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/T5): 适合: 翻译, 对话

tarnsformers 背后.

→ Tokenizer → Model → Post-Processing →

Raw text: "the course is amazing"

Input IDs: [101, 2023, 2607, 2003, 6429, 999, 102]

Logits: [-4.3630, 4.6859]

Predictions: Positive 99.89%, Negative 0.11%

1、我们使用的tokenizer必须与模型在预训练时的一致. ⇒ 词表.

↳ AutoTokenizer 的 from_pretrained 方法

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

2. Model.

以 transformer 基础模型为例子 ("distilbert-base-uncased-finetuned-sst-2-english")

输出就是 hidden states (即文本的向量表示)

AutoModel.from_pretrained(...)

三个维度
1. batch size
2. sequence length.
3. hidden size

["what a nice day! wow ~", "Hello world"] —model→ last_hidden_state

假设转成ID之后有12个(单词的向量)

2×12×768 → hidden_size; batchsize; 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 一般由若干线性层构成

单纯的 Model 不包括 Head, 有些模型 ...for... 包含)

4. Post-Processing

后续处理.

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

常见会出现的词不应该被切分成更小的单位，但不常出现的词应该被切分成更小的单位

step1. 预分词
以英：按空格，标点。
中：按字，jieba ⇒得到：原始词集合词频
e.g ("hug", 10), ("pug", 5), ("pun", 12) ~

step2 构建基础词表并开始学习合并规则
(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。
用时词频统计也变为
("h", "ug", 10), ("p", "ug", 5), ...

step3. 重复执行 step2，直到达到预设的词表规模
(vocab_size)

step4：分词：碰到词表中没有的词
e.g "bug" < "b" "ug"