



**数据挖掘课程设计**

**花园中的学习路径**

|  |  |
| --- | --- |
| 班 级： | 192104 |
| 队 名： | 玛卡巴卡 |
| 学 号： | 19210441, 19210449, 19210403, |
| 姓 名： | 袁仪,章亚宁,毛睿 |

2024年3月10日

# 

目录

[1.项目背景 2](#_Toc161001564)

[2.学习路径预测 2](#_Toc161001565)

[2.1 数据介绍 2](#_Toc161001566)

[2.2 算法介绍 7](#_Toc161001567)

[2.3 特征构造 9](#_Toc161001568)

[2.4 实验结果 12](#_Toc161001569)

[2.5 实验分析 15](#_Toc161001570)

[3. Course Recommendation System 17](#_Toc161001571)

[3.1 功能介绍 17](#_Toc161001572)

[3.2 数据介绍 17](#_Toc161001573)

[3.3 算法介绍 18](#_Toc161001574)

[3.4 特征构造 18](#_Toc161001575)

[3.5 实验结果 19](#_Toc161001576)

[3.6 系统界面 19](#_Toc161001577)

[4.结论 19](#_Toc161001578)

[5.进度安排 20](#_Toc161001579)

[6.团队特色 20](#_Toc161001580)

[参考文献 20](#_Toc161001581)

## 1.项目背景

自适应学习是一种能够根据学生的个人学习能力、学习风格、兴趣和进度来智能调整学习内容和难度的教学方法。在线教育的背景下，自适应学习通过收集和分析大量的学生数据，为学生提供个性化、精准的学习路径和课程。

本次课程设计的主要内容是基于MOOCCubeX数据集实现自适应学习的可行方案，利用机器学习和人工智能技术对数据集进行数据挖掘，基于分析结果最大程度满足学生的个性化需求。

对于该项目，调研发现，有对MOOCCube数据集的课程推荐算法KGAN，对知识追踪任务的DKT、AKT等算法，而对MOOCCubeX数据集的研究较少，且实现学习路径预测和自适应学习系统搭建的成果暂无，我们的项目研究有较广阔的研究前景和价值。

## 2.学习路径预测

##### 2.1 数据介绍

###### 2.1.1 如何切入-数据分析

对于如此庞大并且冗杂的数据组织结构，我们认为有必要针对我们的任务对数据进行一定的筛选。但是如何筛选呢？

我们仔细研究数据集和MoocCubeX的组织方式，可以知道发现整个数据集都是使用**细粒度知识图谱**进行组织（如图1的解释）。并且将知识图谱上的实体和关系分别保存在了文件夹entities和relations当中。

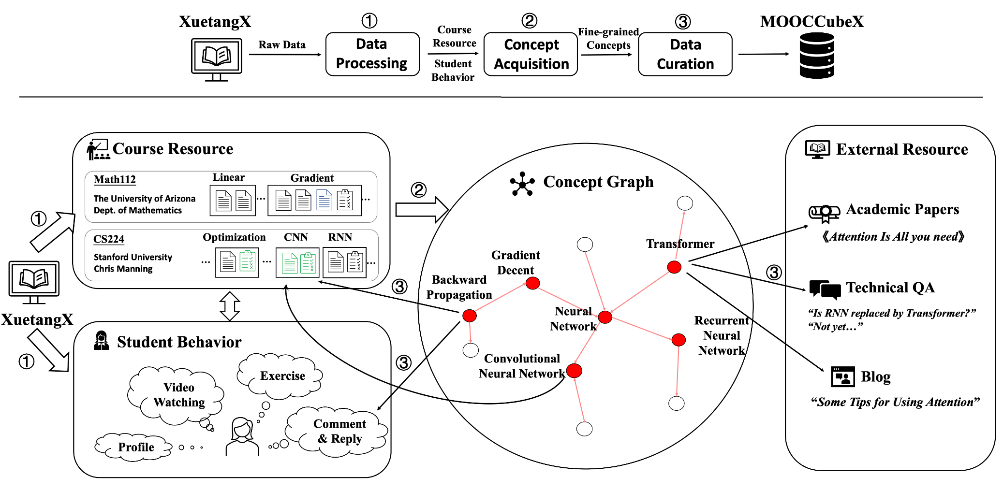


图 1 原始数据集构造的方式

对于一个知识图谱组织的数据，我们需要对这个庞大的架构有大致的把握，而首先需要做的就是，对知识图谱中涉及的基本内容：实体，有初步的分析和感知。因此我们找出我们认为对于学习路径预测重要的数据，并且进行简单的可视化，来分析实体和实体之间的数据的特性，便于我们更好地利用数据。这些数据可视化如图2，3所示，并且附有简单的分析。

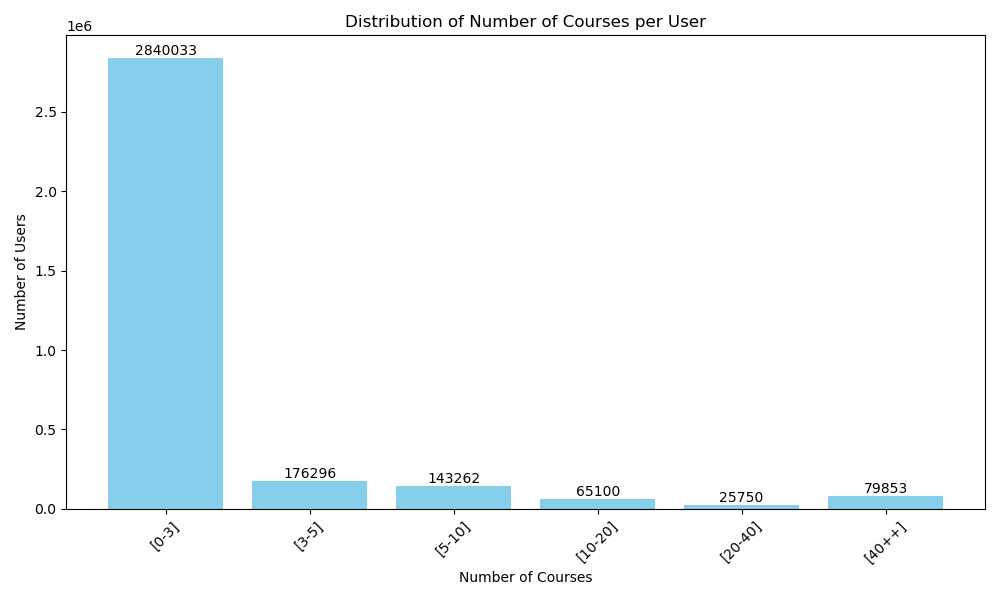


图 2 学生选课情况分布

对于图2中涉及的学生选课情况分布：

* 共有3330294名用户的选课记录。
* 而其中2840033名学生的选课记录是少于3条的，超85%，训练与预测难度都很大。
* 长尾分布

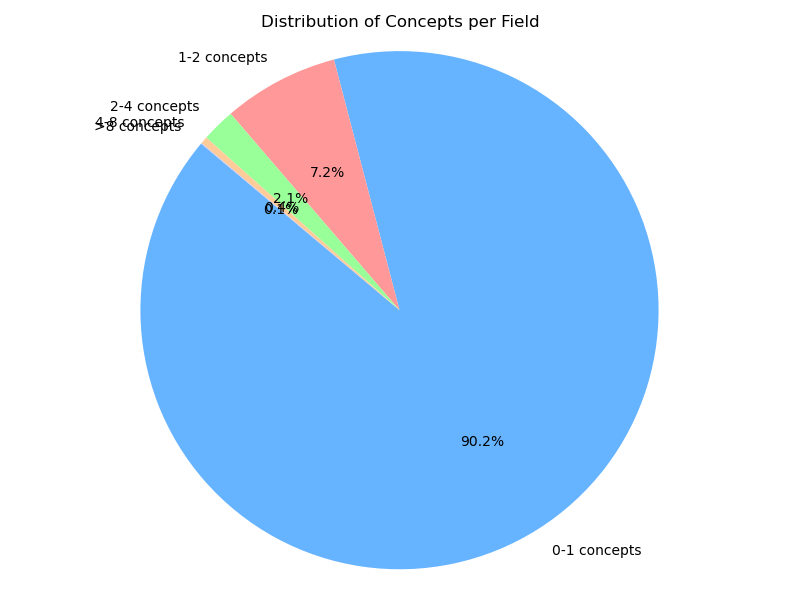


图 3 领域的概念分布情况

对于图3中的领域概念分布情况分析：

* 共有556606个领域，637572个概念。
* 超过97%的领域涵盖概念少于2，利用概念操作难度大
* 长尾分布

有了第一步对数据中实体的分析，我们知道，**大部分的数据能利用的程度较小**。因此，我们自然而然地萌生了筛选数据的想法，我们希望筛选出那一部分最有用的数据，在基于这一小部分数据进行学习路径预测的任务。所以接下来，我们要做的就是，从任务最核心的角度以及根据数据的核心标注筛选出最有用的那一部分数据。

###### 2.1.2 从Concept出发-数据筛选

**对数据的先验观察**：我们已经知道的是，在一个知识图谱中，图上的关系中蕴含的信息和知识往往远多于单独的实体。因此我们首先对relations文件中的数据进行观察（如图4），并且发现，已经组织好的关系中，大多数都是关于concept的关系。也就是许多关系都与知识点核心概念分不开联系，例如concept-course,concept-paper,concept-problem等等。换句话说，概念和所有的其他实体都有一个直接或间接的关联。把实体看做一张拓扑图，对于概念来说，任何一个其他的实体结点都是可达的。**基于此，我们发现，对于数据的筛选任务可以直接转化为对concept的筛选。**

**对Concept的理论分析：**既然作为一个在线教学的数据集，无论应用于任何类型的下游任务，课程中提及到的知识点概念都是任务的关键。从这些概念出发，**我们可以将视频，课程，讨论交流，课后习题等都原子化为多个概念在不同情景下的一种组合**。因此，知识点概念-Concept，可以作为我们在完成该任务时贯穿一整个流程的核心思想，也是我们对于这个庞大的数据架构的手术般精密的切入点。

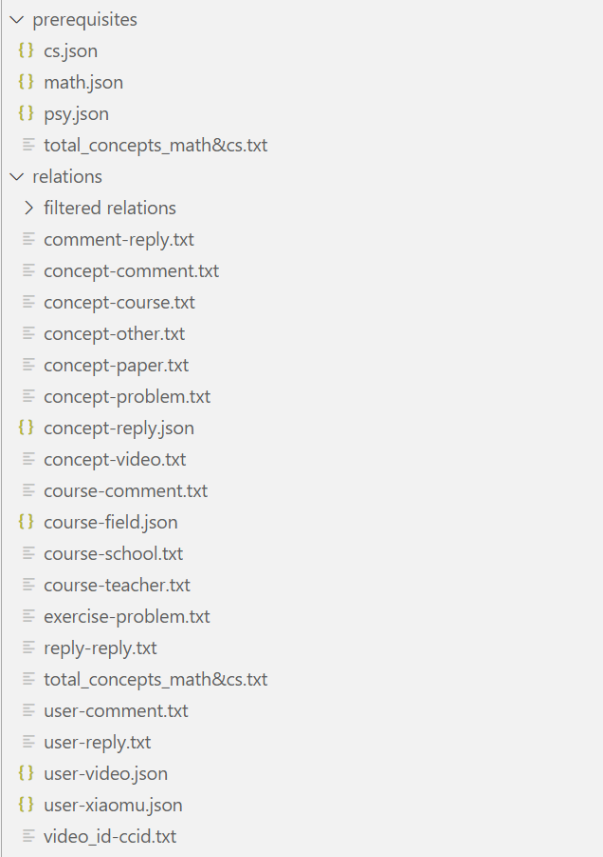


图 4 数据组织结构

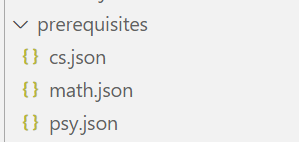
**筛选Concept的依据**：对一个值得注意的点是，原始数据额外在prerequisites文件夹中提供了已标注的数学、心理学和计算机科学三大学科的相关概念的所有先后修关系（图5）。这个是所有数据中唯一不是直接从知识图谱上生成的，而是加入了人为标注的部分。因此，这个部分的数据应该是我们框定数据范围的核心，结合先前做过的对于领域-概念分布的分析（图3），我们发现这三个领域的概念加起来有上千个，远远高于领域-概念的平均值，因此这就是我们想要的概念部分。广阔天地，大有可为。

图 5 先后修关系标注

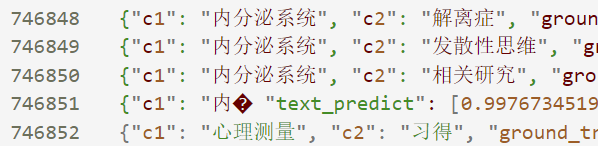
**实际筛选过程-混乱错误的数据格式：**对于数据的筛选，我们首先要获取我们需要的所有concept，组织成一个列表，然后通过这个列表来筛选数据。但是在读取cs.json, math.json, psy.json这三个文件的时候发现，psy.json，也就是心理学的概念先后修标注表格中，存在大量意料之外的数据格式错误（图6）

图 6 psy.json中的格式错误示例

而正是因为这种情况的大量发现，使得我们无法以.json的格式正常读入，也自然无法对错误的值进行自动化地修正。想要正常使用这批数据，只能手动修改，但无异于大海捞针，遂选择放弃使用这部分数据。最终只使用math.json和cs.json。抽取出了1264条数据。如图7。

图 7 抽取出的所有概念

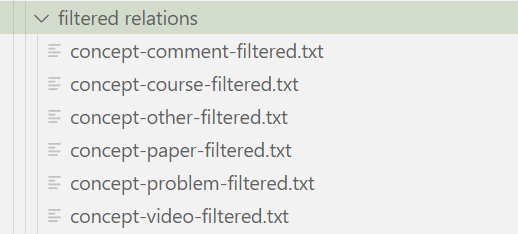
接下来，我们利用这些概念对relations的数据进行筛选，将相关的部分提取。

图 8 筛选过的关系

###### 2.1.3 格式适应-数据集的构造

**学生选课情况数据集-时间序列任务**：

**原始划分**：原始数据中有一个user-course\_order的对应记录（图9）。约80万条。我们选择这个作为需要训练的数据和ground\_truth。

对于这样一个整体的序列，我们使用滑动窗口的方式进行训练和测试。训练测试则是对于所有数据进行按比例划分。但是这样的问题是，我们会用其他用户的数据来预测当前用户的数据，**理论上的正确性无法保证。**

图 9 课程order整体序列构建

**更精密的划分**：上面的数据是所有数据的总体序列，我们从上面的序列中根据我们的concept筛选出所有的user和course的对应记录。构建一个整体的user-course\_order序列，共59226条记录。

我们让数据中的每一个user对应一个课程学习的序列。因此我们进一步对这个整体的有序序列根据user\_id进行划分（如图10）。对于这个数据我们在训练模型时使用的设定是：**使用每个序列的最后一个数据作为label，前面的其他数据作为输入**。这样保证了只用用户的数据来预测他个人接下来的学习路径。

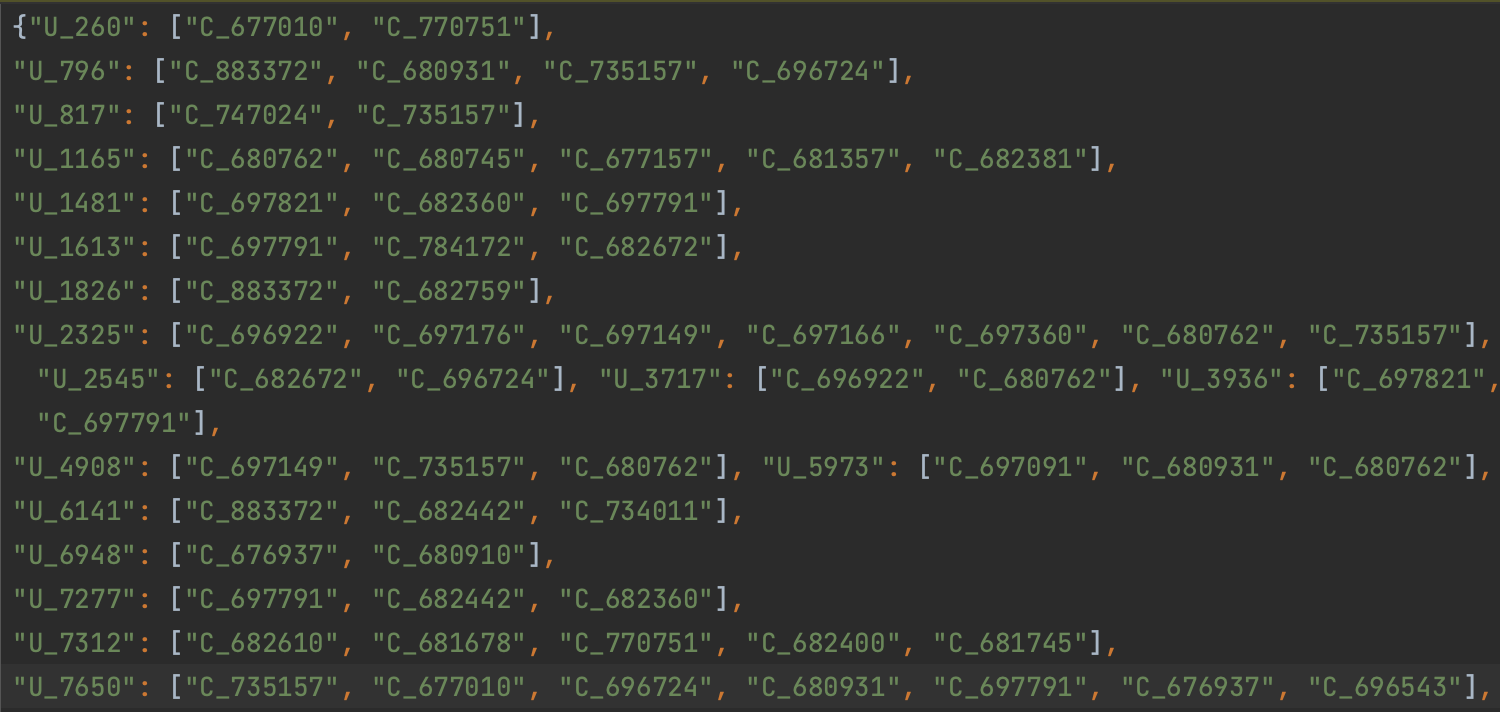
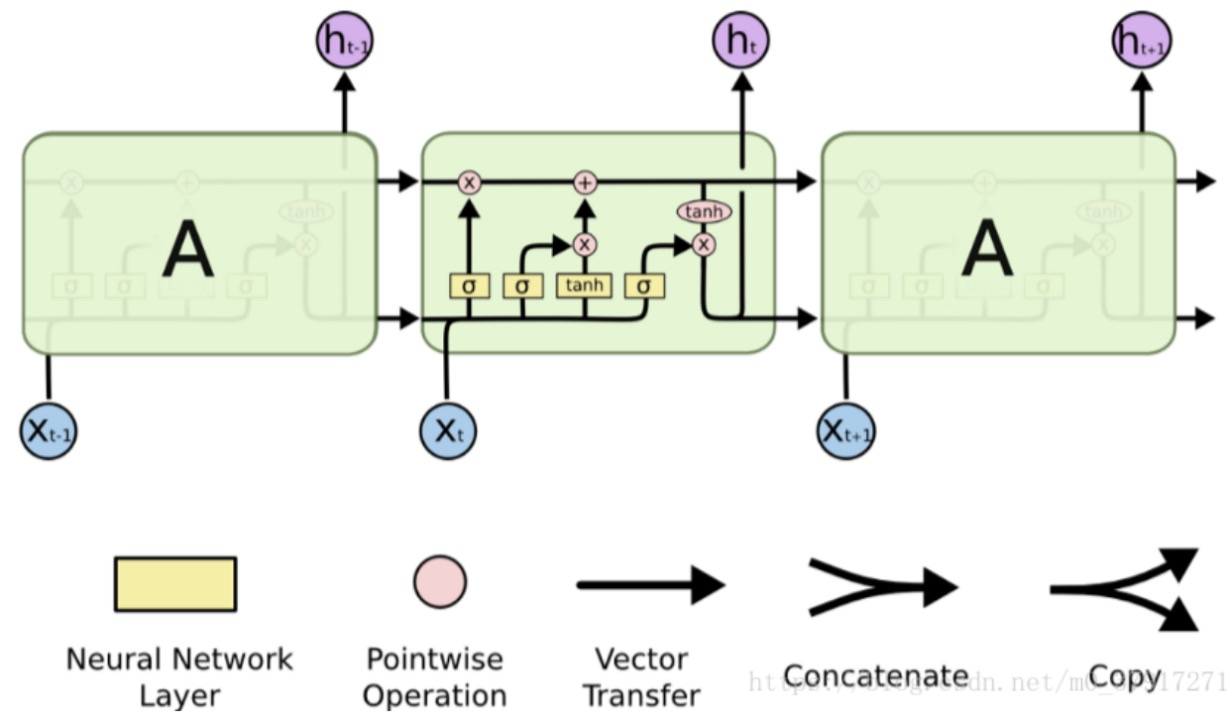
**训练测试集的划分则是根据用户进行划分，**前一部分用户的数据用作训练，后面一部分则用作测试。

图 10 针对每个user的时间序列的数据集

##### 2.2 算法介绍

###### 2.2.1 LSTM&BiLSTM

LSTM算法是一种重要的目前使用最多的时间序列算法，是一种特殊的RNN（Recurrent Neural Network，循环神经网络），能够学习长期的依赖关系。主要是为了解决长序列训练过程中的梯度消失和梯度爆炸问题。



LSTM使用三个门控制对前一段信息、输入信息以及输出信息的记忆状态，进而保证网络可以更好地学习到长距离依赖关系，分别为遗忘门、输入门和输出门：

（1）遗忘门（记忆门）：通过判断当前输入信息的重要程度决定对过去信息的保留度。在我们LSTM中的第一步是决定我们会从细胞状态中丢弃什么信息。这个决定通过一个称为忘记门层完成。该门会读取ht−1和xt，输出一个在 0到 1之间的数值给每个在细胞状态Ct−1中的数字。1表示“完全保留”，0表示“完全舍弃”。

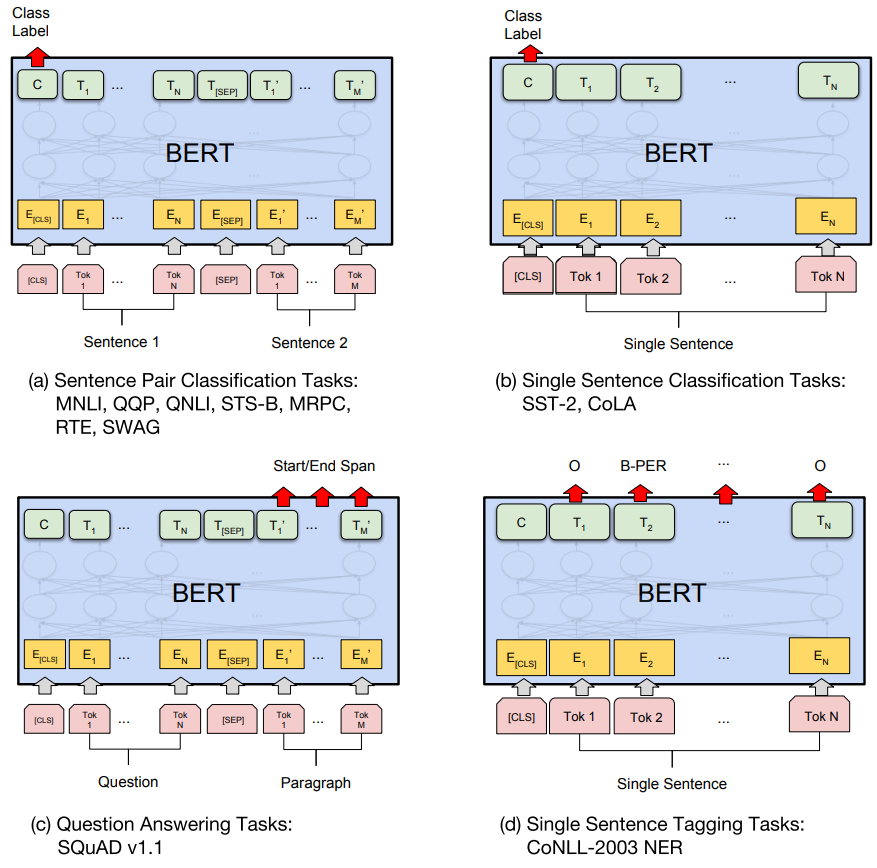
（2）输入门：通过判断当前输入信息的重要程度决定对输入信息的保留度，即决定让多少新的信息加入到cell状态中来。实现这个需要包括两个步骤：首先，一个叫做“input gate layer”的sigmoid层决定哪些信息需要更新；一个tanh层生成一个向量，也就是备选的用来更新的内容。在下一步，我们把这两部分联合起来，对cell状态进行一个更新。

（3）输出门：当前输出有多大程度取决于当前记忆单元。最终，我们需要确定输出什么值。这个输出将会基于我们的细胞状态，但是也是一个过滤后的版本。首先，我们运行一个sigmoid层来确定细胞状态的哪个部分将输出出去。接着，我们把细胞状态通过tanh进行处理（得到一个在-1到1之间的值）并将它和sigmoid门的输出相乘，最终我们仅仅会输出我们确定输出的那部分。

###### 2.2.2 预训练模型进行word\_embedding

BERT的全称是Bidirectional Encoder Representation from Transformers，即双向Transformer的encoder，因为decoder是不能获要预测的信息的。模型的主要创新点都在预训练方法上，即用了Masked LM和Next Sentence Prediction两种方法分别捕捉词语和句子级别的表示。BERT模型实际上是学习到输入序列的特征表示，然后再把学习到的特征表示应用到不同的下游任务。关于为什么要使用双向的结构，作者在论文中的解释是，单向的结构限制了预训练表示的能力，单方向的left-to-right结构或者right-to-left结构，抑或是简单将left-to-right 特征和right-to-left简单拼接的ELMo结构都是不够的，这是因为预训练得到的BERT模型要用于处理不同的任务。因此，不仅需要知道序列中某个词左边的信息，还需要知道右边的信息，并把这两个方向的信息整合起来。

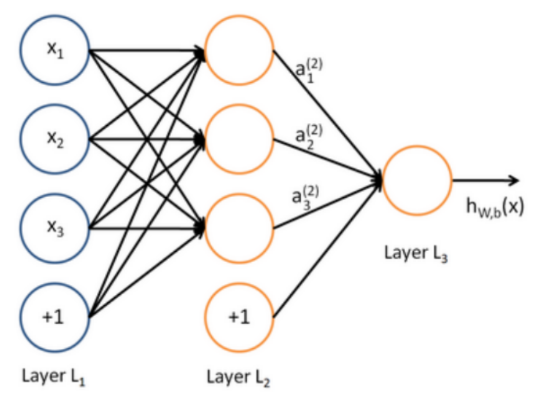
针对特定的下游任务，设计对应的输出层添加到BERT模型的最高层，使用pre-training阶段训练好的模型参数对BERT模型进行初始化，然后在该任务的带标签的数据集上对BERT模型的参数进行微调。下面是BERT模型在不同任务中的应用场景：



###### 2.2.3 MLP+距离度量方法的介绍

多层感知机（Multilayer Perceptron，MLP），是一种由多个神经元组成的前馈神经网络（Feedforward Neural Network，FFN）模型。每个神经元接收来自上一层神经元的输出，并通过激活函数进行非线性变换，然后将结果传递给下一层神经元。这种前馈传播的方式使得MLP能够处理复杂的非线性关系。

MLP由输入层、隐藏层和输出层组成。输入层接收外部输入数据，隐藏层对输入数据进行中间表示的学习，输出层生成最终的预测结果。每个神经元与上一层的所有神经元相连，权重和偏置用于调整信号传递过程中的参数。MLP通过反向传播算法来训练模型，其中使用梯度下降优化算法来调整模型参数，使得模型能够对输入数据进行拟合和预测。



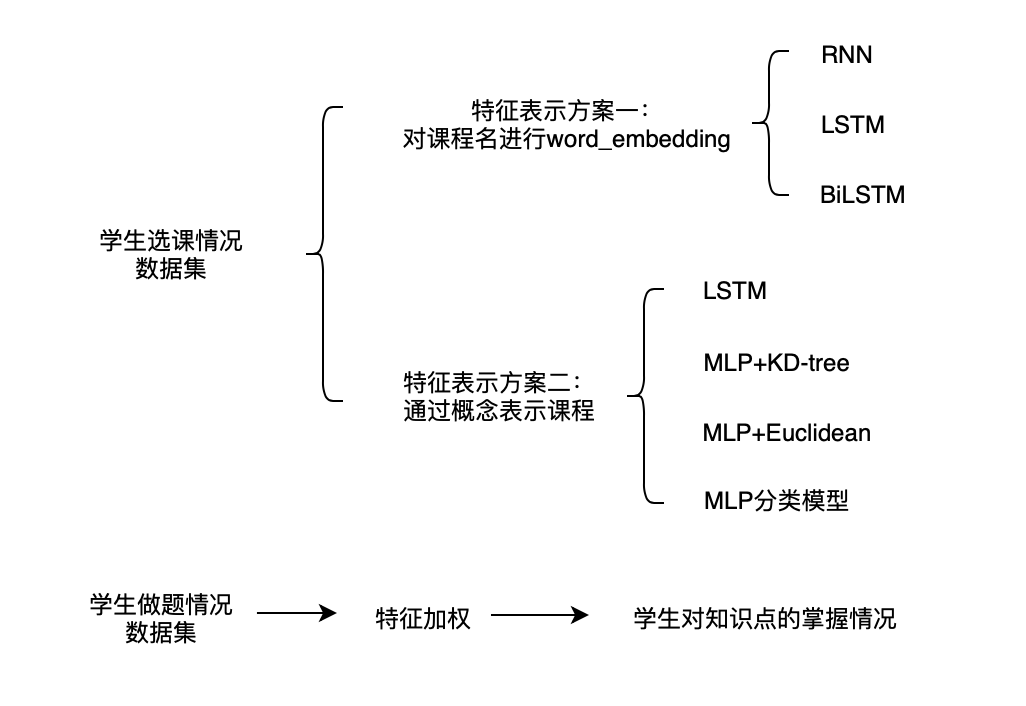
在本项目中，MLP回归模型的目标是拟合一个课程向量，即根据输入的特征向量预测出对应的课程向量。然后，对输出的向量进行距离度量，找到所有课程向量中**最近的向量**作为预测结果。我们使用了kd-tree和欧氏距离两种度量方式。

kd-tree它通过将数据点递归地分割为多个子空间，并在每个子空间中构建一个二叉树来组织数据，从而实现在多维空间中对数据进行高效的最近邻搜索。欧氏距离是最常用的距离度量方法之一，衡量两点在欧氏空间中各个坐标差的平方和的平方根。

而MLP分类模型则是直接预测标签，根据输入的特征向量预测出对应的课程类别，即建立了一个用户向量空间到标签空间的映射。回归模型区别于分类模型，回归模型构建的是从一个向量空间到另一个向量空间的映射，没有向量空间到标签空间的投影关系，而是将这种关系简单设定为距离度量后选择最近邻。我们后续对这两种模型进行了对比实验，同时也对两种距离度量方式进行比较。

##### 2.3 特征构造

下面是我们特征构造和算法使用的整体流程图，在此简单作为一个引入。



###### 2.3.1 特征表示方案一：对课程名word\_embedding

这里我们的任务设定如下：

* 使用未筛选的所有数据构建的序列，近80万条数据，涉及所有领域和概念
* **训练集和测试集直接从单条选课记录数据角度划分**
* 数据未经过用户粒度的划分方案，也就是在时间序列的滑动窗口中会利用别人的数据来预测。
* 特征为课程名本身在词向量空间的嵌入，标签为课程名，也就是用课程名来预测课程名。

对于这样的方案，我们分别采用了RNN，LSTM，以及BiLSTM来直接拟合这样的巨量时序数据。

###### 2.3.2 特征表示方案二：通过概念的embedding进行组合来表示课程的特征

我们首先分析这样的思路，在[**2.1.2**](#_2.1.2_从Concept出发-数据筛选)**-对Concept的理论分析**，我们已经提到一个理论，就是无论是视频，课程，讨论交流，课后习题，或者其他与在线课程相关的数据实体，**都实际上是多个概念在不同情景下的一种组合表征。**

我们借鉴Word2Vec的思路，对于课程来说，**如果两个课程涉及到的概念成分越相近，那我们就希望它们表示出来的特征就在特征空间中距离更近**。因此我们如果用课程涉及到的许多Concept的组合来表示一个课程的特征，那么就完全可以实现这样的效果。

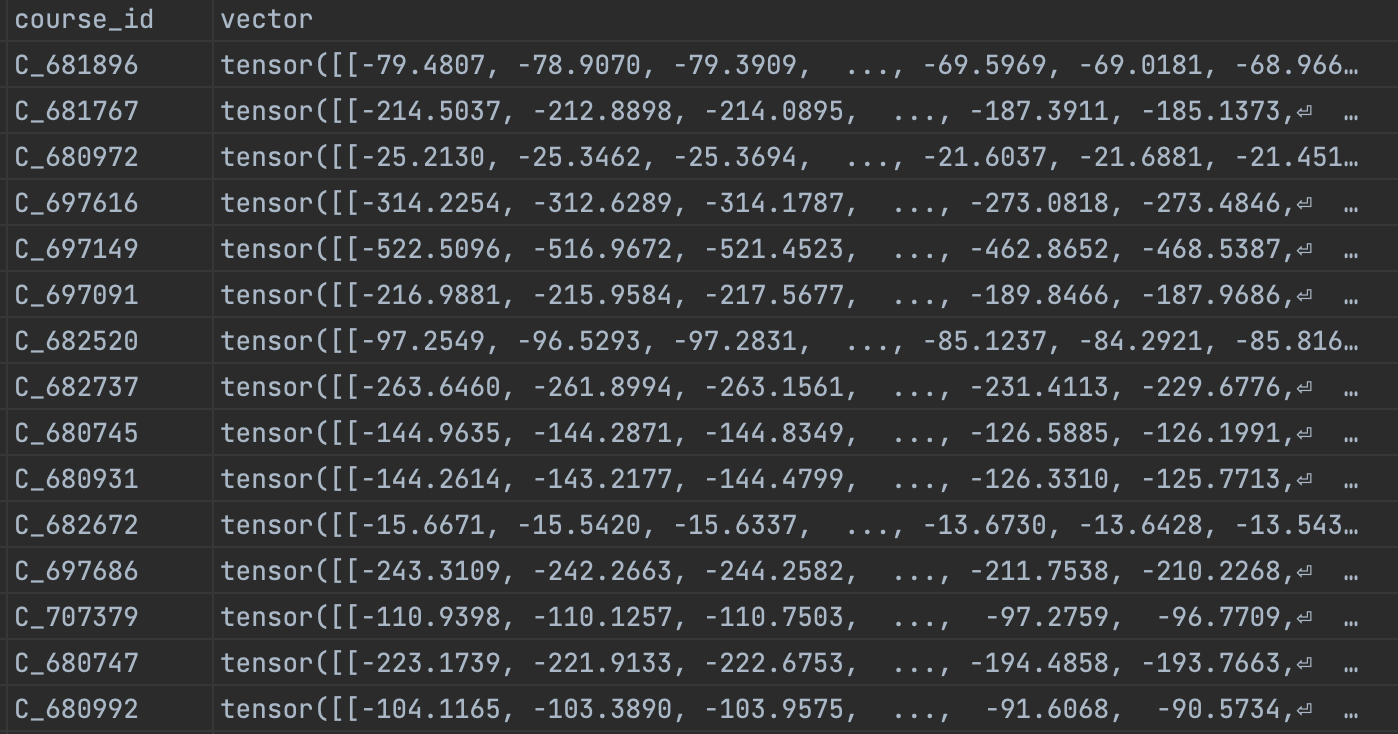
因此我们**直接对概念使用bert-base-chinese的预训练模型进行词向量空间嵌入**，然后为了表示一个课程的特征，我们首先搜寻到这个课程涉及的所有视频，然后在每个视频的字幕文件中搜索这些概念出现的频率，**统计出整个课程中每个概念出现的频率**，然后利用这个频率进行加权，得到一个最终的课程特征。（如图11所示）

图 11 基于概念构建的课程向量

现在已经构造出这样的特征空间，我们就可以在其中大展拳脚。假设一个用户的选课顺序序列中有n条数据，我们选择前n-1条作为特征，最后一条作为我们的标签。

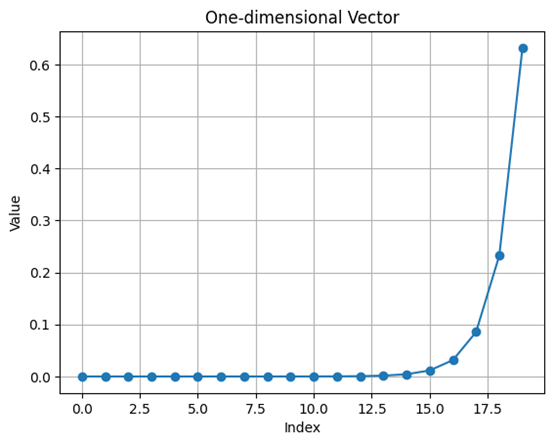
我们接下来再**利用一个用户已经学习的课程向量加权构造一个用户的选课特征**，权重由学习的课程时间的远近来决定，距离当前时间点远的课程自然权重较低，也就是我们不倾向于用一个用户很久以前学习的课程来预测新的课程，而是更倾向于用最近学习的课程来预测。由于遗忘曲线的存在，所以这样的加权无疑是合理的。并且权重用指数滑动平均来获得。权重随时间步变化的曲线如图12所示：

图 12 课程权重随时间步变化

这样，我们就得到了每一个用户已经课程学习情况的特征，接着我们使用了LSTM，MLP分类模型（label设定为标签）,MLP回归模型（label设定为标签课程的向量嵌入）+度量这三种方案来进行拟合我们构造的特征和标签。

总结一下，我们的任务设定如下：

* 使用筛选过的数据，共计约6万条，**训练集和测试集按照用户层次划分。**
* 数据已经过用户粒度的划分方案，在任务中**只会利用一个用户自身的学习数据**来预测该用户未来的学习路径。 理论正确性完全可以保证。
* 输入为上面提及到的基于concept的embedding构建的课程向量经过指数滑动加权获得的用户选课特征。标签可以为标签课程的序号（分类任务），也可以是标签课程的向量（回归任务），我们都进行了实验进行对比。

##### 2.4 实验结果

###### 2.4.1 直接上手尝试—全量数据&特征表示方案一

这是一个最原始的思路：我们使用了特征表示方案一，直接用所有的课程序号来预测课程序号，并且如[2.1.3](#_2.1.3_格式适应-数据集的构造)中原始划分方案所述，使用**不加筛选和处理的数据集**（未经过用户粒度的划分，会导致其他用户的数据用来预测当前用户或者同一个用户的数据分别被划分到训练和测试集当中，理论正确性无法完全保证）。

**主要实验**：三种方案和测试集比例的实验结果如Table 1所示：

**Table 1 Results with only course\_id**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Top1 – Acc | | | | Top5 - Acc | | | |
| 20% | 10% | 1% | 0.05% | 20% | 10% | 1% | 0.05% |
| LSTM | **18.55** | 14.40 | **19.42** | 22.02 | **49.14** | 47.30 | 49.29 | **50.30** |
| RNN | 17.97 | 14.20 | 16.24 | **22.82** | 47.87 | **47.52** | **49.56** | 49.90 |
| BiLSTM | 18.33 | **15.36** | 19.20 | 19.39 | 48.80 | 47.44 | 49.29 | 47.88 |

embedding\_dim = 50, (128), window\_size = 5, epochs=5, batch\_size=256

可以看到效果还挺好，其实这一点与数据分析[2.1.1](#_2.1.1_如何切入-数据分析)中提及到的领域概念图的分布也有关系，即大部分有效的数据都集中在小部分领域，导致用全量数据的效果也不错。

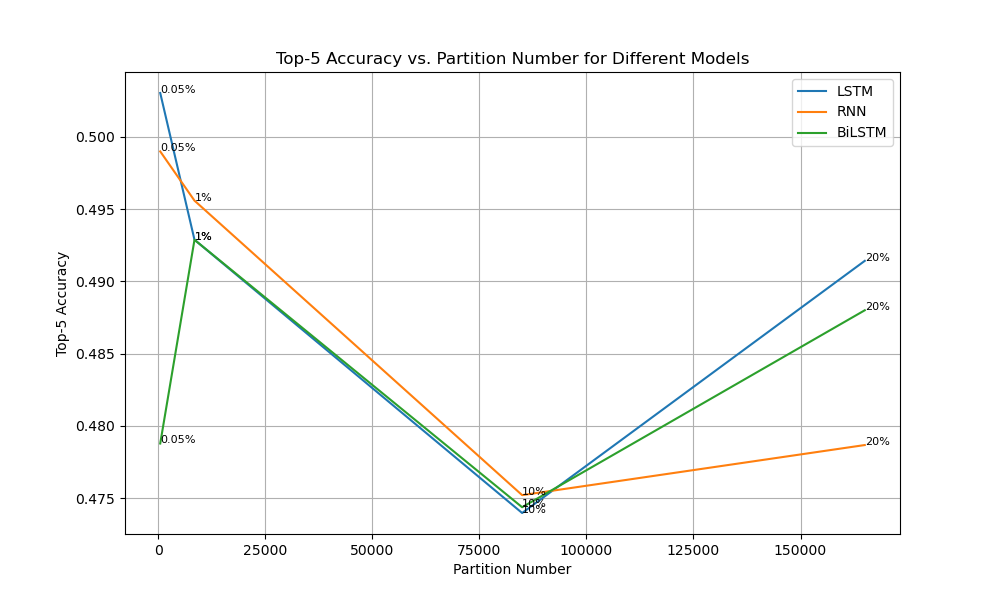
**对比实验：不同算法的性能分析**，如图13所示，LSTM和BiLSTM明显整体更优。

图 13 算法性能对比

**参数调优1：测试集比例**。我们固定算法为LSTM和其他条件不变，发现测试集比例越低效果越好。如Table 2所示

**Table 2 Test size**

|  |  |  |  |
| --- | --- | --- | --- |
| Testing size | Top1 - Acc | Top5 - Acc | Loss |
| 20% | 18.55 | 49.14 | 3.126 |
| 10% | 14.40 | 47.30 | 3.109 |
| 1% | 19.42 | 49.29 | 3.114 |
| 0.05% | **22.02** | **50.30** | 3.098 |

参数：LSTM: **embedding\_dim=50, LSTM(128),window\_size = 5**,epochs=10, batch\_size=256

**参数调优2：BatchSize**。我们固定其他条件,调整batchsize，如Table 3所示。发现Batch size=32时最佳。

**Table 3 Batch size**

|  |  |  |  |
| --- | --- | --- | --- |
| Batch size | Top1 - Acc | Top5 - Acc | Loss |
| 16 | 22.42 | 49.09 | 3.120 |
| 32 | **23.23** | **49.29** | 3.114 |
| 64 | 23.03 | 47.07 | 3.071 |
| 128 | 20.20 | 46.67 | 3.041 |
| 256 | 19.39 | 46.87 | 3.020 |
| 512 | 18.38 | 46.87 | 3.004 |

其他参数:LSTM, embedding\_dim = 50, LSTM(128),window\_size = 5,epochs=10 ,X\_all[:-500]

**参数调优3：Window size**。我们固定其他条件,调整LSTM的滑动窗口大小，如Table 4所示。发现Window size=5时最佳。

**Table 4 Window size**

|  |  |  |  |
| --- | --- | --- | --- |
| Window size | Top1 - Acc | Top5 - Acc | Loss |
| 2 | 14.46 | 47.99 | 3.236 |
| 3 | 17.71 | 48.29 | 3.161 |
| 4 | 20.36 | 49.60 | 3.111 |
| 5 | **22.63** | **51.72** | 3.129 |
| 6 | 21.66 | 49.60 | 3.051 |
| 7 | 21.10 | 49.29 | 3.033 |
| 8 | 21.75 | 49.19 | 3.019 |
| 9 | 21.59 | 46.64 | 3.008 |

其他参数:LSTM, embedding\_dim = 50, LSTM(128), epochs=10, batch\_size=256, X\_all[:-500]

**参数调优4：LSTM unit**。我们固定其他条件,调整LSTM的unit大小，如Table 5所示。发现LSTM unit=64 时最佳。

**Table 5 LSTM unit**

|  |  |  |  |
| --- | --- | --- | --- |
| LSTM unit | Top1 - Acc | Top5 - Acc | Loss |
| 32 | 22.42 | 49.70 | 3.170 |
| 64 | 22.02 | 49.90 | 3.150 |
| 128 | **22.63** | **51.72** | 3.129 |
| 256 | 21.01 | 50.71 | 3.043 |
| 512 | 18.59 | 44.04 | 2.789 |

其他参数:LSTM, embedding\_dim = 50, window\_size = 5 epochs=10, batch\_size=256, X\_all[:-500]

**参数调优5：LSTM嵌入空间的维度**。我们固定其他条件,调整LSTM的Embedding dimension，如Table 6所示。发现Embedding dimension=50 时最佳。

**Table 6 Embedding dimension**

|  |  |  |  |
| --- | --- | --- | --- |
| Embedding dimesion | Top1 - Acc | Top5 - Acc | Loss |
| 25 | 20.40 | 48.48 | 3.136 |
| 50 | **22.63** | **51.72** | 3.129 |
| 100 | 21.62 | 48.89 | 3.123 |
| 200 | 21.41 | 49.49 | 3.118 |
| 300 | 22.63 | 50.71 | 3.117 |
| 400 | 22.63 | 49.90 | 3.117 |
| 500 | 22.02 | 49.09 | 3.118 |

其他参数:LSTM, LSTM(128), window\_size = 5 epochs=10, batch\_size=256, X\_all[:-500]

**参数调优6：Epoch**。我们固定其他条件,调整训练的Epoch数，如Table 7所示。发现Epoch=5 时最佳。证明数据量大，小epoch足够，Epoch大的时候可能过拟合。

**Table 7 Epoch**

|  |  |  |  |
| --- | --- | --- | --- |
| Epoch | Top1 - Acc | Top5 - Acc | Loss |
| 5 | **22.02** | **50.71** | 3.156 |
| 10 | 21.41 | 49.90 | 3.110 |
| 20 | 21.01 | 49.49 | 3.054 |
| 40 | 18.18 | 49.49 | 3.009 |
| 80 | 16.36 | 46.67 | 2.983 |

其他参数:LSTM, embedding\_dim = 50, LSTM(128), window\_size = 5, batch\_size=256, X\_all[:-500]

###### 2.4.2 理论正确性—筛选的数据&特征表示方案二

参考我们[2.3.2](#_2.3.2_特征表示方案二：通过概念的embedding进行组合来表示)中提到的特征表示方法和[2.1.3](#_2.1.3_格式适应-数据集的构造)中的更精密的划分，我们在进行预测和训练的时候，**只用筛选过的数据，并且只用当前用户的数据**来预测当前用户将来的学习路径。

方案的原理[2.2](#_2.2_算法介绍)所示，结果如Table 8所示

**Table 8 Epoch**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Top1 - Acc | | | | Top5 - Acc | | | |
| 20% | 10% | 1% | 0.05% | 20% | 10% | 1% | 0.05% |
| LSTM | 8.0 | 8.0 | 8.2 | 6.6 | **34.7** | **35.5** | 34.7 | 36.6 |
| MLP®+KD-tree | 4.7 | 1.6 | 4.5 | 8.9 | 11.5 | 13.4 | 12.1 | 15.6 |
| MLP®+Euclidean | 4.4 | 1.6 | 5.9 | 8.6 | 11.1 | 13.0 | 14.8 | 17.8 |
| MLP© | **8.6** | **8.5** | **10.7** | **22.0** | 33.2 | 34.3 | **37.9** | **43.4** |

®回归模型（kdtree距离/欧式距离），©分类模型。

我们对两种策略的**对比实验**如Table 9所示：

**Table 9 Epoch**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Feature | Top1 - Acc | | | | Top5 - Acc | | | |
| 20% | 10% | 1% | 0.05% | 20% | 10% | 1% | 0.05% |
| Non-Vectorized | **18.6** | **14.4** | **19.4** | **22.0** | **49.1** | **47.3** | **49.3** | **50.3** |
| Vectorized | 8.0 | 8.0 | 8.2 | 6.6 | 34.7 | 35.5 | 34.7 | 36.6 |

对于当前实验以及两种实验的实验和理论分析我们在2.5中具体阐述

##### 2.5 实验分析

对于单独的用户-课程数据，我们做了两种分类任务，MLP的分类和LSTM的分类。

首先重述一下我们任务的技术路线：

**1. 首先利用概念的向量来获取课程的向量表示**。

**2. 然后对用户的选课数据表示成一个特征用来训练**。我们在每一条用户数据中选择最后一条选课记录作为**标签**，前面的作为**输入**。接着，我们选择将多个课程的向量进行加权混合。思路是我们按照选课顺序对课程进行加权，越早选的课权重越低，使用指数平滑的权重。

**3.我们将得到的（用户向量：标签）一对一对地输入模型，进行训练。**

**4.分类头预测标签，回归任务拟合向量+搜索最近的课程向量来作为分类结果。**

###### 2.5.1 对两种策略训练过程的差异分析

**A.**实验分析：策略二并不是直接考虑全部数据，而是对一个一个用户的数据进行训练，考虑到每一个用户存在个性化差异，所以策略二数据挖掘的目的在于，构建一个鲁棒的模型用于**个性化地推荐用户的选课数据**。而如果是直接全量选课数据的考虑，也就是策略一，则目标是在于“**基于全网数据（包括别人和自己的）对某个用户进行选课推荐**”，这样的话就是**利用到了其他用户的数据来推荐某个用户的选课，更偏向于大规模数据的挖掘**。由于两批实验利用到的数据规模不在一个级别上（策略一利用的是80万条数据，策略二这里只有6万条数据），但是最优的top-5精度相差却只在7%左右（参考我们的实验表格）,**完全可以表明我们策略二存在优越性**。

**B.**理论分析：策略二是每次只输入一个用户的数据训练一轮（梯度回传）,而且每个用户的个性化选课数据大相径庭。也就是说，**每个用户的选课数据相当于都在不同的分布空间（域）**当中，因此我们**相当于在用多个不同的域的数据**来单独地进行训练，**最终测试的任务又是另一个分布空间**，整体为一个域外泛化的任务。而对于使用全部数据的任务，由于数据规模和训练模式的差异，相当于基本学习到了完整的用户选课数据的分布，少量的测试集与我们学习到的分布则基本相差无几。

**用集合的角度来看**，用户-选课数据为全集U，由于我们的策略是针对**每个用户**选择数据，则我们将U划分为不相交的n个子集V\_i（i=1,2……n），从n个子集中按比例选取了n\_train和n\_test个子集组成训练集和测试集。那么每次相当于是我的模型在V\_i（1<=i<=n\_train）上学习一轮，然后梯度回传一轮更新参数，也就对应了我说的用多个不同域的数据进行训练。即多个子集存在差异，我们并不能从这些子集中学到整个集合U的分布。然后测试一个新的集合V\_j ( n\_train+1 <= j <= n)，可以从这个简单的表示中看到，我们实际上是在做**域泛化**的任务。

而全量选择数据，则是对数据集没有任何划分，不分用户地一次性全部喂进去，也就是U的n\_train个训练，n\_test个测试，由于选取的时候没有进行划分，**每一轮喂入的样本点都随机地分布在U的各个无交集的子集V\_i中**。也就是说，训练的时候相当于一次性把整个域都喂进去，也自然能让模型更好地学习到整个分布。

###### 2.5.2 对回归还是分类的理论讨论

MLP相当于对输入向量-输出之间做一个映射，希望更好地建立一个输入向量空间到输出的映射。如果我们用MLP做分类，即直接将输出设定为类别，则我们是在建立一个用户vector-space到label-space的直接映射，实验证明这样效果最好。

而如果用MLP做回归任务，即label是目标课程的向量，我们的输出为一个向量。然后再对输出的向量进行距离度量，找到所有课程向量中**最近的向量**作为预测结果。这样的理论问题是，由于域差异问题，**我们缺少了一个从vector-space到label-space的投影关系，而是将这种投影关系简单设定为一个距离+搜索**。而经过了回归任务的学习，相当于从一个vector-space投影到另一个vector-space，由于这两个空间中的向量交叉情况难以确定（也就是对于投影f-g，f或g中可能有多个重复或者交叉的元素，导致这个投影函数产生了模糊的情况，也就是投影前和投影后可能存在一个相同的向量V，导致难以确定真实情况），**而对于vector-to-label的投影来说，这样就不会存在，因为目标空间根本和原始不是一个空间**。

## 3. Course Recommendation System

##### 3.1 功能介绍

该自适应学习系统的功能为，根据学生之前所选课程和学习情况，推荐下一步可以学习的课程和相关文献。

系统使用流程如下：

学生输入user\_id，点击“Recommend Course”按键，将输出推荐该学生学习的下一门课程，以及相关论文的id，论文区可以滑动浏览。

##### 3.2 数据介绍

通过观察知识追踪领域常用的数据集Synthetic-5、ASSISTments2009、ASSISTments2015、Statics2011等，我们选择user\_id、problem\_id、is\_correct、attempts以及题目涉及的概念来构造学生做题情况数据集。由于学生做题数据记录存在时序关系，因此我们通过设置order\_id列来保存用户做题顺序。对于每道题目涉及到的概念，我们将所有概念进行hash映射为整数，设置skill\_id列来保存当前题目涉及的概念。若当前题目涉及的概念不只一个，则将记录拆成两条，其余列值均不变，仅skill\_id列取值变化。最后，我们将数据整合成Dataframe格式，共计3195条记录。这个任务上训练测试集的划分与任务一相同。

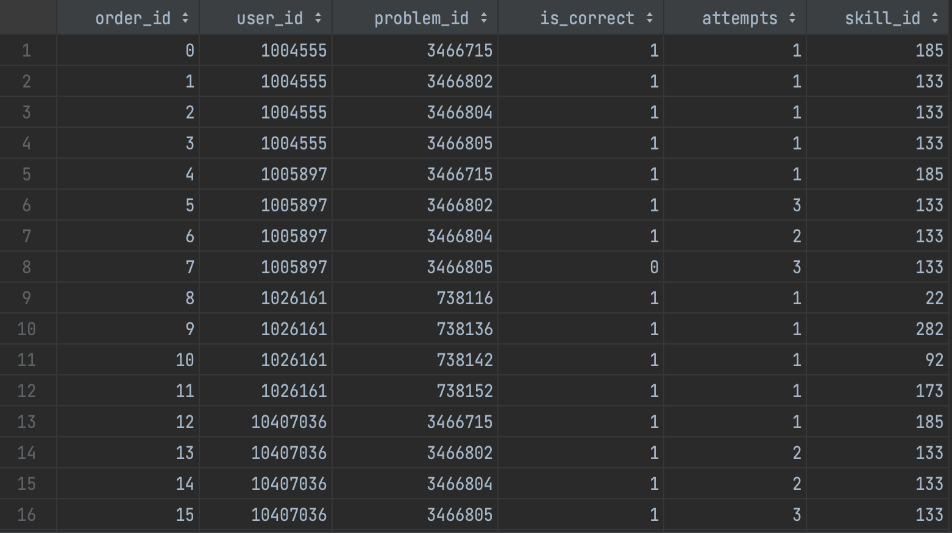


图 14 知识追踪数据集

##### 3.3 算法介绍

起初，我们考虑使用知识追踪模型。知识追踪任务是对学生的知识进行建模，这样我们就可以准确地预测学生在未来的互动中将如何表现。简单来说，就是我们有了学生的历史做题序列数据，可以通过学生与题目的交互结果得出学生当前的知识状态以及题目的信息，这样就可以对学生在题目上的表现进行预测，并且可以动态追踪学生的知识状态变化。

然而，将我们构造的数据与目前知识追踪领域的数据集进行对比后，我们发现对于该数据集难以使用目前主流的知识追踪模型，例如AKT、SimpleKT等，因为该数据集中用户的做题记录过于稀疏，平均每个用户的做题记录在5-10条左右，远远小于这些模型的需求，因此我们采用较为直观的方式来对学生的知识点掌握情况进行建模，即将掌握情况量化为分数来进行加权计算。

我们根据模糊综合评价法来进行特征构造，从而得到该学生对题目涉及的每个知识点的掌握情况。我们认为大于阈值（这里取的阈值为0）代表该学生对该知识点掌握情况较好，则不需要推荐该知识点相关课程给该学生，若小于阈值则代表该学生对该知识点掌握情况不好，则我们为其推荐知识点相关的课程帮助该学生对未掌握好的知识点进行巩固。

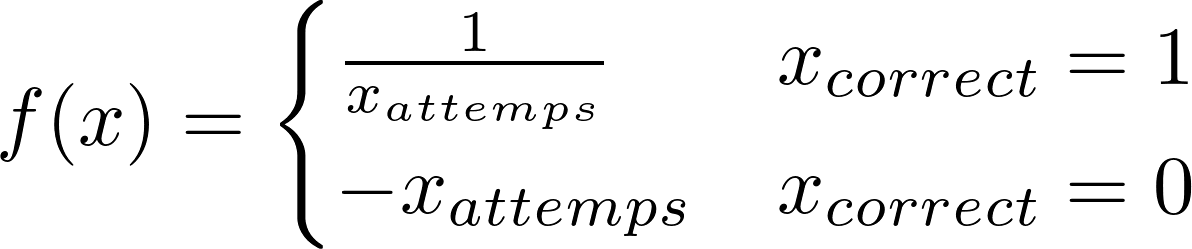
同时，任务一中我们对学生的学习路径进行了预测，并推断出下一时刻学生选择的课程。将二者进行融合，我们得出个性化推荐的课程。

##### 3.4 特征构造

通过观察，我们发现学生做题情况数据集中is\_correct属性对该知识点是否掌握有着决定性的作用。尝试次数与知识点掌握情况呈负相关，即尝试次数越多代表着该学生对该知识点的掌握越不熟练。基于上述发现，我们采用模糊综合评价法来进行特征构造。

模糊综合评价法是一种基于模糊数学的综合评价方法，它通过模糊数学的隶属度理论将定性评价转化为定量评价。这种方法适用于对受到多种因素制约的事物或对象进行总体评价，特别是在处理模糊性和难以量化的复杂问题时表现出色。在模糊综合评价中，首先需要确定评价对象的指标个数，并设置评语集。接着，计算各个指标的权重向量和权重判断矩阵，这些矩阵将用于合成模糊综合评价结果。权重向量的确定是关键步骤，因为它会影响最终的评价结果。权重可以通过多种方法计算，如层次分析法、主成分分析法等。权重向量通常需要满足归一化条件，即所有权重之和等于1。

在该问题上，我们通过以下公式来对不同属性进行加权，建模用户对知识点对掌握情况，并对结果进行归一化，从而进行特征构造。



##### 3.5 实验结果

随机挑选一名学生，user\_id=3491847，在user.json文件中找到该学生的选课序列，course\_order=[681299, 696724, 879040, 734011, 735157, 677133, 696854, 677140].

再在course.json文件中依次查询该学生所选课程，分别是：《统计学》，《概率论与数理统计》，《经济学原理（宏观部分）》，《大数据平台核心技术》，《算法设计与分析》，《数据库系统（上）：模型与语言》，《数据库系统（中）：建模与设计》，《数据库系统（下）：管理与技术》

根据我们的系统，给这位学生推荐的下一门课程是《数据结构（上）》。通过比对这名学生的学习顺序，先学习数学、经济相关知识，又学了算法、数据库，推荐数据结构是合理的。因此证明了我们推荐算法的有效性。

##### 3.6 系统界面

主要使用tkinter搭建GUI界面，创建了一个名为“Course Recommendation System”的主窗口，并设置了背景色为浅蓝色。创建一个标签和一个文本输入框，用于用户输入ID；类似的构建课程推荐的输出框；相关论文推荐的输出框增加了一个滚动条，使得用户能够滚动查看文本内容。推荐按钮“Recommend Course”点击后会触发响应函数，进行课程推荐等操作，输出结果。

## 4.结论

图 15 GUI界面展示结果

本项目

## 5.进度安排

介绍项目的分解情况，以及每个队员的任务分配以及完成情况，推荐以甘特图的形式呈现。

## 6.团队特色

* 基于概念筛选数据
* 基于概念的词向量嵌入表示，来获取课程和用户选课数据的总体特征
* 做了巨量的实验，感谢学院让我蹭到4090
* 使用了玛卡巴卡作为我们的队名，并且希望玛卡巴卡的力量传递

## 参考文献

如果使用了参考文献，附加在这一章节。