人工智能导论 报告2

组员1：肖文璇 学号：2018202163

组员2：雷雨寒 学号：2018202173

1 TFIDF模型优化

1.1 实现思路——增强属于同类的新闻相似度

报告1中实现的TF-IDF算法把标题、摘要和正文作为新闻的关键词，实现了一个有较好的推荐结果。但是没有使用到新闻数据中的大类别与小类别属性，实际上这些类别属性也是非常重要的，同一个类别的新闻相似性也越高。

为了优化我们的推荐结果，在计算TF-IDF值的时候引入权重，降低在同一大类而不在同一小类的新闻所得的TF-IDF权值大小，而在不在同一大类的新闻降低更多的TF-IDF权值，即：

$$Similarity(news, user) = W_{kind} * \sum_{i \in m} w_i * TFIDF_{newsi} * TFIDF_{useri}$$

其中m是新闻与用户喜好库重合关键词的集合，W_{kind}与用户的喜欢类别与该新闻的类别有关，w_i代表关键词i在用户喜好库中的权重。

具体的实现思路是根据用户喜欢的文章，获取用户喜欢的类别，构建用户喜好类型字典，字典的结构为`{Category:[SubCategory,],}`。

在计算新闻与用户喜好库的TFIDF值时，如果这篇新闻的大类别不在用户的喜好类型字典中，则降低它的权重，因为有可能两者的关键词比较相近但是不在用户喜欢的类别之中；如果新闻的大类别在用户的喜好类型字典，但小类别不在用户的喜好类型库中，则小幅降低它的权重；如果新闻的大小类别在用户的喜好类型字典，则增加它的权重。

1.2 代码展示

1.2.1 构建用户喜欢类型字典

```
# 根据用户喜欢的文章，获取用户喜欢的类别，Category结构为{Category:[SubCategory,],}
def Get_like_Category(like,news):
    Category = {}
    # 遍历喜欢的文章
    for like_item in like:
        # 找到喜欢的类别
        key = news[news['News ID'] == like_item]['Category'].values[0]
        # 将小类存放到大类对应的列表中
        if key not in Category.keys():
            Category[key] = []
        Category[key].append(news[news['News ID'] == like_item]
['SubCategory'].values[0])

    return Category
```

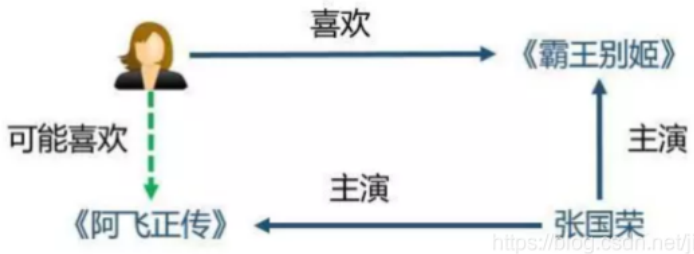
1.2.2 根据用户喜欢类型字典生成推荐字典

```
# 根据用户喜欢的新闻类别，重新生成推荐结果，news为原始的新闻数据，rec为tfidf生成的推荐结果
def re_rec(Category, news, rec):
    # 遍历推荐的文章
    for i in range(len(rec)):
        rec_item = rec.iloc[i]
        news_item = news[news['News ID'] == rec_item['News ID']]
        if news_item['Category'].values[0] not in Category.keys(): #如果不在喜欢的
大类里面，相似值权重降低为0.8
            rec.loc[i, 'similarity_score'] = 0.8*rec_item['similarity_score']
        else:
            # 符合喜欢的大类，不符合喜欢的小类，权重值降低为0.9
            if news_item['SubCategory'].values[0] not in
Category[news_item['Category'].values[0]]:
                rec.loc[i, 'similarity_score'] = 0.9 *
rec_item['similarity_score']
            # 大类小类都符合，权重值提高为1.25
            else:
                rec.loc[i, 'similarity_score'] = 1.25 *
rec_item['similarity_score']
    # 根据新的相似值重新排序
    re_rec = rec.sort_values(by=['similarity_score'], ascending=False)
    print(re_rec[0:10])
    return re_rec
```

2 基于知识图谱的推荐系统

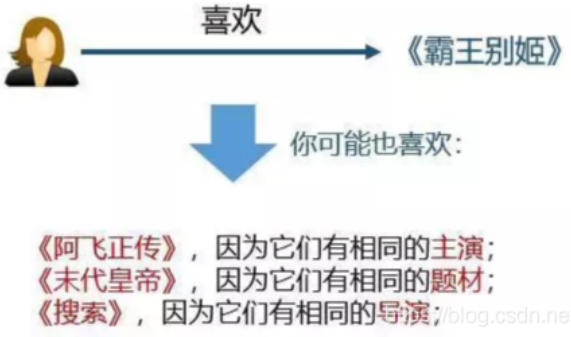
2.1 知识图谱在推荐上的优势

精准性：引入更多的语义关系，深层次的发现用户兴趣，用户喜欢张国荣主演的《霸王别姬》，那也可能喜欢张国荣主演的《阿飞正传》



多样性：通过知识图谱中不同的关系链接种类，有利于推荐结果的发散。用户喜欢《霸王别姬》，那也可能喜欢同题材，同导演等的作品。

可解释性：喜欢《霸王别姬》也可能喜欢《末代皇帝》是因为它们有相同的导演。



2.2 知识图谱与推荐系统的结合方法

2.2.1 基于特征的推荐方法

这类方法主要是从知识图谱中抽取一些用户和物品的属性作为特征，放入到传统的模型中，只引入了**实体特征**，没有引入关系特征。

2.2.2 基于路径的推荐方法

基于路径的推荐方法将知识图谱视为一个**异构信息网络**（heterogeneous information network），然后构造物品之间的基于meta-path或meta-graph的特征。简单地说，meta-path是连接两个实体的一条特定的路径，比如"演员->电影->导演->电影->演员"这条meta-path可以连接两个演员，因此可以视为一种挖掘演员之间的**潜在关系**的方式。

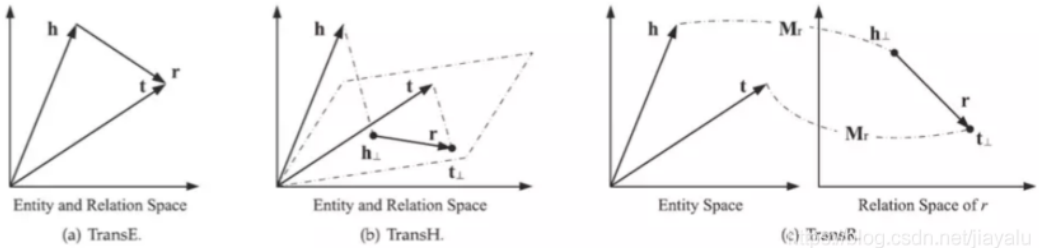
这类方法的**优点**是充分且直观地利用了知识图谱的网络结构，**缺点**是需要手动设计meta-path或meta-graph，这在实践中难以到达最优；同时，该类方法无法在实体不属于同一个领域的场景（例如新闻推荐）中应用，因为我们无法为这样的场景预定义meta-path或meta-graph。

2.2.3 知识图谱特征学习（Knowledge Graph Embedding）

即为实体和关系学习得到一个低维向量。两类模型：**基于距离**的模型和**基于语义**的匹配模型。

- **基于距离的翻译模型（distance-based translational models）**

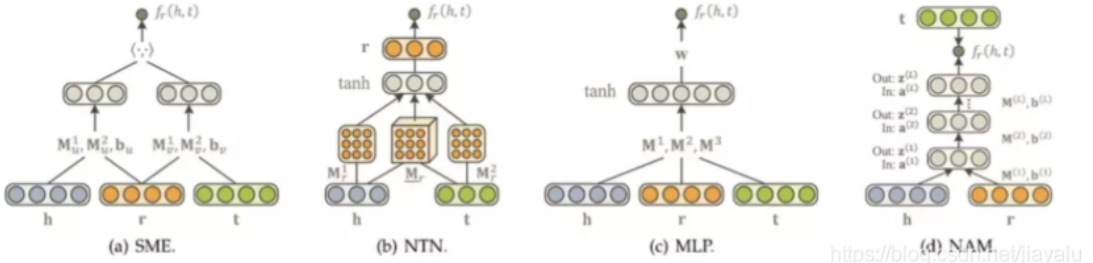
  这类模型使用基于距离的评分函数评估三元组的概率，将尾节点视为头结点和关系翻译得到的结果。这类方法的代表有TransE、TransH、TransR等；



  在空间中，三元组的头节点h、关系r、尾节点t都有对应的向量，我们希望的是h + r = t，如果h + r的结果和t越接近，那么我们认为这些向量能够很好的表示知识图谱中的实体和关系。

- **基于语义的匹配模型（semantic-based matching models）**

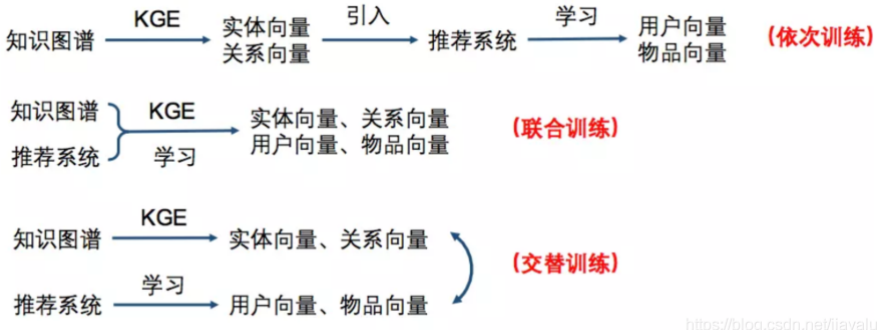
类模型使用基于相似度的评分函数评估三元组的概率，将实体和关系映射到隐语义空间中进行相似度度量。这类方法的代表有SME、NTN、MLP、NAM等。



上述方法的核心是构造一个二分类模型，将h、r和t输入到网络中，如果(h,r,t)在知识图谱中真实存在，则应该得到接近1的概率，如果不存在，应该得到接近0的概率。

2.3 知识图谱特征学习的推荐系统

知识图谱特征学习与推荐系统相结合，往往有以下几种方式：依次训练、联合训练、交替训练。



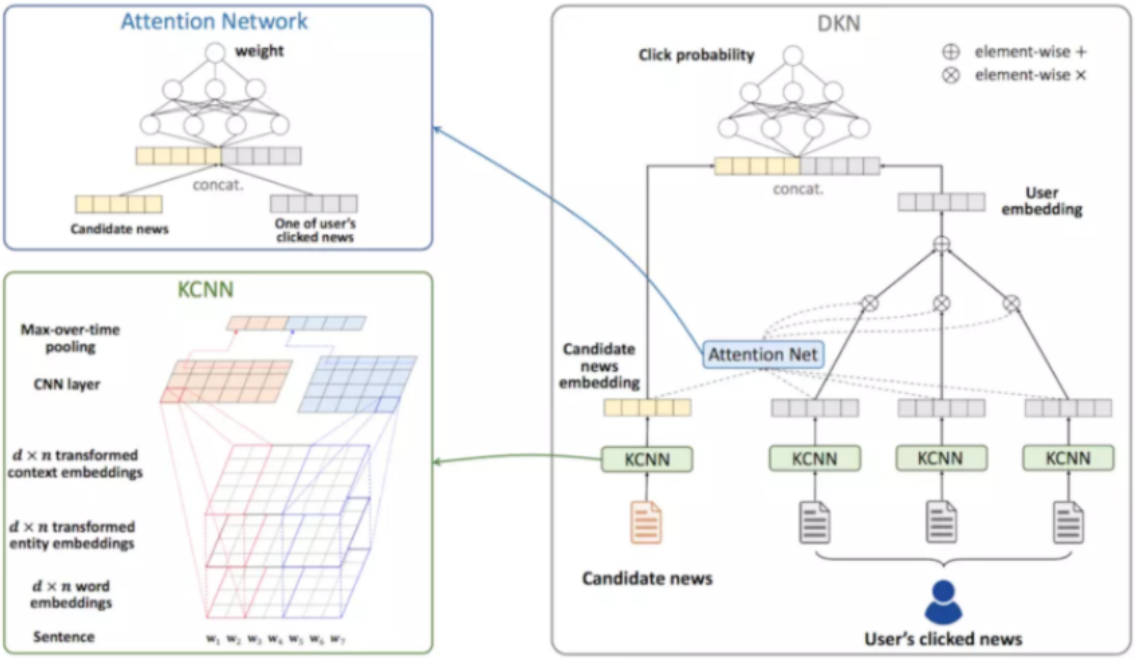
依次训练的方法主要有：Deep Knowledge-aware Network(DKN)
联合训练的方法主要有：Ripple Network
交替训练主要采用multi-task的思路，主要方法有：Multi-task Learning for KG enhanced Recommendation (MKR)

3 DKN 算法原理

3.1 问题描述

给定一个用户$user_i$，他的点击历史$\{t_1, t_2, \ldots t_n\}$是该用户过去一段时间内曾点击过的新闻标题，n代表用户点击过新闻的总数，DKN要解决的问题就是给定用户的点击历史，以及标题单词和知识图谱中实体的关联，预测一个用户i是否会点击一个特定新闻t_j。

3.2 模型框架



可以看到，DKN的网络输入有两个：候选新闻集合，用户点击过的新闻标题序列。输入数据通过KCNN来提取特征，之上是一个attention层，计算候选新闻向量与用户点击历史向量之间的attention权重，在顶层拼接两部分向量之后，用DNN计算用户点击此新闻的概率。接下来，我们介绍一下DKN模型中的一些细节。

3.3 知识提取

1、标题中每个单词的Embedding（可以通过预训练的word2vec模型得到）

2、获取标题中每个单词对应的实体的Embedding

3、得到每个单词的上下文Embedding

3.3.1 标题单词embedding

每个标题都是一个词序列$t = \{w_1, w_2, \dots w_n\}$，标题中的单词有的对应知识图谱中的一个实体，如在标题《Trump praises Las Vegas medical team》中，Trump与知识图谱中的实体"Donald Trump"对应，Las和Vegas与实体Las Vegas对应

3.3.2 标题实体embedding

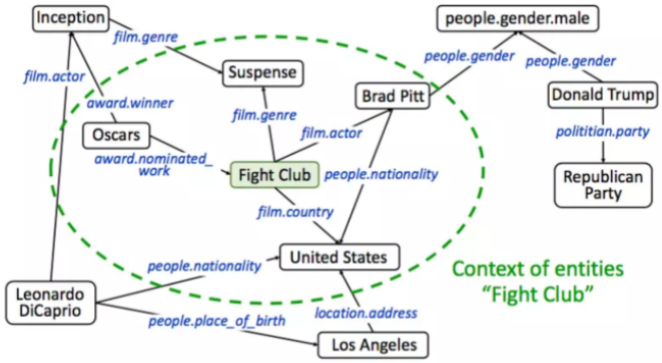
实体特征即标题中每个单词对应的实体的特征表示，通过下面四个步骤得到：

2. 识别出标题中的实体并利用实体链接技术消除歧义
3. 根据已有知识图谱，得到与标题中涉及的实体链接在一个step之内的所有实体所形成的子图。
4. 构建好知识子图以后，利用基于距离的翻译模型得到子图中每个实体embedding。
5. 得到标题中每个单词对应的实体embedding。

3.3.3 上下文Embedding

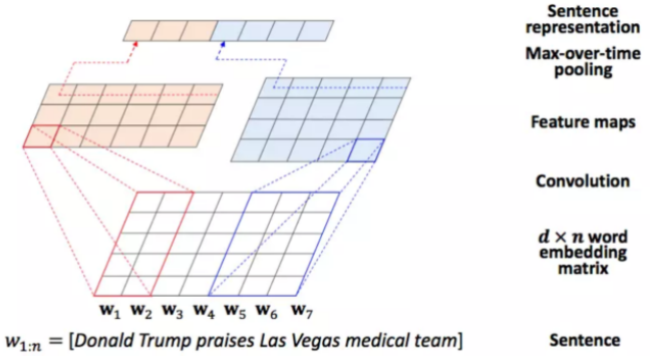
尽管目前现有的知识图谱特征学习方法得到的向量保存了绝大多数的结构信息，但还有一定的信息损失，为了更好地利用一个实体在原知识图谱的位置信息，文中还提到了利用一个实体的上下文来进一步的刻画每个实体，具体来说，即用每个实体相连的实体embedding的平均值来进一步刻画每个实体，计算公式如下：

$$\bar{e} = \frac{1}{|context(e)|} \sum_{e_i \in conext(e)} e_i$$



3.4 基于CNN的句子特征提取

DKN中提取句子特征的CNN，用句子所包含词的词向量组成的二维矩阵，经过一层卷积操作之后再做一次max-over-time的pooling得到句子的向量，如下所示：



3.5 新闻特征提取 KCNN (Knowledge-aware CNN)

在知识抽取部分，我们得到了三部分的embedding，一种最简单的使用方式就是直接将其拼接：

但这样做存在几方面的限制：

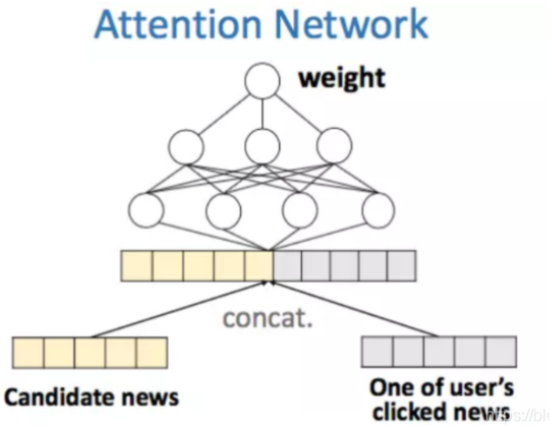
1. 连接策略打破了单词和相关实体之间的联系，并且不知道它们的对齐方式。
2. 单词的embedding和对应实体的embedding是通过不同的方法学习的，这意味着它们不适合在单个向量空间中将它们一起进行卷积操作。
3. 连接策略需要单词的embedding和实体的embedding具有相同的维度，这在实际设置中可能不是最优的，因为词和实体embedding的最佳维度可能彼此不同。

因此使用的改进方式为mutlti-channel和word-entity-aligned KCNN，具体做法是先把实体的embedding和实体上下文embedding映射到一个空间里，映射的方式可以选择线性方式g(e)=Me，也可以选择非线性方式g(e) = tanh(Me+b)，这样我们就可以拼接三部分作为KCNN的输入：

$$w = [[w_1 g(e_1) g((\bar{e_1}))][w_2 g(e_2) g((\bar{e_2}))] \dots [w_n g(e_n) g((\bar{e_n}))]] \in R^{d*n*3}$$

3.6 基于注意力机制的用户兴趣预测

获取到用户点击过的每篇新闻的向量表示以后，计算候选文档对于用户每篇点击文档的attention，再做加权求和，计算attention：



4 DKN 应用实现

4.1 dkn环境配置

dkn需要安装tensorflow，本次实验使用的是`tensorflow1.15.0-CPU`环境。

```
import sys
sys.path.append("../../")

import os
from tempfile import TemporaryDirectory
import papermill as pm
import tensorflow as tf

from reco_utils.recommender.deeprec.deeprec_utils import
download_deeprec_resources, prepare_hparams
from reco_utils.recommender.deeprec.models.dkn import DKN
from reco_utils.recommender.deeprec.io.dkn_iterator import DKNTextIterator
```

4.2 数据处理与参数初始化

从`behavior.tsv`中得到用户历史文件`user_history_file`。

```
train_session, train_history = read_clickhistory(train_path, "behaviors.tsv")
valid_session, valid_history = read_clickhistory(valid_path, "behaviors.tsv")
get_train_input(train_session, train_file)
get_valid_input(valid_session, valid_file)
get_user_history(train_history, valid_history, user_history_file)
```

从新闻标题获得标题单词的embedding `word_embeddings_5w_100.npy` 与对应实体的embedding `entity_embeddings_5w_100.npy`，并从新闻文档信息中得到新闻的特征 `doc_feature.txt`。

```
train_news = os.path.join(train_path, "news.tsv")
valid_news = os.path.join(valid_path, "news.tsv")
news_words, news_entities = get_words_and_entities(train_news, valid_news)
train_entities = os.path.join(train_path, "entity_embedding.vec")
valid_entities = os.path.join(valid_path, "entity_embedding.vec")
news_feature_file = data_path + '\\doc_feature.txt'
word_embeddings_file = data_path+'\\word_embeddings_5w_100.npy'
entity_embeddings_file = data_path+'\\entity_embeddings_5w_100.npy'
```

设定DKN模型要使用的参数 `hparams`，包括上面生成的 `embedding` 文件，还有训练的次数 `epochs`，历史大小 `history_size` 与批次大小 `batch_size`。

```
hparams = prepare_hparams(yaml_file,
                          news_feature_file=news_feature_file,
                          user_history_file=user_history_file,
                          wordEmb_file=word_embeddings_file,
                          entityEmb_file=entity_embeddings_file,
                          epochs=epochs,
                          history_size=history_size,
                          batch_size=batch_size)
```

4.3 模型训练与预测

利用我们设定的参数 `hparams` 初始化DKN模型，然后使用我们的训练数据 `train_file` 与验证数据 `valid_file` 对模型进行反复训练

```
model = DKN(hparams, DKNTextIterator)
model.fit(train_file, valid_file)
```

训练的运行过程如下图所示：

```
at epoch 1
train info: logloss loss:0.6707635026867107
eval info: auc:0.5601, group_auc:0.5114, mean_mrr:0.172, ndcg@10:0.2394, ndcg@5:0.1708
at epoch 1 , train time: 76.3 eval time: 11.4
at epoch 2
train info: logloss loss:0.6216596258898913
eval info: auc:0.5653, group_auc:0.5307, mean_mrr:0.1798, ndcg@10:0.2427, ndcg@5:0.1852
at epoch 2 , train time: 74.7 eval time: 11.6
at epoch 3
train info: logloss loss:0.5895718016614349
eval info: auc:0.5854, group_auc:0.5445, mean_mrr:0.1818, ndcg@10:0.2448, ndcg@5:0.1909
at epoch 3 , train time: 75.0 eval time: 11.5
```

使用 `predict` 函数即可对用户与新闻的匹配程度打分，分数越高则代表用户可能对这则新闻越感兴趣。

```
model.predict(test_file,output_file)
```

4.5 结果展示

我们设定一个用户历史阅读的新闻内容为关于皇室的新闻，部分新闻描述如下：

	News ID	Category	SubCategory	Title	Abstract	URL	Title Entities	Abstract Entities
0	N3112	lifestyle	lifestyleroyals	The Brands Queen Elizabeth, Prince Charles, an...	Shop the notebooks, jackets, and more that the...	https://www.msn.com/en-us/lifestyle/lifestyler...	[{"Label": "Prince Philip, Duke of Edinburgh",...	[]
17	N25192	lifestyle	lifestyleroyals	Every outfit Duchess Kate has worn in 2019	See Kate Middleton's style choices this year f...	https://www.msn.com/en-us/lifestyle/lifestyler...	[]	[{"Label": "Catherine, Duchess of Cambridge", ...
66	N1172	lifestyle	lifestyleroyals	The surprising age differences between your fa...	Here are the age differences between Meghan Ma...	https://www.msn.com/en-us/lifestyle/lifestyler...	[]	[{"Label": "Meghan, Duchess of Sussex", "Type"...
197	N16959	lifestyle	lifestyleroyals	Queen Elizabeth's Favorite Beauty Products Hav...	Here, all the brands the British monarch swear...	https://www.msn.com/en-us/lifestyle/lifestyler...	[]	[]

因为实验的设备限制与时间限制，我们只将模型的训练次数设置为3次，次数较少，所以效果未达到我们的预期，最终结果展示出前几条新闻推荐结果如下，大多数是关于足球与音乐的新闻。

	News ID	Category	SubCategory	Title	Abstract	URL	Title Entities	Abstract Entities
7324	N26059	sports	football_nfl	Odell Beckham's Comments After Browns-Patriots...	When the New York Giants shockingly decided to...	https://www.msn.com/en-us/sports/football_nfl/...	[{"Label": "Odell Beckham Jr.", "Type": "P", "...	[{"Label": "Odell Beckham Jr.", "Type": "P", "...
18355	N14982	sports	football_nfl	Browns, Steelers brawl at end of Cleveland's 2...	Cleveland defense end Myles Garrett ripped off...	https://www.msn.com/en-us/sports/football_nfl/...	[{"Label": "Cleveland Browns", "Type": "O", "W...	[{"Label": "Cleveland Browns", "Type": "O", "W...
18847	N13428	sports	football_nfl	Packers' Preston Smith buying Rolexes for defe...	Preston Smith's 10th sack of the season could ...	https://www.msn.com/en-us/sports/football_nfl/...	[{"Label": "Preston Smith (linebacker)", "Type...	[{"Label": "Preston Smith (linebacker)", "Type...
20893	N18291	news	newscrime	Father, daughter charged with raping woman, le...	A father and daughter were arrested in Los Ang...	https://www.msn.com/en-us/news/newscrime/fathe...	[]	[{"Label": "North Las Vegas, Nevada", "Type": ...
21434	N18892	music	music-celebrity	Wynonna Judd's Daughter, 23, Released from Pri...	Wynonna Judd's Daughter Released from Prison 6...	https://www.msn.com/en-us/music/music-celebrit...	[]	[]
23633	N14475	tv	tvnews	Rip Taylor's Cause of Death Revealed, Memorial...	The comedian died at the age of 84 last month.	https://www.msn.com/en-us/tv/tvnews/rip-taylor...	[{"Label": "Rip Taylor", "Type": "P", "Wikidat...	[]
26197	N22990	music	musicnews	Broadway Actress Laurel Griggs Dies at Age 13	Broadway star Laurel Griggs has died, accordin...	https://www.msn.com/en-us/music/musicnews/broa...	[{"Label": "Broadway theatre", "Type": "F", "W...	[{"Label": "Broadway theatre", "Type": "F", "W...
26414	N12985	music	music-celebrity	Broadway Star Laurel Griggs Suffered Asthma At...	Teen star Laurel Griggs, who passed away on No...	https://www.msn.com/en-us/music/music-celebrit...	[{"Label": "Broadway theatre", "Type": "F", "W...	[{"Label": "Once (musical)", "Type": "W", "Wik...

5 改进思路

1. TFIDF与DKN模型混合优化

   我们已经完成了TF-IDF模型与DKN模型的新闻推荐实验，其中TF-IDF模型的效果较好，DKN模型结果还需进一步优化，下一步，我们将继续优化DKN模型，并尝试共同使用两个模型，通过内容相似度与知识图谱的混合，达到更优推荐结果。

2. 基于协同过滤的新闻推荐

   当前的新闻推荐没有考虑用户与用户之间的关系，而如果两个用户的新闻阅读历史十分相似，则说明这两个用户有很相似的兴趣爱好，则可以为一位用户推荐另一位用户看过的新闻。

3. 热点新闻推荐

   热点新闻是在当前快速发展与变化的世界中经常出现的一类新闻，这类新闻的特点是产生的速度很快，关注的人很多，因此，我们可以根据某段时间内新闻的点击量来判断热点新闻，并将这些热点新闻推荐给更多用户。

4. 关注度衰减机制

   我们尚未用到用户点击新闻的时间，实际上随着时间的推移，用户对这类新闻的关注度也会逐渐下降，因此我们可以考虑在用户阅读新闻的时间对不同新闻上加一些权重的改变来改善推荐效果。

参考资料

1. 知识图谱推荐，DKN推荐算法：https://blog.csdn.net/jiayalu/article/details/100536068
2. DKN算法实现：https://github.com/microsoft/recommenders/blob/master/examples/00_quick_start/dkn_MIND.ipynb