文章链接：

https://scontent-lga3-1.xx.fbcdn.net/v/t39.8562-6/246795273_2109661252514735_2459553109378891559_n.pdf?*nc_cat=105&ccb=1-5*&nc_sid=ad8a9d&*nc_ohc=uxRbkRelrzwAX8nvpiK*&nc_ht=scontent-lga3-1.xx&oh=00_AT8syOkf-tBPOkMbyre3mWz6dk1lrAfhXKIs8taZzhK4Hw&oe=61E258E3scontent-lga3-1.xx.fbcdn.net/v/t39.8562-6/246795273_2109661252514735_2459553109378891559_n.pdf?*nc_cat=105&ccb=1-5*&nc_sid=ad8a9d&*nc_ohc=uxRbkRelrzwAX8nvpiK*&nc_ht=scontent-lga3-1.xx&oh=00_AT8syOkf-tBPOkMbyre3mWz6dk1lrAfhXKIs8taZzhK4Hw&oe=61E258E3

本文参考：KDD'21 | 揭秘Facebook升级版语义搜索技术

1. 简介

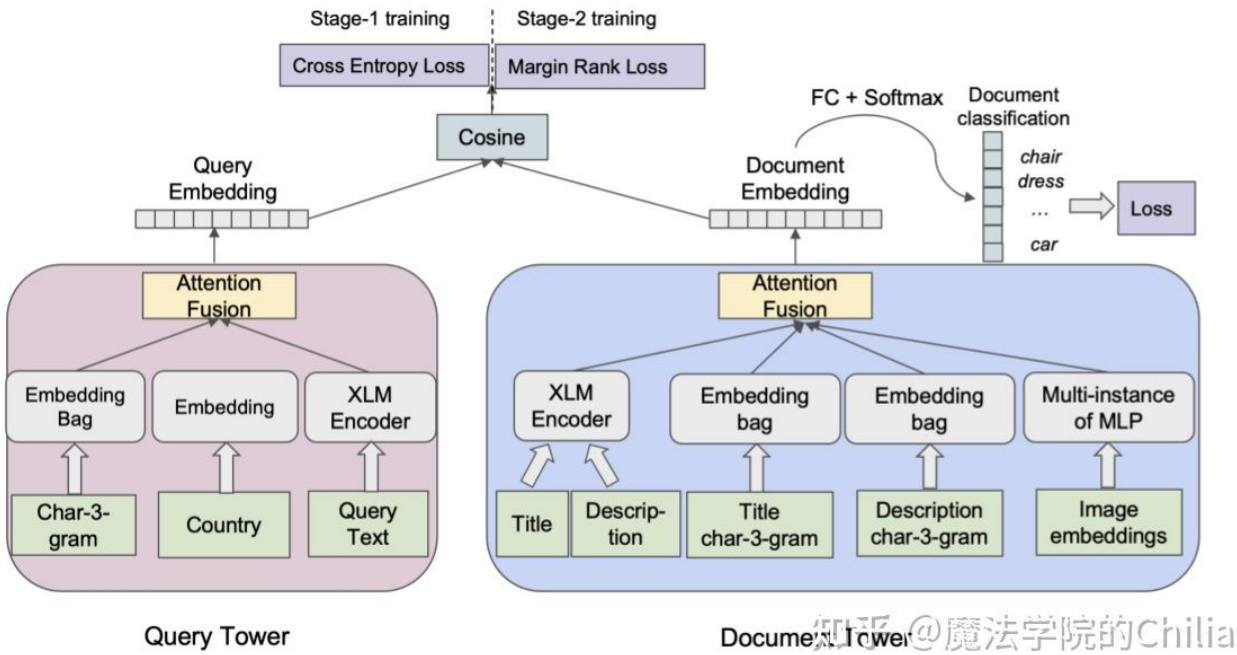
本篇文章介绍了query和商品的语义理解系统Que2Search。之前的商品向量检索大多只使用n-gram的稀疏特征做embedding，没有使用BERT类模型，所以对自然语言的理解能力不够好。但是，如果使用Transformer类模型，时间复杂度较高，对搜索的时延要求也更加严苛。

本论文提出的模型在性能上实现了CPU上的实时推理99线小于1.5ms；在效果上，离线相关性指标提升5%，线上参与度指标提升4%。能够有如此提升的原因大致是：

- 在双塔上引入了用document预测query的多任务学习，强迫模型理解query意图；
- 采用两阶段训练范式，第一阶段用in-batch负样本，第二阶段用难负例进行课程学习；
- 使用了包括多模态、多语言在内的多种特征，并且用注意力机制融合这些特征。

2. 模型结构

模型依旧是采用双塔结构：

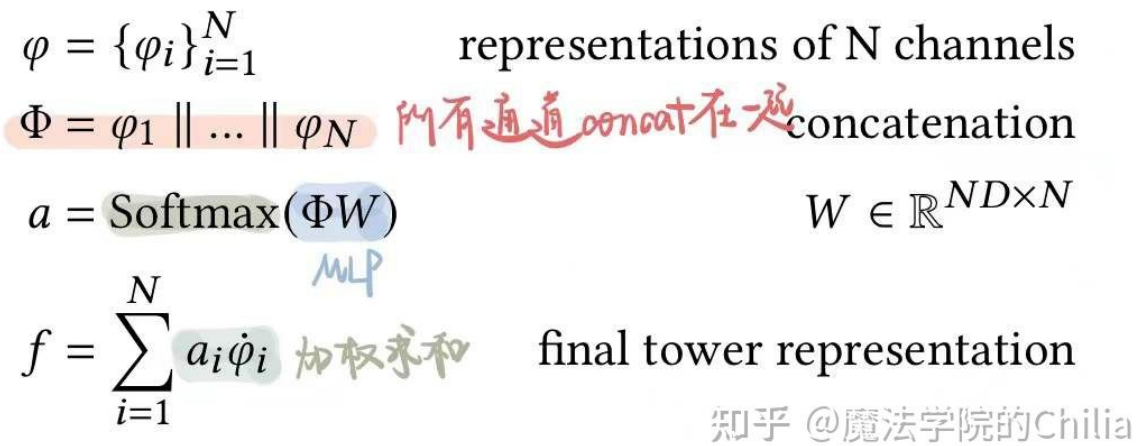


2.1 Query 塔

输入的特征包括：

- 3-gram稀疏特征，然后用哈希映射到一个embedding table中的某一行，整个query的embedding就是所有其所有3-gram的sum-pooling。这个思想和最早的双塔模型DSSM是类似的。
- 用户的国家ID embedding
- query的文本直接输入一个两层的XLM encoder (4个attention head, hidden size = 128，这样做是为了达到准确率和效率的平衡)，然后用[CLS]的向量表征经过一个MLP（为了降维）得到整个句子的embedding。XLM相当于做了跨语言对齐的BERT，能够把不同语言的相同意思映射到相似的向量空间。

朴素的方法就是把所有field的embedding都拼接到一起，然后过若干层MLP得到query塔最终的embedding。但是，这个模型用了一个简单的attention机制来赋予不同通道以不同的注意力权重：



这个想法和SENET比较类似，都是赋予不同特征以不一样的注意力权重。这也带来了良好的可解释性，例如文章发现XLM encoder得到的embedding的平均权重系数为0.64, 而3-gram得到的embedding平均权重系数为0.36. 这说明XLM encoder对于模型学习的帮助更大。进一步，文章发现query长度小于5的时候，模型的确更加关注3-gram；而当query更长的时候，XLM的权重几乎接近1，即起了主导作用。

2.3 Document塔

输入特征包括：

- 标题(title)和描述(description)的文本经过6层XLM-R encoder做embedding
- 标题、描述的3-gram稀疏特征
- 商品关联的图片的表征

同样使用attention机制来融合这些特征。

2.4 多任务学习

除了预测该商品是否相关，文章还使用了另外一个**辅助任务**，即用document来预测query的类别（见图右上角）。我们找到和这个document相关的query作为label。

将document的embedding经过MLP+softmax之后变成了一个多分类任务，这个任务的label就是此document对应的query。具体地，使用了频率最高的45k个query，因此这就是个45k多分类任务；一个document可以对应多个label，所以这是一个多分类问题，使用多个交叉熵损失求平均。所以，这样对于每个<query,document>对，我们的主loss是sampled softmax loss，这个loss需要负采样+带温度系数的softmax来完成；辅助loss就是正

document预测query的关键词label。这样做的目的是强迫模型去根据document来推测**用户意图**。

2.5 训练过程

整个训练过程可以分为两部分：使用In-batch negative负样本的训练阶段、使用难负例的课程学习(curriculum training)阶段。

从search log中，我们只能获得<query, document+>正样本对。文中描述一个"正样本"需要具备如下条件：（1）用于搜索某个query （2）点了某个商品 （3）进去和商家咨询 （4）商家回复了

只有在24h之内发生如上连续事件，才算得上一个正样本。使用如此严苛的正样本筛选措施是为了让正样本真正的相关，防止在正样本中引入噪声。当然了，其实把impression或者click当成正样本也是可行的办法。

至于负样本，则需要我们自己去构造。

2.5.1 使用In-batch negative负样本的训练阶段

这个负采样方法十分简单，就是使用一个batch中其余B-1个document作为负样本。对于某个正样本

$$(q_i, d_i)$$

, 其损失为：

$$\text{loss}_i = -\log \frac{\exp(s \cdot \cos\{q_i, d_i\})}{\sum_{j=1}^{B} \exp(s \cdot \cos\{q_i, d_j\})}$$

s值越大，越能拉开正样本和负样本的差距，收敛越快。"拉近正样本、推开负样本"，这其实就是对比学习的思想。

2.5.2 课程训练

上文所用的in-batch negative是普通的随机样本，模型区分两个毫不相干的document自然比较容易；那么为了让模型能够更加精细的区分，还需要一些**难负例**训练作为第二阶段的训练。这就像学生学习的时候，需要由简单到困难来学习课程一样。

一般的方法是用另外一个模型来挖掘困难负样本，然后喂给我们的模型来学习。但是这样需要单独维护另外一个模型，不够简洁。文章的做法是还是使用in-batch的方法来获得难负例。具体的做法是：

计算一个batch中每对<query,document>的相似度，构成 B*B大小的矩阵。对于每一行（也就是每个query），都抽取除了对角线正样本之外的具有**最高相似度分数**的document作为难负例。

有两种损失函数都能达到"拉近正样本，推开负样本"的目的，分别是交叉熵损失和pair-wise hinge loss：

$$\text{loss}_i = -log(\sigma(s \cdot cos(q_i, d_i))) + log(1 - \sigma(s \cdot cos(q_i, d_{nq_i}))) \quad (4)$$

$$\text{loss}_i = max(0, -[cos(q_i, d_i) - cos(q_i, d_{nq_i})] + margin) \quad (5)$$

文章发现使用margin位于0.1~0.2之间的pairwise hinge loss表现最佳，即强迫模型把正样本的得分打的至少比负样本高0.1~0.2。

这两阶段的学习需要串行进行，即先用in-batch negative随机采样法训练，使得第一阶段收敛之后才能进行第二阶段的课程学习。如下图可以看到，在第二阶段有明显的指标跳变，比起不用课程学习有了1%左右的AUC提升。

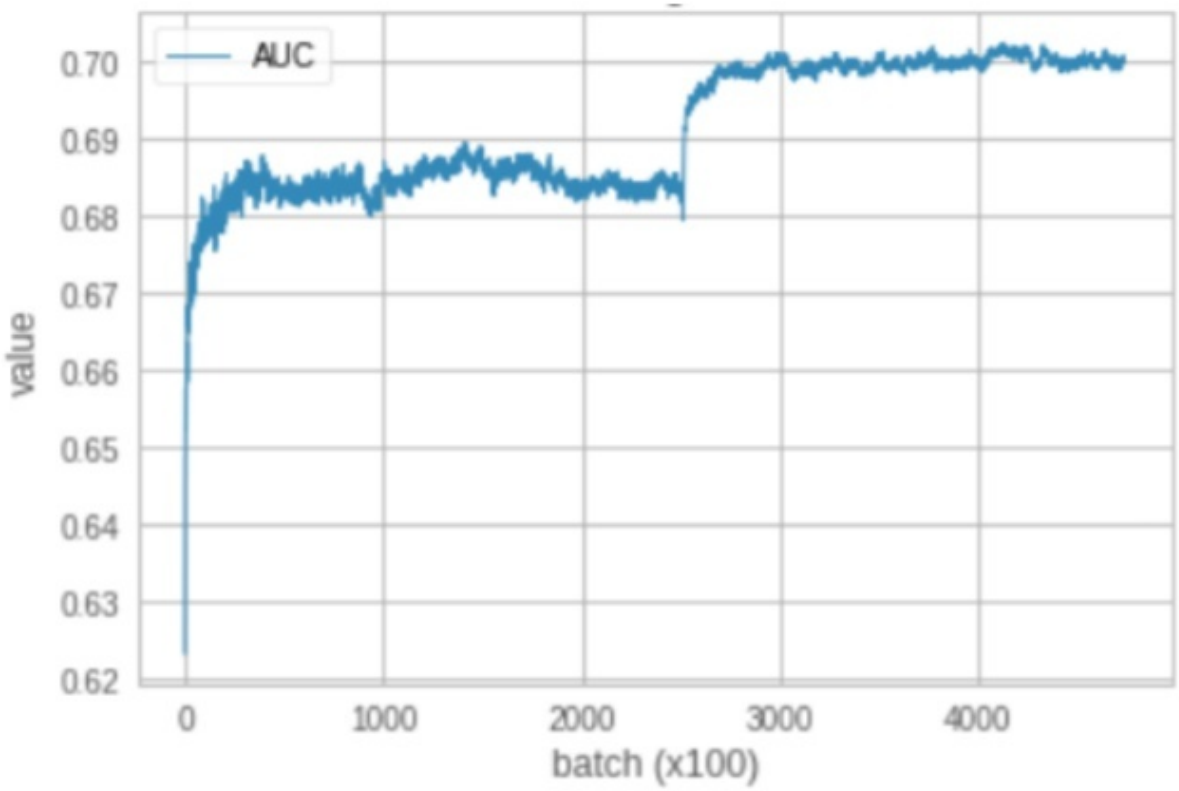


Figure 3: ROC-AUC curve for multi-stage training.