Today: improvements on BERT

1. training improvements \Rightarrow \text{RoBERTa}
   \text{more data}

2. longer sequences
   \text{Transformer XL (XLNet)}

3. more efficient pretraining objectives
   \text{ELECTRA}

4. smaller models
   \text{ALBERT}

\text{RoBERTa: very simple}
\text{Collection of modifications}

1. train w/ bigger batches
   \Rightarrow \text{smaller \# of batches w/}
   \text{larger batch size}

   \Rightarrow \text{gradient accumulation}
   \text{to bypass GPU mem. limitations}

2. no next sentence prediction
   \Rightarrow \text{downstream perf. unaffected}
   \Rightarrow \text{[CLS] token gets no pretraining}

3. pretrain on more data
16 GB → 160 GB

→ common crawl
→ URLs from Reddit

4. pretrain for longer (more batches/epochs)
→ 500K steps

**TransformerXL**

BERT has a fixed token limit of 512
for its inputs. How can we model longer sequences?

→ idea: add a recurrent mechanism
that connects adjacent segments

→ no gradient flow to previous segments
hidden states from prev. segment are cached

→ practical limit to their extended context window
→ 900 words for TransformerXL

**ELECTRA** - cheaper obj. fn
Jane goes to [MASK] practice

predict real?

real real fake real

Jane goes to tree practice

how do I decide which words to replace and with what?

"generator" ➞ coming up w/ fake words

train a small BERT model

Jane goes to [MASK] practice

football
basketball
baseball
5 sampled words from generator form fake words for ELECTRA
- every single token is associated with a prediction of real/fake, not just 15% of words as in BERT

**ALBERT** - more params ≠ better model

- cross-layer param sharing
  - Q, K, V projection matrices shared across
  - W matrices in FF layers across all layers

**BERT - large**: 334M params
**ALBERT - large**: 18M params

- what if we make our shared set of params bigger?

**ALBERT - XXL**: 235M params, 4096d hidden state size
- outperforms BERT - large