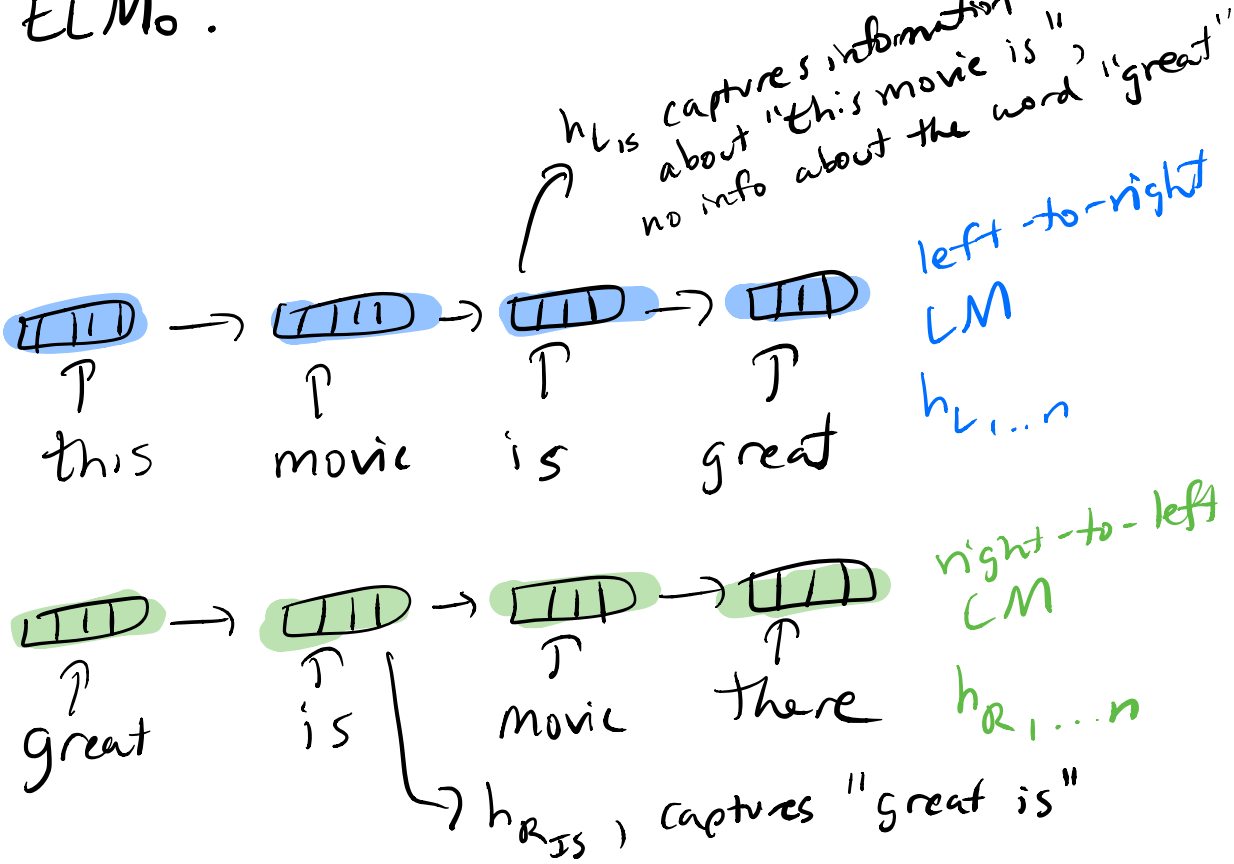
Today: - from ELMo to BERT
- from language modeling to masked LM

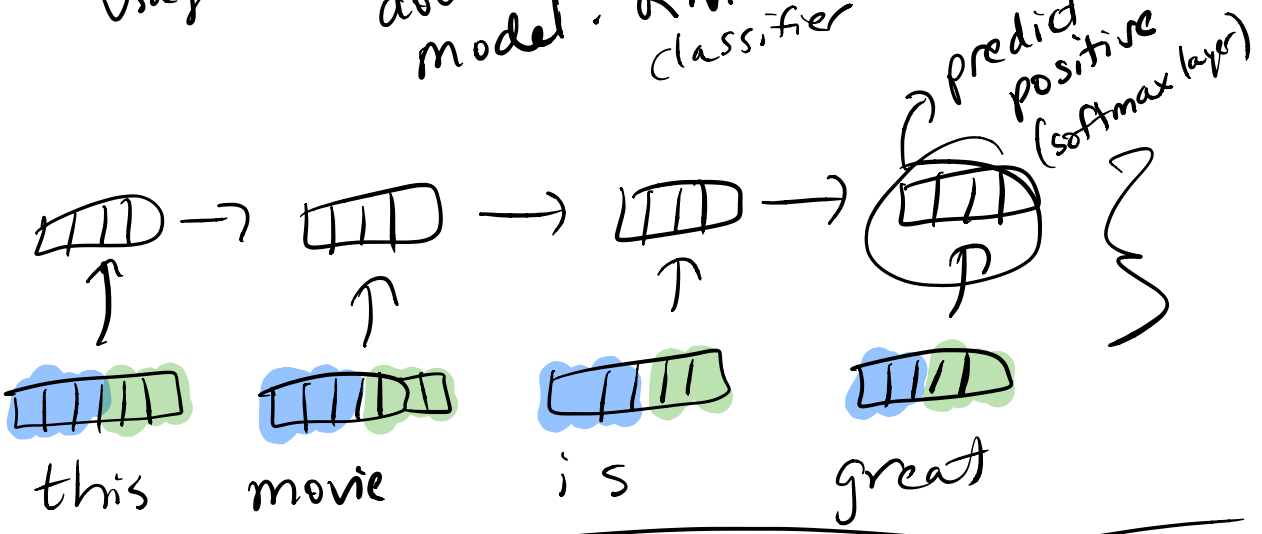
Goal of pretraining: use these big LMs as text encoders. Their goal is to enable downstream models to focus on the task at hand, instead of learning how language works.

ELMo:



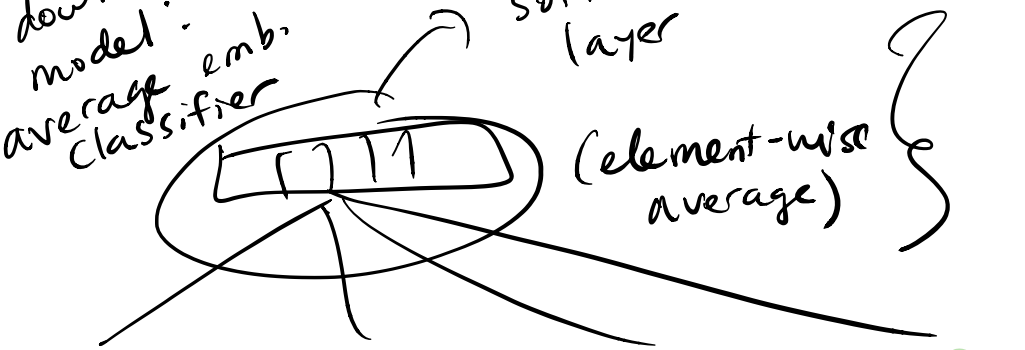
downstream usage

downstream model: RNN classifier



"neural bag of words"

downstream model: average emb. classifier



ex: (in response to question)

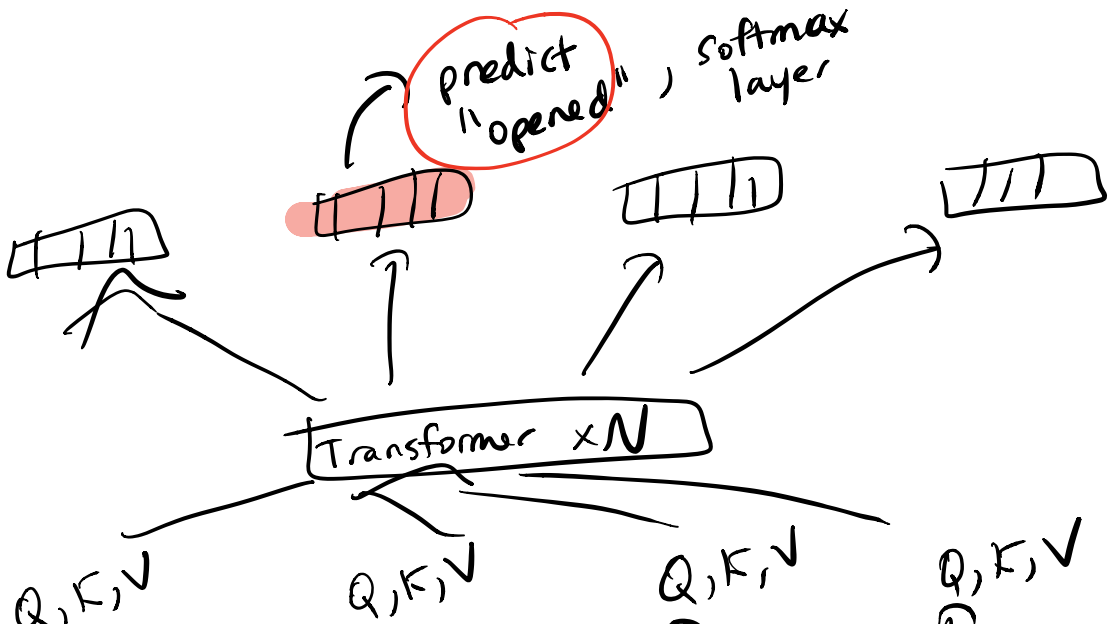
i loved the acting, but (the rest of the movie was terrible).

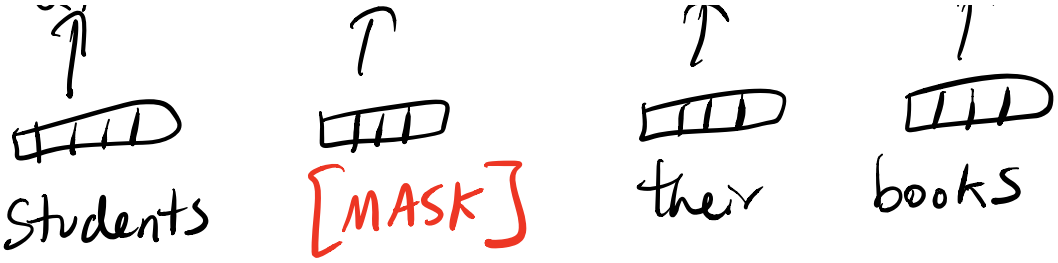
the ELMo approach of two separate LMs that are then concat together is a little hacky ...

- can we accomplish the same goal within a single model
- change pretraining obj.
  - from LM to masked LMs

masked LM:

- given a full sequence of words (not just prefix) where X% of the words have been masked out
- instead of predicting the next word, we only predict masked words

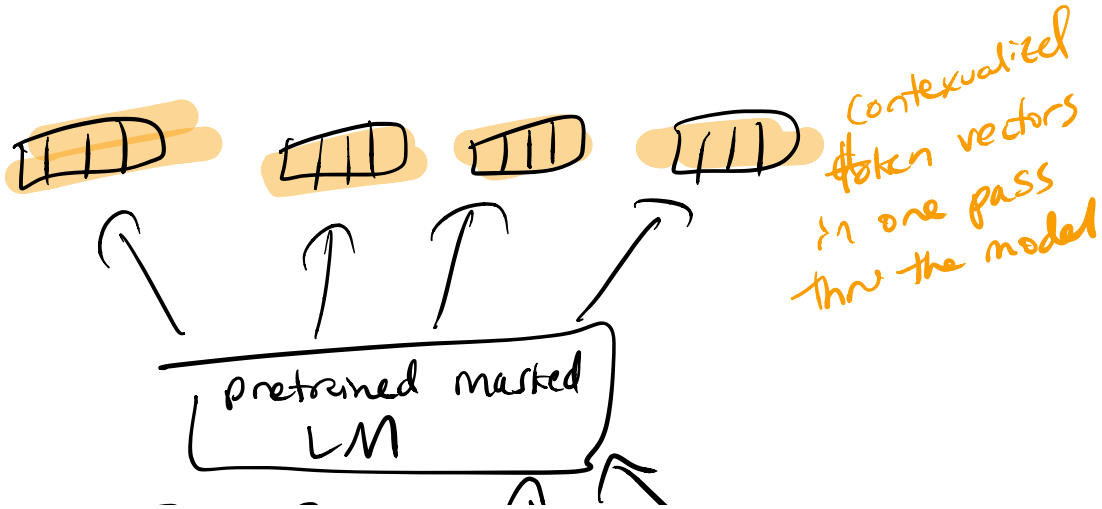


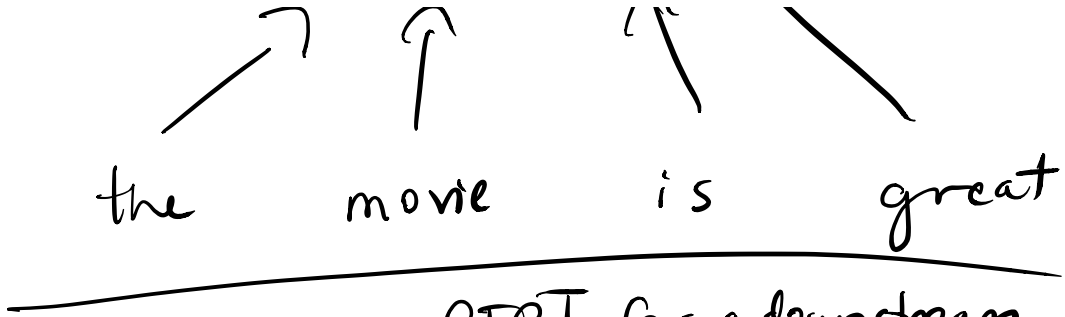


- all of the final layer representations are fully contexualized
  - "aware" of words in the past as well as words in the future

- Same training loss as NLMs
  - minimizing neg. log likelihood of the ground-truth (unmasked) tokens

ELMo → BERT:
- → 2 unidirectional LMs → 1 masked LM
- → recurrent models to Transformers
- → BERT was pretrained on a LOT more data

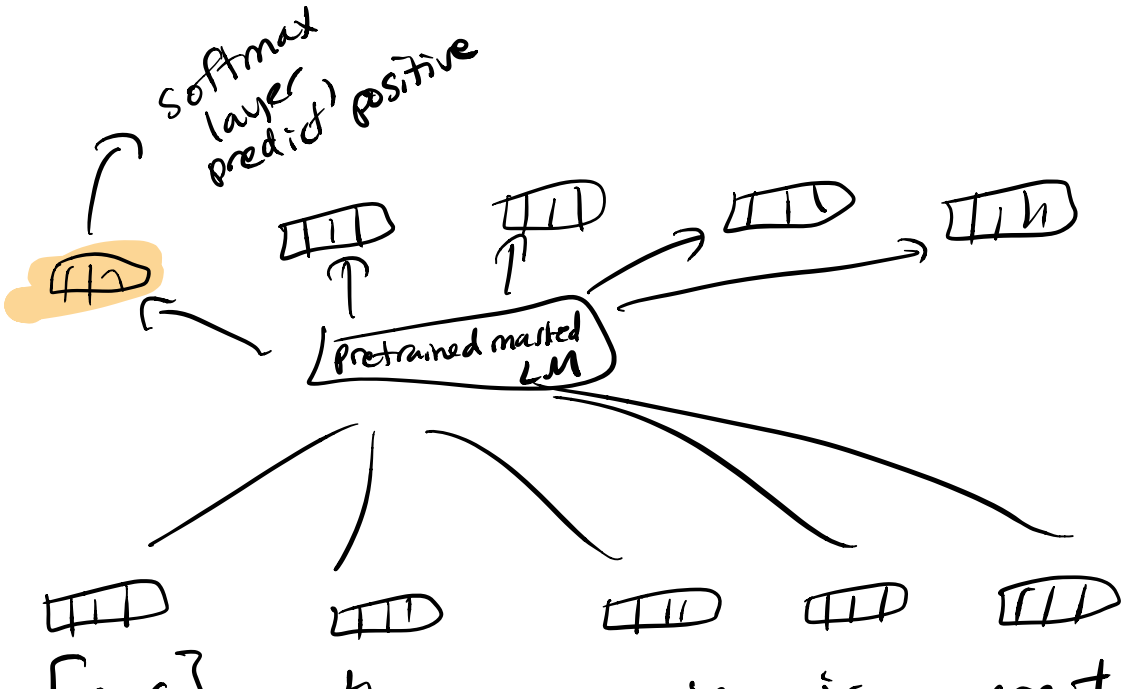




how do we use BERT for a downstream task? the pretrained architecture is <u>almost</u> the same as the downstream model

e.g. sentiment analysis

- add a special token to the beginning of every sequence
  - [CLS] token



[CLS] this movie is great

- backprop the error signal from the sentiment classifier through the entire pretrained masked LM
  - "fine-tuning"
  - no external downstream model
    - only new component is a single softmax layer