

01. BERT

Why was BERT way ahead of its time?

-
-
-

Because it was a masked language model even during pre-covid days!



CS 1671 / CS 2071

Human Language Technologies

Session 20: BERT

Michael Miller Yoder

October 29, 2026

Assessments: project deliverable 2

- [Project deliverable 2](#) is due **tonight, Thu Oct 29 at 11:59pm**
- Part 1: Basic data analysis (if any updates are required from the proposal)
- Part 2: Result from baseline approach
 - Ideally performance metric result from the baseline system you proposed
- Part 3: Advanced system (LLM) proposal
 - How might you use an LLM programmatically to attempt your task?
 - Zero-shot and at least 1 more advanced approach

Assessments: exam

- There will be an exam in class on **Tue Nov 10**
- Covers everything in the class up to that point
- In-class, paper exam
- Closed-book except for 1 letter page of double-sided notes
- Conceptual
- No programming
- Class next Thu Nov 5 will be a review session for the exam
 - Final in-class quiz moved to Nov 5 since Nov 3 is **Election Day**

Assessments: homework

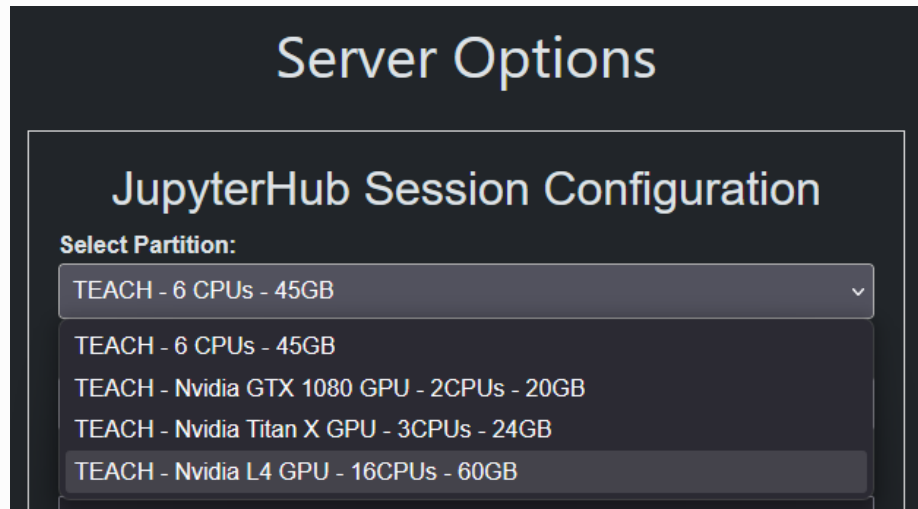
- [Homework 3](#) has been released and is **due Nov 17**
- Run Jupyter notebooks from templates on the CRCD
- Part 1: LLM prompting
- Part 2: Instruction tuning of an LLM
 - CRCD GPUs

Overview: BERT

- Coding activity from last time: finetune GPT-2 on Shakespeare
- BERT and masked language modeling
- Encoder-decoder models: T5
- Finetuning BERT for classification
- Coding activity: finetuning BERT for text classification

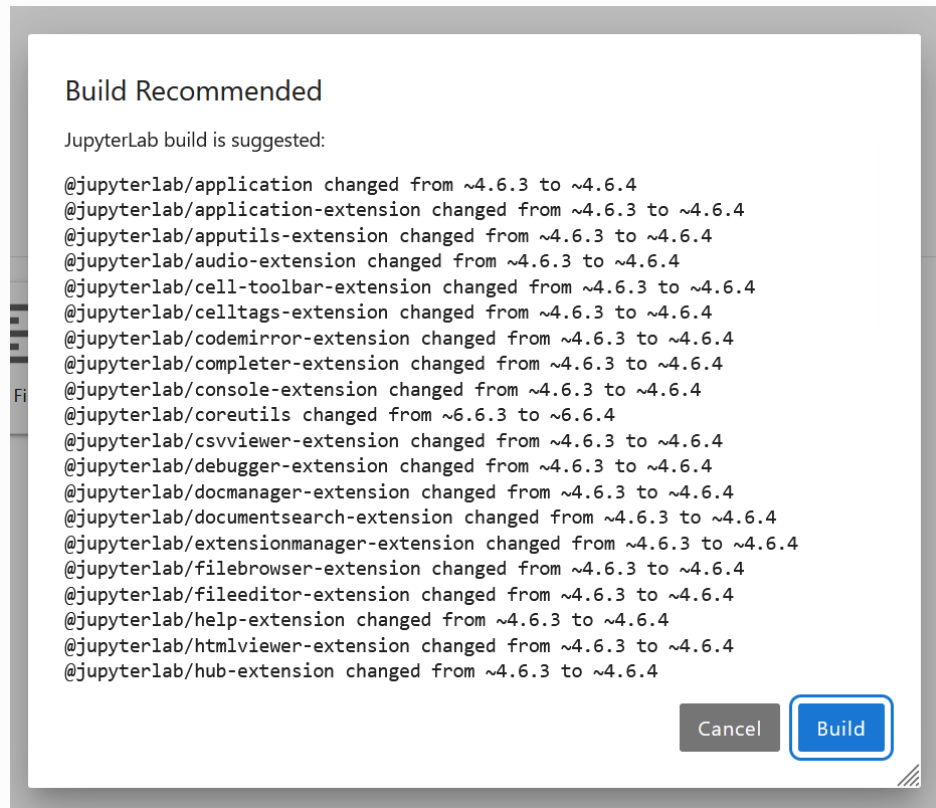
Load notebook: finetune GPT-2 on Shakespeare

1. Click this [nbgitpuller link](#) (also available on course website)
2. **Important difference from normal:** Start a server with 'TEACH – Nvidia L4 GPU – 16 CPUs – 60GB' server
3. Load **new** custom environment at `/vast/cs1671-2026f/class-env/.pixi/envs/default`



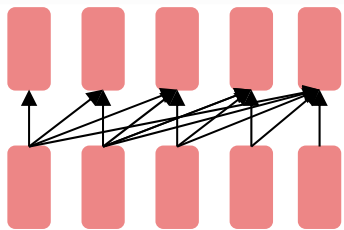
Load notebook: finetune GPT-2 on Shakespeare

4. Open `session19_gpt2_shakespeare.ipynb`
5. Push **Cancel** on Build Recommended window
6. Run the Setup cells



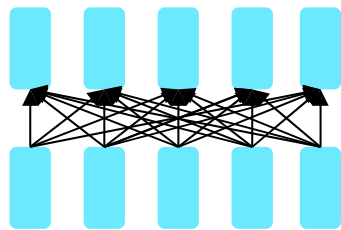
Review: Describe encoder, decoder, and encoder-decoder architectures

Three architectures for large language models



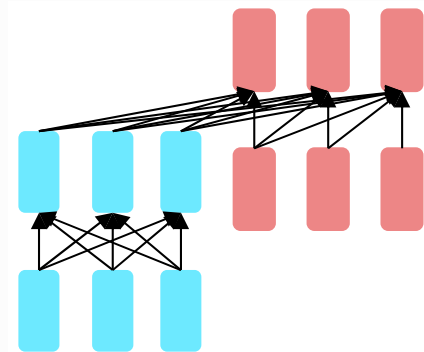
Decoders

GPT, Claude,
Llama, Mixtral



Encoders

BERT family,
RoBERTa

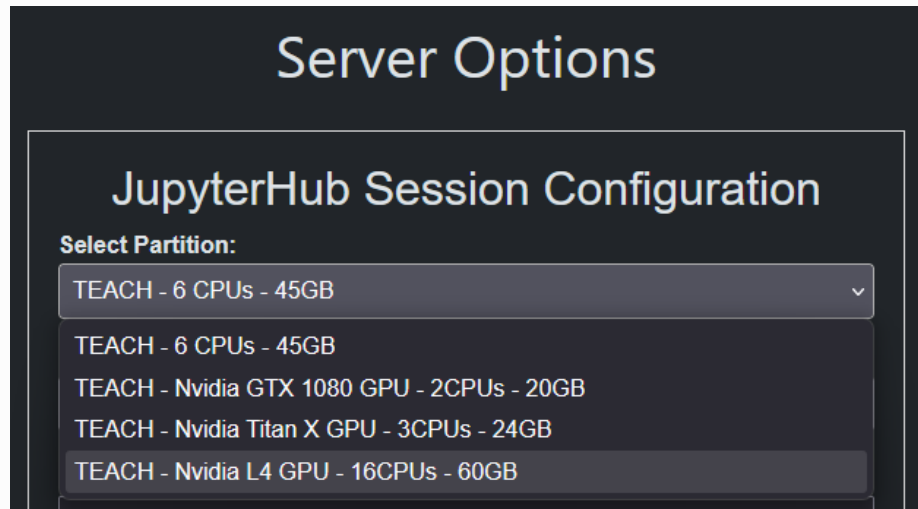


Encoder-decoders

Flan-T5, Whisper

Notebook: finetune GPT-2 on Shakespeare

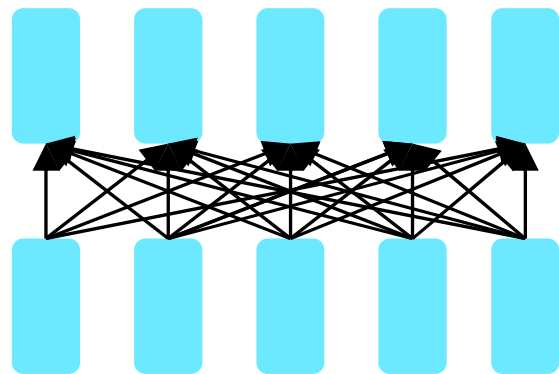
1. Click this [nbgitpuller link](#) (also available on course website)
2. **Important difference from normal:** Start a server with 'TEACH – Nvidia L4 GPU – 16 CPUs – 60GB' server
3. Load **new** custom environment at `/vast/cs1671-2026f/class-env/.pixi/envs/default`
4. Open `session19_gpt2_shakespeare.ipynb`



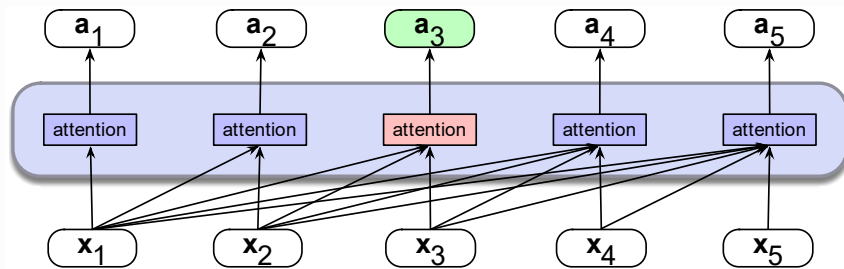
- Transformer encoder: BERT family

Encoders

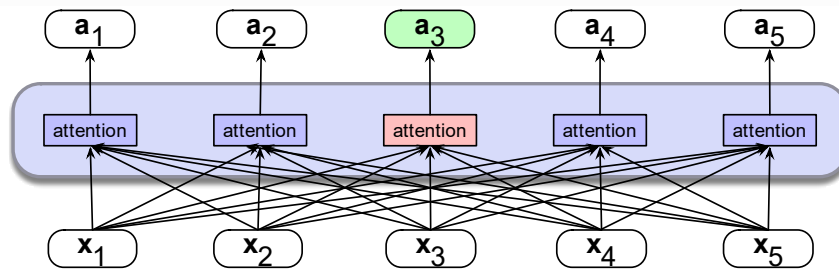
- So far, we've looked at (causal, left-to-right) language model pretraining
- But what about tasks where we want to peek at future tokens?
- Encoders can access bidirectional context
- Map sequences of input embeddings to sequences of output embeddings that have been contextualized using information from the entire sequence
- No “masking” of future words in self-attention



Bidirectional Self-Attention



a) A causal self-attention layer



b) A bidirectional self-attention layer

Pretraining encoders: masked language modeling

- BERT (Devlin et al. 2019) is pretrained with 2 objectives
 - Masked language modeling
 - Next sentence prediction (not as important, not covered in class)

The Cloze Task

- The **cloze** task comes from psycholinguistics (the branch of linguistics and cognitive science that uses experimental methods to study how language works in human brains).
- It is a fill-in-the-blank task:

He drove the yellow _____ into the front of our house.

- Subjects are presented with these frames and asked to fill in the missing words
- This allows experimenters to assess what a speaker understands about grammar, semantics, etc.
- According to the original BERT paper, this task provided the inspiration for BERT's masked language modeling (MLM) training task.
- But compare various kinds of denoising algorithms.

MLM training in BERT

15% of the tokens are randomly chosen to be part of the masking .

Example: "Lunch was **delicious**", if delicious was randomly chosen:

Three possibilities:

1. 80%: Token is replaced with special token [MASK]

Lunch was **delicious** -> Lunch was **[MASK]**

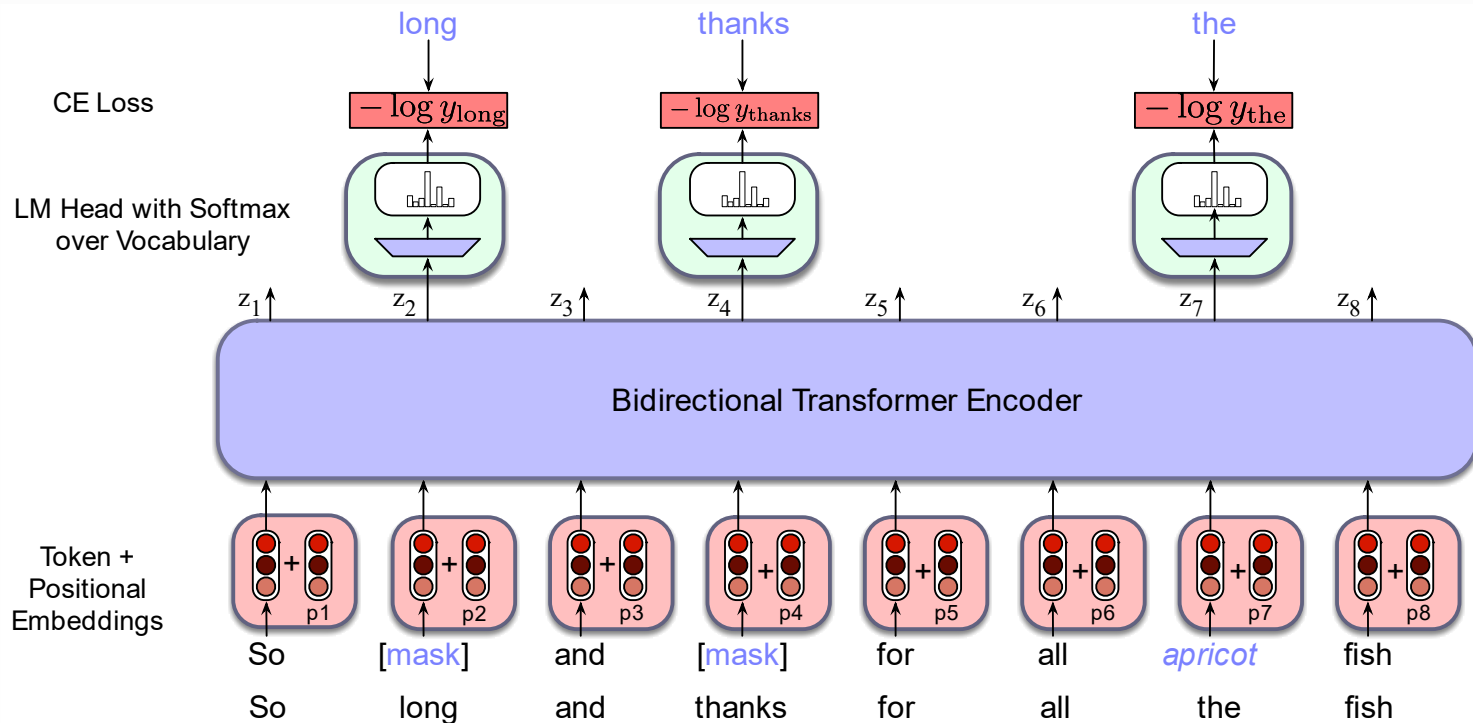
2. 10%: Token is replaced with a random token (sampled from unigram prob)

Lunch was **delicious** -> Lunch was **gasp**

3. 10%: Token is unchanged

Lunch was **delicious** -> Lunch was **delicious**

In detail



BERT: Bidirectional Encoder Representations from Transformers

Details about BERT

- Two models were released:
 - BERT-base: 12 layers, 768-dim hidden states, 12 attention heads, 110 million params.
 - BERT-large: 24 layers, 1024-dim hidden states, 16 attention heads, 340 million params.
- Trained on:
 - BooksCorpus (800 million words)
 - English Wikipedia (2,500 million words)
- Pretraining is expensive and impractical on a single GPU.
 - BERT was pretrained with 64 TPU chips for a total of 4 days.
(TPUs are special tensor operation acceleration hardware)
- Finetuning is practical and common on a single GPU
 - “Pretrain once, finetune many times.”

Extensions of BERT

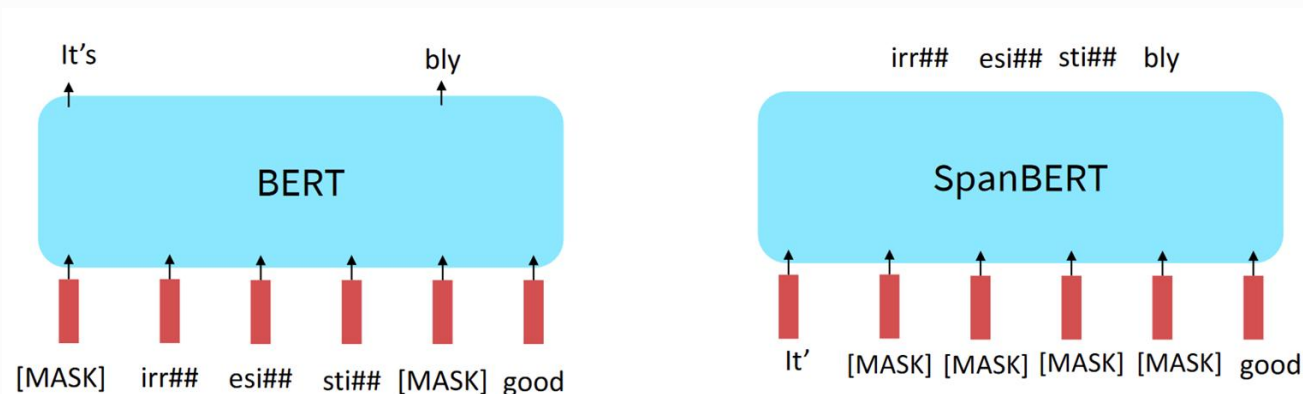
You'll see a lot of BERT variants like RoBERTa, SpanBERT, etc

Contemporary versions of BERT:

- ModernBERT
- XLM-R
- MiniLM

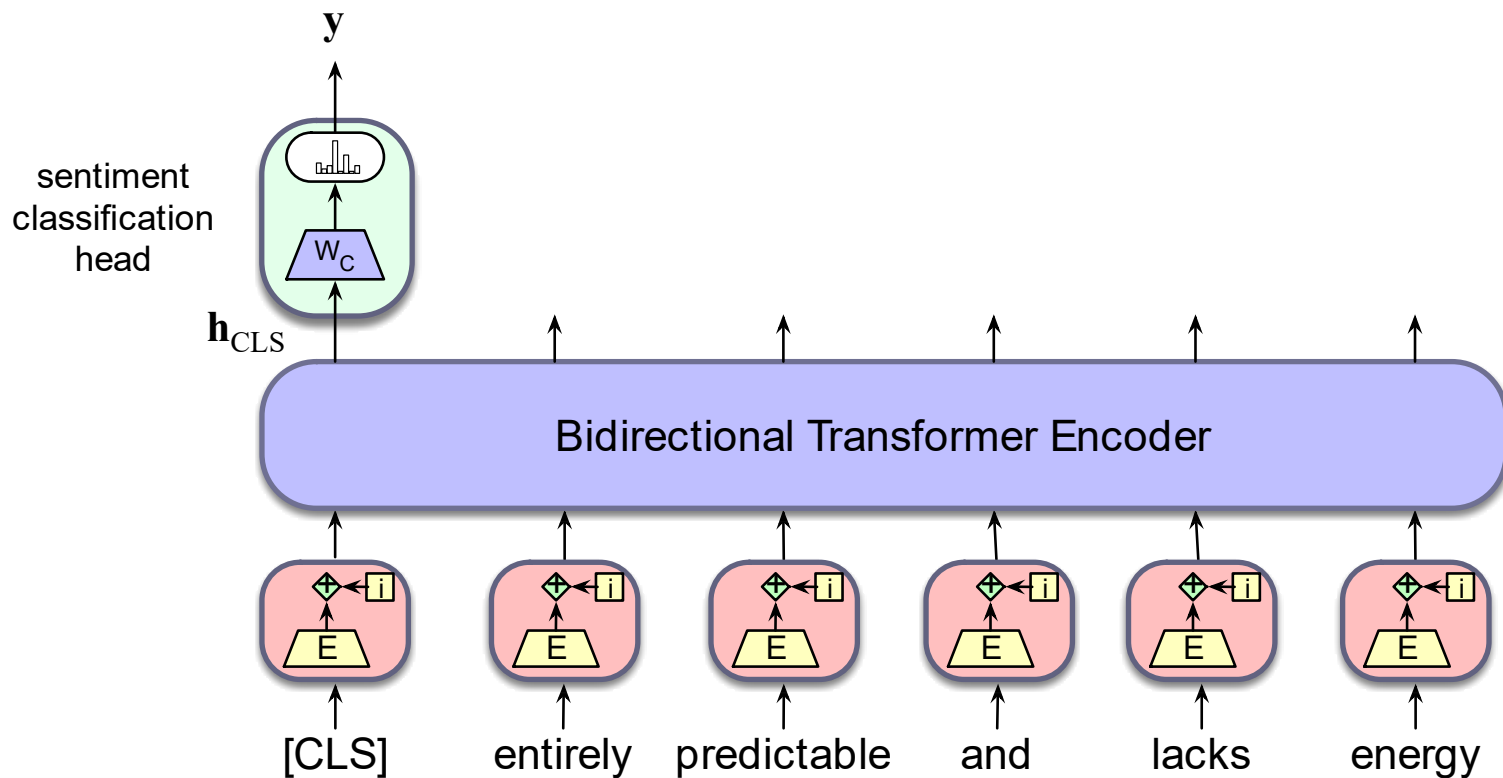
Some generally accepted improvements to the BERT pretraining formula:

- RoBERTa [Liu et al. 2019]: mainly just train BERT for longer and remove next sentence prediction!
- SpanBERT [Joshi et al. 2020]: masking contiguous spans of words makes a harder, more useful pretraining task



Finetuning BERT for classification

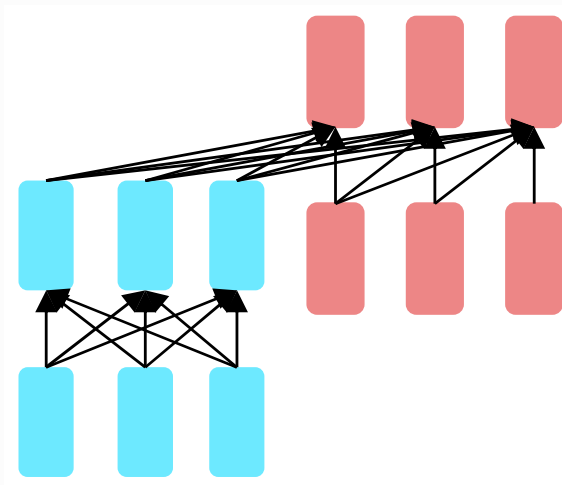
Finetuning for classification



Encoder-decoders (T5)

Encoder-Decoders

- Trained to map from one sequence to another
- Popular for:
 - machine translation (map from one language to another)
 - speech recognition (map from acoustics to words)
- See T5 model (Raffel et al 2020)
- Decoder transformer attends to encoding



Conclusion

- BERT is an encoder transformer model that produces an output embedding for every input token
- BERT is pretrained on the task of masked language modeling, learning to predict masked words in the middle of sentences
- BERT is often finetuned for classification with a neural classification layer on the embedding output for the special [CLS] token (taken to represent the whole sentence)
- Encoder-decoder models like T5 map one sequence to another sequence

Notebook: finetune BERT for politeness classification

1. Click this [nbgitpuller link](#) (also available on course website)
2. **Keep using a GPU:** Start a server with 'TEACH – Nvidia L4 GPU – 16 CPUs – 60GB' server
3. Load **new** custom environment at `/vast/cs1671-2026f/class-env/.pixi/envs/default`
4. Open `session20_bert_politeness.ipynb`

