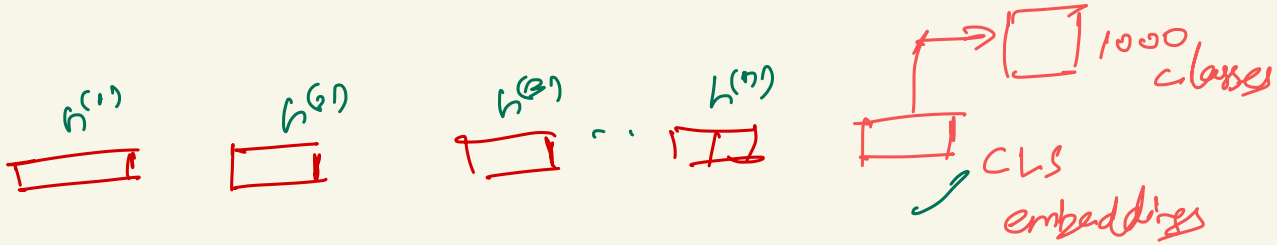
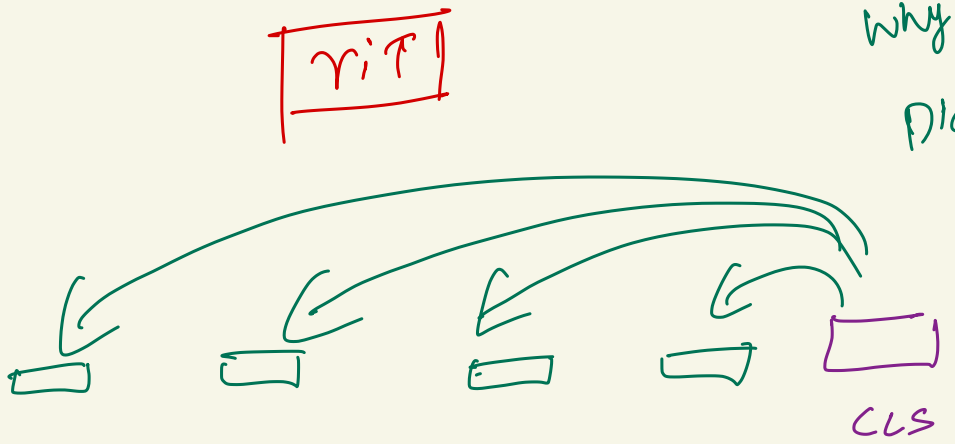
More Details about BERT

- 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."

BERT vocabulary: 30,522





Why is Transformers?
Didn't need in RNN.

Encoders vs. Decoders

Same # of parameters

Same model

Bi-directional attn

Causal attention

Classification

Seq labeling

Sentence rep. (2 sentences are similar?)

Generation

Essential for Pretraining (

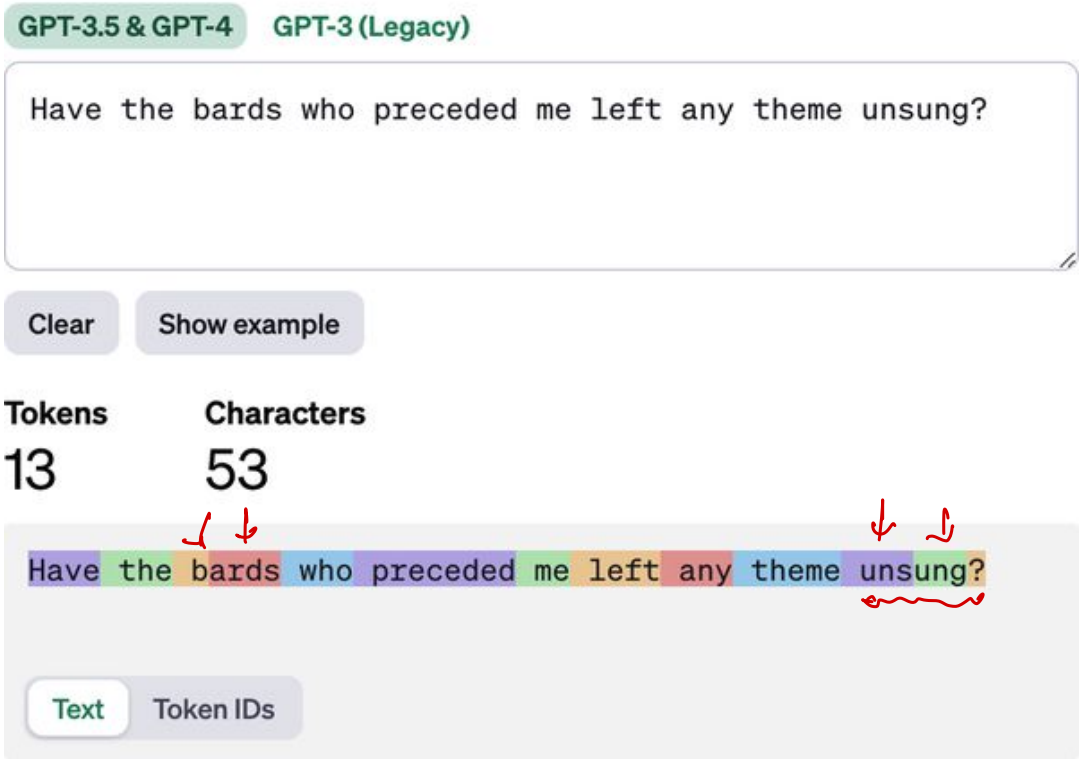
1. Data (lots & lots of)

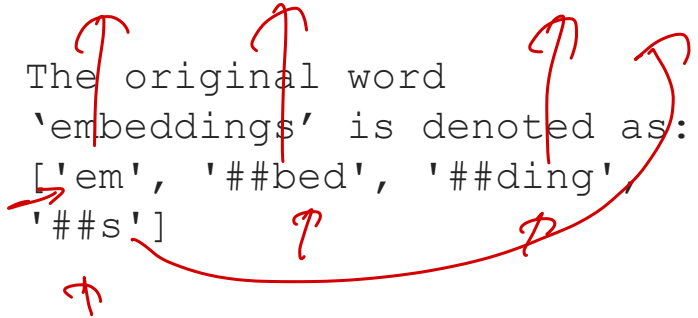
2. Compute

Model Architectures

self-supervision →

BERT uses subwords





Source: Alammar, J., & Grootendorst, M. (2024). Hands-On Large Language Models. O'Reilly.

Byte-Pair-Encoding (BPE)

[coined by Gage et al., 1994; adapted to the task of word segmentation by **Sennrich et al., 2016**; see Gallé (2019) for more]

Main idea: Use our data to automatically tell us what the tokens should be

Token learner:
Raw train corpus ⇒ Vocabulary (a set of tokens)

BPE

Token segmenter:
Raw sentences ⇒ Tokens in the vocabulary

Types are distinct tokens in a corpus

Corpus vs. **Dataset**

[Jurafsky & Martin (2023)]

Byte-Pair-Encoding (BPE) – **Token learner**

[coined by Gage et al., 1994; adapted to the task of word segmentation by **Sennrich et al., 2016**; see Gallé (2019) for more]

Raw train corpus ⇒ Vocabulary (a set of tokens)

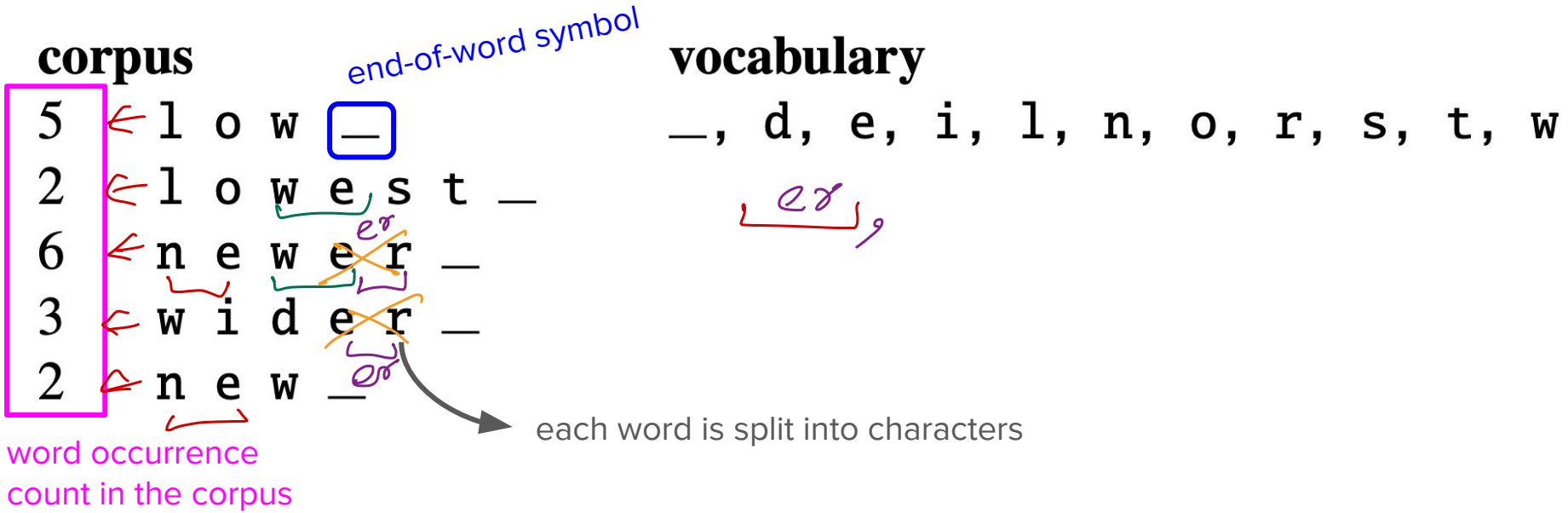
- Pre-tokenize the corpus in words & append a special end-of-word symbol _ to each word
- Initialize vocabulary with the set of all individual characters
- Choose 2 tokens that are most frequently adjacent ("A", "B")
  - Respect word boundaries: Run the algorithm inside words
- Add a new merged symbol ("AB") to the vocabulary
- Change the occurrence of the 2 selected tokens with the new merged token in the corpus
- Continues doing this until k merges are done

All k new symbols and initial characters are the final vocabulary

What's k? Open research question, see Mielke et al., 2021 Sec 6.6; 30K is seen frequently

[Jurafsky & Martin (2023)]

Byte-Pair-Encoding (BPE) – **Token learner *Example***



[Example from Jurafsky & Martin (2023), pages 18–19]

Byte-Pair-Encoding (BPE) – **Token learner *Example***

- Counts all pairs of adjacent symbols
- The most frequent is the pair `e r` [a total of 9 occurrences]
- Merge these symbols, treating `er` as one symbol, & add the new symbol to the vocabulary

corpus

```
5   l o w _
2   l o w e s t _
6   n e w er _
3   w i d er _
2   n e w _
```

vocabulary

_, d, e, i, l, n, o, r, s, t, w, er

[Example from Jurafsky & Martin (2023), pages 18–19]

Byte-Pair-Encoding (BPE) – **Token learner *Example***

- Counts all pairs of adjacent symbols
- The most frequent is the pair `er _`
- Merge these symbols, treating `er_` as one symbol, & add the new symbol to the vocabulary

corpus

```
5   l o w _
2   l o w e s t _
6   n e w er_
3   w i d er_
2   n e w _
```

vocabulary

_, d, e, i, l, n, o, r, s, t, w, er, er_

[Example from Jurafsky & Martin (2023), pages 18–19]

Byte-Pair-Encoding (BPE) – **Token learner *Example***

- Counts all pairs of adjacent symbols
- The most frequent is the pair n e
- Merge these symbols, treating ne as one symbol, & add the new symbol to the vocabulary

corpus

5	l o w _
2	l o w e s t _
6	ne w er_
3	w i d er_
2	ne w _

vocabulary

_, d, e, i, l, n, o, r, s, t, w, er, er_, ne

[Example from Jurafsky & Martin (2023), pages 18–19]

Byte-Pair-Encoding (BPE) – **Token learner *Example***

merge	current vocabulary
(ne, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new
(l, o)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo
(lo, w)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low
(new, er_)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_
(low, _)	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne, new, lo, low, newer_, low_

Final vocabulary

[Example from Jurafsky & Martin (2023), pages 18–19]

Byte-Pair-Encoding (BPE) – **Token segmenter**

[coined by Gage et al., 1994; adapted to the task of word segmentation by **Sennrich et al., 2016**; see Gallé (2019) for more]

The token segmenter is used to tokenize a test sentence

- Just runs on the test data the merges we've learned from the training data, greedily, in the order we learned them

First we segment each test sentence word into characters

Then we apply the first merge rule

- E.g., replace every instance of `e r` in the test corpus with `er`

Then the second merge rule

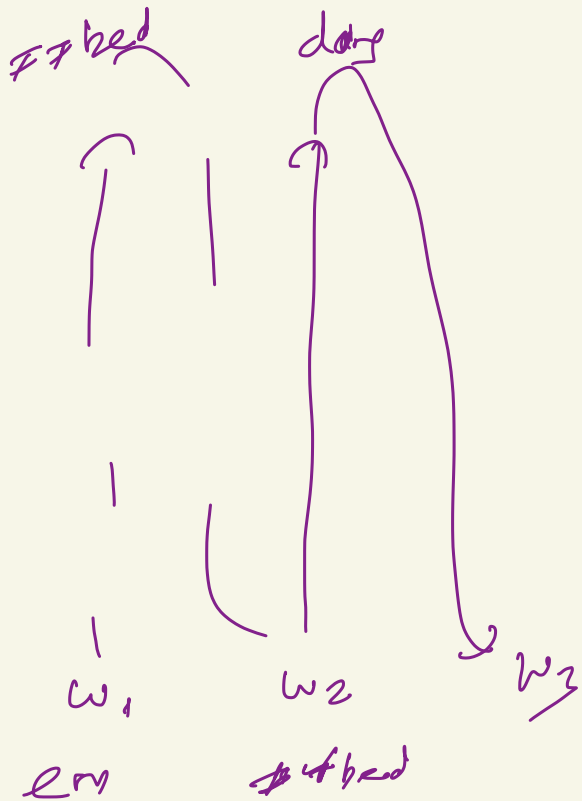
- E.g., replace every instance of `er _` in the test corpus with `er_`

And so on

[Example from Jurafsky & Martin (2023), pages 18–19]

Vocabulary size of various models

Model	Tokenizer	Vocabulary Size
BERT base (uncased) [2018]	WordPiece	30,522
BERT base (cased) [2018]	WordPiece	28,996
GPT-2 [2019]	BPE	50,257
Flan-T5 [2022]	SentencePiece	32,100
GPT-4 [2023]	BPE	> 100,000
StarCoder2 [2024]	BPE	49,152
Llama2 [2023]	BPE	32,000


fixed
dog
W1
W2
W3

Coopies

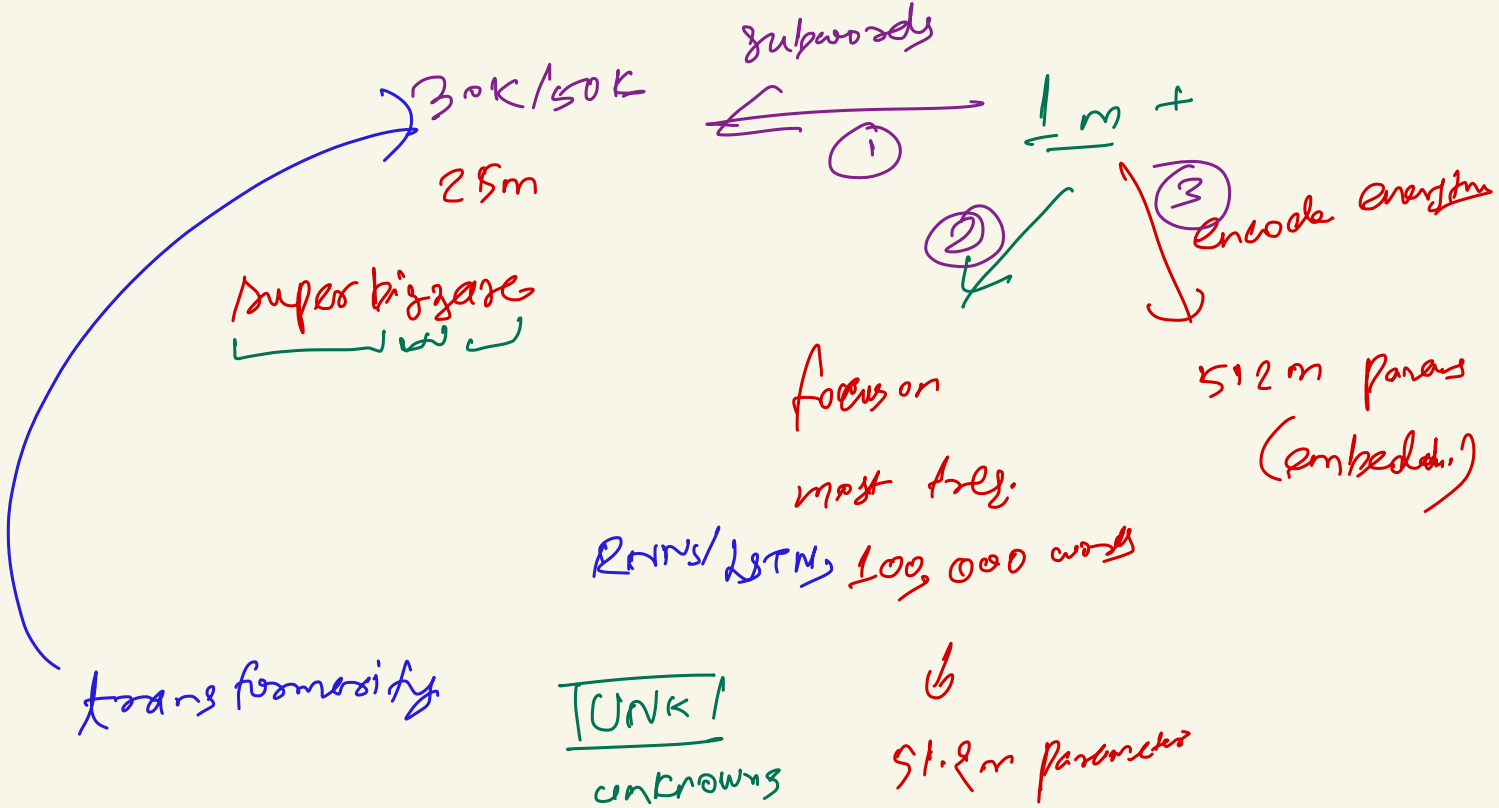
S1. Glove embeddig is of 300 dimensions

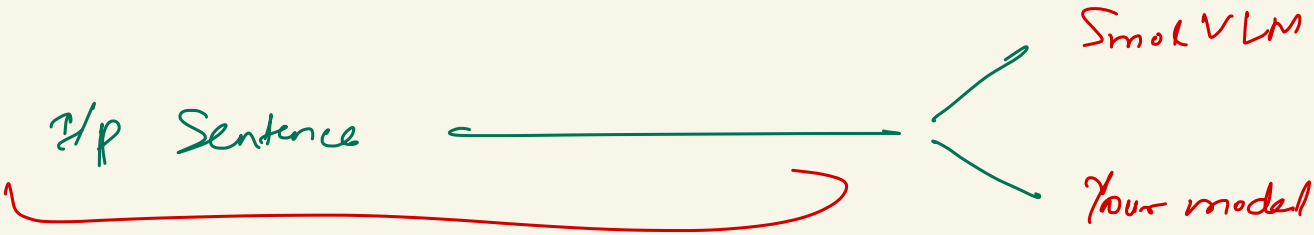
↓ BPE Tokens

↑

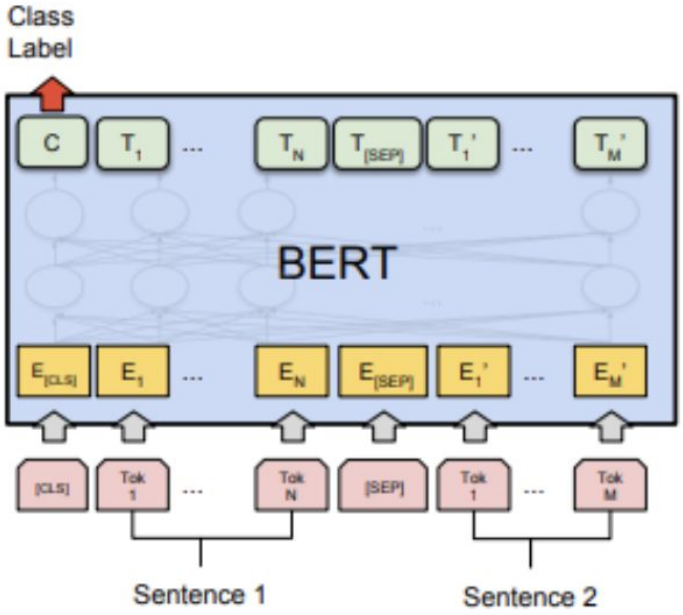
Glove em ~~Dod~~ dig s is of —

(MASK)

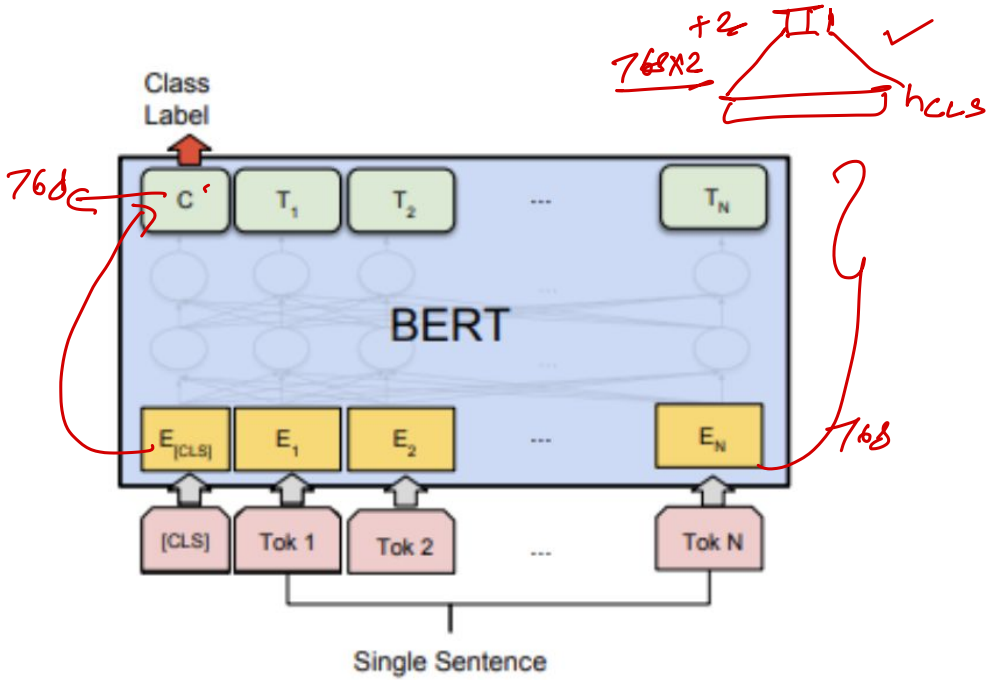

subwords
30K/50K
25m
1m+
1
2
3
encode everything
Super bizzare
focus on
most freq.
RNNs/LSTMs 100,000 words
512m params
(embedding)
transformers
[UNK]
unknowns
51.2m parameters


I/p Sentence
SmolVLM
Your model

Using BERT for different tasks

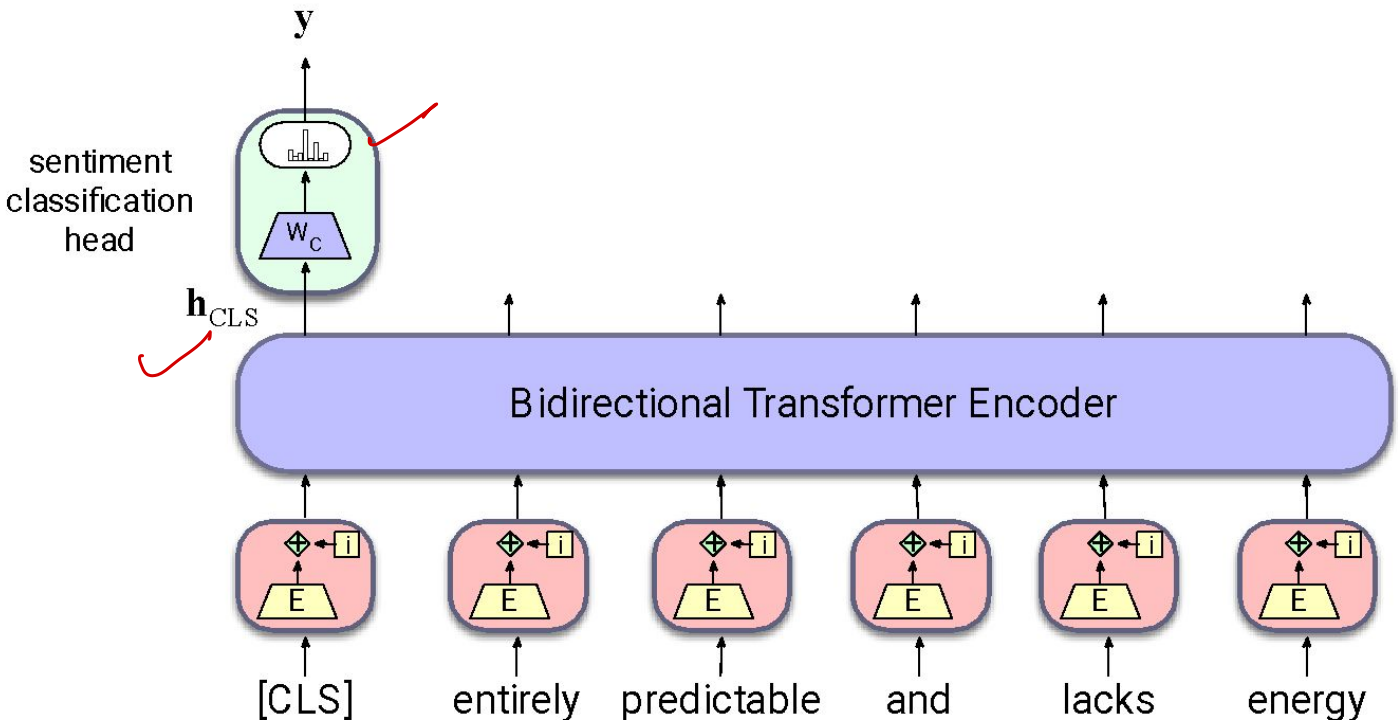


(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

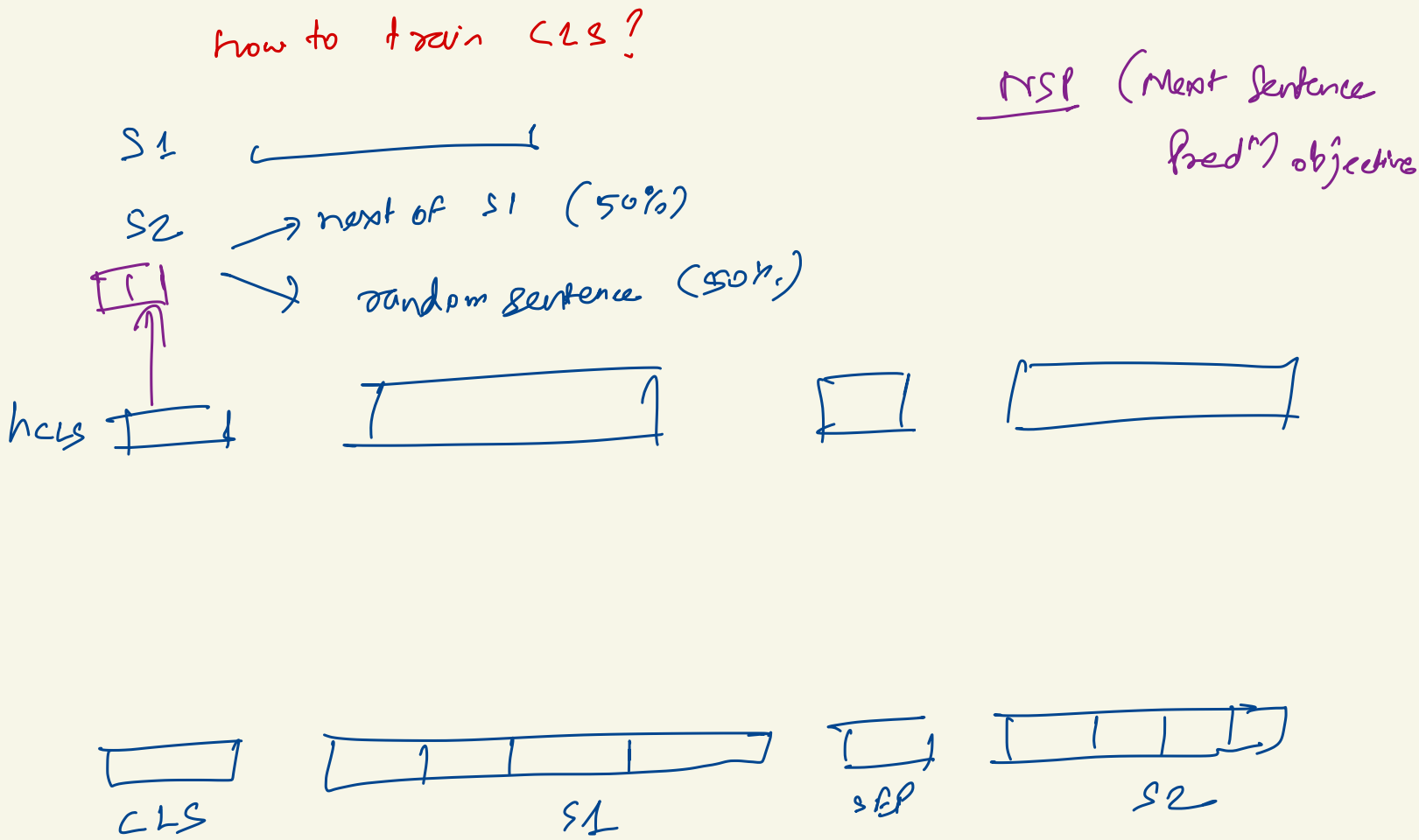


(b) Single Sentence Classification Tasks: SST-2, CoLA

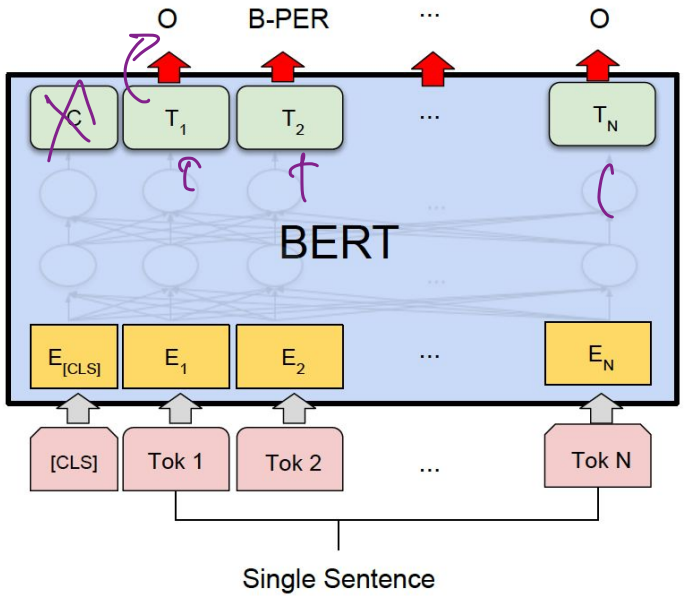
Fine Tuning for Sequence Classification



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


how to train CLS?
NSP (Next Sentence Pred^n objective
S1
S2
next of S1 (50%)
random sentence (50%)
hCLS
CLS
S1
SEP
S2

Sequence Labeling



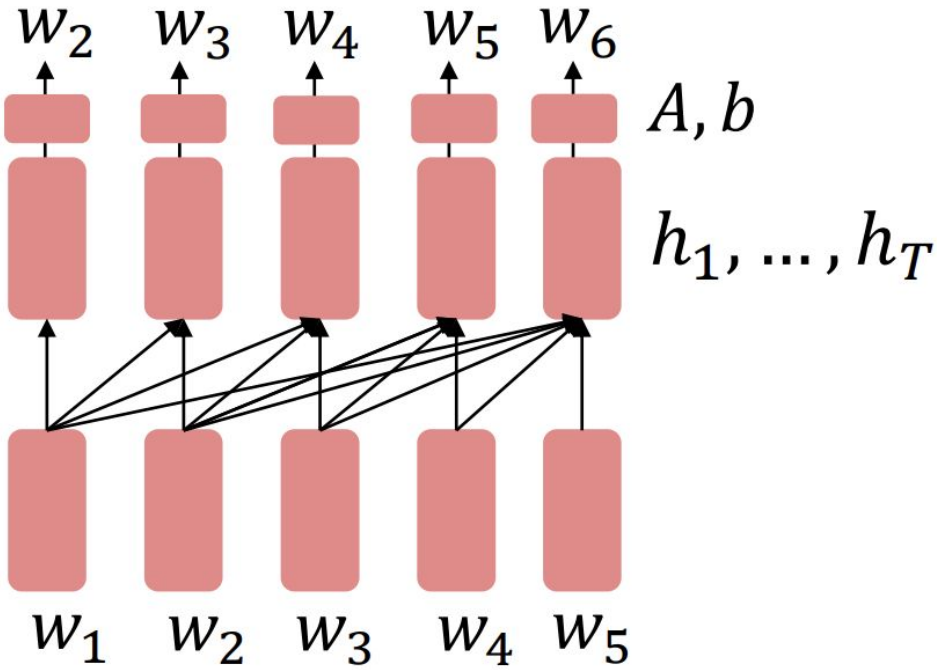
(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

- Here, the final output vector corresponding to *each input token* is passed to a classifier that produces a softmax distribution over the possible set of tags.
- The set of weights to be learned for this additional layer is $W_K \in R^{k \times d_h}$, where k is the number of possible tags for the task.

$d_h = 768$

Pretraining Decoders

It's natural to pretrain decoders as language models



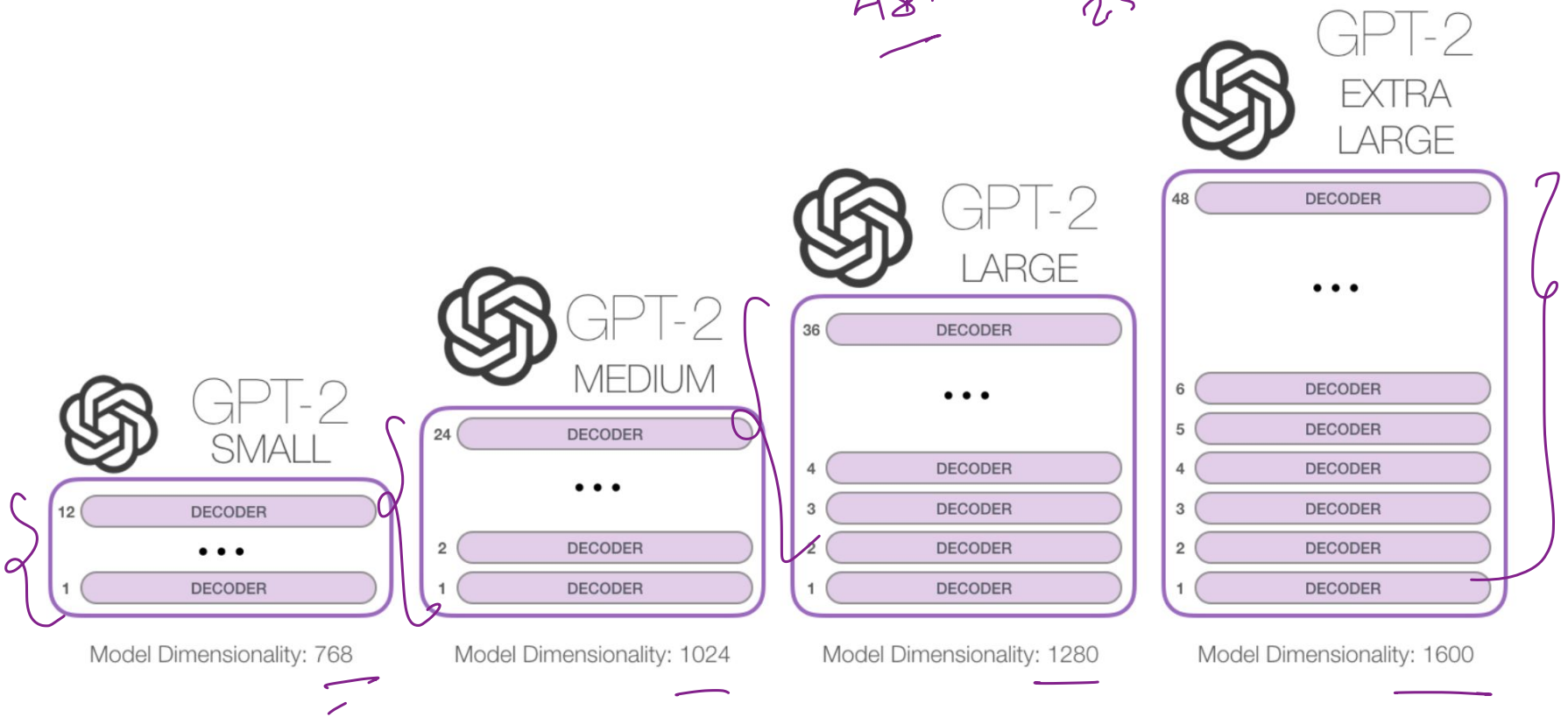
≈ RNN LM training

Generative Pretrained Transformer (GPT)

- Transformer decoder with 12 layers, 117M parameters.
- 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
- Byte-pair encoding with 40,000 merges
- Trained on BooksCorpus: over 7000 unique books.
- Contains long spans of contiguous text, for learning long-distance dependencies.

Vocabulary larger than BERT

Pretrained Decoders



GPT-2: Beyond Language Modeling

Basic Idea

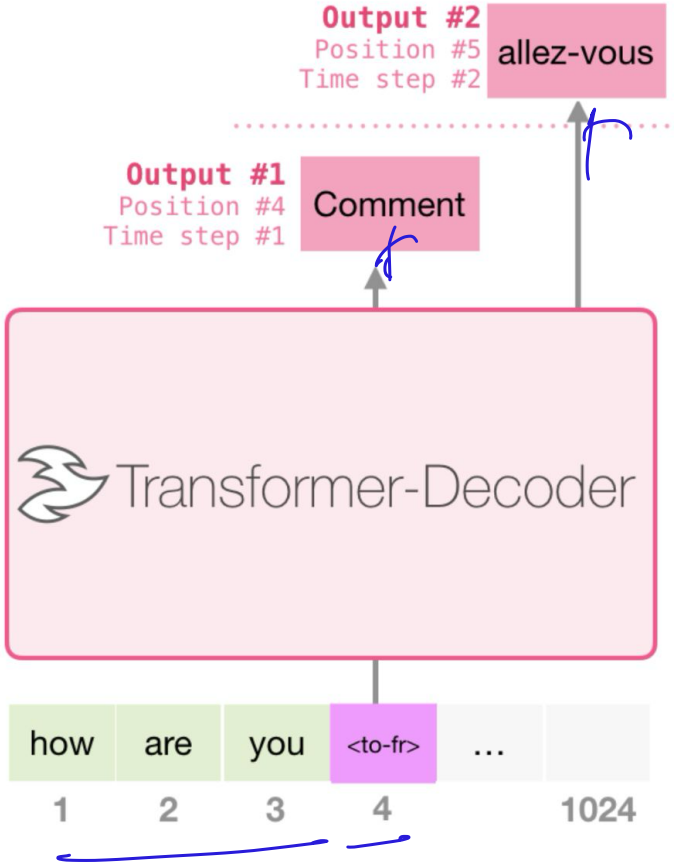
The decoder can work with a prompt!

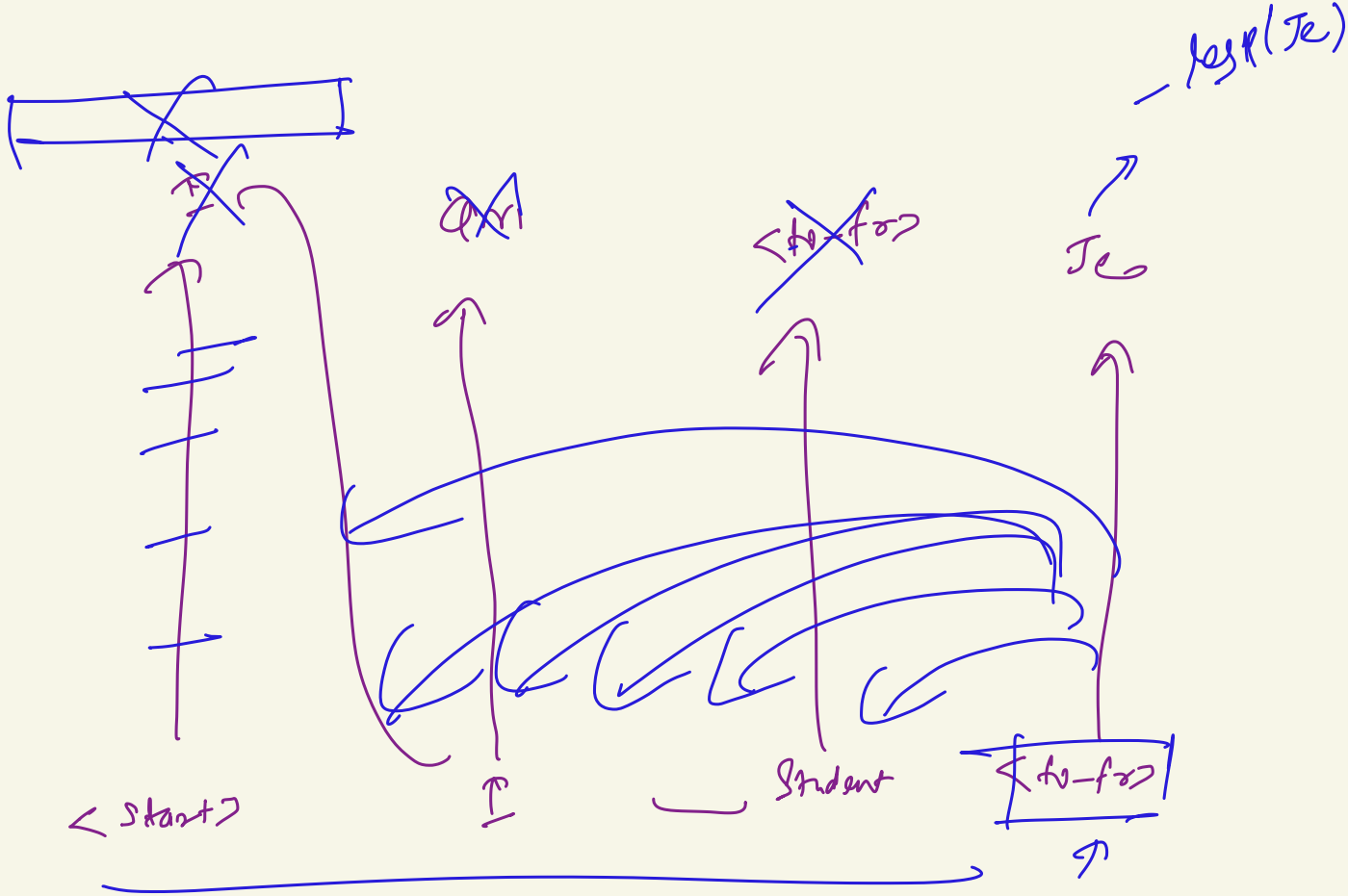
Any NLP task can be expressed in a probabilistic framework as estimating a conditional distribution $p(output|input)$.

GPT-2: Machine Translation

Training Dataset

I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				




log(Je)
Je
<start>
Student

GPT-2: Summarization

Training Dataset

Article #1 tokens		<summarize>	Article #1 Summary
Article #2 tokens	<summarize>	Article #2 Summary	padding
Article #3 tokens		<summarize>	Article #3 Summary