CIS 6930 Topics in Computing for Data Science
Week 10: More Pre-trained Language Models

Tue 11/9/2021

Yoshihiko (Yoshi) Suhara

2pm-3:20pm

Term Project
Project Presentation & Project Report

Term Project

- Fri 11/12: Feedback & Pre-finalize the project plan
- **Fri 11/18:** Finalize the project title and plan
- Tue 11/23: Project presentations (1)
- Tue 11/30: Project presentations (2)
- **Tue 12/7:** Project report deadline

I'll reach out to each of you (via email) to help finalize the project plan.

Each student must work on their own term project (**No group work**)

Project Report (due Tue 12/7)

The project report must contain

- 1) Motivation
- 2) Clear problem definition + related background
- 3) Methodology: Description of the solution/baselines and the dataset
- 4) Evaluation: Results and Discussion
- 5) Personal reflection: Challenges encountered, lessons learned

The ideal project report should also answer the following questions

- How is your solution different from other solutions (baselines)?
- How better is your solution than the baselines?
- Why is your solution better than the baselines?

Project Presentation (5min + 3min QA)
Tue 11/23 & Tue 11/30

The presentation should contain

- 1) Motivation (30 seconds)
- 2) Clear problem definition (1 minute)
- 3) Methodology (1-2 minutes)
- 4) Evaluation: Results and Discussion (1-2 minutes)

Notes

- Presentations will NOT be evaluated by the performance of your solution
- Students are encouraged to share in-progress work to get feedback from the instructor.
- Please use this opportunity to improve the quality of your final project report.

Any Questions?

Week 10: More Pre-trained Language Models

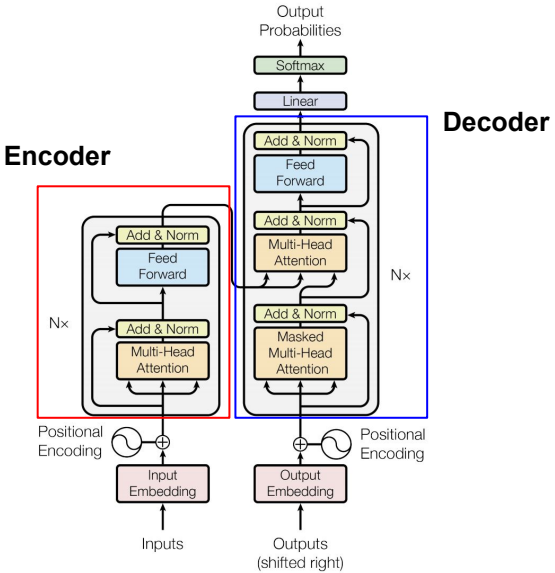
- ~~Week 8: Transformers~~
- ~~Week 9: Pre-trained Language Models~~
- **Week 10: More Pre-trained Language Models**
- Week 11: More Deep Learning Techniques for NLP (Text generation, Text summarization, Information Extraction etc.)
- Week 12: Advanced Techniques and Challenges
- Week 13: Final project presentations

How we ended the last session

Pre-trained Language Models: Categorization

- Most pre-trained LMs can be categorized into the following 3 patterns
  - 1) Transformer **Encoder-only** Models (e.g., BERT, RoBERTa, ALBERT)
  - 2) Transformer **Decoder-only** Models (e.g., GPT-2/3)
  - 3) Transformer **Encoder-Decoder** Models (e.g., BART, T5)

Next →

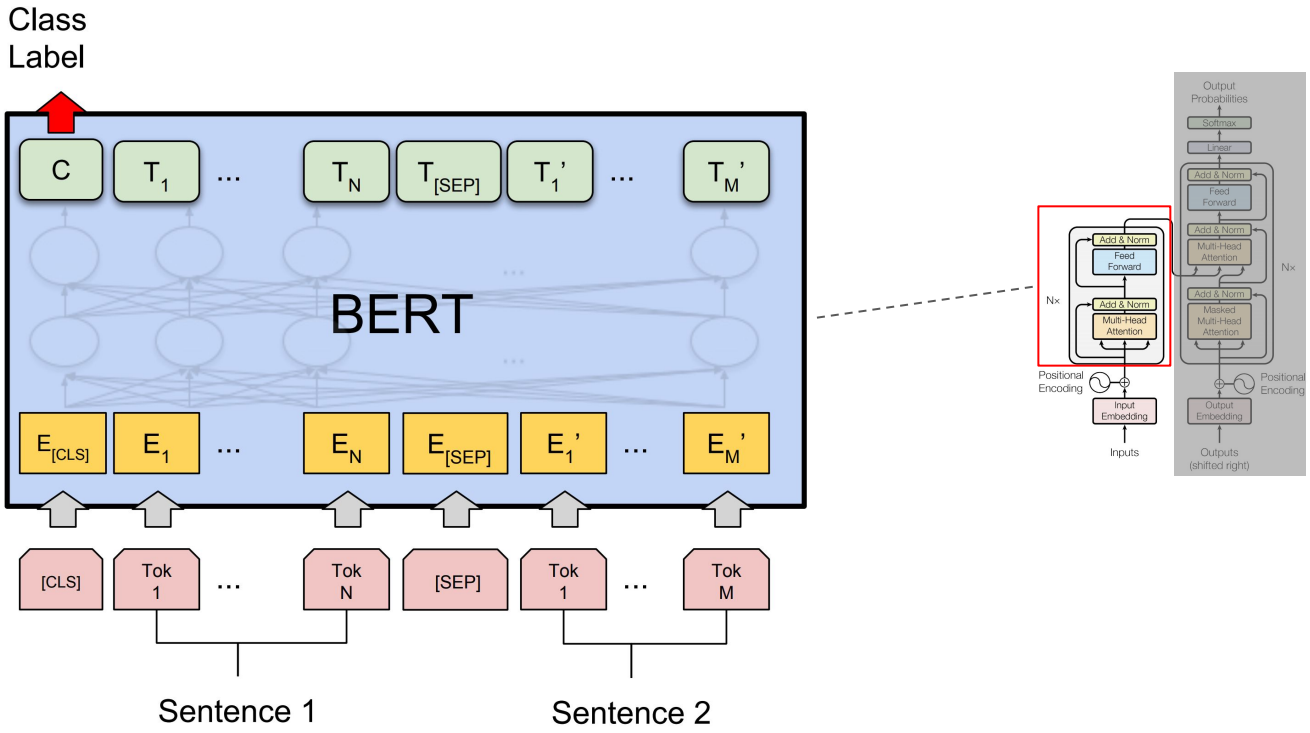


Today’s Agenda

- Other types of Pre-trained Language Models
  - Transformer Decoder-only Models (e.g., GPT-2/3)
  - Transformer Encoder-Decoder Models (e.g., BART, T5)
- Transformers for Computer Vision
  - Vision Transformer
  - Detection Transformer
  - TransGAN

Recap: BERT Basics

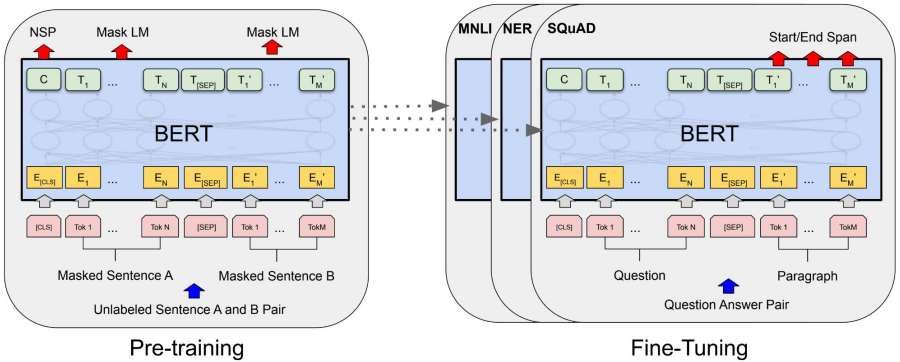
BERT = (pre-trained) Transformer Encoder
(= Stacked Transformer Encoder Blocks)



What is BERT? Why BERT?

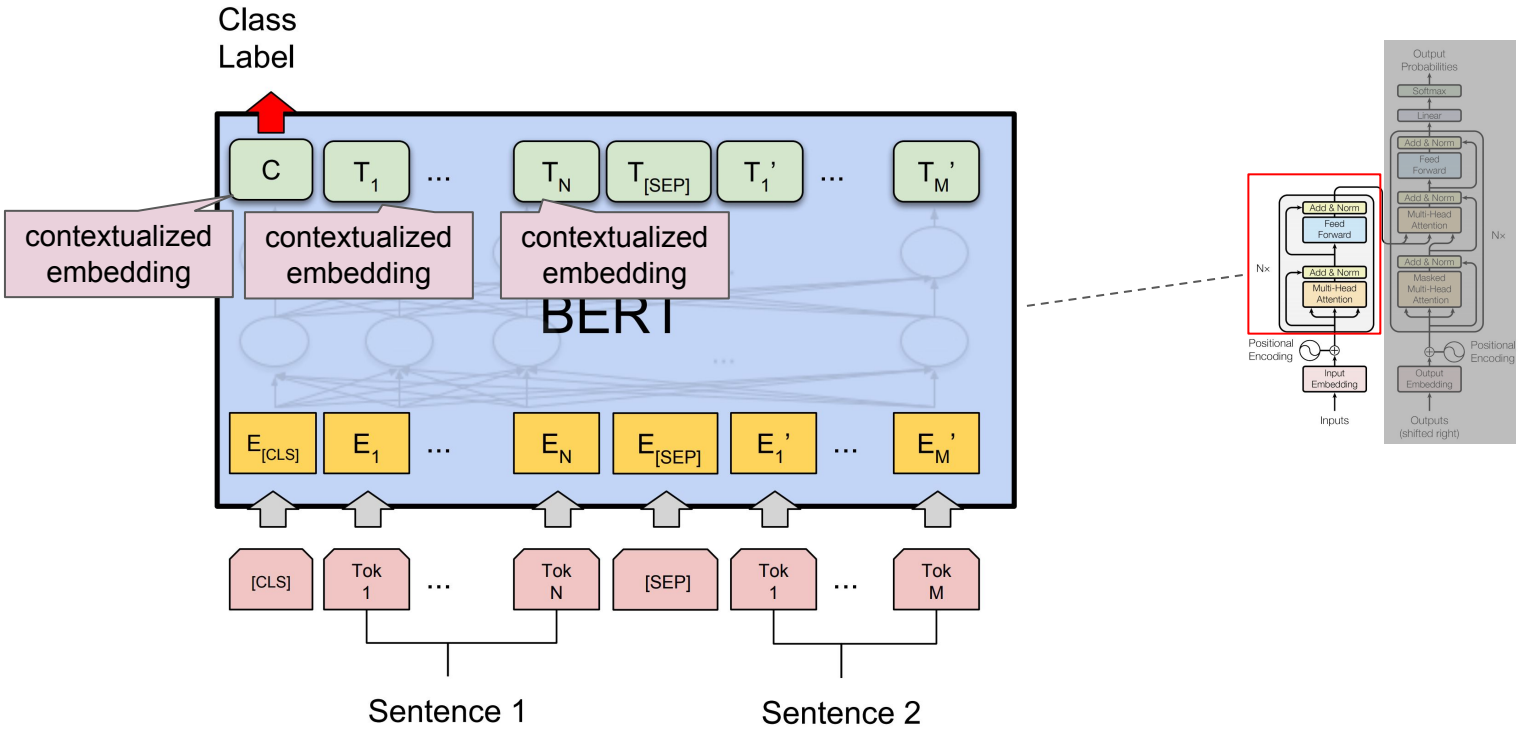
- **BERT: Bidirectional Encoder Representations from Transformers**
  - The most popular pre-trained language model in NLP

- A pre-trained **Transformer Encoder** model, which can be **fine-tuned** for a variety of NLP tasks



Key Point: Self-Attention

- Any input tokens attend **all input tokens** (i.e., “full” attention)



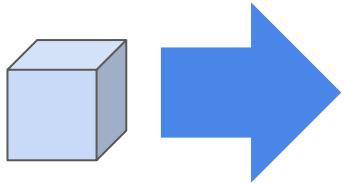
Pre-training & Fine-tuning: Intuition

Pre-training
(Basic training)

Pre-training Task 1

Pre-training Task 2

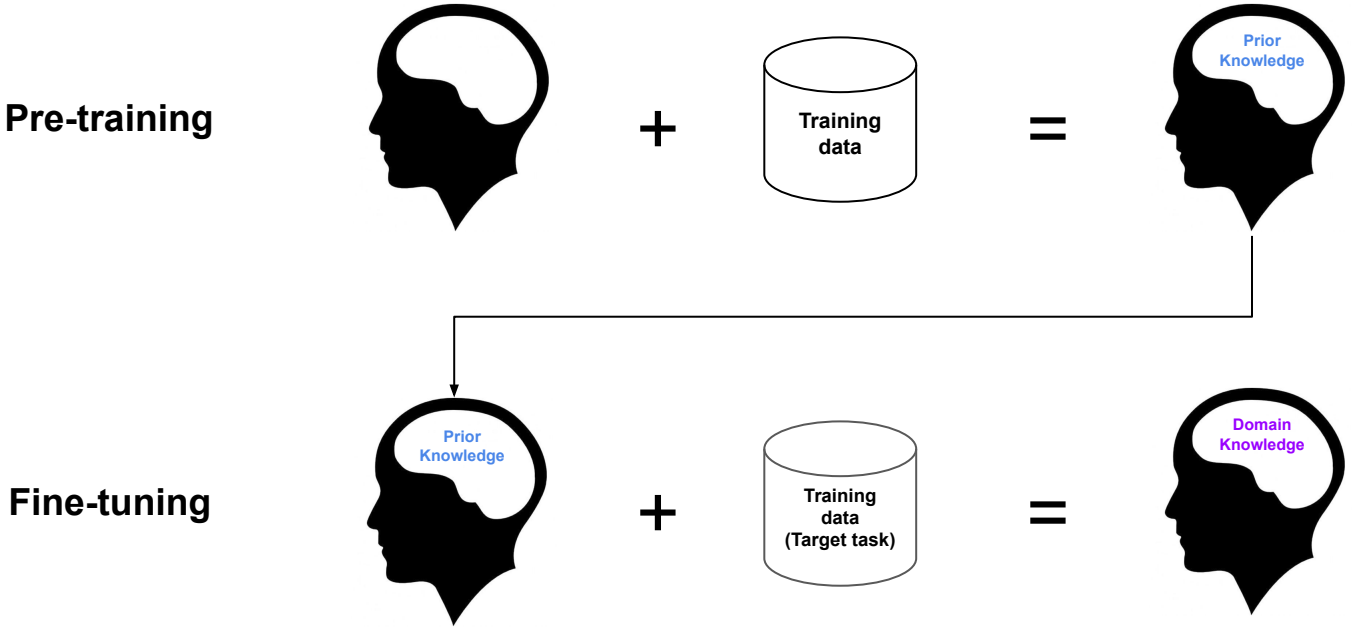
Fine-tuning
(Skill training)



(e.g., BERT)

Downstream tasks
(Sports)

Pre-training & Fine-tuning Framework



Pre-training = (A) Pre-training Method(s) + (B) Corpus

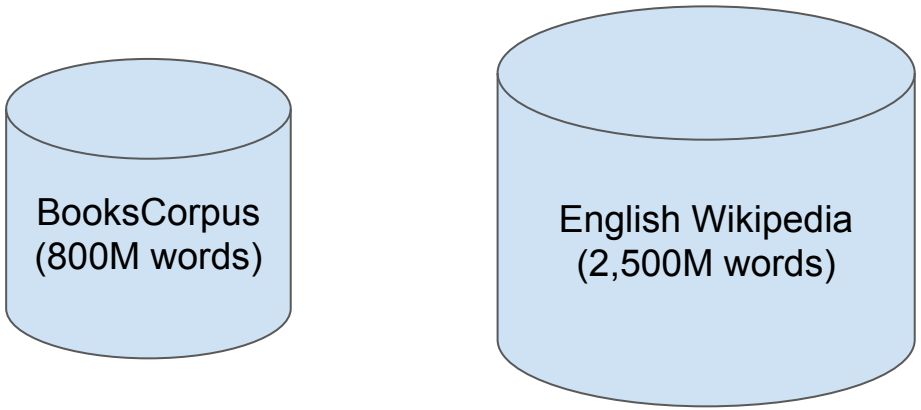
- (A) **How to learn**: Pre-training method(s)
- (B) **What to learn from**: Textual corpora

Pre-training

BERT Pre-training

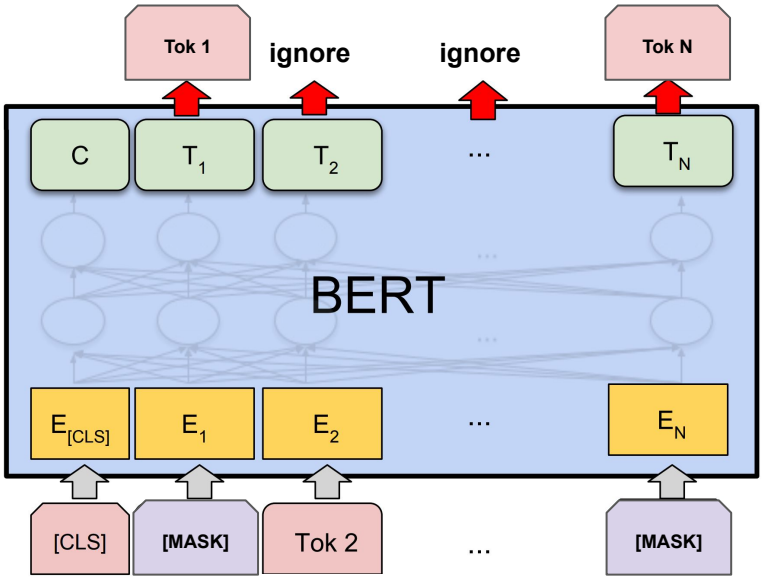
- Pre-training Methods
  - 1) Masked Language Model
  - 2) Next Sentence Prediction

- Pre-training corpus



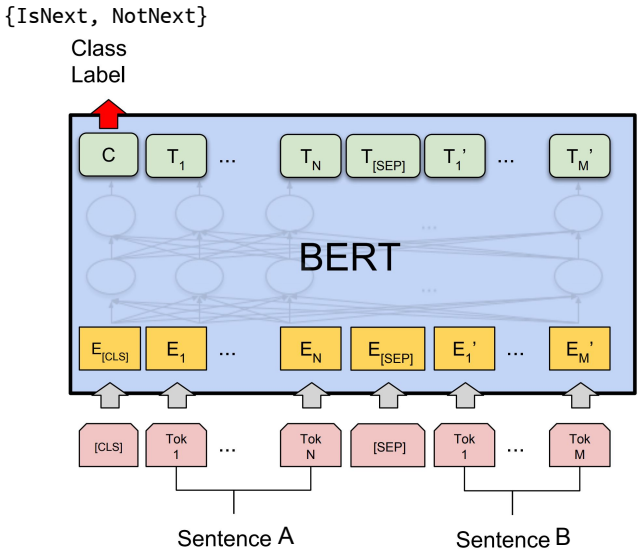
Masked Language Model (MLM)

- Step 1: Replace **randomly selected tokens** with **[MASK] tokens**
- Step 2: Train the model to **reconstruct the original token**



Next Sentence Prediction (NSP)

- Step 1) Sample sentences A and B such that
  - 50%: B is **actual next sentence** that follows A
  - 50%: random sentence from the corpus
- Step 2) Train the model to predict the correct label



Tokenizers for BERT (Pre-trained LMs): Takeaway

- A subword tokenizer (i.e., No out-of-vocabulary issue)
- [important!] Need to be “trained” on text data

  → Pre-trained language model & **pre-trained** tokenizer model are coupled

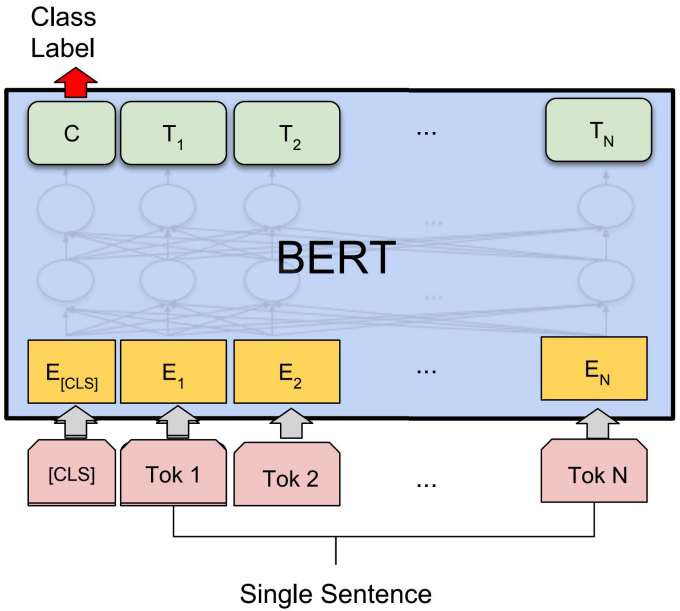
```
>>> from transformers import AutoTokenizer, AutoModel

>>> tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
>>> model = AutoModel.from_pretrained("bert-base-uncased")

>>> inputs = tokenizer("Hello world!", return_tensors="pt")
>>> outputs = model(**inputs)
```

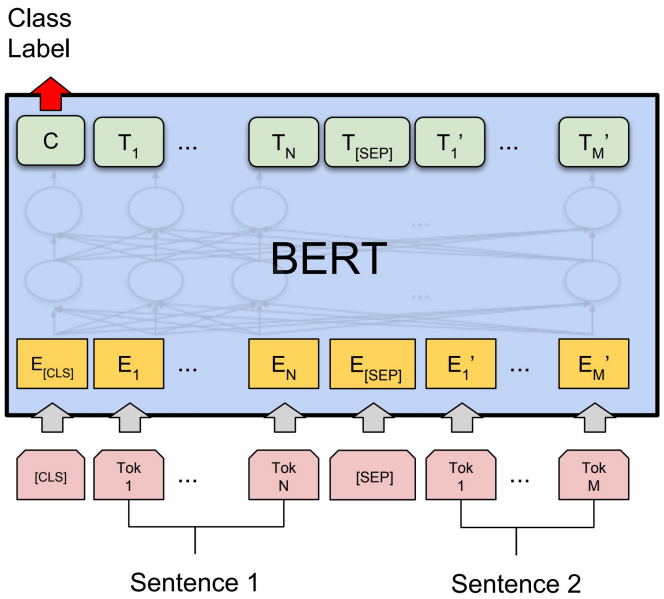
Single-sentence Classification

- Text classification tasks



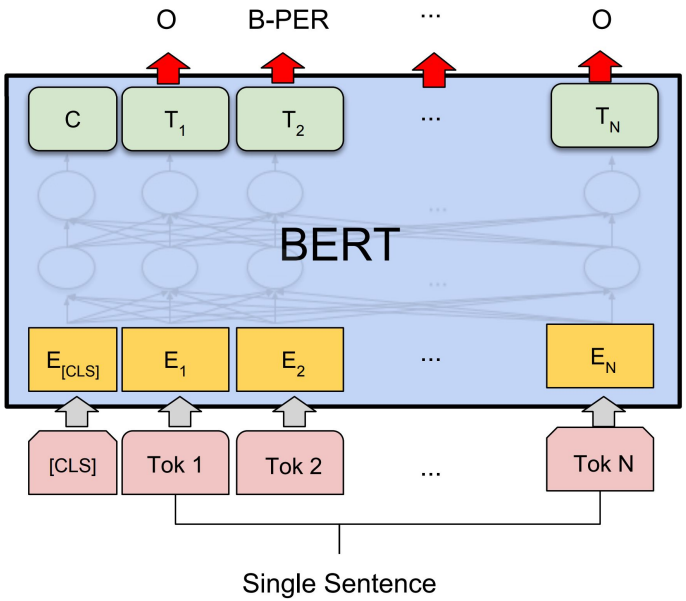
Sentence-pair Classification

- Textual entailment recognition and duplicate question detection tasks



Token Classification

- Sequential tagging problems (e.g., Named Entity Recognition)



Hugging Face's Transformers Library

https://github.com/huggingface/transformers

- You can use a variety of pre-trained language models just with a few lines of Python code

```
>>> from transformers import AutoTokenizer, AutoModel

>>> tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
>>> model = AutoModel.from_pretrained("bert-base-uncased")

>>> inputs = tokenizer("Hello world!", return_tensors="pt")
>>> outputs = model(**inputs)
```

Pre-trained Language Models Need Fine-tuning

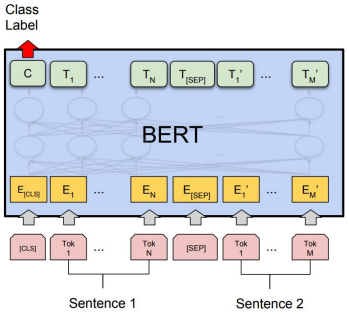
❗

- to be used for downstream tasks

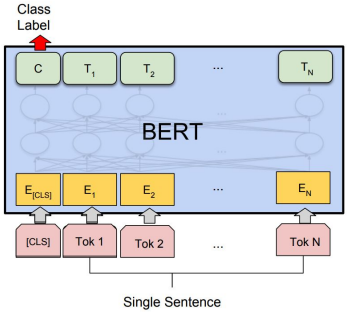
- Pre-training methods (e.g., Masked Language Models, Next Sentence Prediction) **do NOT directly teach** to solve sentiment classification tasks
  - Pre-training methods are still helpful, though
  - Imagine a sport player, who's physically very strong 💪 but never had any skill-training

Questions?

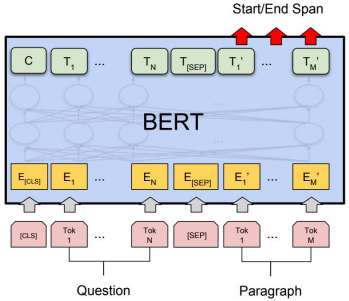
What are the limitations of BERT?



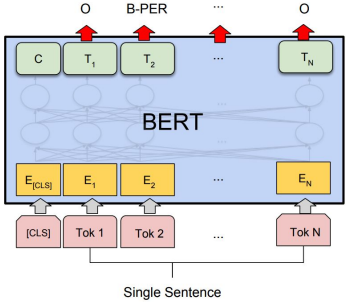
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



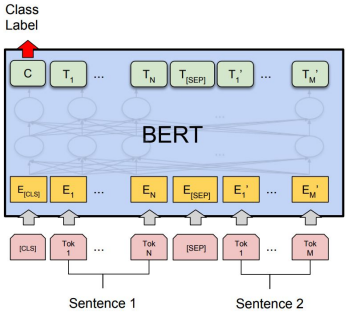
(c) Question Answering Tasks: SQuAD v1.1



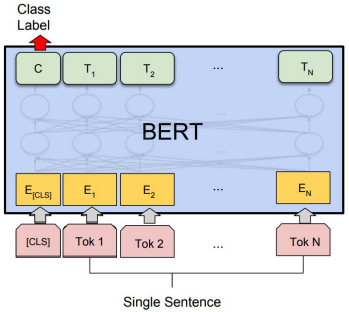
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

It looks versatile. Really? 🤔

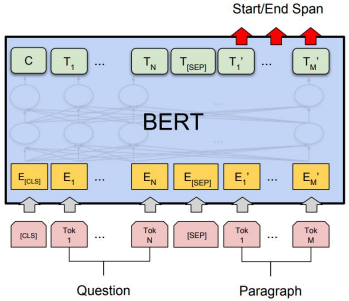
What are the limitations of BERT?



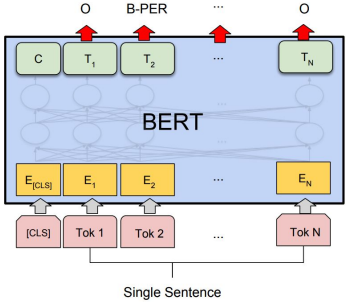
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1



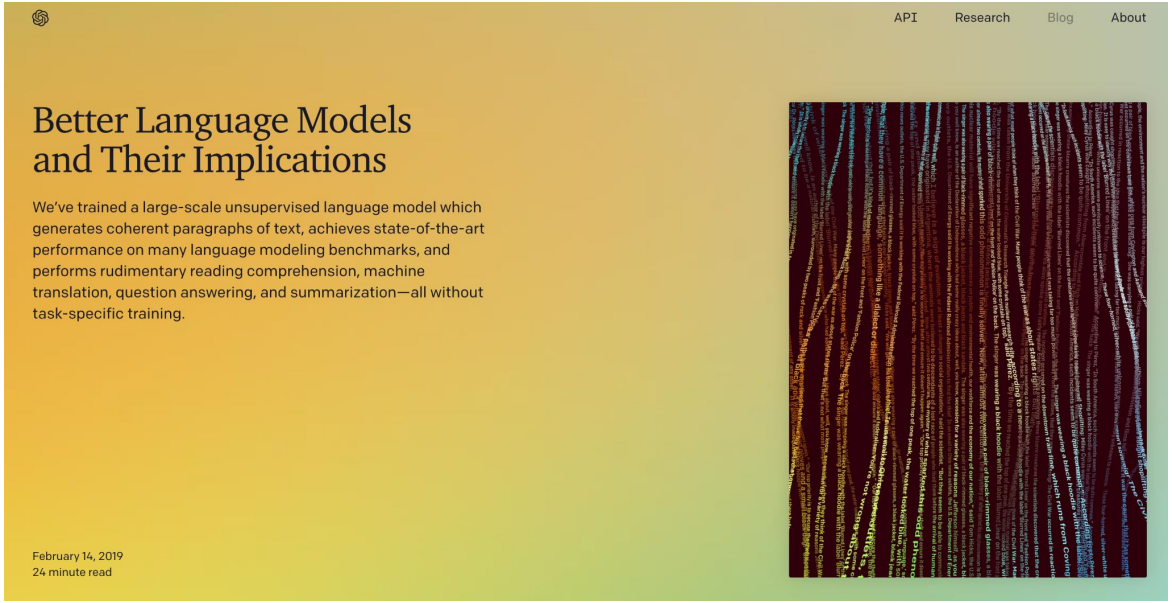
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

It looks versatile. Really? 🤔

Can it be used for text generation?

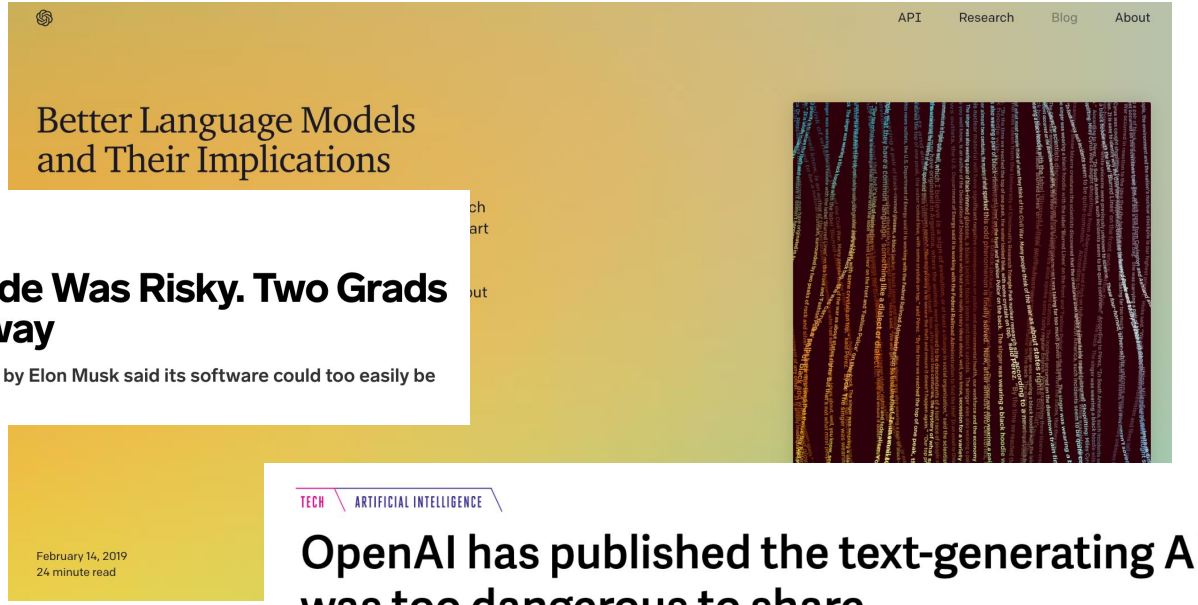
GPT-2

https://openai.com/blog/better-language-models/



GPT-2

https://openai.com/blog/better-language-models/

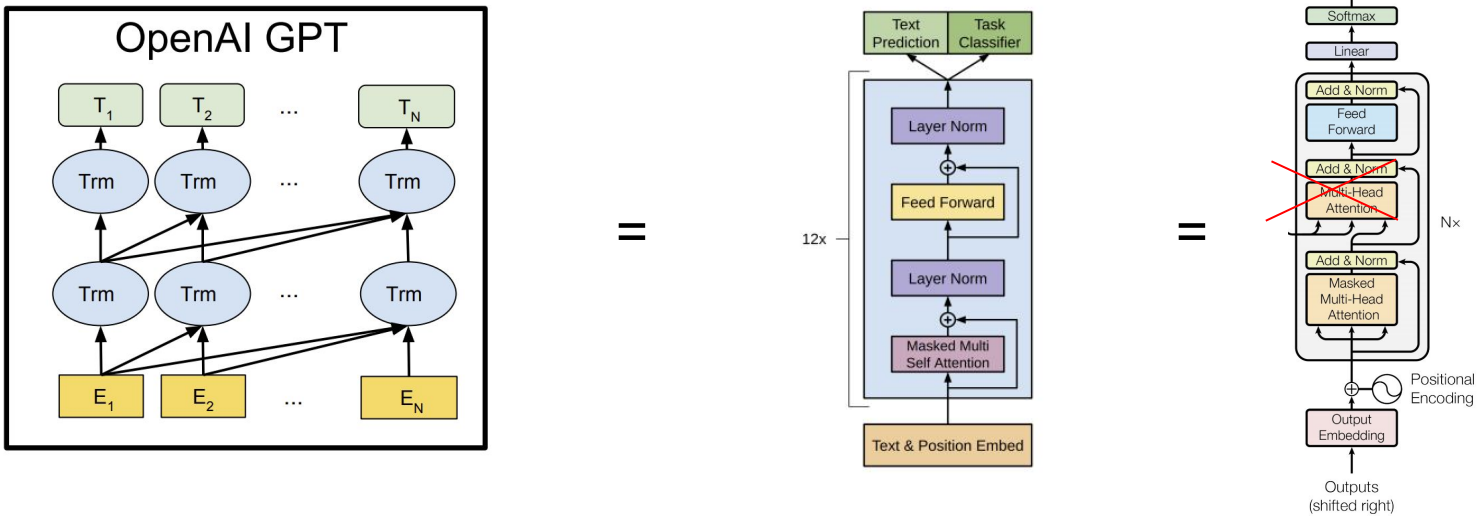


GPT History

- 2018: GPT (117M parameters)
- 2019: GPT-2 (1.5B parameters)
- 2020: GPT-3 (175B parameters)
- Coming soon?: GPT-4
  - https://towardsdatascience.com/gpt-4-will-have-100-trillion-parameters-500x-the-size-of-gpt-3-582b98d82253
  -

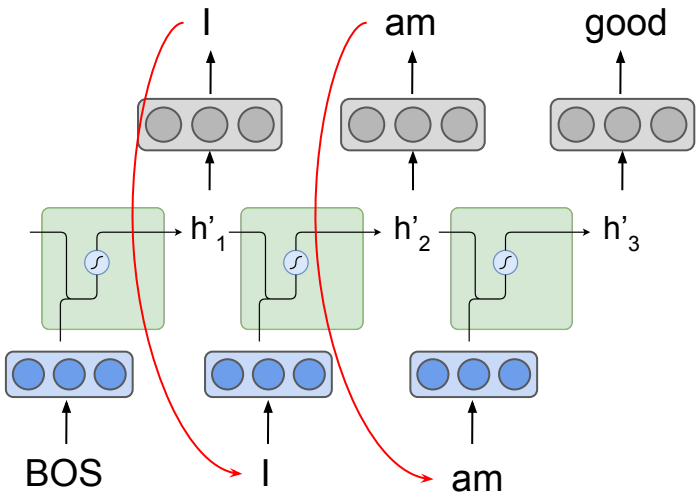
OpenAI GPT (Generative Pre-trained Transformer)

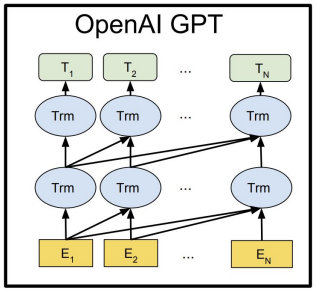
- GPT = Stacked **Transformer Decoder-only** Model
  - Autoregressive model

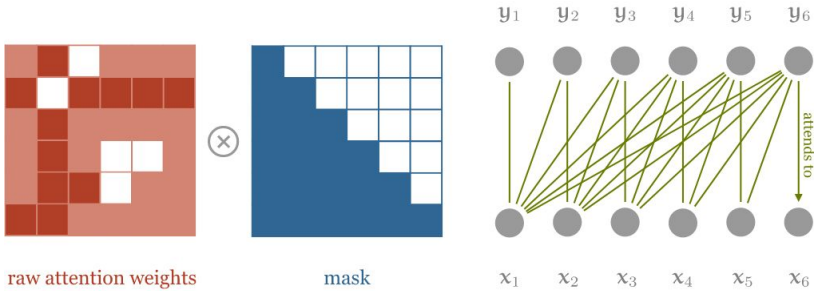


Technically, it does not use Encoder-Decoder Attention

Autoregressive Model





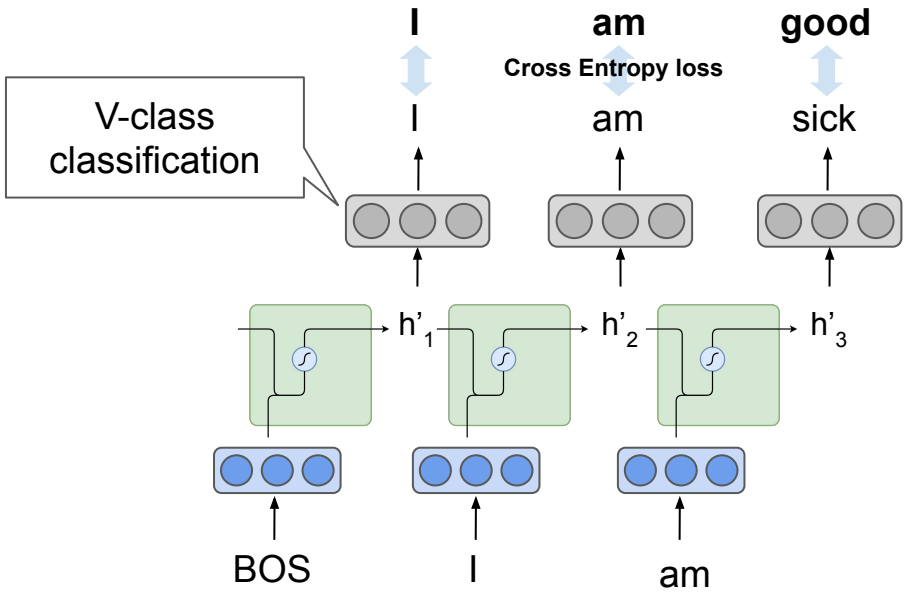


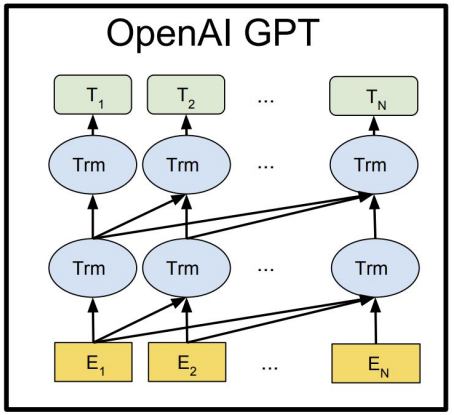
Decoder

Pre-training Corpus: BooksCorpus

- 7,000 books in various genres

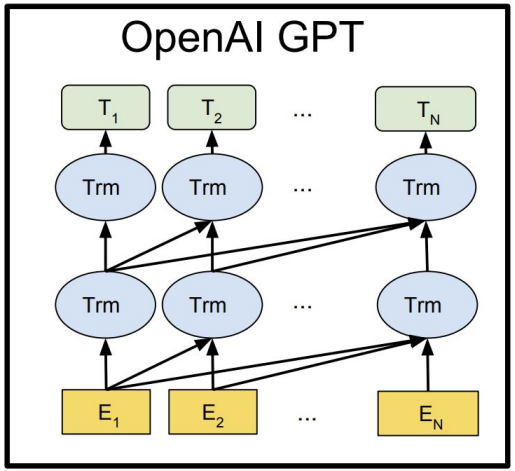
Pre-training: Causal Language Model = Teacher forcing



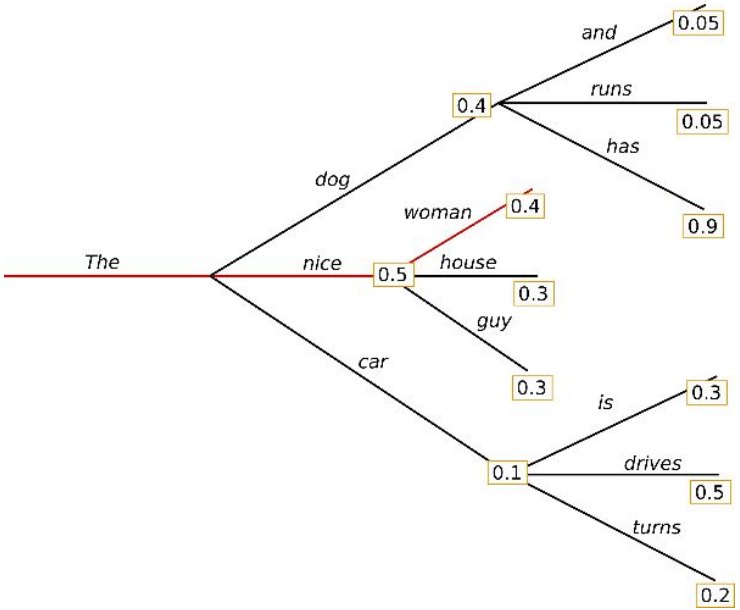


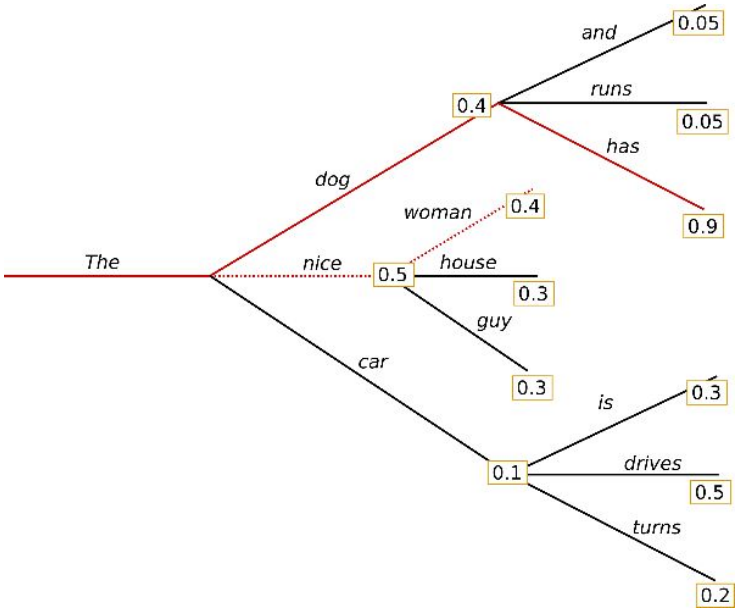
Can we Fine-tune GPT-2?

- Yes, do the same thing as Causal Language Model
  - 1) Give **input sequence** as prefix (i.e., not used for loss calculation)
  - 2) Train the model with teacher forcing using the **target sequence**



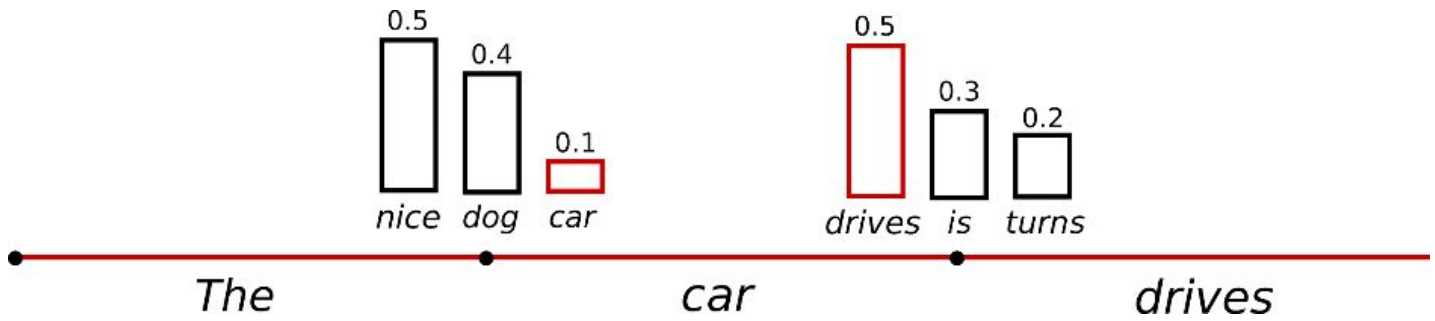
Decoding Algorithm: Greedy Search & Beam Search





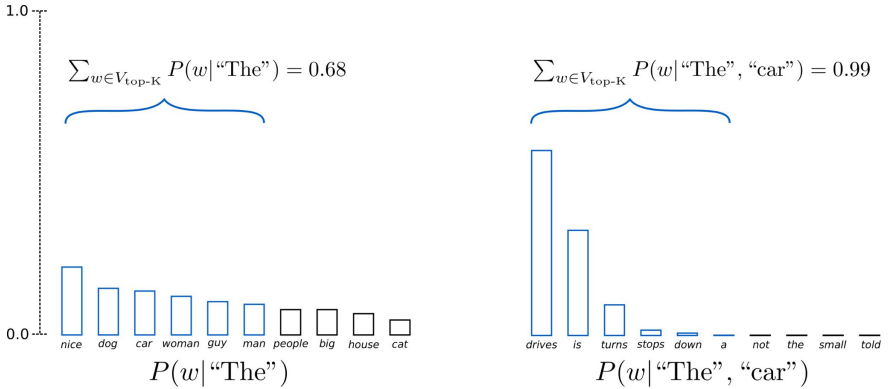
Yet Another Decoding Algorithm: Sampling

- At each step, **randomly choose a token** according to the token probability distribution
  - Note: we need to fix the random seed for reproducibility

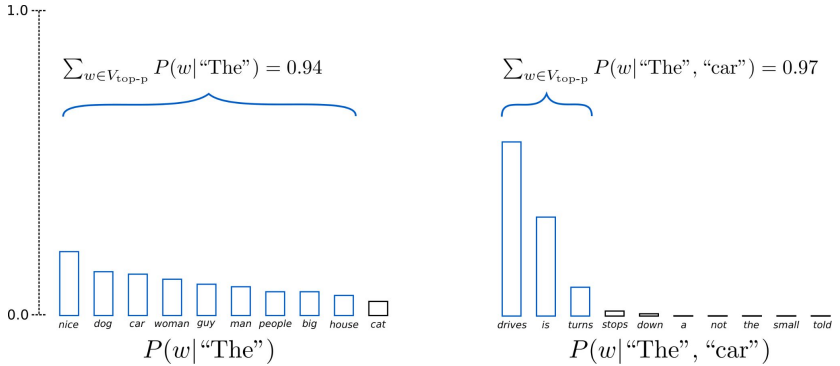


Top-k & Top-p (Nucleus) Sampling

- Use the top-k (the cumulative probability) to determine the token candidates



Top-k sampling



Top-p sampling

N-gram Blocking

- The technique excludes any sequence that contains the same N-gram tokens from the candidate sequence

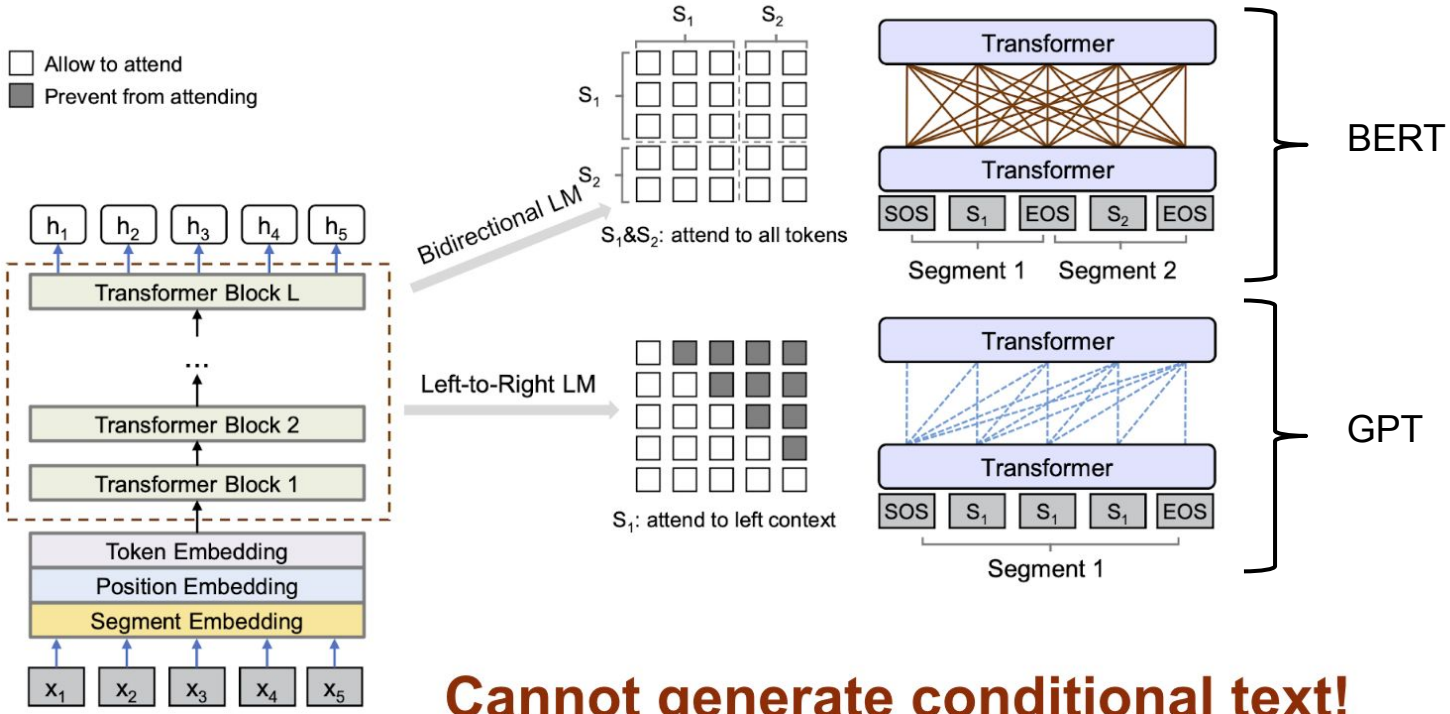
- e.g., 2-gram blocking
  - The dog and the [ ]
    - ~~dog (0.55)~~
    - cat (0.30)
    - boy (0.10)
    - ...

2 or 3-gram blocking are commonly used to avoid the repetition issue

Hands-on: Playing with GPT-2 & Decoding algorithms

- Google Colab

UniLM: Limitations of BERT & GPT



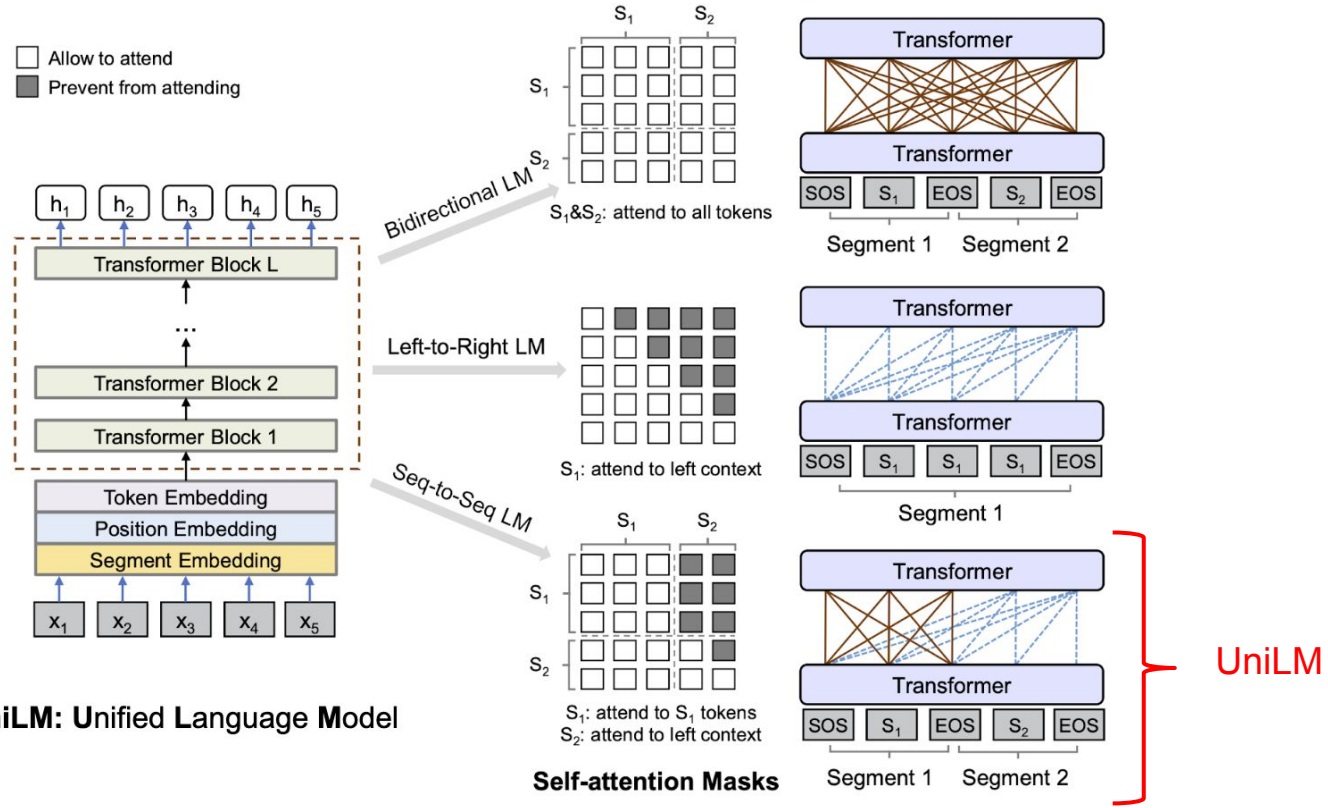
Cannot generate conditional text!

GPT: Generative Pre-Training

https://docs.google.com/presentation/d/1FWqQDqi1edA3ivobQk93gq_NBEITKq8WzG1RIJzEDJM/edit#slide=id.g59b5860c59_0_564

UniLM: Unified Language Model

UniLM = BERT + GPT

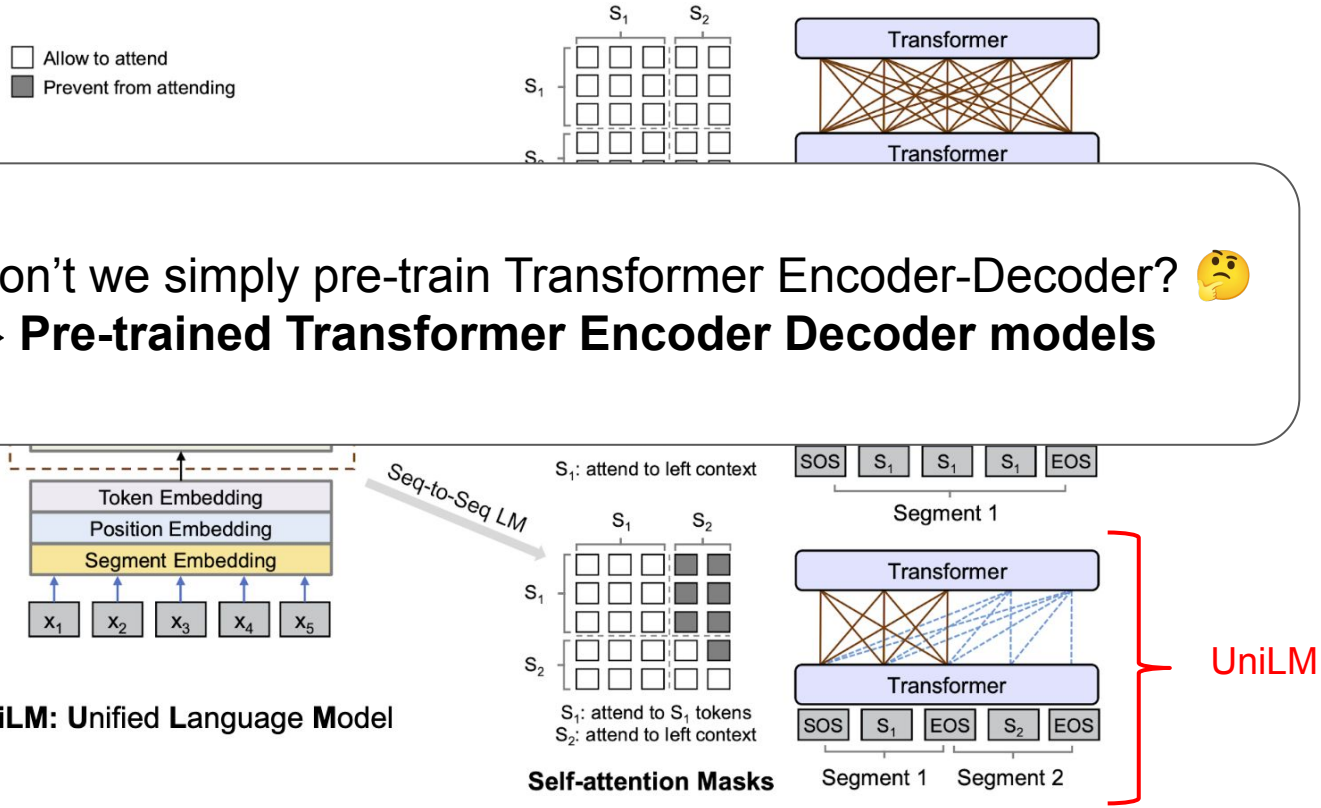


UniLM: Unified Language Model

https://docs.google.com/presentation/d/1FWqQDqi1edA3ivobQk93gq_NBEITKq8WzG1RIJzEDJM/edit#slide=id.g59b5860c59_0_564

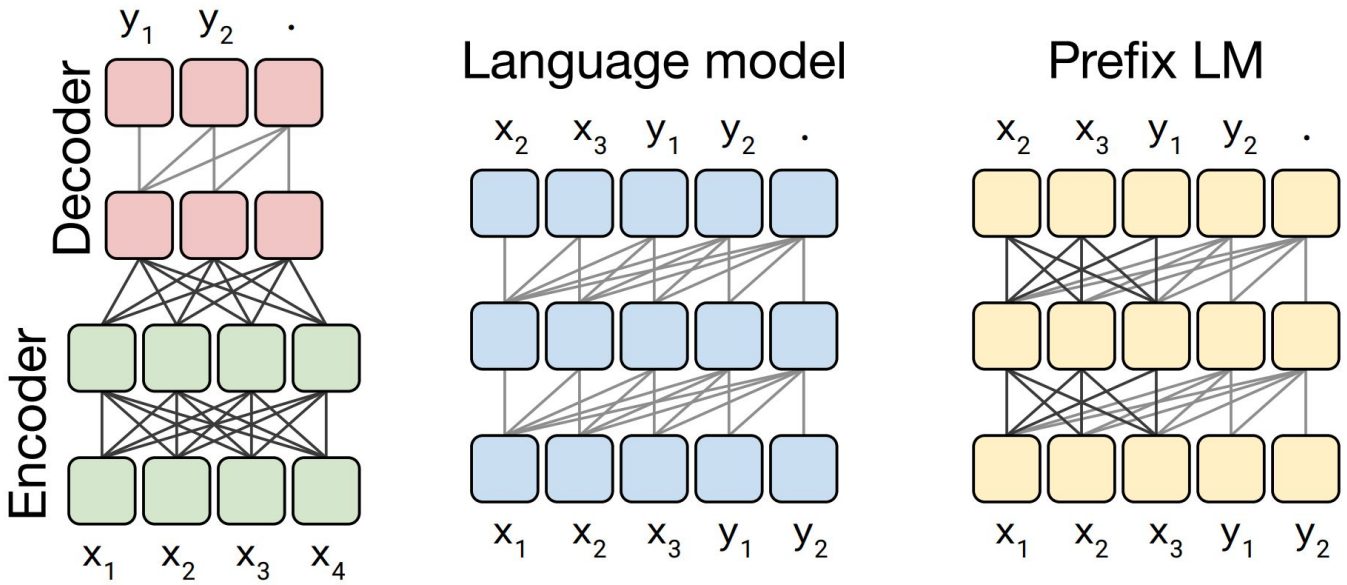
UniLM: Unified Language Model

UniLM = BERT + GPT



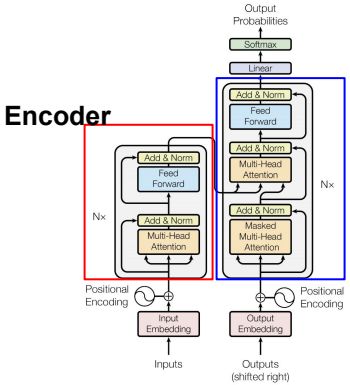
https://docs.google.com/presentation/d/1FWqQDqi1edA3ivobQk93gq_NBEITKq8WzG1RIJzEDJM/edit#slide=id.g59b5860c59_0_564

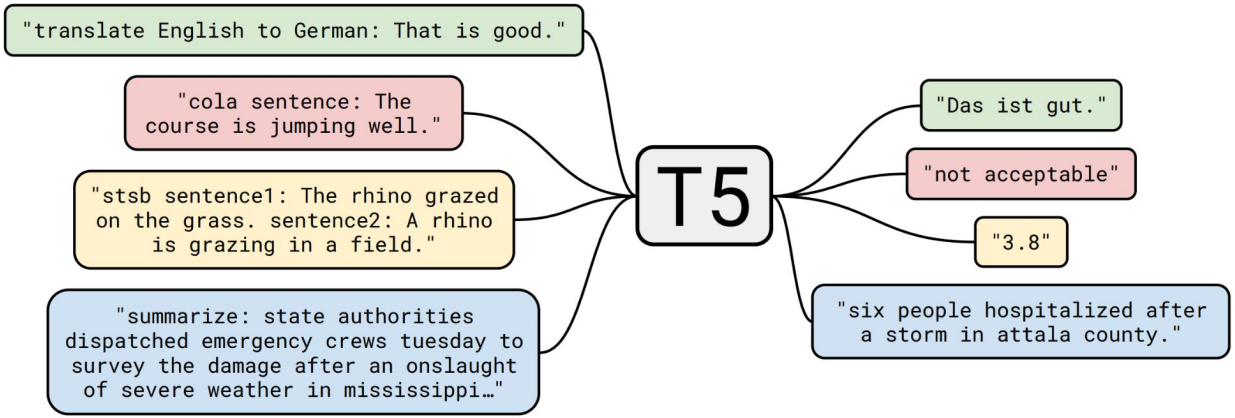
Encoder-Decoder vs Prefix LM



Text-to-Text Transfer Transformer (T5)

- Transformer Encoder-Decoder





T5's "Multi-task" Pre-training

- The Colossal Clean Crawled Corpus (C4): **20TB** web corpus
- Multiple pre-training methods

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

More is Better … ? No

- We should let the model “read good books”

Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
Wikipedia	16GB	81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
Wikipedia + TBC	20GB	83.65	**19.28**	**82.08**	**73.24**	**26.77**	39.63	**27.57**

T5 Models

- Small (60M): d=512, L=6, H=8
- Base (220M): d=1024, L=12, H=16
- Large (770M): d=1024, L=24, H=16

Can you explain how the model designs look like?

A List of Pre-trained Encoder-Decoder Models

- T5 by Google
- BART by Facebook
- MASS by Microsoft
- PEGASUS by Google
- MARGE by Facebook
- OPTIMUS by Microsoft
- ...

They are *all* Transformer Encoder-Decoder Models and can be used in the same manner

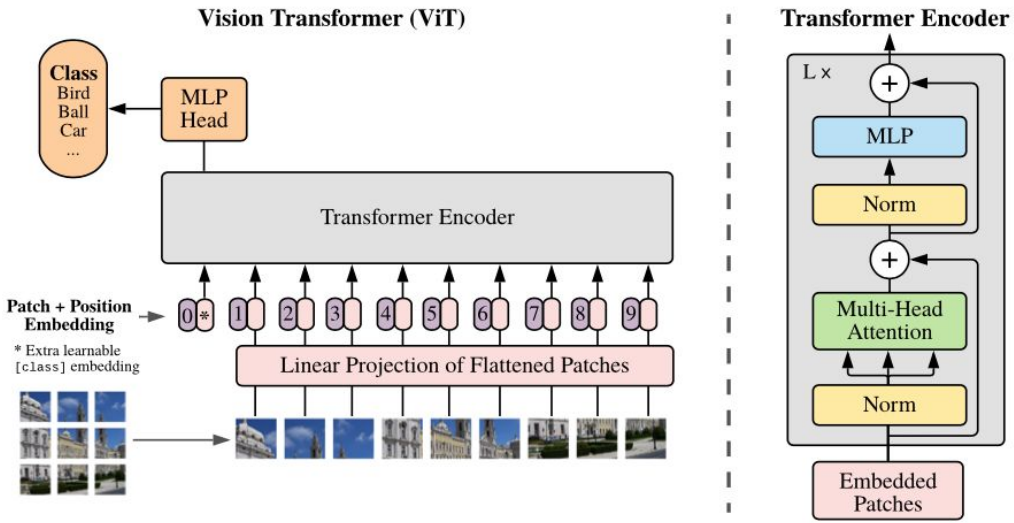
Hugging Face Model Hub

- https://huggingface.co/models
- Some models are **just pre-trained**. Some are **also fine-tuned**

Transformers for Computer Vision

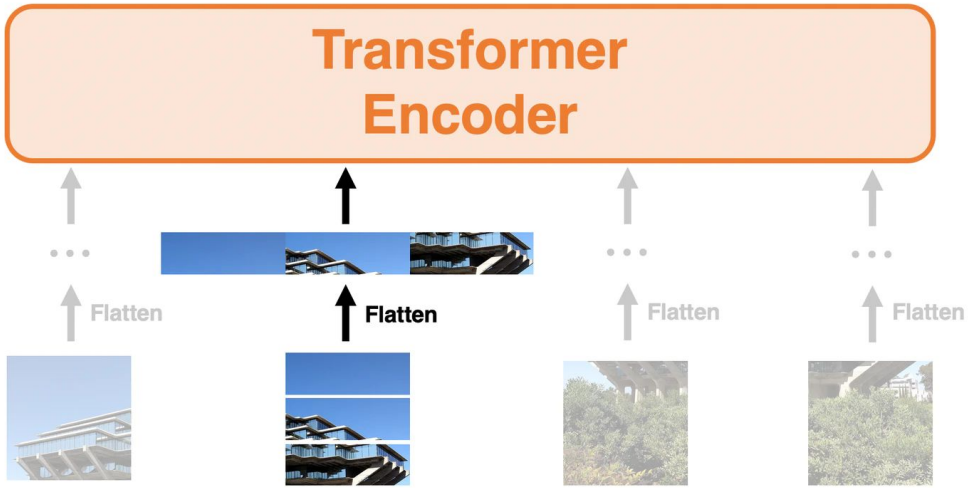
Vision Transformer (ViT) [Dosovitskiy et al. 2021]

- RGB Image → N Patches → Flattening patches
  - Each patch = (3 * P^2)-dimensional vector



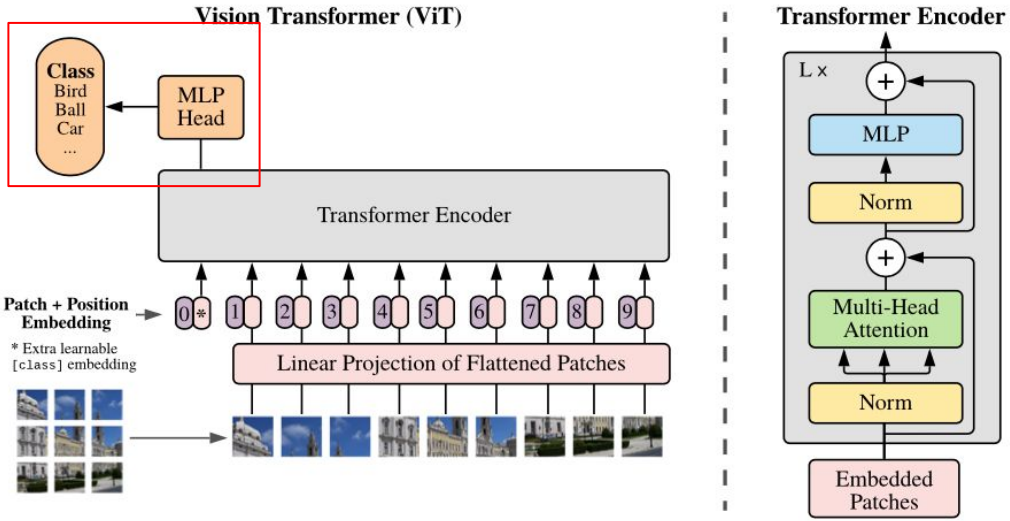
A. Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, ICLR ‘21

Input to Vision Transformer



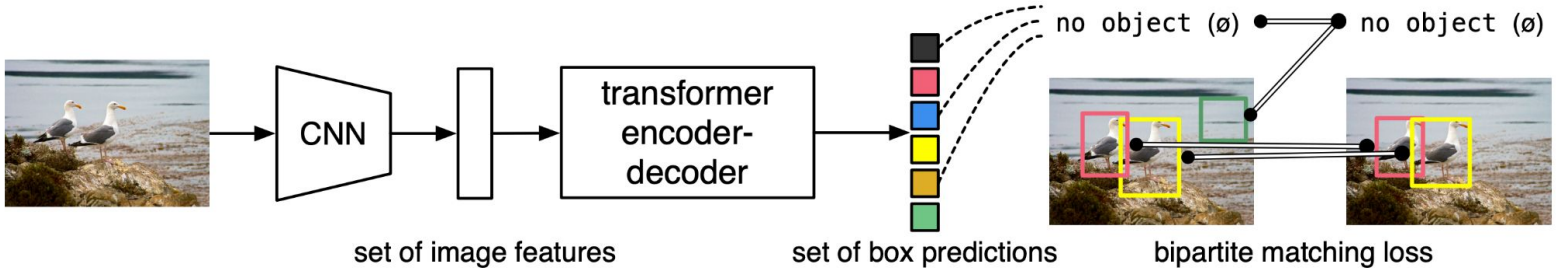
Vision Transformer Pre-training

- **“Supervised”** pre-training
- on the JFT-300M private image dataset
  - 18k image categories (!)



Detection Transformer (DETR) [Carion et al. 2020]

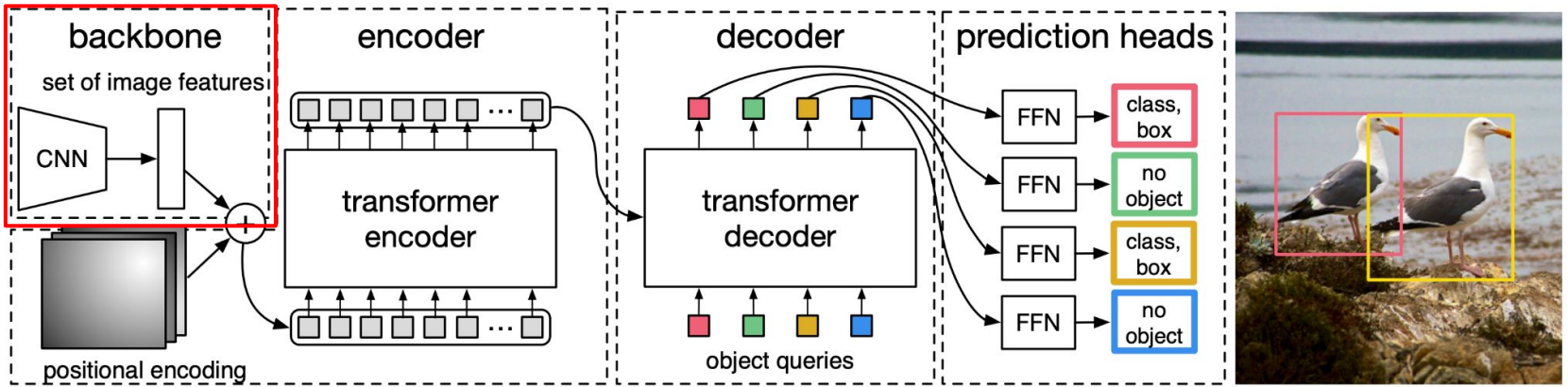
- CNN + Transformer for object detection



N. Carion et al. “End-to-End Object Detection with Transformers”, arXiv 2020.

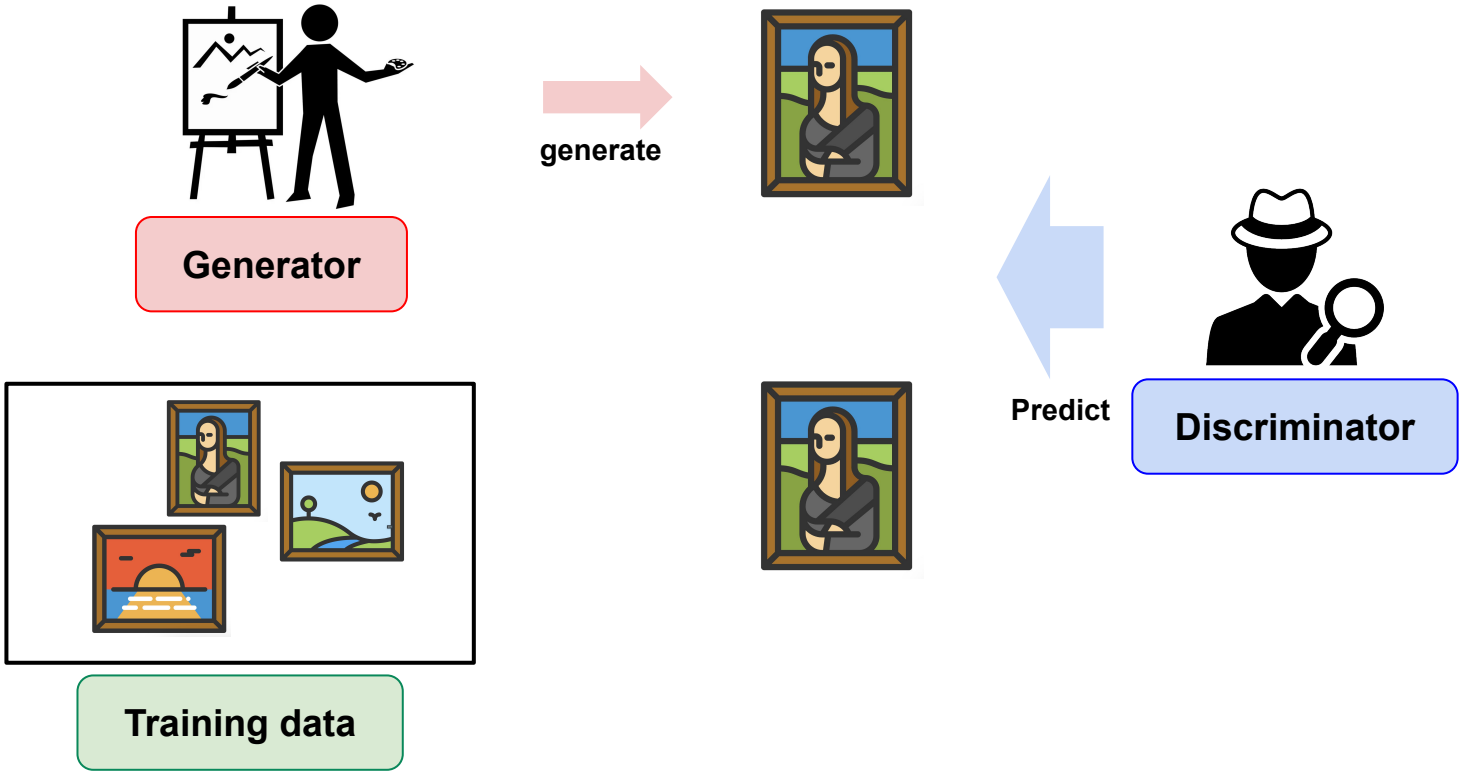
Detection Transformer (DETR) [Carion et al. 2020]

- CNN + Transformer for object detection

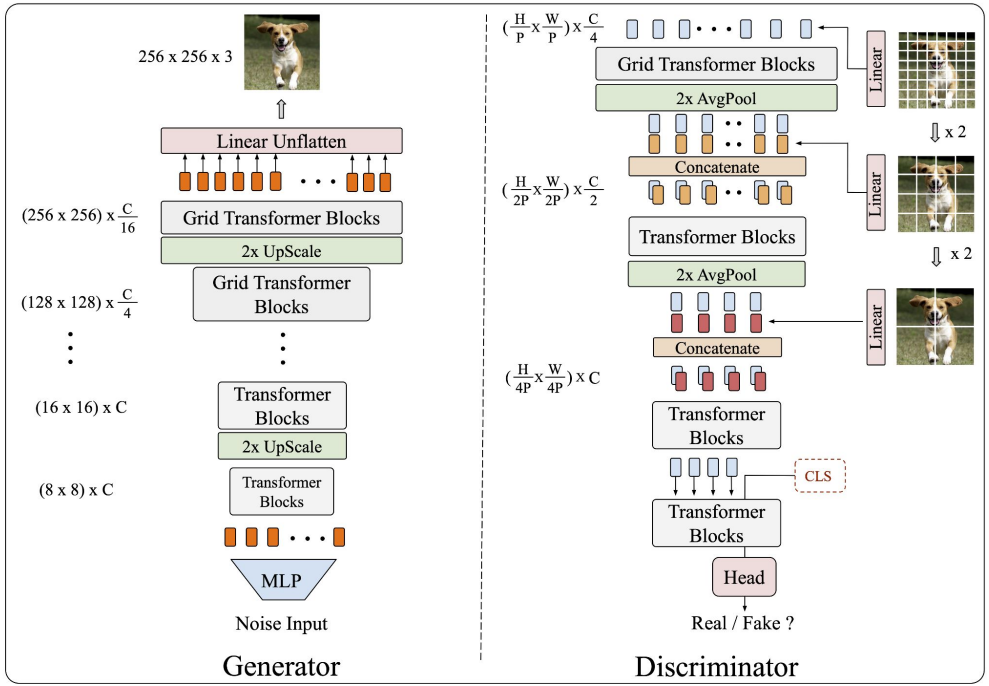


N. Carion et al. “End-to-End Object Detection with Transformers”, arXiv 2020.

Recap: GAN Framework



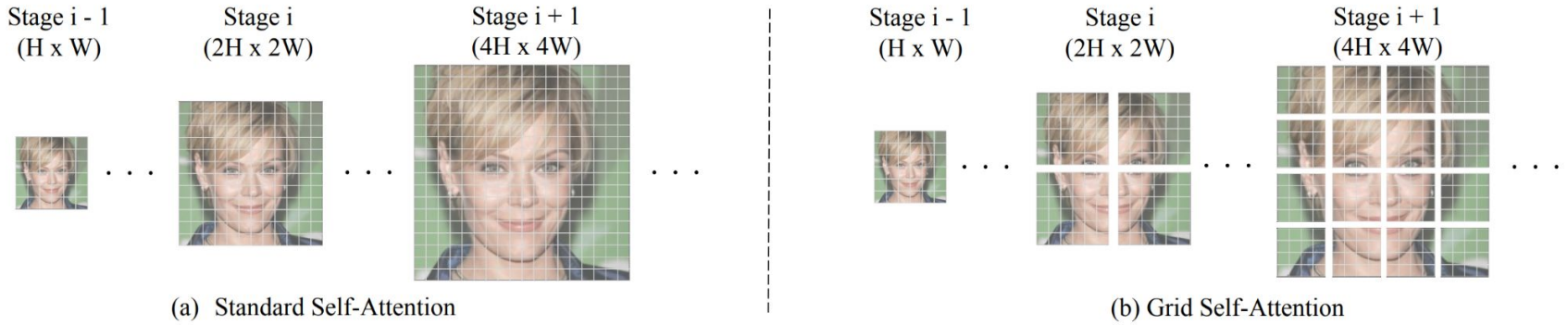
TransGAN [Jian et al. 2021]



Yifan Jiang, Shiyu Chang, Zhangyang Wang, “TransGAN: Two Pure Transformers Can Make One Strong GAN, and That Can Scale Up”, arXiv 2021.

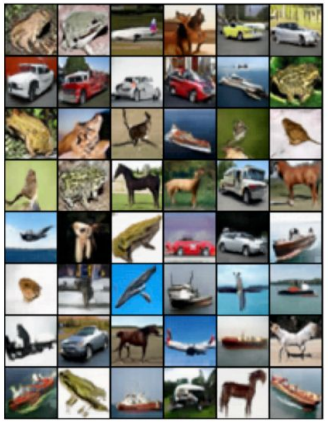
Grid Self-Attention

- Introducing a structure to the self-attention mechanism

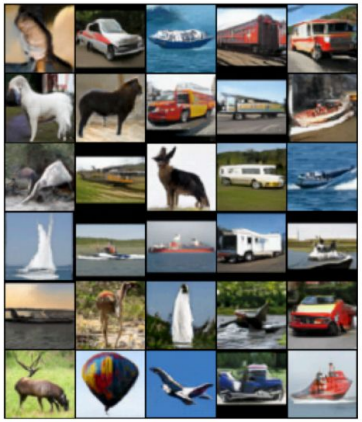


TransGAN Outputs

CIFAR-10
32 x 32

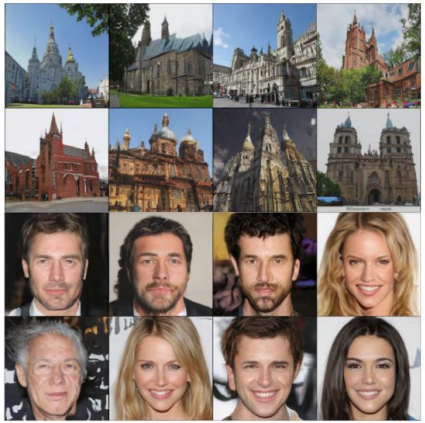


STL-10
48 x 48



CelebA
128 x 128

CelebA-HQ & Church
256 x 256



Should We Forget About CNNs?

- **No.** Always keep alternative solutions in your hands

ResNet strikes back: An improved training procedure in timm

Ross Wightman° Hugo Touvron*,† Hervé Jégou*

°Independent researcher *Facebook AI †Sorbonne University

Summary

- Other types of Pre-trained Language Models
  - Transformer Decoder-only Models (e.g., GPT-2/3)
  - Transformer Encoder-Decoder Models (e.g., BART, T5)
- Transformers for Computer Vision
  - Vision Transformer
  - Detection Transformer
  - TransGAN

No Class on Thu 11/11 & 2x Sessions Next Week (!)

- Tue 11/16 **(2x 80min)**
- Thu 11/18 **(2x 80min)** (* Finalize the term project title & plan!)
- Tue 11/23 Project presentation (1)
- Thu 11/30 Project presentation (2) (* Final session!)