

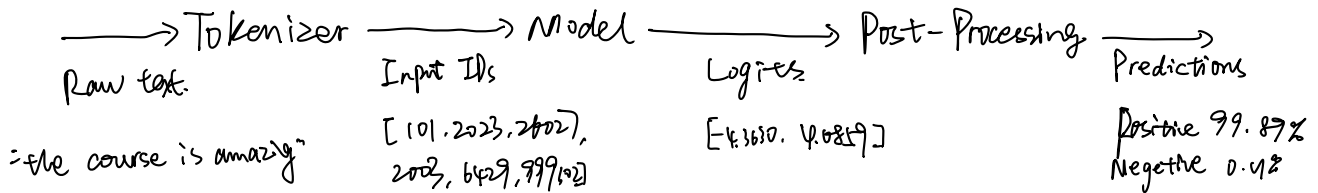
transformer 家族架构

Encoder-only (bert): 适合文本分类、实体识别、关键信息抽取

Decoder-only (GPT): 适合文本生成

Seq2Seq (BART/TL): 总结、翻译、对话

transformers 架构



1. 我们使用 tokenizer 必须与模型在训练时的一致。⇒ 词表。

↳ AutoTokenizer 的 from_pretrained 方法

参数:

- text: string, string list, list of list ...
- padding: 填充
- truncation: 截断
- max_length: 设置最大句长
- return_tensor: 返回类型

2. Model.

w/ transformer 基础模型为 `distilbert-base-uncased-finetuned-sst-2-english`

输出是 hidden states (即文本的向量表示)

AutoModel.from_pretrained(...)

3个维度:

- 1. batch size
- 2. sequence length
- 3. hidden size

Example: ["what a nice day! wow ~", "Hello world"]

↓ 假设取成 2D 张量 (2个句子)

model → last_hidden_state

$2 \times 12 \times 768 \rightarrow \text{hidden-size}$

batch size → seq-len

3. Model-Head.

接在基础模型后面, 将 hidden-states 文本进一步处理, 用于具体任务

Head - 开发由若干线性层构成

基础的 Model 不包括 Head, 有些模型 ... for ...

4. Post-Processing

后续处理.

BPE or "Byte-Pair Encoding" (用来得到标准的 subword tokenization)

step 1. 预分词

→ 英: 按空格, 标点.

中: 按字, jieba

⇒ 得到: 原始词集合 合词频

e.g. ("hug", 10), ("pug", 5), ("pun", 12) ...

step 2 构建基础词表 并添加组合规则

(base vocab)

(merge rules)

e.g. 对英文, 我们选择字母, 生成基础词表 ["b", "g", "h", "n" ...]

(("h", "u", "g", 10), ("p", "u", "g", 5), ("p", "u", "n", 12) ...)

从上述统计结果中, 找出出现次数最多的符号对.

$h+u = 10+5 = 15$ 次

$u+g = 10+5+5 = 20$ 次

$p+u = 12$ 次

...

发现 $u+g$ 最多, 把 "ug" 加入基础词表.

同时词频统计也变了

(("h", "ug", 10), ("p", "ug", 5), ...)

step 3. 重复执行 step 2, 直到达到预设的词表规模

(vocab_size)

step 4: 分词: 碰到词表中没有的词

e.g. "bug" < "b" "ug"