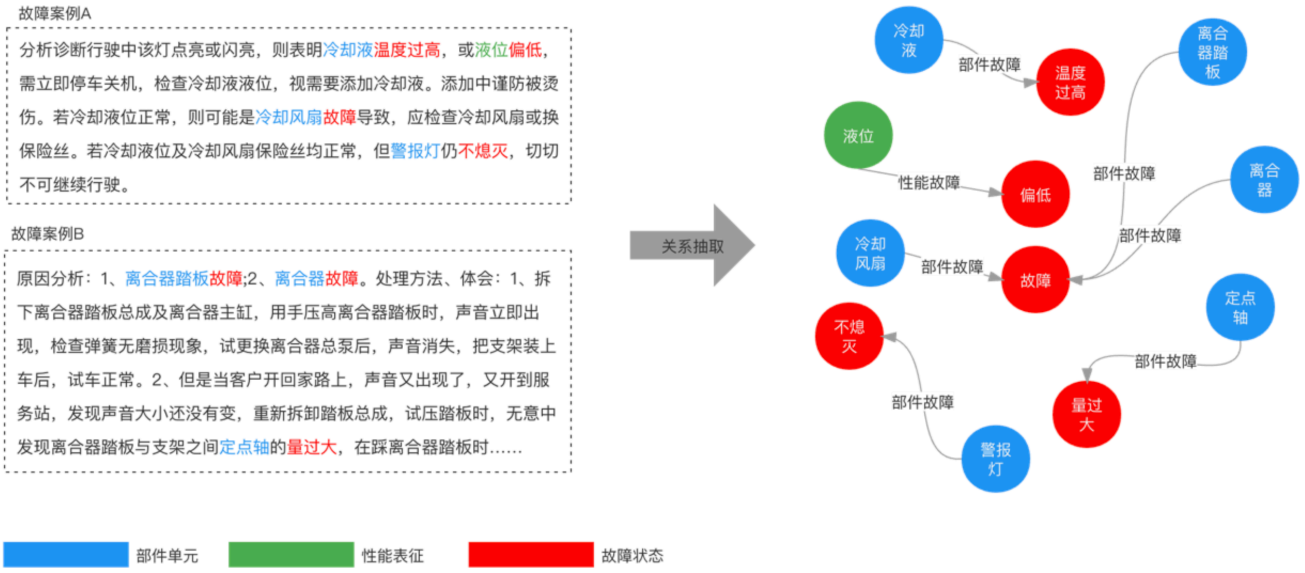
一、选题

赛题任务：[高端装备制造知识图谱自动化构建技术评测任务]

通过从大量故障案例文本抽取出部件单元、性能表征、故障状态、检测工具等实体及其关系，可以为后续高端装备制造业故障知识图谱构建和故障智能检修和实时诊断打下坚实基础。本任务需要从故障案例文本自动抽取4种类型的关系和4种类型的实体。关系类型为：部件单元的故障状态、性能表征的故障状态、部件单元和性能表征的检测工具、部件单元之间的组成关系。



通过从大量故障案例文本抽取出部件单元、性能表征、故障状态、检测工具等实体及其关系，可以为后续高端装备制造业故障知识图谱构建和故障智能检修和实时诊断打下坚实基础。本任务需要从故障案例文本自动抽取4种类型的关系和4种类型的实体。关系类型为：部件单元的故障状态、性能表征的故障状态、部件单元和性能表征的检测工具、部件单元之间的组成关系。

实体类型

实体类型名称	说明	示例
部件单元	高端装备制造领域中的各种单元、零件、设备	“燃油泵”、“换流变压器”、“分离器”
性能表征	部件的特征或者性能描述	“压力”、“转速”、“温度”
故障状态	系统或部件的故障状态描述，多为故障类型	“漏油”、“断裂”、“变形”、“卡滞”
检测工具	用于检测某些故障的专用仪器	“零序互感器”、“保护器”、“漏电测试仪”

关系类型

主体	客体	关系	主体示例	客体示例
部件单元	故障状态	部件故障	发动机盖	抖动
性能表征	故障状态	性能故障	液面	变低
检测工具	性能表征	检测工具	漏电测试仪	电流
部件单元	部件单元	组成	断路器	换流变压器

二、目录树

```
├─── extra
│     ├── pretrain_models                                  # 预训练模型
├─── src
│     ├─── dataset                                         # 数据集文件夹
│          ├── train_bdci.json          # 比赛提供的原始数据
│          ├── rel2id.json              # 关系到id的映射
│          ├── evalA.json               # A榜评测集
│          ├── submit_A.json            # A榜提交结果
│          ├── train.json               # 数据处理的中间文件
│          ├── test.json                # 数据处理的中间文件
│     ├─── scripts                                          # 代码集文件夹
│          ├── data_generator.py        # 数据处理工具
│          ├── predict.py               # 预测代码
│          ├── requirements.txt         # 依赖库
│          ├── train.py                 # 训练代码
│          ├── data_utils.py            # 数据处理类
│          ├── model.py                 # 模型类
│          └── util.py                  # 工具类
```

由于pretrain_models大于50M，放在百度网盘中链接: https://pan.baidu.com/s/1gfq8kE1mOO7cjks2BzoMvg
提取码: xz8w

三、运行方式

3.1 安装环境依赖

使用pip指令安装requirement.txt文件中的依赖包

3.2 生成模型数据

执行python data_generator.py生成train.json和test.json

3.3 模型训练

执行 python train.py

3.4 模型预测

```
执行 python predict.py
```

4. 5 注意事项

```
1.所用模型：GRTE；github：https://github.com/neukg/GRTE
2.batch_size不可设置过大。本实验采用的运行环境是GPU RTX 3090*1（显存24GB）和24核CPU，batch_size设置为4。
```

四、模型构建

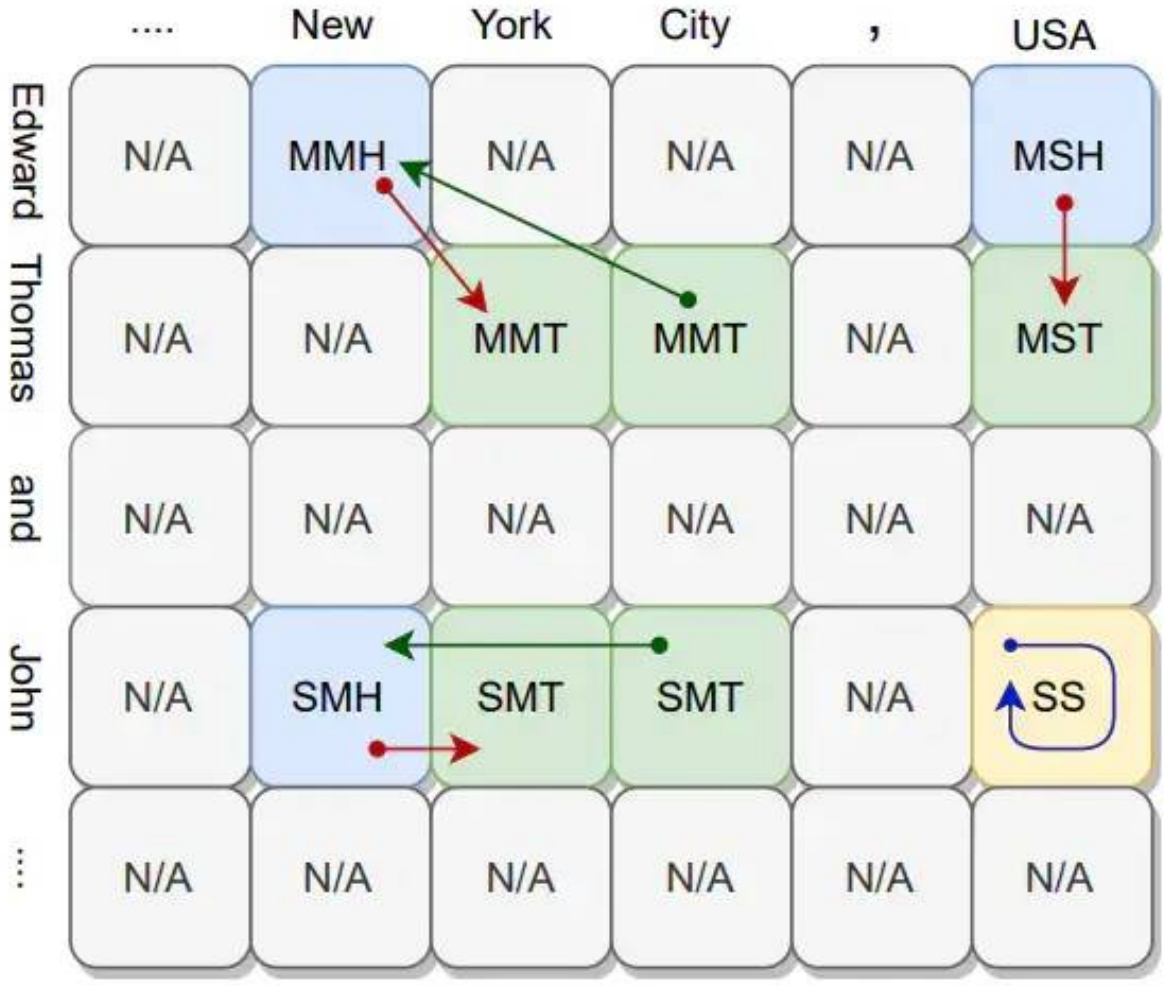
4.1 表格填充

三元组一般以(subject, relation, object)的形式表示客观存在的一个知识。比如，(中国，首都，北京)可以表示“中国的首都是北京”这一事实。**在三元组中，subject和object均为实体，relation为关系。**三元组抽取任务是在给定输入文本（一般以句子为单位）的条件下，从中自动地抽取出文本所包含的三元组信息，和知识图谱自动构建等下游任务非常契合。

对一个文本“Edward Thomas and John are from New York City, USA.”例子来说，其中关系live in存在以下(subject,object) pair: (EdwardThomas, New York City), (Edward Thomas, New York), (Edward Thomas, USA), (John, New York City), (John, New York) and (John, USA)

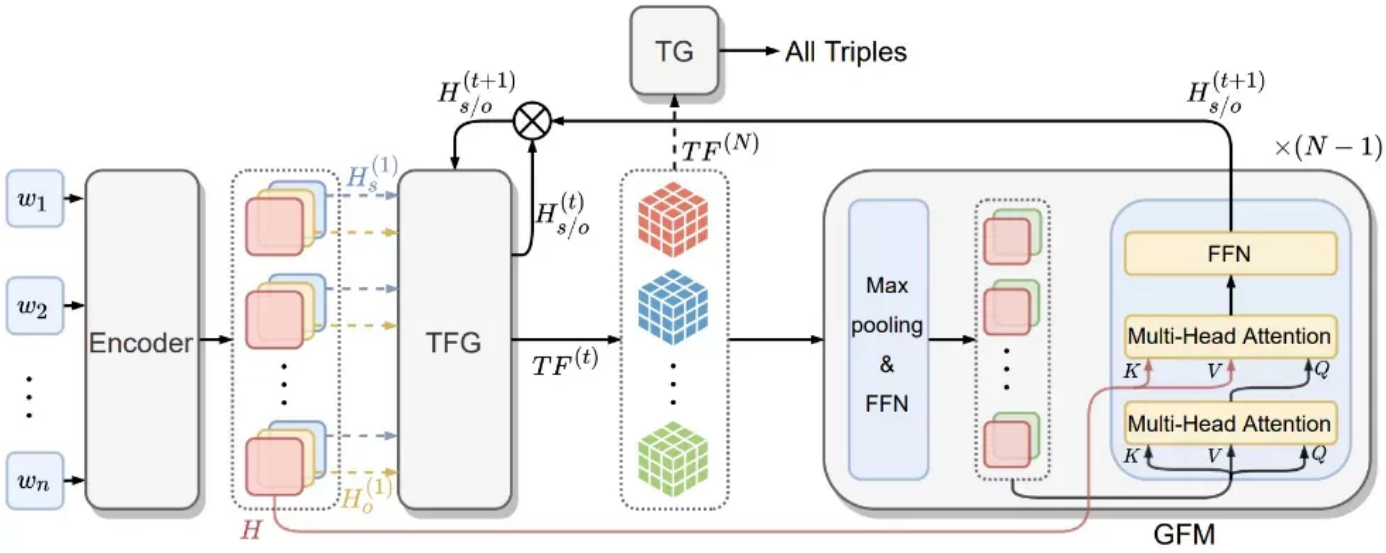
接着按这样的方式定义：将文本句子横纵排行，形成一个的表格，用**(w_i,w_j)**代表第i行与第j列的token pair，且**w_i**代表subject token，**w_j**代表object token，则**(w_i,w_j)**对应单元格填充的标签集合为 **{"N/A", "MMH", "MMT", "MSH", "MST", "SMH", "SMT", "SS"}**，标签中字母含义为：

- 第一个字母代表subject是多个token(M)，还是单个token(S);
- 第二个字母代表object是多个token(M)，还是单个token(S);
- 第三个字母代表token pair(w_i,w_j)是两个实体的开头(H)或结尾(T);
- 不存在上述关系的用N/A填充



需要注意的是H与T代表token pair同时是subject与obecct实体的开头或结尾。拿 (Edward Thomas, New York City)来说， token pair(Edward,New)是两实体的开头，两个实体又是多个token组成，则该token pair对应的label为MMH，token pair(Thomas,City)是两实体的结尾，则标签为MMT，token pair(John, USA)两实体都是单个token组成(single)，即label为SS。完整的转化对应关系可见上图。

4.2 模型构建



模型中包含4个部分：an Encoder module, a Table Feature Generation (TFG) module, a Global Feature Mining (GFM) module, and a Triple Generation (TG) module，分别对应为句子编码模块，表格特征学习模块，全局特征学习模块，三元组生成模块。另外，TFG与GFM是一个迭代学习过程。

给定一个输入句子，我们首先对其进行编码，抽取出句子特征。之后句子特征被输入进表特征生成模块中，生成初始的表特征。接着全局特征挖掘器利用max pooling和transformer进行表格和句子的交互，用以捕获全局特征，并将全局特征和句子特征进行信息融合作为下一次迭代时的句子特征输入进表特征生成模块。至此，**整个迭代过程形成了一个闭环**。经过多次迭代后，每个表对应的特征将被逐渐细化，我们依据最后一次迭代生成的表特征使用三元组抽取器进行表填充和表解码以得到最终的三元组结果。

特点：**可以大幅减少模型需要填充的元素个数(详细情况可以参考论文中对应的分析部分)**。

(1)关于句子编码

输入的文本是利用BERT进行编码，得到表征向量**H**，然后将其对接两个分开的全连接层分别表示subject 特征向量(**H_s**)和object特征向量(**H_o**)，这也代表迭代的第一步，即为：

$$
\begin{aligned}
H_s^{(1)} &= W_1 H + b_1 \\
H_o^{(1)} &= W_2 H + b_2
\end{aligned}
\qquad (1)
$$

(2)关于表格学习

对于一个token pair**(w_i,w_j)**，文中利用(**H_s**)与(**H_o**)进行Hadamard Product operation与全连接操作，来学习其对应的标签特征，即为

$$
TF_r^{(t)}(i,j) = W_r \,\mathrm{ReLU}(H_{s,i}^{(t)} \circ H_{o,j}^{(t)}) + b_r \quad (2)
$$

其中r代表第r种关系，也可理解为第r个表格，t代表迭代的轮次，圆圈代表Hadamard Product计算。

(3)关于全局特征的学习

关于全局特征前面已提及过，可直接理解为句子中三元组之间是存在相互影响的关系，下面就是展示如何学习这类特征的。具体分三步，假设迭代到第t轮:

- 第一步，将前面学习的每个表格特征进行concatenate操作形成一个融合的变量TF(t)，然后进行最大池化+FFN操作，分别生成t轮subject导向的表格向量TF_s和TF_o导向的表格向量，即为：

$$
\begin{aligned}
TF_s^{(t)} &= W_s \underset{s}{\mathrm{maxpool}}\left(TF^{(t)}\right) + b_s \\
TF_o^{(t)} &= W_o \underset{o}{\mathrm{maxpool}}\left(TF^{(t)}\right) + b_o
\end{aligned}
\qquad (3)
$$

- 第二步，使用Multi-Head Self-Attention进行交互计算，学习token pair与relation的全局关系信息，即为：

$$
\begin{aligned}
\hat{TF}_{s/o}^{(t)} &= \mathrm{MultiHeadSelfAtt}(TF_{s/o}^{(t)}) \\
\hat{H}_{(s/o)}^{(t+1)} &= \mathrm{MultiHeadAtt}(\hat{TF}_{s/o}^{(t)}, H, H) \\
H_{(s/o)}^{(t+1)} &= \mathrm{ReLU}(\hat{H}_{(s/o)}^{(t+1)} W + b)
\end{aligned}
\qquad (4)
$$

这里要说明下是，s/o代表TF_s和TF_o分别按这种方式计算，最后对应得到t+1步两个表征向量。此外，进行了两次Multi-Head Self-Attention计算，第一次是自我编码，第二次是将原始文本特征信息(H)融合起来编码。

- 第三步，考虑模型的深度和迭代的复杂度，导致训练中容易出现梯度消失的问题，论文在第二步的计算结果上加了一个残差网络，用LayerNorm方式来实现，即：

$$H_{(s/o)}^{(t+1)} = \text{LayerNorm}\left(H_{(s/o)}^{(t)} + H_{(s/o)}^{(t+1)}\right) \quad (5)$$

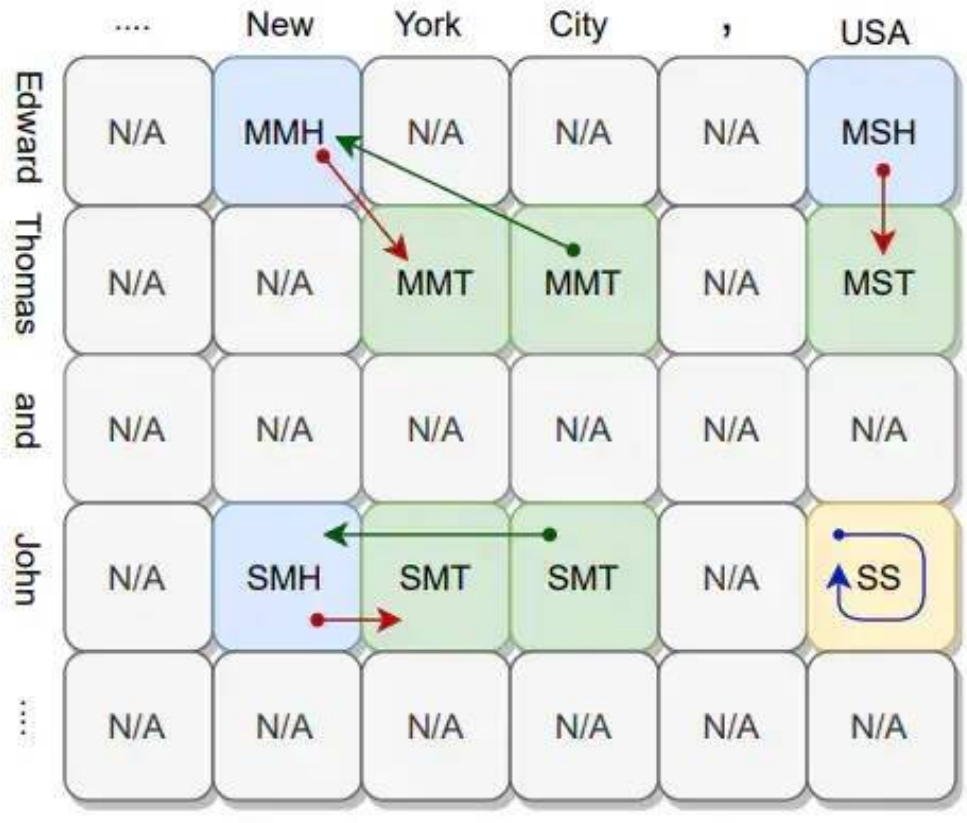
(4)关于三元组抽取

在前面迭代计算结束后，用最后一个轮学到的表格特征，预测各个关系表格中单元格值，按一定解码方式生成最终的三元组，即为：

$$\begin{aligned} \hat{table}_r(i,j) &= \text{softmax}\left(TF_r^{(N)}(i,j)\right) \\ table_r(i,j) &= \underset{l \in L}{\text{argmax}}\left(\hat{table}_r(i,j)[l]\right) \end{aligned} \quad (6)$$

2.3 解码输出

在学习到各个关系表格中的标签值后，进行解码，输出三元组的形式。文中提出三种解码方式，对照下图，在关系 live in 表中可解码为：



- **方式1: 前向解码**，subject与object都按前向的方向进行搜索。如(Edward,New) 识别为MMH，按行列前向方向搜索，搜索到(Thomas,York)为MMT，遇到T结束，则三元组为(Edward Thoms，live in，New York)，对应图中的红色箭头。
- **方式2：后向解码**，subject与object都按后向的方向进行搜索 (Edward, City)识别为MMT，按行列反方向搜索，则搜索的三元组为(Edward Thoms，live in，New York City)，对应图中的绿色箭头。
- **方式3：single解码**， 遇到SS标签，直接解析，如(John,USA)识别为SS，则直接解析三元组为(John, live in ,

USA)，对应图中的蓝色箭头。

一般情况，前向解码和single解码就够处理。

五、细节处理

1. 三种关系的标签数量非常不均衡，考虑使用数据增强的方式来改进。
2. 原始数据需要进行清洗，比如需要统一对空格进行替换。当使用bert 的 tokenizer 进行分词时，空格会被处理为空，如tokenizer.tokenize(' ') = []，这样就会造成 index 错误。

六、运行结果