Representation Models II

HSS 510 / DS 518: NLP for HSS

Taegyoon Kim

May 20, 2025

Agenda

Things to be covered

- Recap: Transformer and BERT architecture
- Training BERT
  - Masked language modeling
  - Next sentence prediction
- Fine-tuning BERT

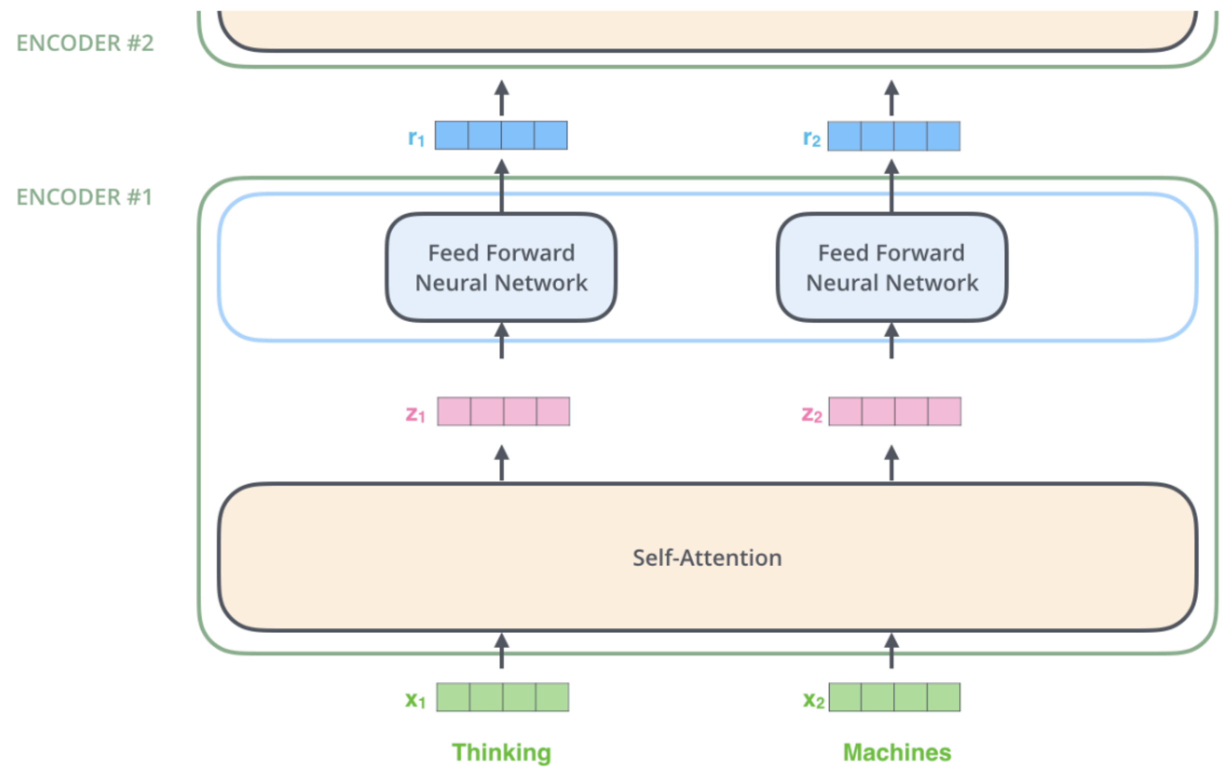
Overview of Transformer

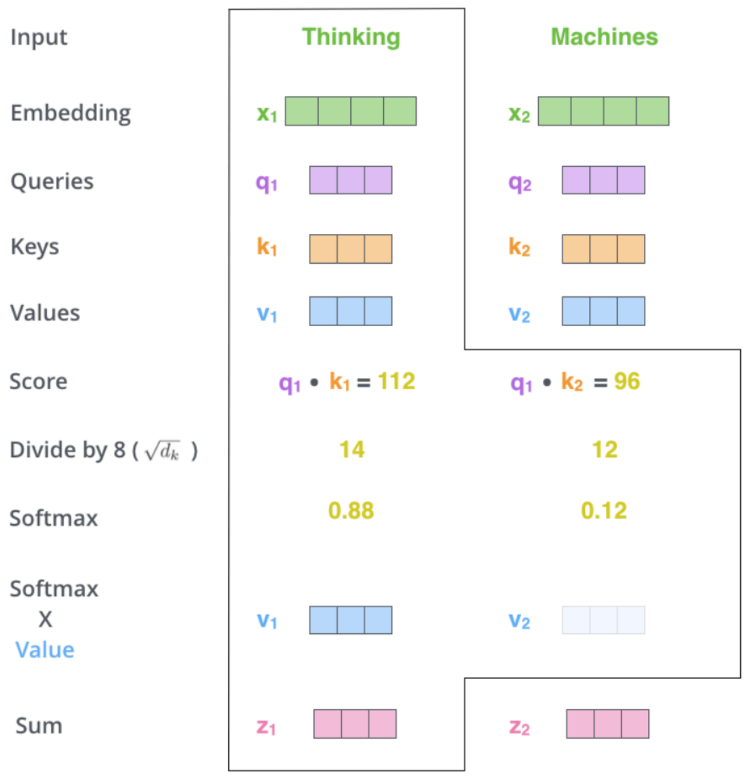
What is Transformer

- Transformer is a type of neural network architecture with self-attention mechanisms
  - Introduced in 2017: Vaswani et al. (2017)
  - Cited 179,358 times as of May 12 2025
- Transformers consist of a stack of (encoder/decoder) layers main consisting of self-attention and feed forward neural networks
  - See [JM] Chp.7 for a discussion of neural networks in NLP

Overview of Transformer

Transformer encoder





Overview of BERT

Bi-directional **E**ncoder **R**epresentations from **T**ransformers

- A form of transformer trained as a language model
  - Two tasks: masked token prediction, next sentence prediction
  - Introduced in 2019: Devlin et al. (2018)
  - Cited 130,271 times as of May 12 2025
- Pre-trained on huge data sets (BookCorpus and Wikipedia)
- Two versions of the model introduced in the paper
  - BERT BASE (12 layers)
  - BERT LARGE (24 layers)

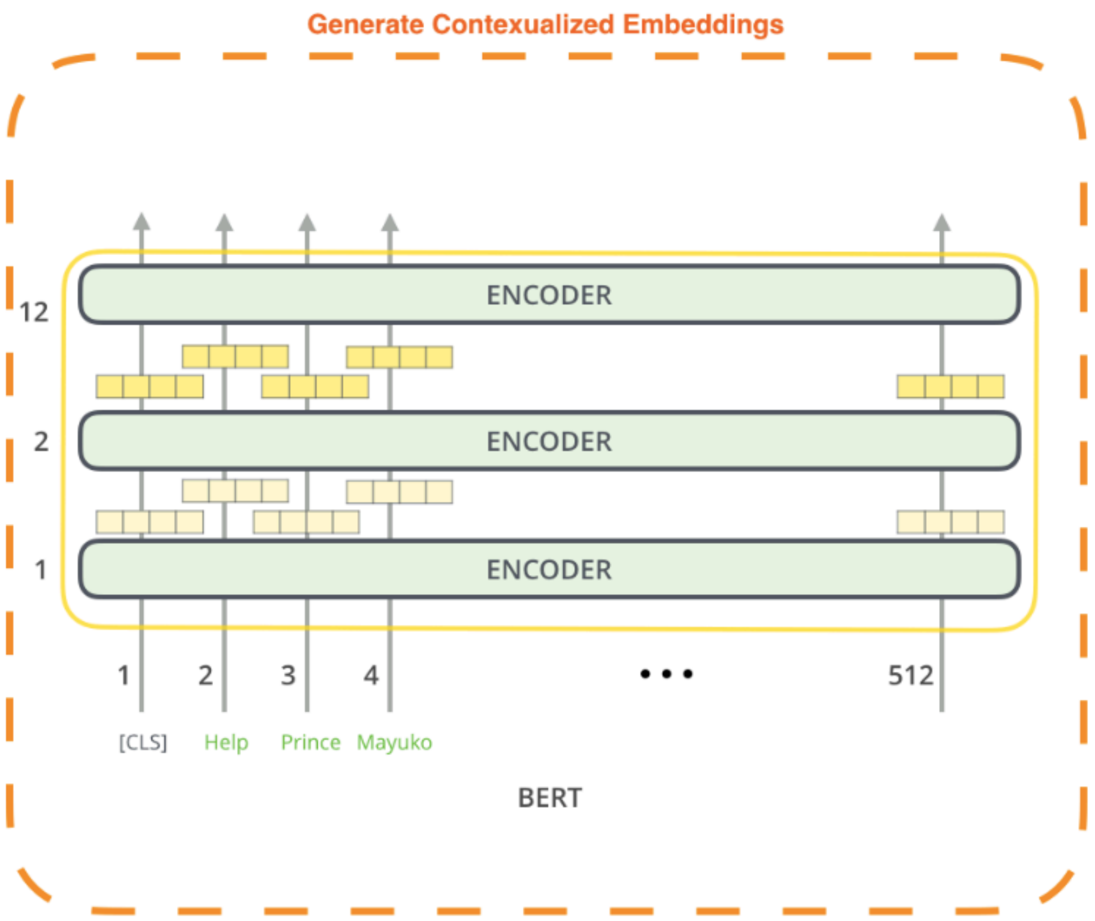
Overview of BERT

Peak at under the hood

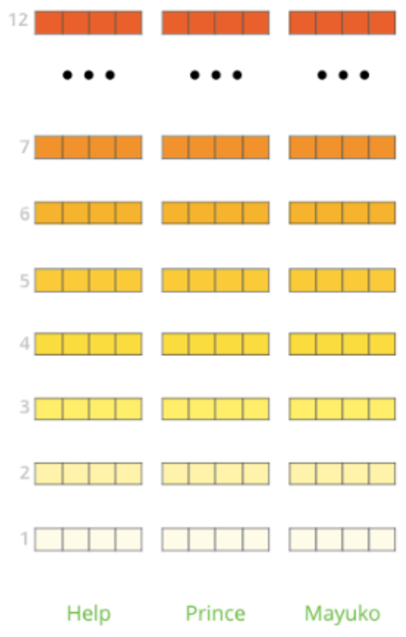
- BERT generates its own embeddings (from scratch) as part of its training process
- Stack of Transformer encoder layers involving self-attention mechanisms
- Each layer passes its results through a feed-forward network, and then hands it off to the next encoder in the stack
- Each position outputs a vector of size 768 (BASE) or 1024 (LARGE)

Overview of BERT

Extracting embeddings



The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

Training BERT

Overview

- BERT is trained using two unsupervised objectives:
  - Masked Language Modeling for learning word-level representations
  - Next Sentence Prediction for learning sentence-level relationships
- MLM trains BERT to understand context by predicting missing words in a sentence
- NSP trains BERT to model discourse coherence by predicting if one sentence follows another
- Together, these tasks help BERT learn deep, contextual understanding of language

Training BERT

Task 1: Masked Language Model

- The MLM training objective is to predict the original inputs for each of the masked tokens
- Comparison with left-to-right/causal models
  - Please turn your homework ____.
  - MLM: Please turn ____ homework in.
- A form of denoising

Training BERT

Task 1: Masked Language Model

- 15% of the input tokens in a training sequence are sampled for learning
- Out of these:
  - 80% are replaced with `[MASK]`
  - 10% are replaced with another token from the vocabulary
    - randomly sampled based on token uni-gram probabilities
  - 10% are left unchanged

Training BERT

Task 1: Masked Language Model

- All tokens go through the transformer layers (output: contextualized vector for every token in the sequence)
- Only selected tokens (15%) are targets for prediction
  - Their contextual vectors are passed through a feed-forward layer + softmax
  - This produces a probability distribution over the entire vocabulary
  - Loss is computed by comparing predicted word to the true word
  - Gradients from loss are used to update model parameters

Training BERT

Task 1: Masked Language Model

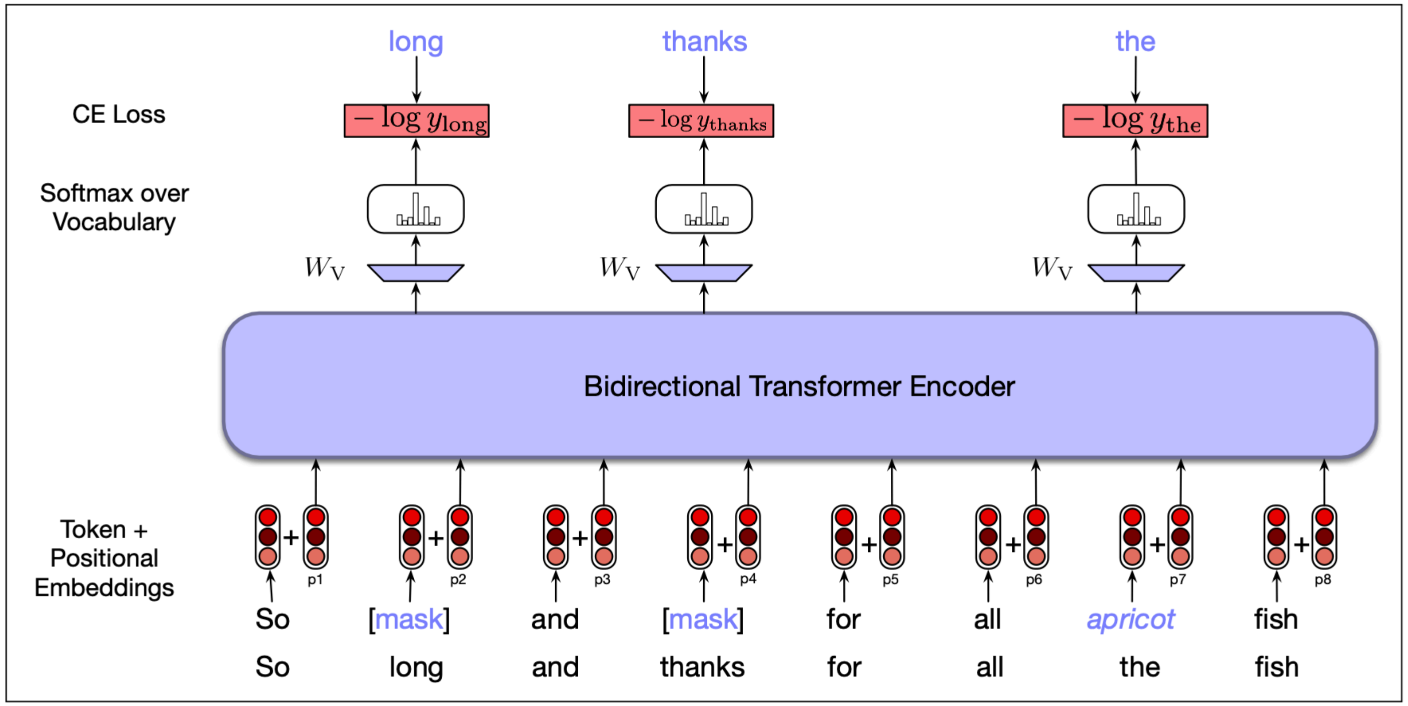


Figure 11.5 Masked language model training. In this example, three of the input tokens are selected, two of which are masked and the third is replaced with an unrelated word. The probabilities assigned by the model to these three items are used as the training loss. (In this and subsequent figures we display the input as words rather than subword tokens; the reader should keep in mind that BERT and similar models actually use subword tokens instead.)

Training BERT

Task 2: Next Sentence Prediction

- MLM focuses on word-level representations
- However, many NLP applications rely on sentence-level understanding
  - E.g., natural language inference
- With NSP (Next Sentence Prediction) task
  - The model is trained to predict whether one sentence follows another
    - 50% of training pairs are actual consecutive sentences
    - 50% are random mismatches

Training BERT

Task 2: Next Sentence Prediction

- Input uses special tokens:
    - `[CLS]` at the beginning
        - `[SEP]` between and after the two sentences
- E.g., “ `[CLS]` cancel my flight `[SEP]` And the hotel `[SEP]` ”
- `[CLS]` token’s final vector is used for NSP prediction
- Classification head (≈ a feed-forward neural network + softmax) outputs a binary label (next or not)

Training BERT

Task 2: Next Sentence Prediction

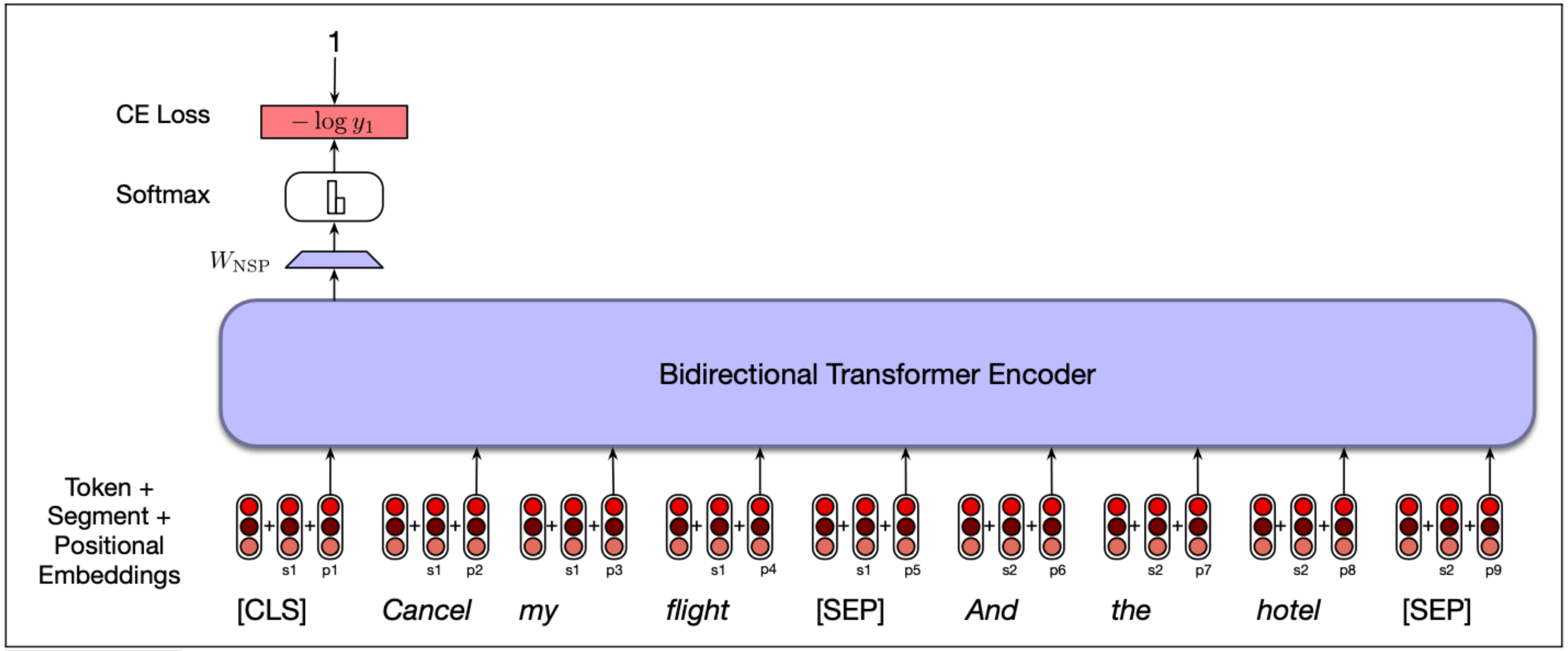


Figure 11.7 An example of the NSP loss calculation.

Fine-tuning BERT

What is fine-tuning, and why?

- Pre-trained language models (like BERT) capture general, transferable knowledge from massive text corpora
- This knowledge can be adapted for a wide variety of downstream tasks (e.g., classification, NER)
- Fine-tuning adds task-specific layers (on top of the pre-trained model) and updates the model using supervised data for the target task
- This training will
    - Make adjustments to the pertained language model parameters (or some of them)
    - Or simply freeze them

→ Stronger performance than training a new model from scratch, especially with limited data

Fine-tuning BERT

Fine-tuning BERT for classification

- `[CLS]` token represents the entire input sequence
- It is a special token added to the start of input during fine-tuning (also during pre-training)
- An vector is learned for this token (a.k.a. “sentence embedding” since it refers to the entire sequence)
- The output vector in the final layer for `[CLS]` serves as representation of the entire input sequence (a summary representation)

Fine-tuning BERT

Fine-tuning BERT for classification

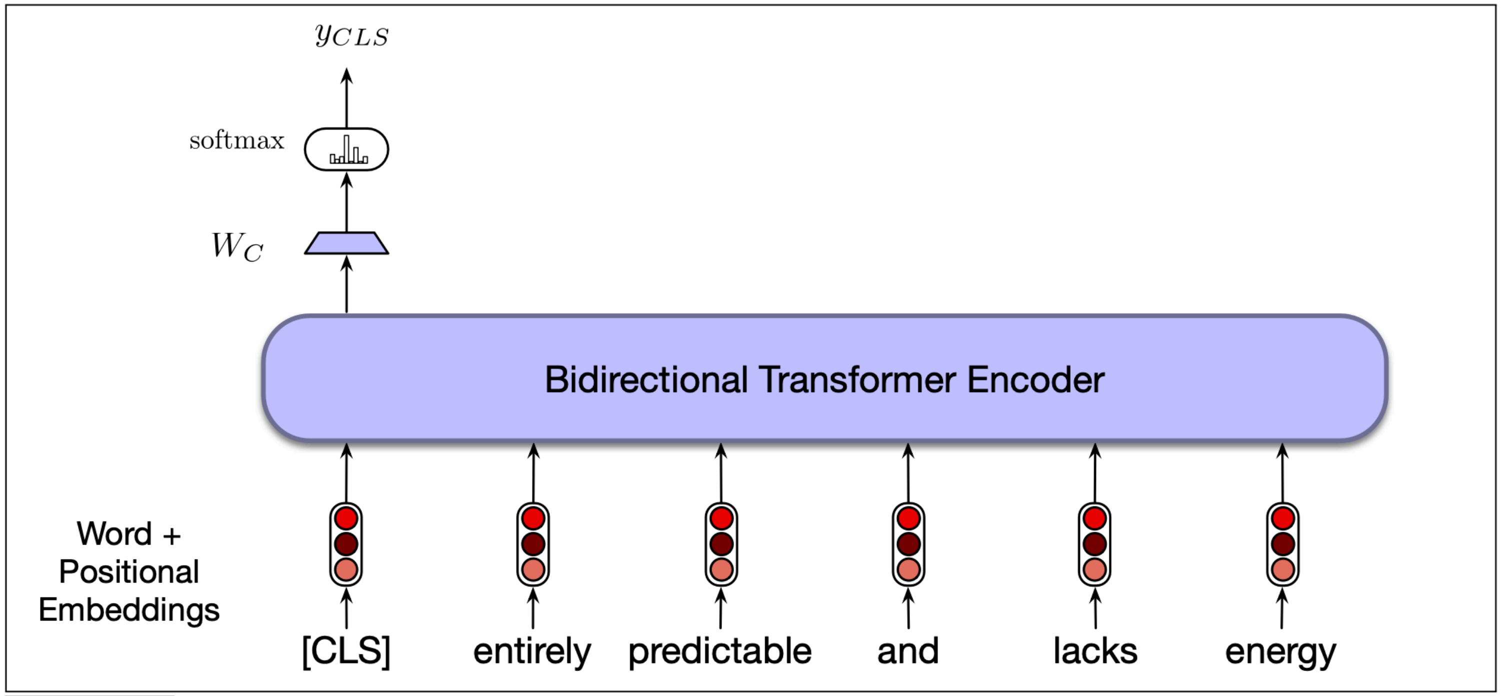

y_{CLS}
softmax
W_C
Bidirectional Transformer Encoder
Word + Positional Embeddings
[CLS]
entirely
predictable
and
lacks
energy

Figure 11.8 Sequence classification with a bidirectional transformer encoder. The output vector for the `[CLS]` token serves as input to a simple classifier.

Summary

Pre-trained transformer models (not just BERT) can a highly useful and versatile tool for text analysis; some of them include

- Contextual embeddings to study word meaning (last week)
- Topic modeling (last week)
- Other downstream tasks (e.g., text classification)

Guided Coding

- Identifying violent threats with BERT fine-tuning (simpletransformers): here
- Stance detection on abortion with BERT fine-tuning: here