03

实战演练之机器阅读理解

（1）机器阅读理解任务介绍

（2）基于Transfromers的解决方案

（3）代码实战演练（上）

（4）代码实战演练（下）

机器阅读理解任务介绍

机器阅读理解简介

- **什么是机器阅读理解任务**
  - 机器阅读理解（Machine Reading Comprehension，简称MRC）是一项通过让机器回答基于给定上下文的问题来测试机器理解自然语言的程度的任务，简单来说即**给定一个或者多个文档P，以及一个问题Q，输出问题Q的答案A。**
  - 机器阅读理解任务的形式是较为多样化的，常见的类型包括完型填空式、答案选择式的、片段抽取式的、自由生成式，本次课程讲解的内容为**片段抽取式的机器阅读理解，即问题Q的答案A在文档P中，A是P中的一个连续片段。**

机器阅读理解任务介绍

机器阅读理解简介

- **机器阅读理解任务样例**

工商协进会报告，12月消费者信心上升到78.1，明显高于11月的72。另据《华尔街日报》报道，2013年是1995年以来美国股市表现最好的一年。这一年里，投资美国股市的明智做法是追着“傻钱”跑。所谓的“傻钱”策略，其实就是买入并持有美国股票这样的普通组合。这个策略要比对冲基金和其它专业投资者使用的更为复杂的投资方法效果好得多。

Q1：消费者信心指数由什么机构发布?

A1：工商协进会

Q2：12月的消费者信心指数是多少?

A2： 78.1

Q3：什么是傻钱策略?

A3： 买入并持有美国股票这样的普通组合

如何为机器阅读理解任务建模？

机器阅读理解任务介绍

机器阅读理解简介

- **机器阅读理解任务样例**

工商协进会报告，12月消费者信心上升到78.1，明显高于11月的72。另据《华尔街日报》报道，2013年是1995年以来美国股市表现最好的一年。这一年里，投资美国股市的明智做法是追着“傻钱”跑。所谓的“傻钱”策略，其实就是买入并持有美国股票这样的普通组合。这个策略要比对冲基金和其它专业投资者使用的更为复杂的投资方法效果好得多。

Q1：消费者信心指数由什么机构发布?

A1：工商协进会

Q2：12月的消费者信心指数是多少?

A2： 78.1

Q3：什么是傻钱策略?

A3： 买入并持有美国股票这样的普通组合

如何为机器阅读理解任务建模？

定位答案在文档中的起始位置和结束位置

机器阅读理解任务介绍

机器阅读理解简介

- **机器阅读理解数据集格式**
  - CMRC2018

```
{
    "context": "《战国无双3》是由光荣和ω-force开发的战国无双系列的正统第三续作。本作以三大故事为主轴，分别是以武田信玄等人为主的《关东三国志》，织田信长等人为主的《战国三杰》，石田三成等人为主的《关原的年轻武者》，丰富游戏内的剧情。此部份专门介绍角色，欲知武...",
    "id": "DEV_0_QUERY_0",
    "question": "《战国无双3》是由哪两个公司合作开发的？",
    "answers": {
            "answer_start": [11, 11],
            "text": ["光荣和ω-force", "光荣和ω-force"]
     },
}
```

机器阅读理解任务介绍

机器阅读理解简介

- **评估指标**
  - 精准匹配度（Exact Match，EM）：计算预测结果与标准答案是否完全匹配。
  - 模糊匹配度（F1）：计算预测结果与标准答案之间字级别的匹配程度。
- **简单示例**
  - 数据
    - 模型预测结果：北京
    - 真实标签结果：北京天安门
  - 计算结果
    - EM = 0， P = 2/2 R=2/5 F1 = (2 * 2/2 * 2/5) / (2/2 + 2/5) = 4/7 ~=0.57

基于Transformers的解决方案

- **数据预处理**
  - 数据处理格式

  - 如何准确定位答案位置
    - start_positions / end_positions
    - offset_mapping
  - Context过长如何解决
    - 策略1 直接截断，简单易实现，但是会损失答案靠后的数据，因为无法定位答案
    - 策略2 滑动窗口，实现较为复杂，会丢失部分上下文，但是综合来看损失较小

代码实战演练

基于滑动窗口的数据处理

- **滑动窗口**
  - Context

  - Query

  - Input

代码实战演练

基于滑动窗口的数据处理

- **带有overflow的滑动窗口**
  - Context

  - Query

  - Input

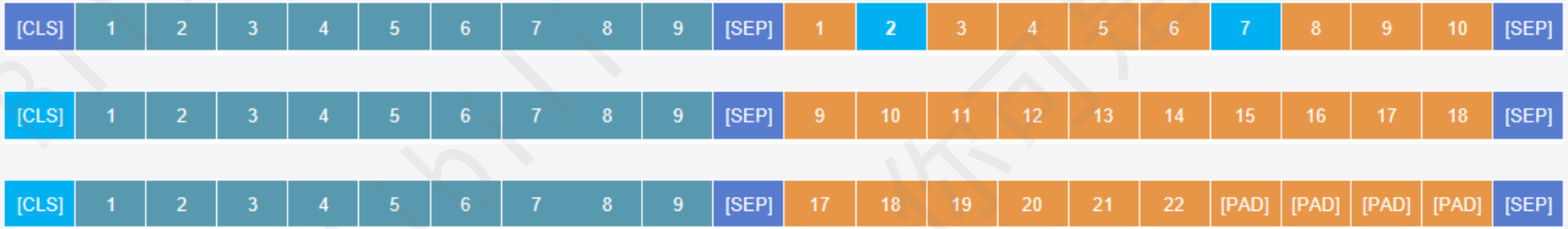
代码实战演练

基于滑动窗口的结果预测

- **结果预测**
  - Context

  - Query

  - Input



基于Transformers的解决方案

- **模型结构**
  - *ModelForQuestionAnswering

```python
class BertForQuestionAnswering(BertPreTrainedModel):
    _keys_to_ignore_on_load_unexpected = [r"pooler"]

    def __init__(self, config):
        super().__init__(config)
        self.num_labels = config.num_labels

        self.bert = BertModel(config, add_pooling_layer=False)
        self.qa_outputs = nn.Linear(config.hidden_size, config.num_labels)

        # Initialize weights and apply final processing
        self.post_init()
```

基于Transformers的解决方案

- **模型结构**
  - *ModelForQuestionAnswering

```
outputs = self.bert(
    input_ids,
    attention_mask=attention_mask,
    token_type_ids=token_type_ids,
    position_ids=position_ids,
    head_mask=head_mask,
    inputs_embeds=inputs_embeds,
    output_attentions=output_attentions,
    output_hidden_states=output_hidden_states,
    return_dict=return_dict,
)

sequence_output = outputs[0]

logits = self.qa_outputs(sequence_output)   # [bs, seq_len, 2]
start_logits, end_logits = logits.split(1, dim=-1)  # [bs, seq_len, 1], [bs, seq_len, 1]
start_logits = start_logits.squeeze(-1).contiguous()     # [bs, seq_len]
end_logits = end_logits.squeeze(-1).contiguous()         # [bs, seq_len]
```

基于Transformers的解决方案

- **模型结构**
  - *ModelForQuestionAnswering

```
total_loss = None
if start_positions is not None and end_positions is not None:
    # If we are on multi-GPU, split add a dimension
    if len(start_positions.size()) > 1:
        start_positions = start_positions.squeeze(-1)
    if len(end_positions.size()) > 1:
        end_positions = end_positions.squeeze(-1)
    # sometimes the start/end positions are outside our model inputs, we ignore these terms
    ignored_index = start_logits.size(1)
    start_positions = start_positions.clamp(0, ignored_index)
    end_positions = end_positions.clamp(0, ignored_index)

    loss_fct = CrossEntropyLoss(ignore_index=ignored_index)
    start_loss = loss_fct(start_logits, start_positions)
    end_loss = loss_fct(end_logits, end_positions)
    total_loss = (start_loss + end_loss) / 2
```

代码实战演练（上，截断策略版本）

- **数据集**
  - cmrc2018
- **预训练模型**
  - hfl/chinese-macbert-base
- **数据集处理方式**
  - 对context进行截断处理

代码实战演练（下，滑动窗口版本）

- **数据集**
  - cmrc2018
- **预训练模型**
  - hfl/chinese-macbert-base
- **数据集处理方式**
  - 对context进行滑动窗口处理
- **相关包安装**
  - `pip install nltk`
  - `nltk.download("punkt")`