任务二 USTC_MINE 解决方案

1 数据预处理

相比原始文本数据，我们做了以下预处理：

（1）对 info 文件的 item_pvs 字段进行合并同类项，例如，某个 item_pvs 中值为 “产地:江苏省;附加功能:储藏;附加功能:推拉;附加功能:多功能;”，合并后为“产地:江苏省;附加功能:储藏、推拉、多功能、拆装、可升降、可移动;”。

（2）从 item_pvs 中提取出“品牌”、“货号”、“型号”、“产地”、“材质”、“风格”属性作为新的字段，并从 item_pvs 中删除“品牌”、“配送安装地区”属性。

（3）去除 item_pvs 里的部分符号。

图片数据未做处理。

2 模型方法

2.1 文本模型

我们选择 roberta_base 和 roberta_large 作为文本的 encoder，使用 industry_name, cate_name, brand(预处理提取), title, item_pvs, sku_pvs，将这些字段直接拼接成为字符串,每个属性中间用“*”分隔,设置 maxlen 为 256，输出时我们有四种方案

（1） 第一层和最后一层的隐层取出，然后经过平均池化

（2） 取最后一层的隐层，然后平均池化

（3） 只取最后一层的 CLS 输出

（4） 普通的 bert 输出方式

最后我们采用方式二，即取最后一层的隐藏层，再过平均池化操作，然后经过 p=0.1 的 dropout，再接 128 维的全连接层，得到最终 embedding。

训练时，我们只开放后三层，其余层参数冻结。按照 8:1:1 划分训练、验证和测试集。设置学习率为 2e-5，阈值为 0.8，按此阈值在验证集上的最优 f1 值保存模型。

2.2 图像模型

经过各种尝试，我们最终选择 huggingface 中开源模型 Swin-Transformer (https://huggingface.co/microsoft/swin-large-patch4-window12-384-in22k)作为我们的 image encoder，后面接 FCN 层将 image encoder 输出结果 pooler_output 映射到 512 或 256 维作为 image embedding。由于图像模型训练起来较慢，故使用 autocast 与 GradScaler 进行混合精度计算以提升速度并降低显存。此外，图像模型的训练过程中也用到了针对于学习率的 warmup 和 linear decay 操作，使模型后期训练较为稳定。

训练时，我们只开放后两层，其余层参数冻结。数据集划分同文本模型，但是学习率改为 1e-5，阈值修改为 0.76，按照此阈值在验证集上的最高 f1 值保存模型。

2.3 基于样本对对比学习的 Loss 函数

由于该任务基于样本对进行训练与预测，我们采用苏剑林提出的 CoSENT 作为损失函数。即

$$\log\left(1+\sum_{(i,j)\in\Omega_{pos},(u,v)\in\Omega_{neg}} e^{\lambda(\cos(emb_u,emb_v)-\cos(emb_i,emb_j))}\right)$$

其中，Ω_{pos}表示正样本对集合，Ω_{neg}表示负样本对集合，$\cos(\cdot,\cdot)$表示计算样本对之间的 cos 相似度，emb_u表示商品 u 的 embedding，$\lambda > 0$表示超参数，一般取 20。

从本质上看，CoSENT 的优化目标是降低负样本对 cos 相似度、增加正样本对 cos 相似度，进而采用负样本对相似度减正样本对相似度的做法来扩大正负样本对之间的距离。这种做法的优点是无需预先确定判断某个样本对是正样本对还是负样本对的阈值。因为最优阈值在训练过程中动态变化，所以让模型自己确定阈值比我们事先设定阈值的效果要好。

其实上述 CoSENT 的核心属于 Circle Loss 理论，故根据 Circle Loss 理论可对 CoSENT 进行优化，为 CoSENT 中正负样本对分别加上权重与边距：

$$\log(1+\sum_{(i,j)\in\Omega_{pos},(u,v)\in\Omega_{neg}} e^{\lambda(\omega_{neg}(\cos(emb_u,emb_v)+m_{neg})-\omega_{pos}(\cos(emb_u,emb_v)-m_{pos}))})$$

其中，

$$\omega_{neg}=\frac{(\cos(emb_u,emb_v)+1)}{2}$$

$$\omega_{pos}=1-\frac{(\cos(emb_i,emb_j)+1)}{2}$$

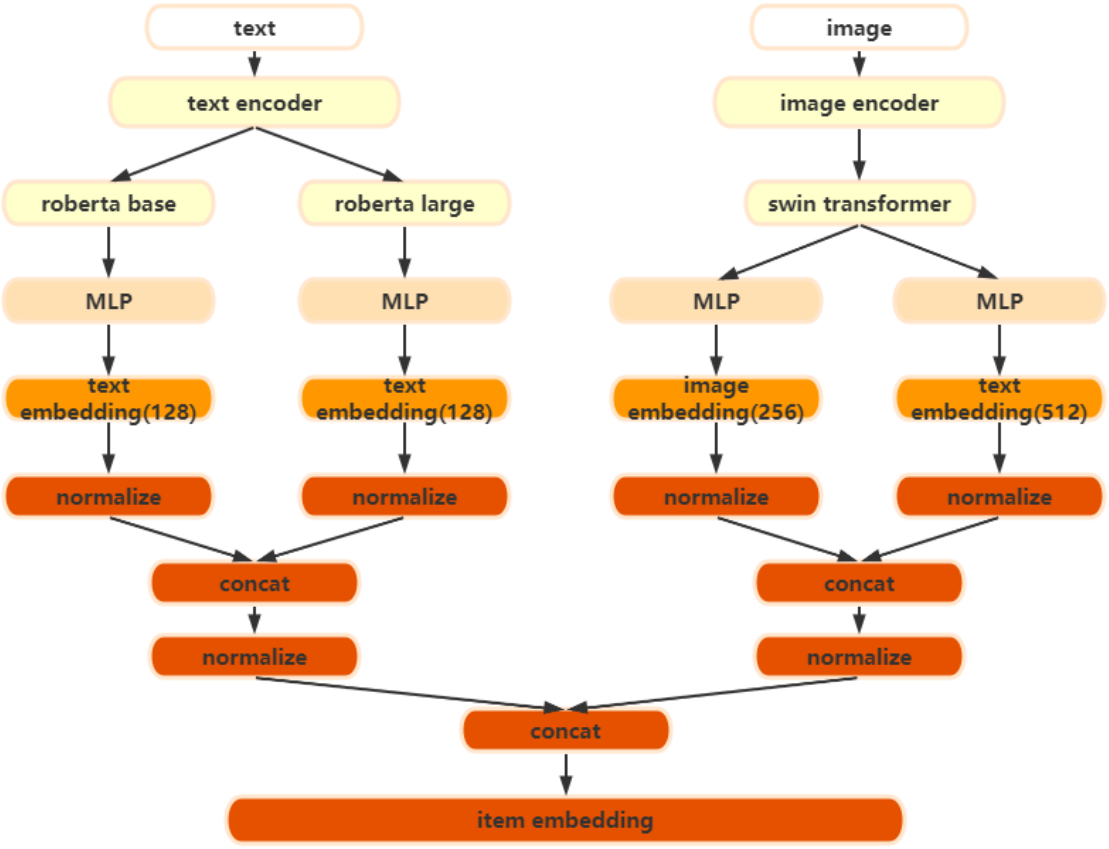
分别代表负样本对权重与正样本对权重，m_{neg}与m_{pos}分别代表负样本对边距与正样本对边距。由于 cos 相似度的取值范围为(-1,1)，为了将其范围调整到(0,1)，直接采用$\frac{(\cos+1)}{2}$。我们将样本对分为简单样本对与困难样本对，显然，标签为 0 的样本对其预测结果越靠近 0 越是简单样本，越靠近 1 越是困难样本；同理，标签为 1 的样本对其预测结果越靠近 1 越是简单样本，越靠近 0 越是困难样本。遵循简单样本对权重较小，困难样本对权重较大的思想，ω_{neg}为$\frac{(\cos+1)}{2}-0$，ω_{pos}为$1-\frac{(\cos+1)}{2}$。对于$m_{neg}>0$与$m_{pos}>0$，我们希望预测的负样本对 cos 相似度即使加上m_{neg}也被预测为负样本对，预测的正样本对 cos 相似度即使减去m_{pos}也被预测为正样本对，进一步拉大正样本对与负样本对的距离。

在实际实验中发现，虽然加上权重与边距后的 CoSENT 在本地验证集 F1 稍有提升，但线上无明显变化，故在最后的提交结果中只使用了 CoSENT。

3 模型融合

对于文本特征，分别使用 roberta_base 和 roberta_large 生成两个 128 维的 embedding，并将文本的两个 emb 分别 normalize 后 concat 成一个 256 维的 embedding 文件。

对于图像特征，使用 swin_transformer_large 生成一个 256 维和 512 维的 embedding，分别 normalize 后 concat 成一个 768 维的 embedding 文件。



4 后处理

在生成最终的 jsonl 格式提交文件时，首先设置一个基础阈值，例如 0.776，然后分别将文本的 256 维和图片的 768 维 emb 进行 normalize 再 concat，并分别计算出文本和图片的相似度。再根据以下规则对相应的数据修改阈值：

（1）图片相似度比较高 或者 文本相似度比较高且品牌相同的数据，降低其阈值。

（2）相似度不是特别低的，且型号相同、品牌相同的 或 货号相同的数据，降低阈值。

（3）如果文本相似度和图片相似度都不是很高，且品牌不同的数据，提高阈值。

（4）如果品牌不同、型号不同，且图片相似度不是很高的数据，提高阈值。

（5）对于“投资贵金属”品类的数据，我们从标题中抽取出 生肖、年份、金额。若生肖或年份有一个不同，或年份相同、金额不同的数据，提高其阈值；若生肖相同，或年份相同、金额相同或无法比较的数据，降低其阈值；若上述条

件都不满足，且金额不同的数据，提高其阈值。

（6）对于“洗烘套装”和“洗衣机”品类的数据，我们从 item_pvs 中抽取出“洗衣机型号”、“烘干机型号”。 若洗衣机型号、烘干机型号有一个相同，降低其阈值；若品牌不同 或 洗衣机型号、烘干机型号都不同，提高其阈值。

具体代码在 submit_rules.py 文件中。