Neural NLP II

HSS 510: NLP for HSS

Taegyoon Kim

May 29, 2024

Agenda

Things to be covered

- BERT pre-training and fine-tuning
- Fine-tuning pre-trained model
- BERTopic

Attention and Transformer

Example from last week

- *Open a bank account* $\rightarrow e_{bank_{s1}}$: [0.3, 0.9, …]
- *On the river bank* $\rightarrow e_{bank_{s2}}$: [0.8, 0.1, …]

To capture contextual meanings, the embeddings (i.e., representations) of the tokens are updated based on the relationships between tokens

- $e_{bank_{s1}}$ should be similar to $e_{account_{s1}}$, and $e_{bank_{s2}}$ should be similar to $e_{river_{s2}}$

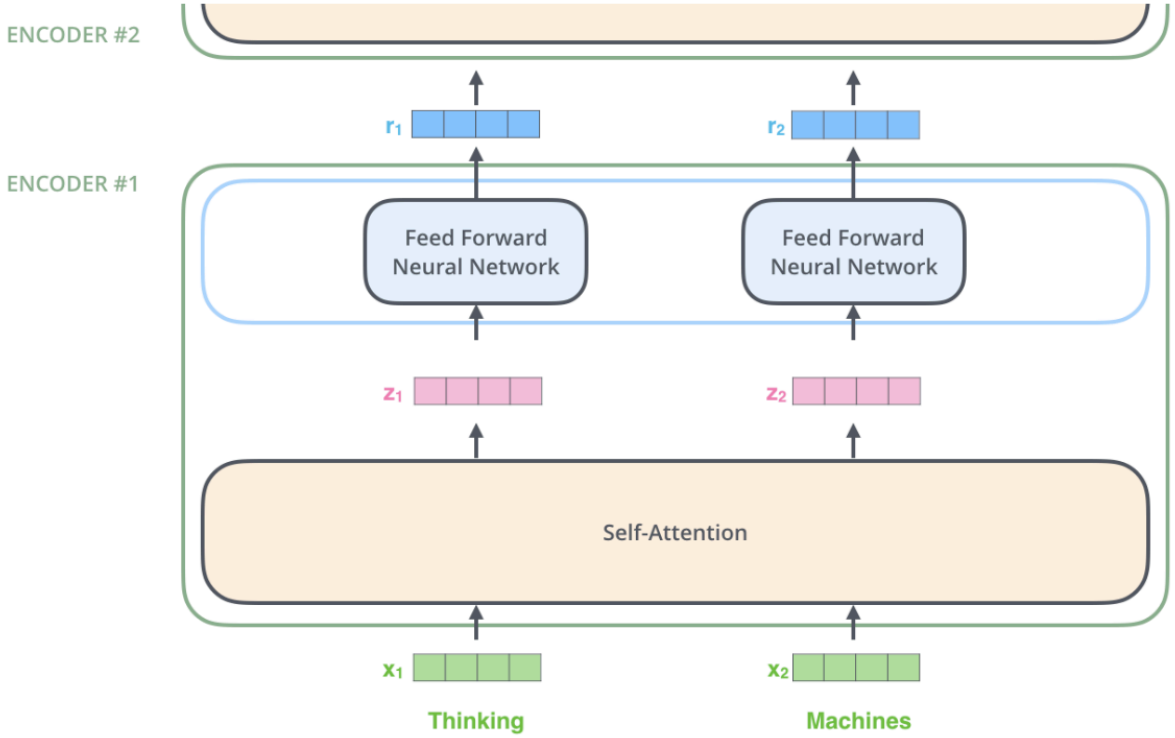
Attention and Transformer

What is Transformer

- Neural network architecture with self-attention mechanisms to process and generate sequences of data
- Consists of a stack of layers (either encoders, decoders, or both, with each layer incorporating self-attention mechanisms and FNN)
- Transformers dispense with recurrence and rely entirely on attention mechanisms
  - Capturing long-range dependencies
  - Parallelizing computations efficiently

Attention and Transformer

Transformer encoder (attention layer + FNN)



BERT

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

- A form of transformer trained as a language model
- A stack of bi-directional transformer encoders (no decoder)
  - C.f., GPT family is uni-directional
- Two versions of the model introduced in the paper
  - BERT BASE (12 transformer encoders)
  - BERT LARGE (24 transformer encoders)
- Each token is represented with 768/1024 dimensions (BASE/LARGE)

BERT

Task 1: Masked Language Model

- The MLM training objective is to predict the original inputs for each of the masked tokens
- Comparison with regular LM
  - Regular LM: Please turn your homework ____.
  - MLM: Please turn ____ homework in.
- 15% of the input tokens in a training sequence are sampled for learning
  - Out of these, 80% are replaced with `[MASK]`

BERT

Task 1: masked language model

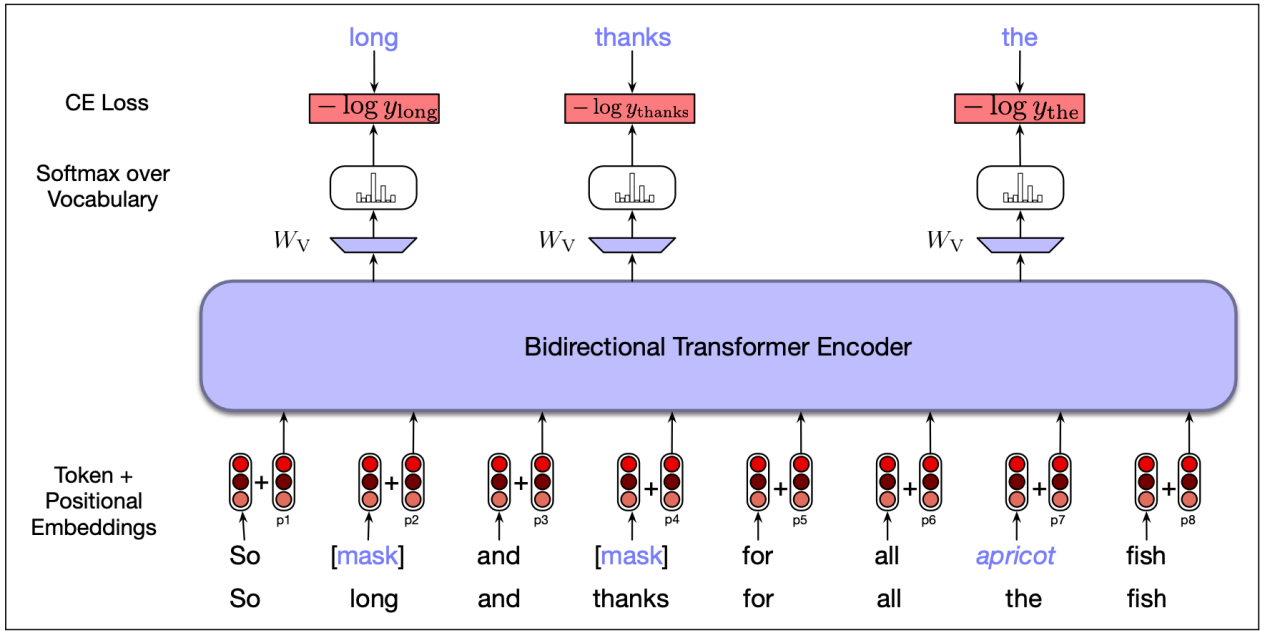


Figure 11.5 Masked language model training. In this example, three of the input tokens are selected, two of which are masked and the third is replaced with an unrelated word. The probabilities assigned by the model to these three items are used as the training loss. (In this and subsequent figures we 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.)

BERT

Task 2: Next Sentence Prediction

- MLM is designed for effective *word-level representations*
- An important class of applications involves knowledge of *relationships between sentences*
- Thus, the model also learns to predict whether one sentence follows another
- The model is presented with pairs of sentences and predicts whether each pair consists of an actual pair of adjacent sentences

BERT

Task 2: Next Sentence Prediction

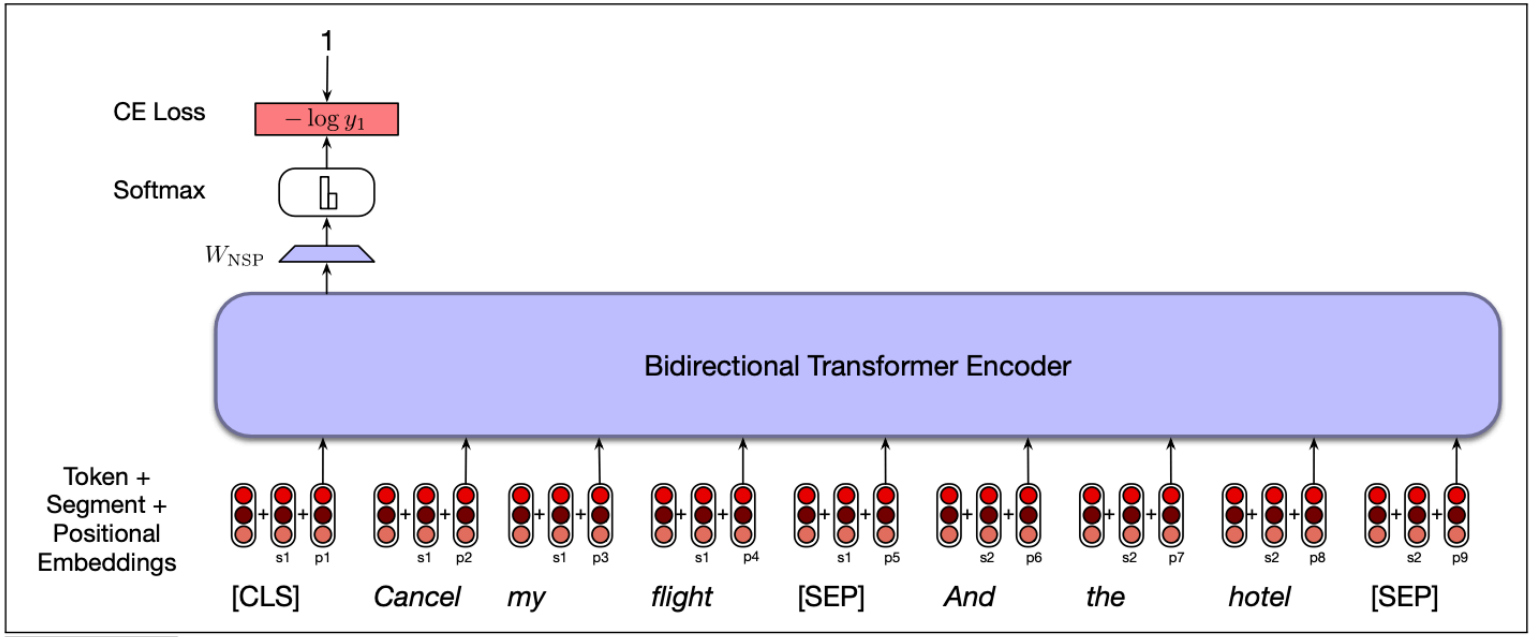
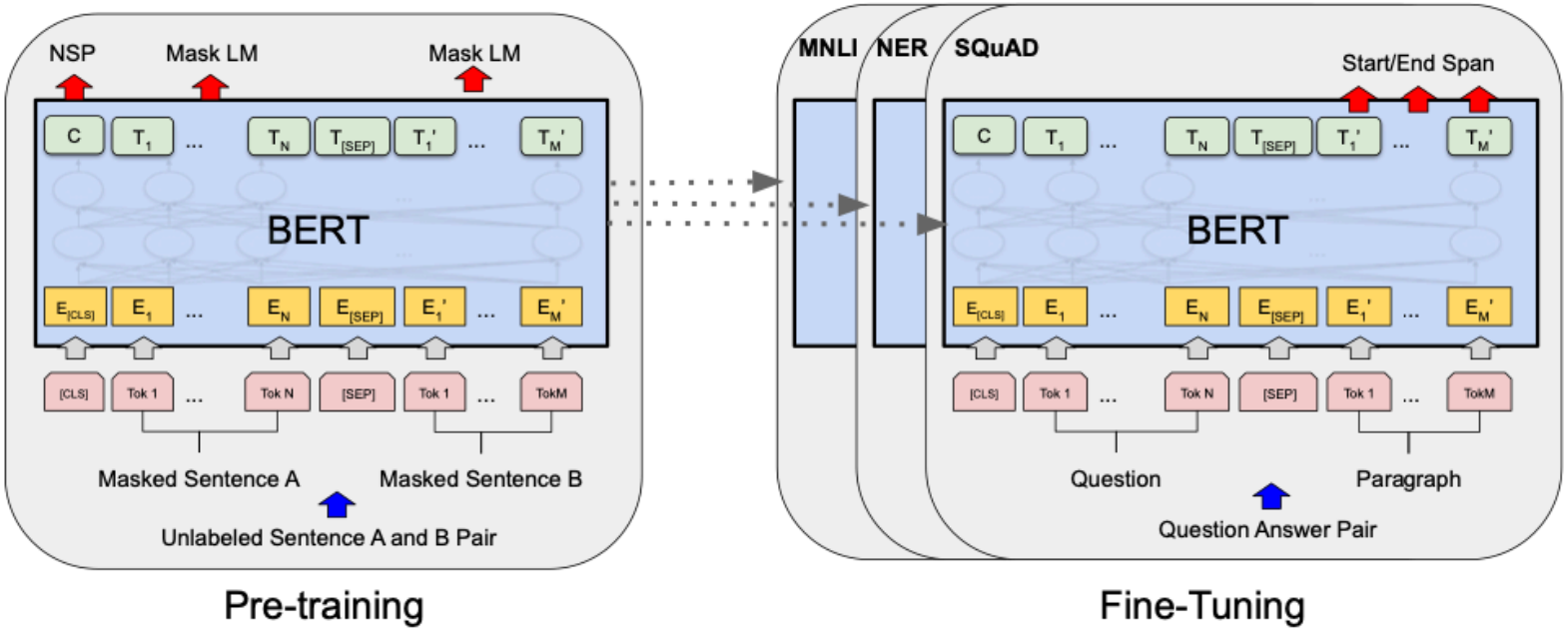


Figure 11.7 An example of the NSP loss calculation.

BERT

Pre-training vs. Fine-tuning



Fine-tuning

What is fine-tuning, and why?

- Through pre-training, language models extract generalizable knowledge from massive training data
- This can be highly useful for a wide range of down-stream tasks
- Fine-tuning facilitates the creation of applications *on top of* pre-trained models by adding a set of application-specific parameters

→ It tends to outperform models trained from scratch given the same amount of (application-specific) training data

Fine-tuning

Overall process of fine-tuning BERT

- `[CLS]` token plays the role of representing the entire sequence
- It is added to the vocabulary and is pre-pended to the start of all input sequences
- An additional 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
  - The input to the added/last layer that makes classifications

Fine-tuning

Overall process of fine-tuning BERT

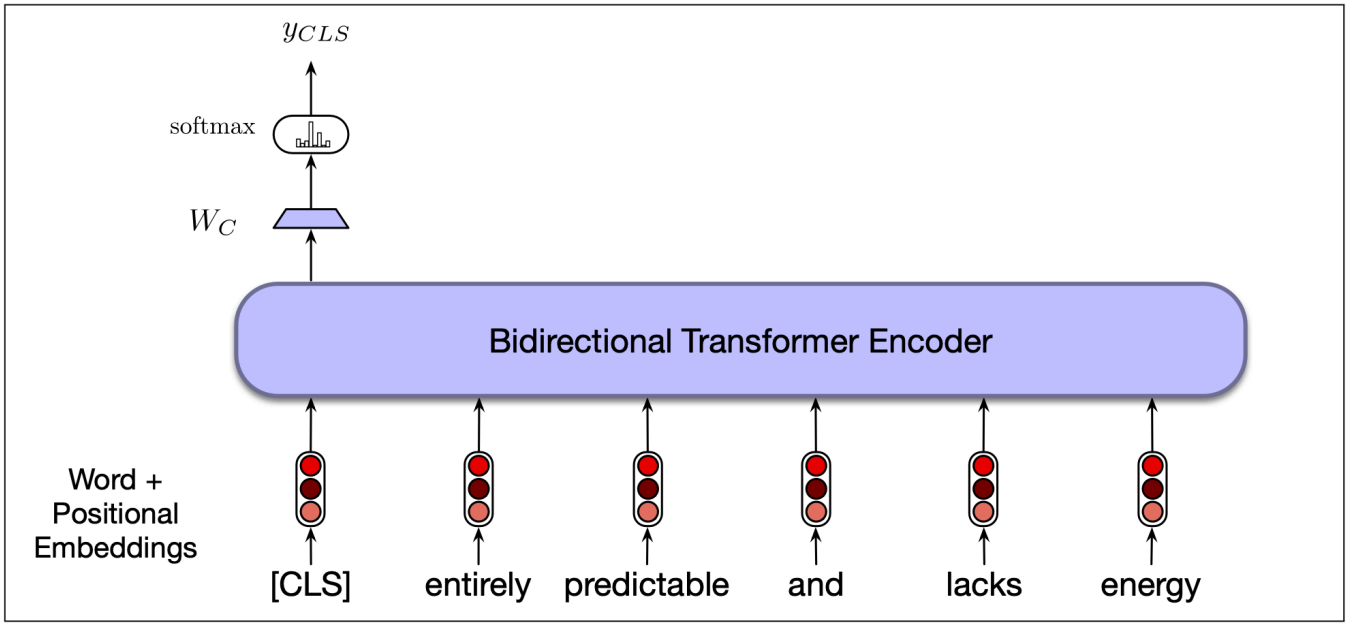
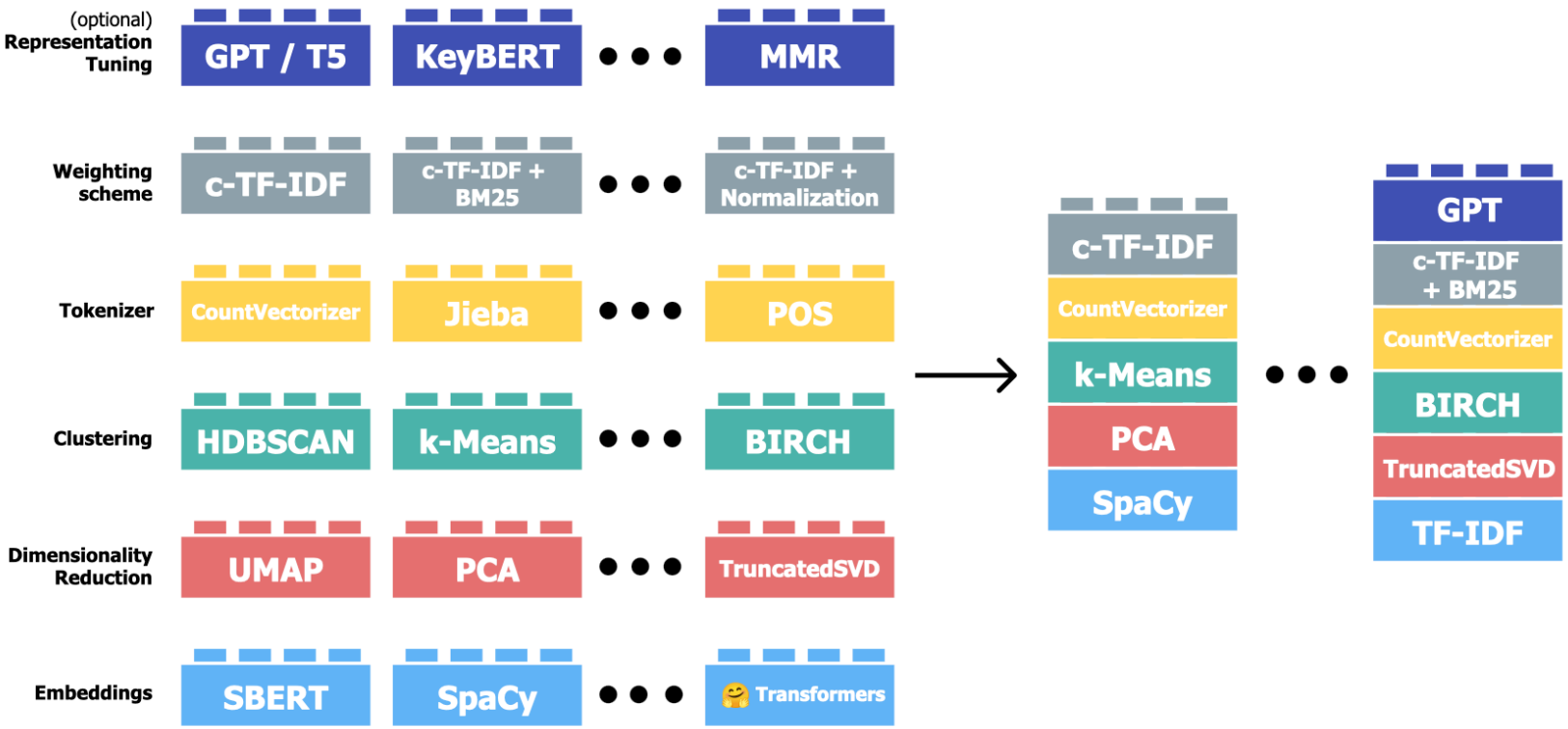


Figure 11.8 Sequence classification with a bidirectional transformer encoder. The output vector for the [CLS] token serves as input to a simple classifier.

BERTopic

Modular approach to topic modeling



BERTopic

Main steps

- Embedding documents
- Reducing dimensionality of embeddings
- Clustering reduced embeddings into groups
- Represent topics with one or multiple representations

BERTopic

Main steps

- Embedding documents (**S-BERT**)
- Reducing dimensionality of embeddings (**UMAP**)
- Clustering reduced embeddings into groups (**HDBSCAN**)
- Represent topics with one or multiple representations (**c-TF-IDF**)

BERTopic

- Assumes no generative process
  - Unlike LDA, CTM, STM, etc.
- Semantically rich: good for short texts
- With embeddings form proprietary models, it is not quite transparent
- Guide coding involves detailed discussions of BERTopic in general and its modules, but ***is modularity good?***

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
- Topic modeling
- Classification and scaling

Guided Coding

- BERTopic: here
- Text classification with BERT fine-tuning: here