Stanford University

CS 329P : Practical Machine Learning (2021 Fall)

11.3 Fine-Tuning in NLP

Qingqing Huang, Mu Li, Alex Smola

https://c.d2l.ai/stanford-cs329p

Self-supervised pre-training

- No large-scale labeled NLP dataset
- Large quantities of unlabeled documents
  - Wikipedia, ebooks, crawled webpages
- Self-supervised pre-training
  - Generate “pseudo label” and use supervised learning task
  - Common tasks for NLP
    - Language model (LM): predict next word. e.g. I like your hat
    - Masked language model (MLM): random masked word prediction. e.g. I like your hat

Pre-trained Models

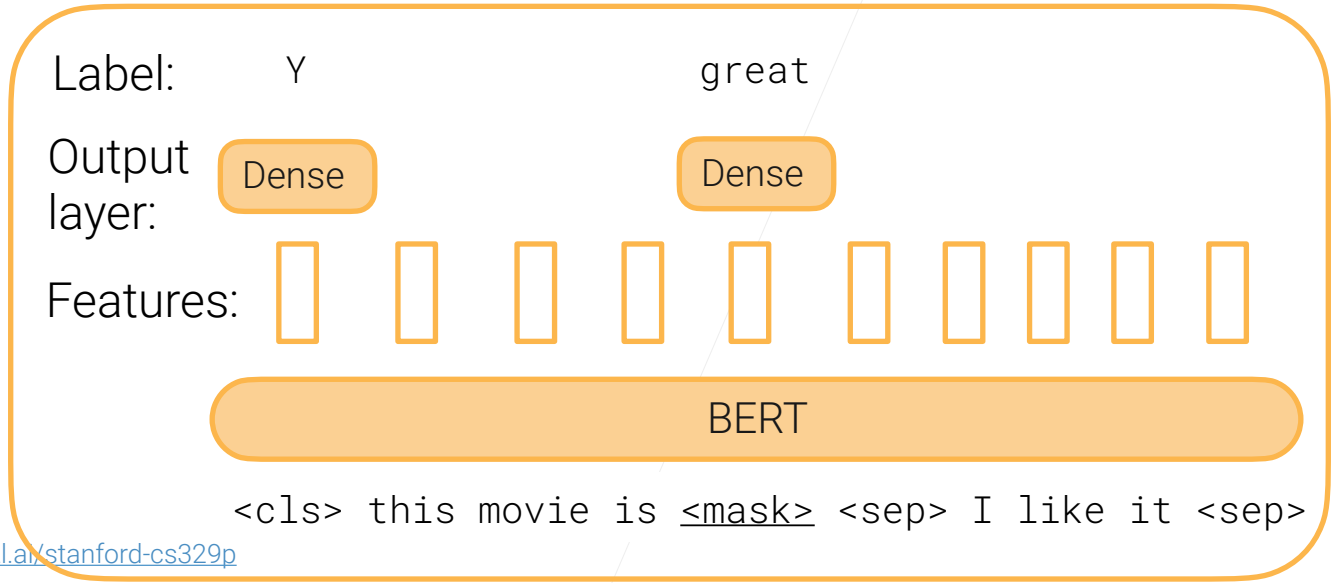
- Word embeddings: learn embeddings $\mathbf{u}_w$ and $\mathbf{v}_w$ for each word w
  - The masked word y can be predict by context words $x_1, \ldots, x_n$ via

$$\text{argmax}_y \, \mathbf{u}_y^T \sum_i \mathbf{v}_{x_i} \quad \text{(CBOW)}$$

  - Embeddings $\mathbf{u}$ are used by other applications
- Transformer based pre-trained models
  - BERT: a transformer **encoder** trained with masked words prediction and next sentence prediction
  - GPT: a transformer **decoder** (will cover in prompt learning)
  - T5: a transformer **encoder-decoder** trained to fill span of text from documents

BERT

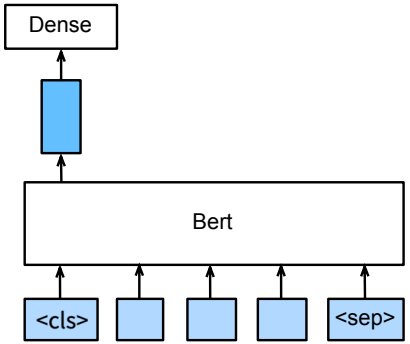
- Pre-training tasks: masked token prediction, next sentence prediction
- Pre-trained on Wikipedia and BookCorpus (>3B words)
- Many versions: base / large, English / multilingual, cased / uncased
- Multiple variants
  - ALBERT
  - ELECTRA
  - RoBERTa
  - ....



BERT Fine-tuning

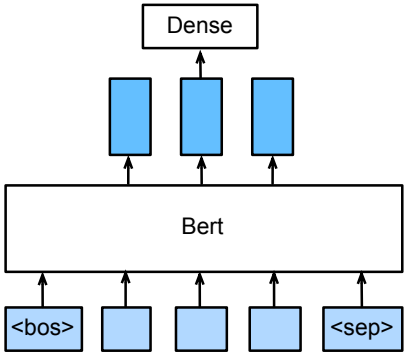
- Randomly initialize the last layer, train a few epochs with small lr
- Downstream task examples

Sentence classification



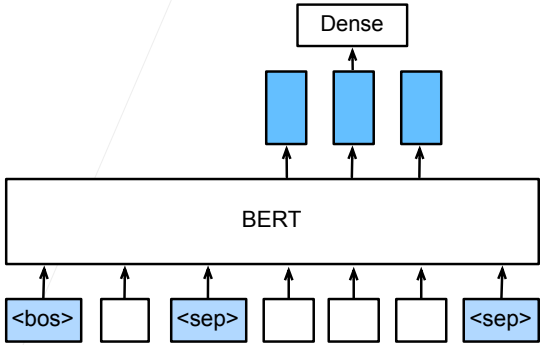
Use CLS embedding as linear classifier input.

Named-entity recognition



For each token, use the hidden layer output to predict named entity tag.

Question Answering



First sentence is the question, second sentence is the reference, predict span in the reference as the answer.

Practical Considerations

- BERT fine-tuning on small datasets can be unstable
  - BERT removed bias correction steps in Adam
  - Too few (=3) epochs
- Randomly initializing some top transformer layers help
  - Features learned by top layers are too specific to the pre-training tasks
  - The cutoff depends on downstream tasks

Where to Find Pre-trained Models

- HuggingFace: a collection of pre-trained transformer models on both PyTorch and TensorFlow

```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification

tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")
inputs = tokenizer(sentences, padding="max_length", truncation=True)
model = AutoModelForSequenceClassification.from_pretrained(
  “bert-base-cased", num_labels=2)
# Train model on inputs as a normal training job
```

Applications

- "(BERT) obtains new state-of-the-art results on eleven natural language processing tasks", including
  - If a sequence of words is a grammatical English sentence
  - Sentiment of sentences from movie reviews
  - Sentences/questions in a pair are semantically equivalent, or similar
  - If the premise entails the hypothesis
  - Find the span of the answer for a question
- "(T5) achieve state-of-the-art results on many benchmarks covering summarization, question answering, text classification, and more"

Summary

- Self-supervised pre-training for NLP models
  - A common task is (masked) language model
- BERT is a giant transformer encoder
- Downstream tasks fine-tune BERT with a consistent manner