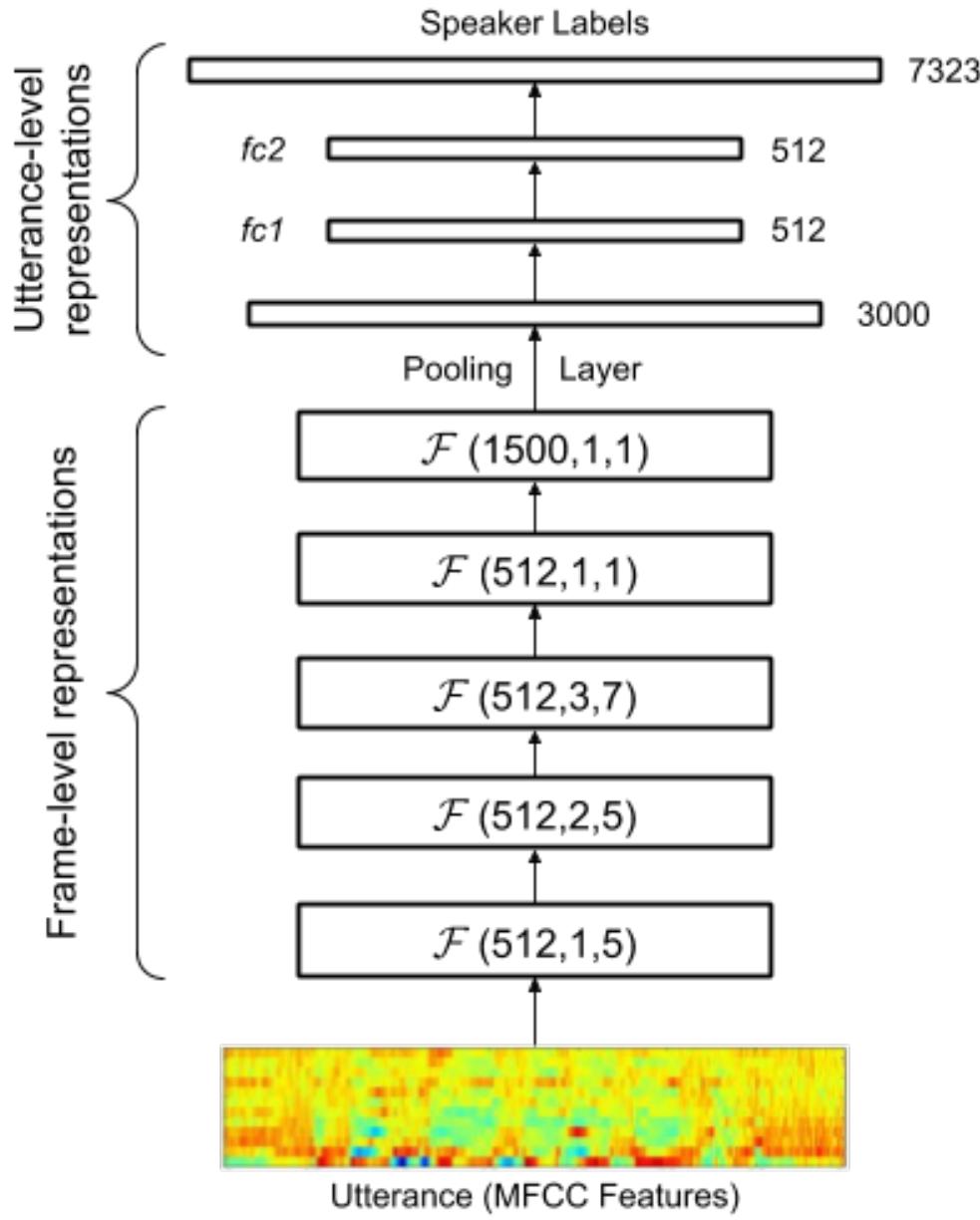
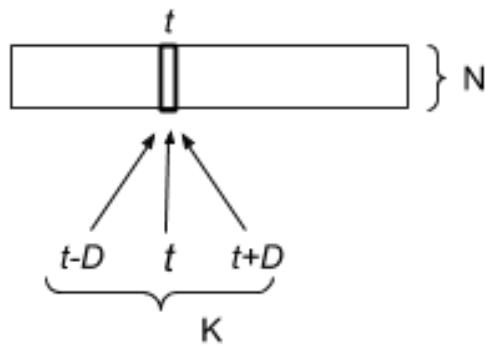
Too many LLMs to talk about...



# Audio/Speech Processing

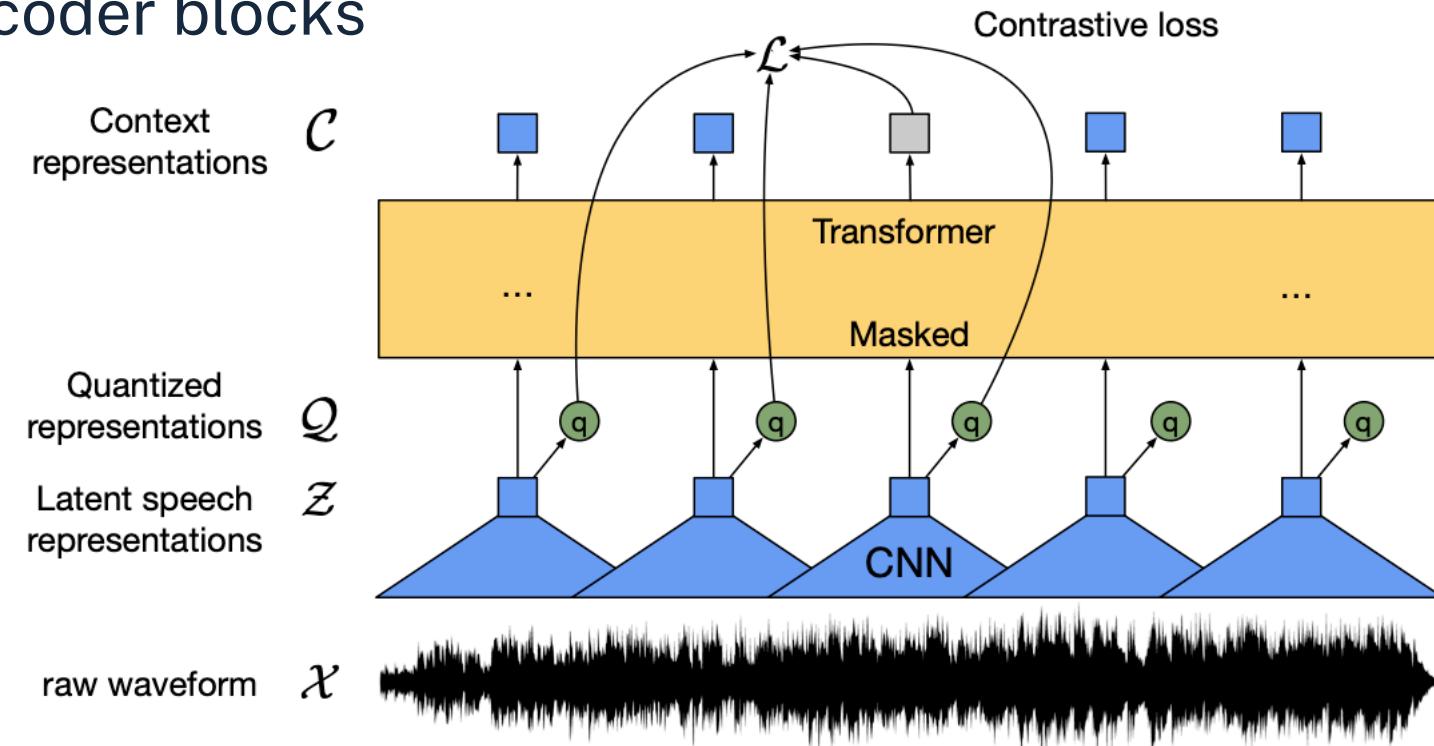
- Snyder et al., 2018: “X-Vectors: Robust DNN Embeddings for Speaker Recognition”.
  - [https://www.danielpovey.com/files/2018\\_icassp\\_xvectors.pdf](https://www.danielpovey.com/files/2018_icassp_xvectors.pdf)
  - Use TDNN features and global pooling
  - Classification of Age/Sex, medical conditions, etc.

# X-Vectors



# Audio/Speech Processing

- Baevski et al. 2020, “wav2vec2.0: A Framework for Self-Supervised Learning of Speech Representations”
  - 7 convolution layers on raw input
  - 12 (24) transformer encoder blocks
  - Masked training
  - Contrastive loss
- Used for...
  - ASR
  - SID
  - LID ...



The screenshot shows the Huggingface website interface. On the left, there's a sidebar with sections for Tasks (Fill-Mask, Question Answering, Summarization, etc.), Libraries (PyTorch, TensorFlow, JAX), Datasets (wikinli, common\_voice, wikipedia, etc.), Languages (en, bn, es, fr, de, sv, fi, zh), and Licenses (apache-2.0, mit, cc-by-4.0). The main area displays a grid of 20 pre-trained model cards. Each card contains the model name, its purpose (e.g., Fill-Mask, Text Classification), last update date, and number of downloads. Some models shown include bert-base-uncased, distilbert-base-uncased, roberta-base, and gpt2.

Model	Type	Last Updated	Downloads
bert-base-uncased	Fill-Mask	May 18	11,768k
xlm-roberta-base	Fill-Mask	Dec 11, 2020	4,775k
distilbert-base-uncased	Fill-Mask	Dec 11, 2020	3,038k
roberta-base	Fill-Mask	Dec 11, 2020	2,831k
allegro/herbert-base-cased	Updated	May 28	2,627k
bert-large-uncased-whole-word-masking-finetuned-sst-2-english	Question Answering	May 18	2,569k
bert-base-cased	Fill-Mask	May 18	2,402k
sentence-transformers/paraphrase-xlm-rand122	Sentence Similarity	Updated yesterday	2,121k
distilbert-base-uncased-finetuned-sst-2-english	Text Classification	Feb 9	1,450k
google/bert_uncased_L-12_H-512_A-8	Updated	May 19	1,345k
bert-base-chinese	Fill-Mask	May 18	1,337k
distilbert-base-cased	Updated	Dec 11, 2020	1,038k
gpt2	Text Generation	May 19	986k
roberta-large	Fill-Mask	May 21	890k
deepset/roberta-base-squad2	Question Answering	May 20	711k
bert-base-multilingual-cased	Fill-Mask	May 18	652k
bert-large-uncased	Fill-Mask	May 18	546k
hfl/chinese-bert-wwm-ext	Fill-Mask	May 19	516k
deepset/bert-large-uncased-whole-word-masking-finetuned-sst-2-english	Question Answering	May 19	509k
sshleifer/distilbart-cnn-12-6	Summarization	Updated 10 days ago	507k

# Summary

- We can learn representations from large corpora
  - using dedicated labels (eg. ImageNet, VoxCeleb)
  - using surrogate labels, ie. by reconstructing masked inputs (c4)
- For (most) of the models we discussed, fine-tuning to the actual task only involves training a single final layer (or maybe a few iterations of BP)
- Auto-encoder bottle-necks are also (basic) representations
  - typically much smaller/simpler architectures
  - resulting representations require typically much more sophisticated architectures for the actual tasks
- LLMs are tempting, but should only be used if approximate answers are ok!

# Recommendations

- If you can avoid it, do not train the discussed models yourself
  - ...it's complicated, time-consuming and expensive
  - Download base (and adapted) models from Huggingface
- For NLP tasks, use transformer models such as BERT or FLAN-T5
- For audio/speech classification tasks regarding speaker properties, use x-vectors
- For general audio/speech classification tasks, use wav2vec2 (or HuBERT)