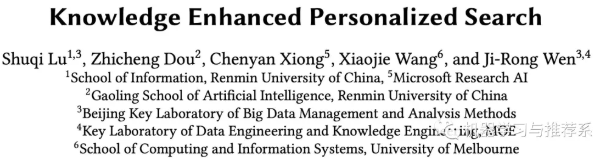
当知识图谱遇上个性化搜索

本文来自微信公众号：机器学习与推荐系统（ml-recsys）

欢迎关注微信公众号：鸿煊的学习笔记（Techs\_AI）

# 1、前言

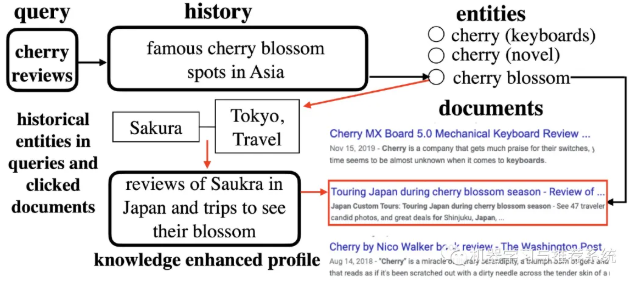
今天给大家分享一篇SIGIR 2020的文章：KEPS，用图谱来辅助优化个性化搜索。整篇文章首先充满学术风，充分且完备的设计了模型和实验，其次很多思路也可以在工业界中进行借鉴。一开始我只是大概扫了一下作者列表，看的过程中就觉得应该是学术界和工业界合作的一篇文章，果然是人大和微软都有署名~



顾名思义，这篇文章的主要工作是一个基于知识图谱来优化的个性化搜索模型，命名为 KEPS。整个工作可以分为四部分：首先是构建一个个性化的实体链接网络；然后再进行用户画像的构建；接下来就可以利用搜索意图和用户画像对文档进行个性化排序；最后再根据用户的点击结果对实体链接进行调整。

首先是个性化搜索的情况，本质上是根据用户的历史行为来判断现在搜索的目的。这种做法的缺点是不能利用到一些外部信息，比如樱花和日本之间的关系就无法获得；另外一方面是利用实体链接的方法，可以获得外部信息，但是不善于处理模糊语义，比如cherry到底是在说樱桃，还是说键盘？

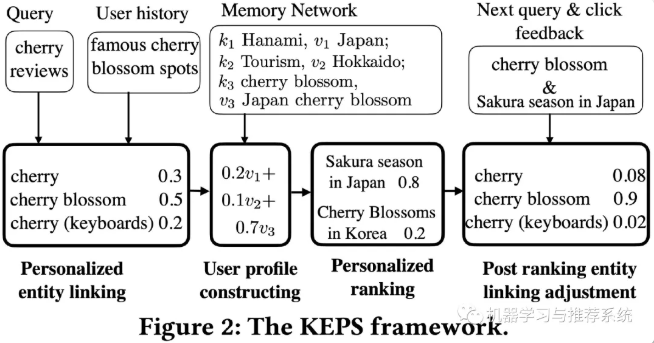
直觉上很容易想到，将这两者结合起来，就可以互相弥补，实现一个有效的个性化搜索功能。



上面这幅图展示了用户输入“cherry review”时的一个流程，从用户的历史行为可以看到这里cherry这个实体指的是樱桃花。接着通过实体链接图谱，找到樱花和Tokyo，Sakura之间的联系，进而就可以得到一些相关的文档了。

# 2、整体流程

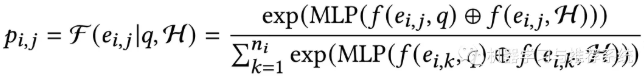
接下来我们简单的介绍一下文章整体的思路，下面这幅图给出了本文四部分主要的流程框架：



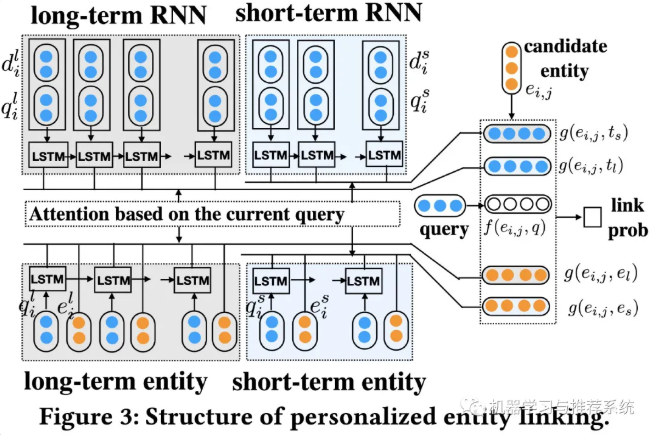
首先是personalized entity linking，主要是为了更好的识别query意图；接下来是user profile constructing，根据query意图来构建用户偏好模型；第三部分是personalized ranking，根据前面得到的query意图和用户偏好来构建一个对文档相关性进行个性化排序的模型；最后是post ranking entity linking adjustment，根据用户点击的反馈来调整前面query实体链接的概率，进一步优化本次搜索序列中接下来的搜索结果。

# 3、personalized entity linking

首先我们搞清楚一个实体链接网络是什么形式的，一张图，节点是不同的实体，边是实体之间的链接关系。在本文中边就是两个实体之间链接的概率，更具体的说，本文中的边表示的是**「用户输入的query中提到的实体i和某个实体意图j链接的概率」**：



上面的式子是一个计算过程，在给定query和历史行为H的情况下，实体i是意图j的概率为p\_ij。所谓的个性化的实体链接，其实就是针对不同的用户历史，这里的p\_ij都会是不同的取值，那么问题就转变到了如何构建出来个性化的p\_ij，具体的内容通过下面这幅图进行介绍：



以中间为界，可以将上下两部分分为query的实体和文档的历史交互记录，以及query 的实体和对应意图之间的历史交互记录。而每一部分又可以左右分为长期的历史行为和短期的历史行为。

在这里需要补充一个作者提到的关键点，作者认为用户的历史查询可以分为多个 session，每个session是一个短期历史行为，而整个历史行为则是作为长期历史。比如用户当前想搜索一些关于旅游的信息，那么当前这个session中他搜索的行为都是一个主要目的，本次搜索就对下一次起到重要影响。

上面p\_ij公式中的f(\*)函数的内容就如下：



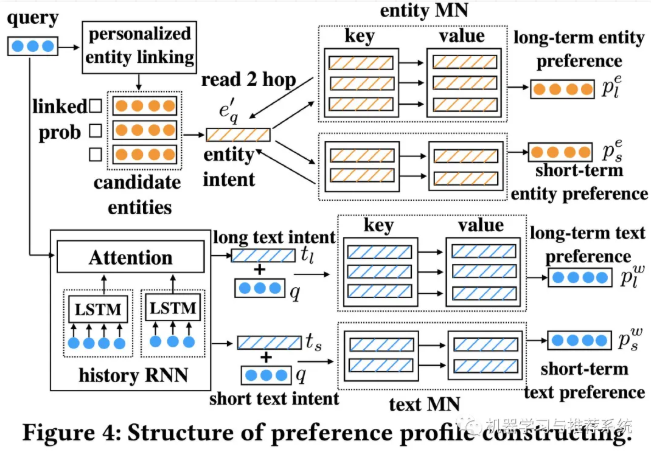


这两个式子分别计算了当前e\_ij和当前query的关联程度，以及e\_ij和历史行为的关联程度，在历史行为部分也分别考虑了长期和短期。

通过这种办法可以构建出一个个性化的实体链接图，实体与意图之间的边就是p\_ij。接下来介绍用户画像的构建方法。

# 4、user profile constructing

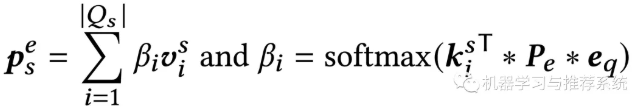
使用前面的实体链接来反映用户的搜索意图，对应的来检索用户与此意图相关的搜索历史，通过相应的文档点击历史来构建用户偏好。在本文中，使用key-value memory network来保存用户的历史。与第一部分类似，本文既考虑实体的历史记录，也考虑文档的历史记录。整体结构如下：



这里作者利用memory network来构建用户偏好，分别从实体的memory network和文本的memory network两部分来构建，以前者为例，我们简单的介绍一下。

## 4.1 entity memory network

这里将用户历史行为中的query**「实体」**向量作为key，然后点击过的文本向量作为value，然后利用下面的公式来计算用户短期历史的实体偏好，长期历史只需要将公式中所有的s换成l，同样的计算思路。



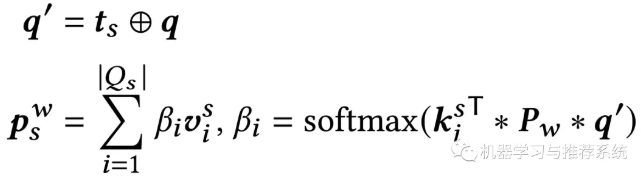
这里的k表示memory network中的key，v表示value，所以简单点理解计算思路就是，利用实体key和前面的实体链接概率图来作为权重，再对value进行加权，获得最终的偏好表示。

文章中对得到结果进行了进一步处理，但是和这个思路一致，将得到的偏好和当前 query结合起来再查询一次memory network，也就是图中标注的read 2 hop。

## 4.2 text memory network

文本的memory network也是类似的思路，不同之处在于前面的entity memory network 将query的实体向量和文档的向量分别作为key和value，而这里将query本身的向量作为key，将对应点击过的所有文本向量的平均值作为value。

不同的key和value其实是提供了不同的特征表达方式，而考虑到直接用query的 embedding不能充分的表示当次搜索的意图，作者还添加了t\_s短期兴趣作为query的补充。所以计算方法就成了如下这样：



计算思路和前面一致，这里是用短期历史作为例子，长期也是同样的思路。

# 5、personalized ranking

通过前面的两个部分，可以得到用户的意图概率，用户的偏好画像，然后就可以进一步来对候选的item进行排序，这里以文档排序为例，计算公式可以简化为这样：



相当于分别计算文档和用户意图的相关性，候选文档和用户偏好的相关性，以及候选文档与输入的query之间的相关性。三部分的embedding向量进行拼接，经过一个MLP层就是最终的排序得分。

以文档与意图的相关性进行示例：



每次的计算中，同时考虑了文档和长期与短期，文本与实体，共两个维度，四种组合的相关性，并将最终结果拼接起来作为最后的计算结果。

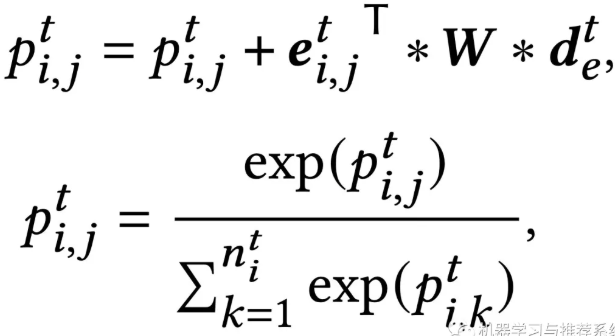
这样就充分考虑了前面的工作成果，并将其结合起来作为对候选文档的个性化打分的依据，输出一个个性化的排序结果。

# 6、post ranking entity linking adjustment

在给用户展示了排序好的文档后，根据用户的当次点击情况，可以再作为标签反馈，对实体链接图进行调整。

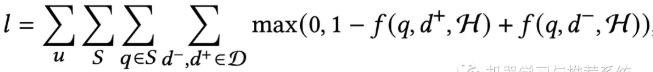
作者在文中举了这样一个例子，首先，每次搜索的一个session中，用户搜索的意图背景可以看做是相似的。在这样的前提下，假如用户搜了Java相关的内容，当他再搜编程书的时候，我们就可以优先展示Java相关的编程书籍。

具体的调整思路，也就是根据用户点击的文档，以及其中提到的实体，来更新前面构建的个性化实体链接图：



# 7、训练与实验

最终作者将所有的内容整合起来，用一个pair-wise的损失函数来直接进行训练，一次传播中对涉及的所有参数都进行更新，具体的loss就比较简单了：



d+和d-两个分别表示正样本和负样本，也就是给用户展示了以后，用户点击与没点击的文档。

实验部分作者做了很多工作，感兴趣的同学可以去找论文看看，作者除了对比了一些 baseline的搜索方法，也对比了一些sota的工作。

除了与别的工作作对比之外，作者也进行了很多自身的对比，比如删除自己模型的不同部分，分别进行效果的验证。大家可以在后台输入关键词获得论文，再自行欣赏。

# 8、总结

这篇文章整体的思路还是比较清晰，很多设计不是非常新颖，但却考虑的很周到，事业设计也很完备。我们组做搜索，感觉对于长短期历史行为的设计，的确得到了一些启发，可以做的更精细些。相关从业者的小伙伴可以灵活借鉴其中的一些思路，对自己的工作提供帮助，

想看论文的小伙伴，可以关注微信公众号：鸿煊的学习笔记（Techs\_AI），在后台回复：KEPS，获得论文的网盘链接。