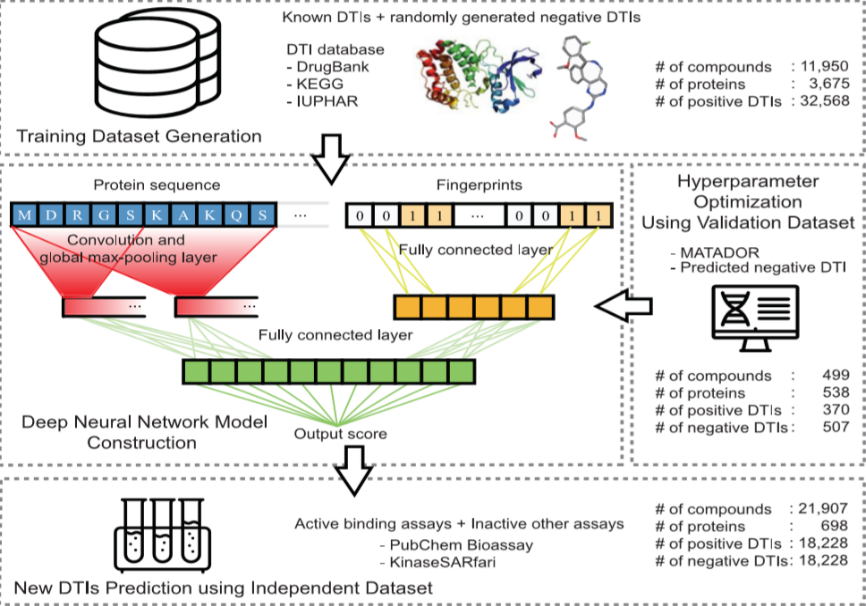
**生物信息学project 2 项目报告**

1. 复现论文简介

药物靶点相互作用（dti）的识别在药物发现的早期阶段起着关键的作用，它有利于药物开发人员筛选出与特定靶点相互作用的化合物。然而，在大规模的化学或生物实验中，dti的鉴定通常需要2~3年的实验，相关费用较高。因此，随着药物、靶点和相互作用数据的积累，各种各样的计算方法被开发出来用于预测可能的dti，以帮助药物发现。在一些计算模型中，传统的蛋白质描述符被证明没有足够的信息来预测准确的DTI。

因此，本文提出了一个基于深度学习的DTI预测模型，捕捉参与DTI的蛋白质的局部残留模式。作者使用了大规模的DTI信息训练模型，并使用独立的数据集展示该模型的性能。 因此，作者的模型比以前基于蛋白质描述符的模型表现得更好。此外，作者的模型比最近发展的深度学习模型在DTIs的大规模预测方面表现更好。通过对混合卷积结果的检验，作者证实了他们的模型可以预测DTIs。总之，作者的预测模型成功地丰富了原始蛋白质序列的蛋白质特征，得到了更好的结果。

模型的网络架构如下图所示：



其中，输入为靶标蛋白的氨基酸序列和药物分子的摩根指纹，靶标蛋白的氨基酸序列经过embedding矩阵处理后与摩根指纹进行拼接(concatenate)，之后输入卷积层和全连接层进行训练。

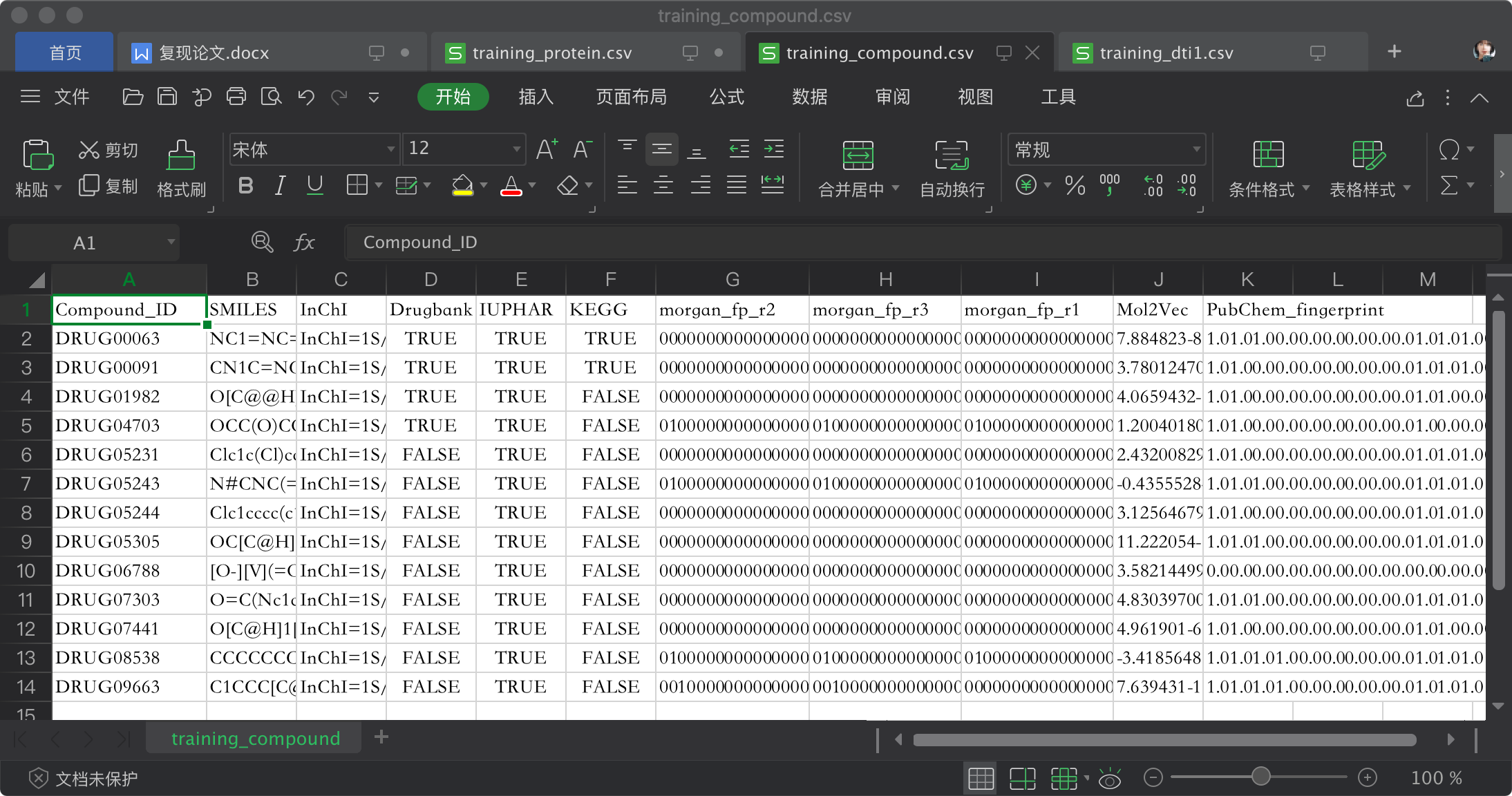
二、data的下载与展示

1、原作者给出的data样例——toy\_example

按照以往复现论文的经验，作者提供的源代码中往往会包含相关数据集或打包完成的数据集的链接，而复现者所需要做的仅仅是下载相关数据集、配置环境、运行源代码既可以得到论文中相关模型的结果。然而，本文的作者仅提供了极少的样例数据集，完整的数据集需要复现者从drugbank、kegg等数据库自行下载并做数据清洗和数据预处理的操作，大幅增加了复现的工作量。我们首先来看作者给出的样例数据。

整个数据集分为测试集、验证集和训练集，每个集合都包含三个文件，分别为compound、protein和dti。其中，compound用来存储药物（drug）的相关信息和特征，protein用来存储药物的靶向蛋白的相关信息和特征、dti用来存储药物-靶标关系。下面我们详细展示数据的各个字段信息。

1. **compound文件**



其中，各个字段的含义如下：

Compound\_ID:药物ID。由于作者采用的药物数据来源于多个数据库，所以直接采用数据库自带的ID(例如，drugbank的DB000x，pubchem的CID等)会造成ID形式不统一。所以作者对来自不同数据库的数据统一为DRUG000x的形式。

SMILES:简化分子线性输入规范(Simplified molecular input line entry specification)。用ASCII字符串来明确描述分子结构，是药物特征（feature）之一。

InChI:国际化合物标识(International Chemical Identifier)。作用类似于SMILES，用来描述化合物的分子结构。由于SMILES更容易被程序处理，故InChI在这里属于冗余信息。

Drugbank/IUPHAR/KEGG:表示三个数据库，字段信息表示该药物能不能在相应的数据库检索到，该字段不属于药物特征信息。

Morgan\_fp\_r2,Morgan\_fp\_r3,Morgan\_fp\_r1:摩根指纹，即扩展连通性指(ECFP)，是用于构建化合物定量构效关系(QSAR)模型的分子指纹。该指纹通过python的rdkit模块以及化合物的SMILES计算得到。摩根指纹是一种环形指纹，在计算过程中需要指定半径n(即迭代次数)。数据集中对应的三个摩根指纹分别是半径为1、2、3时计算得到的摩根指纹。

Mol2Vec:是由*Peng et al*在另一篇文章中仿照word2vec构造的分子向量嵌入。该文章最终通过优化一个支持向量机来实现分类。这里作者为了比较不同化合物的描述符(descriptor)对最终预测结果对差异，故引入了该属性。我们不复现其他文章对模型，故该属性不需要。

PubChem\_fingerprint:由pubchem数据库提供的区别于摩根指纹的另一种分子指纹。目的仍是比较不同化合物的描述符(descriptor)对最终预测结果对差异。作者已经明确指出，当使用摩根指纹且半径设定为2时，模型取得最好的效果，故该属性我们也不使用。

1. **protein文件**



Protein\_ID:靶标蛋白ID。该ID对应drugbank中的蛋白质ID。

Uniprot\_id:靶标蛋白ID。该ID对应UniProt蛋白质数据库中的蛋白质ID。

Taxonomy:蛋白质分类。图中数据均属于人类蛋白质。

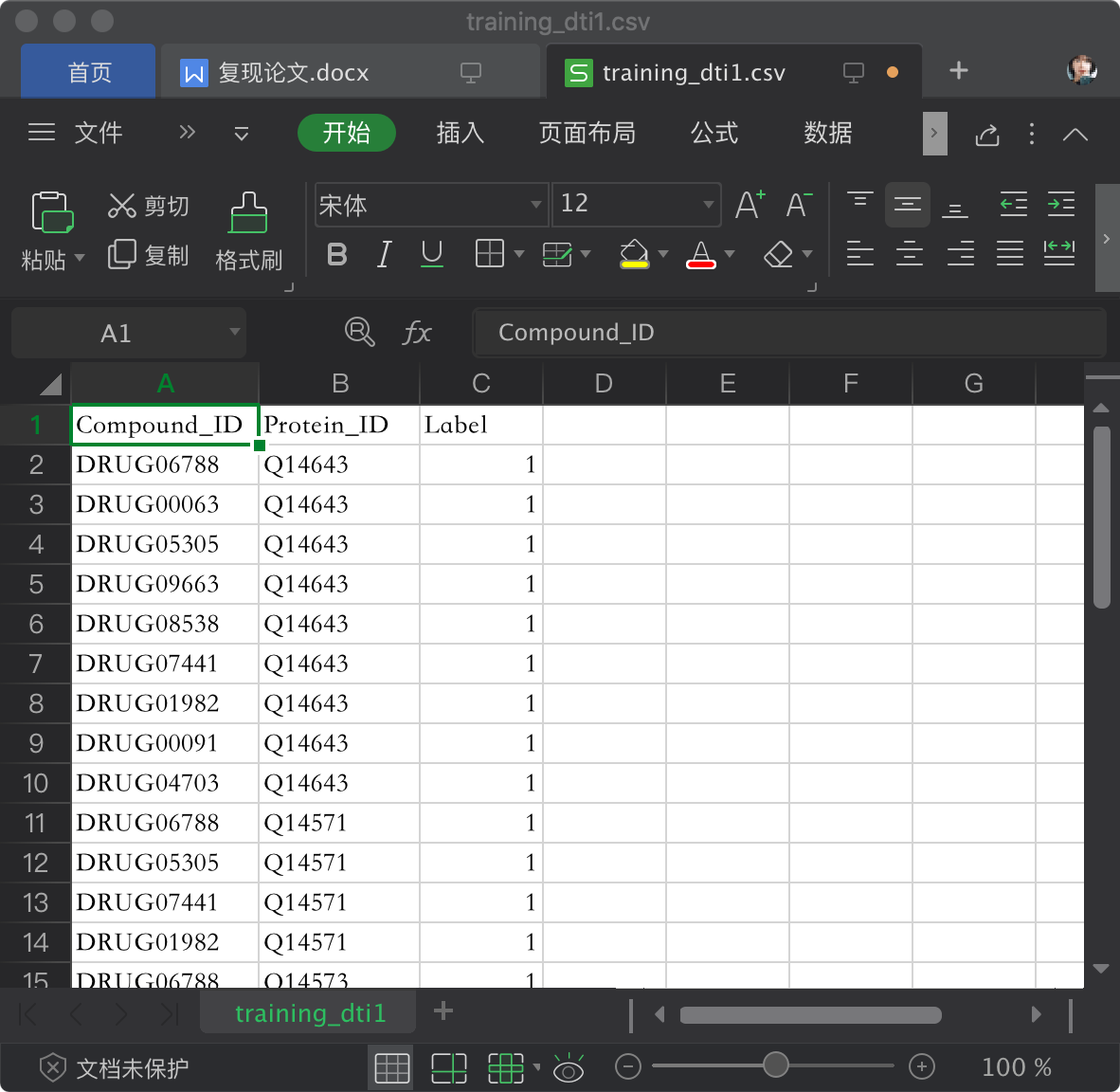
Sequence:蛋白质氨基酸序列。论文中氨基酸序列经过embedding后输入到网络中。

AAComposition:氨基酸组成描述符(Amino Acid Composition)。另一种蛋白质氨基酸组成描述符，作者为了比较模型与其他使用不同蛋白质描述符(descriptor)模型的性能差异，故引入了该属性。

CTD:组成(composition)、转变(transition)、分布(distribution)描述符，一种蛋白质特征描述符，作者为了比较模型与其他使用不同蛋白质描述符(descriptor)模型的性能差异，故引入了该属性。

Similarity: 蛋白质相似性描述符。作者为了比较模型与其他使用不同蛋白质描述符(descriptor)模型的性能差异，故引入了该属性。

1. **dti文件**



Compound\_ID：药物ID，用于连接compound和protein文件。

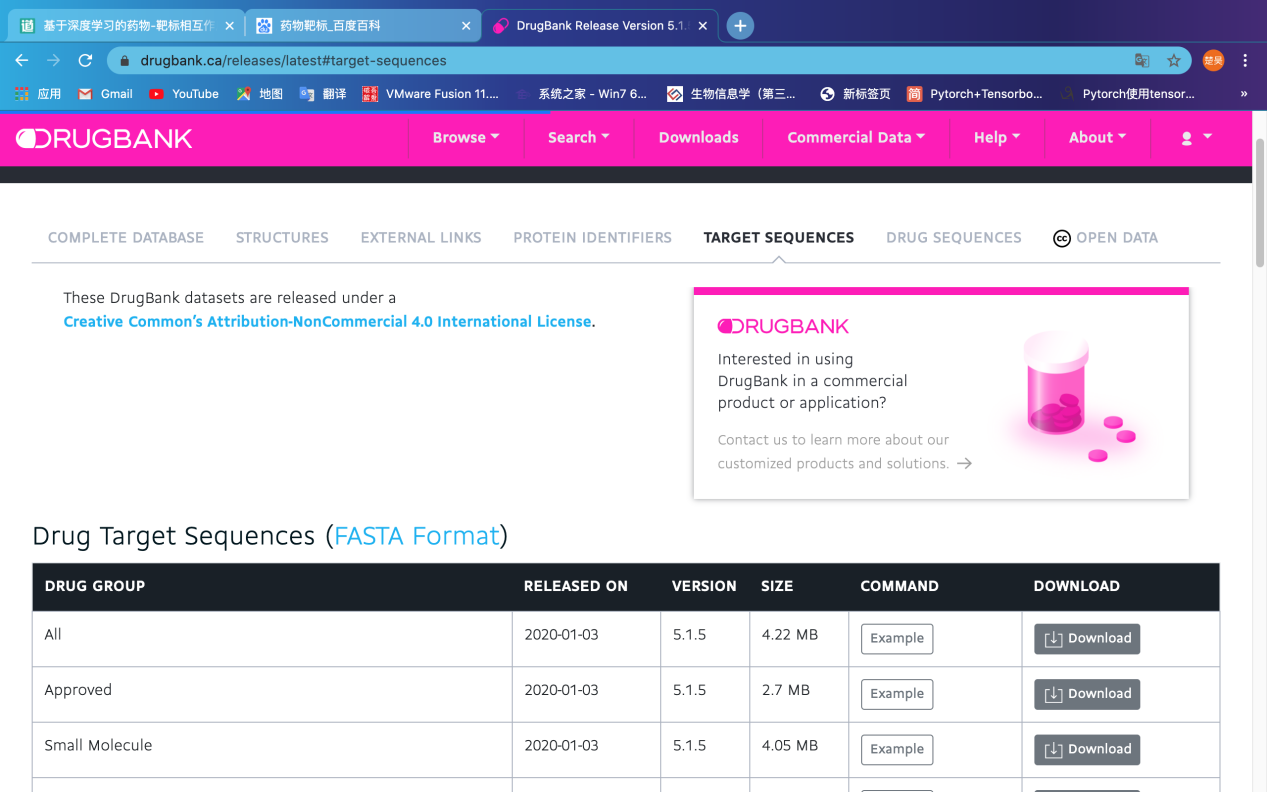
Protein\_ID：蛋白质ID,用于连接compound和protein文件。

Label：表示药物分子能否与靶标蛋白质结合。可以结合，label为1，为正例(positive);不能结合，label为0，为负例(negative)。

1. 数据下载——drugbank

由于时间关系，我们没有能够解析所有数据库中的数据。本次复现中采用的所有数据均来自drugbank。

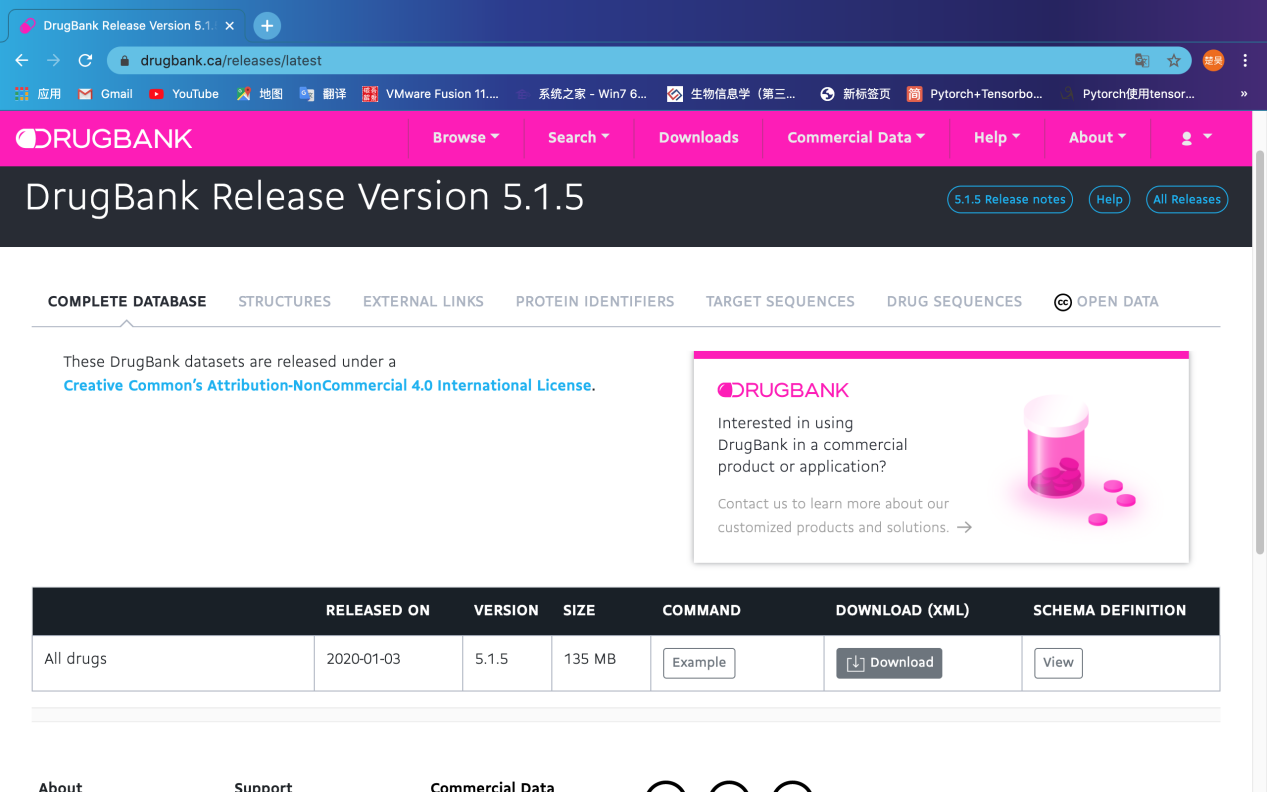
1. 从drugbank中下载target\_sequence，选择approved下载，该文件表示所有已经被美国FDA认证过的药物。



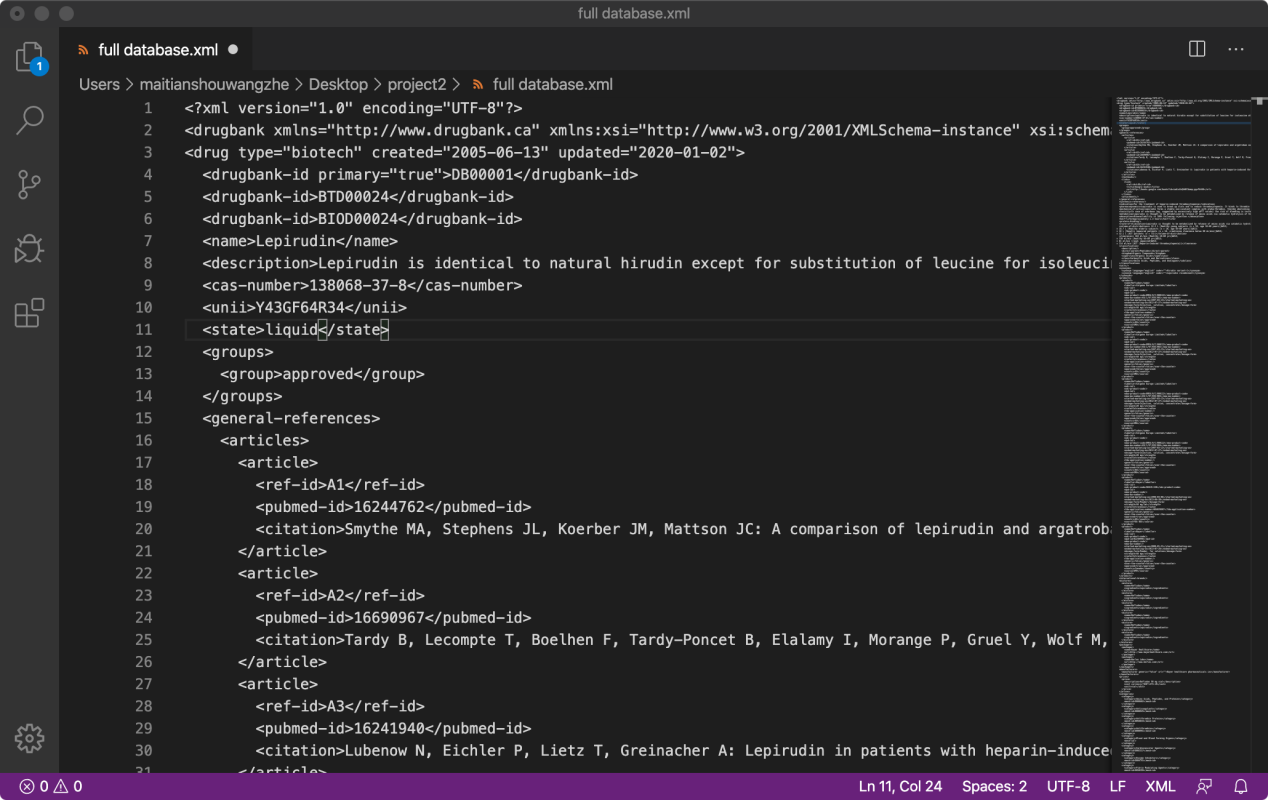
1. 下载得到的靶标蛋白与药物关系文件如图所示。文件中包含靶标蛋白与药物关系、靶标蛋白氨基酸序列信息。

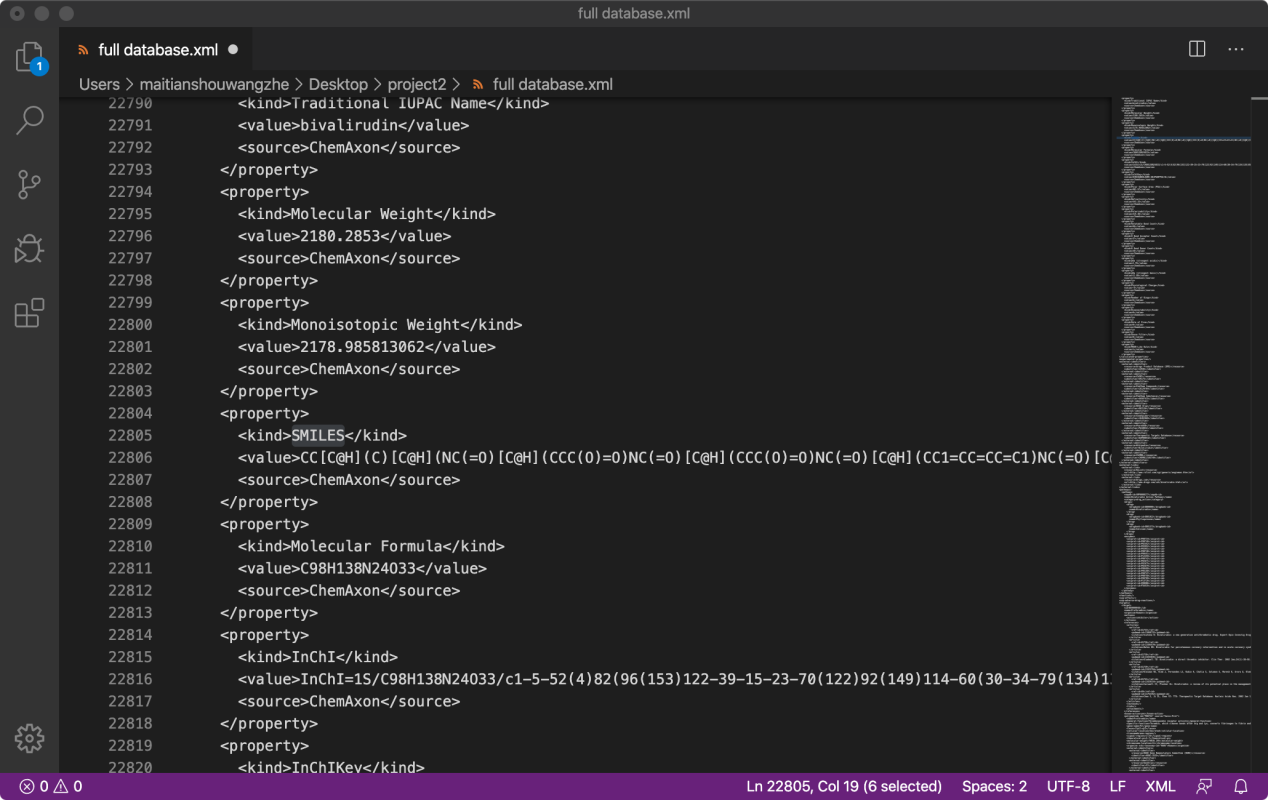


1. 从drugbank中下载complete database



1. 下载得到的drugbank完整数据库文件如图所示。文件中包含大量信息，我们只需要(1)文件中含宥的药物的drug\_ID和SMILES表示两个信息。

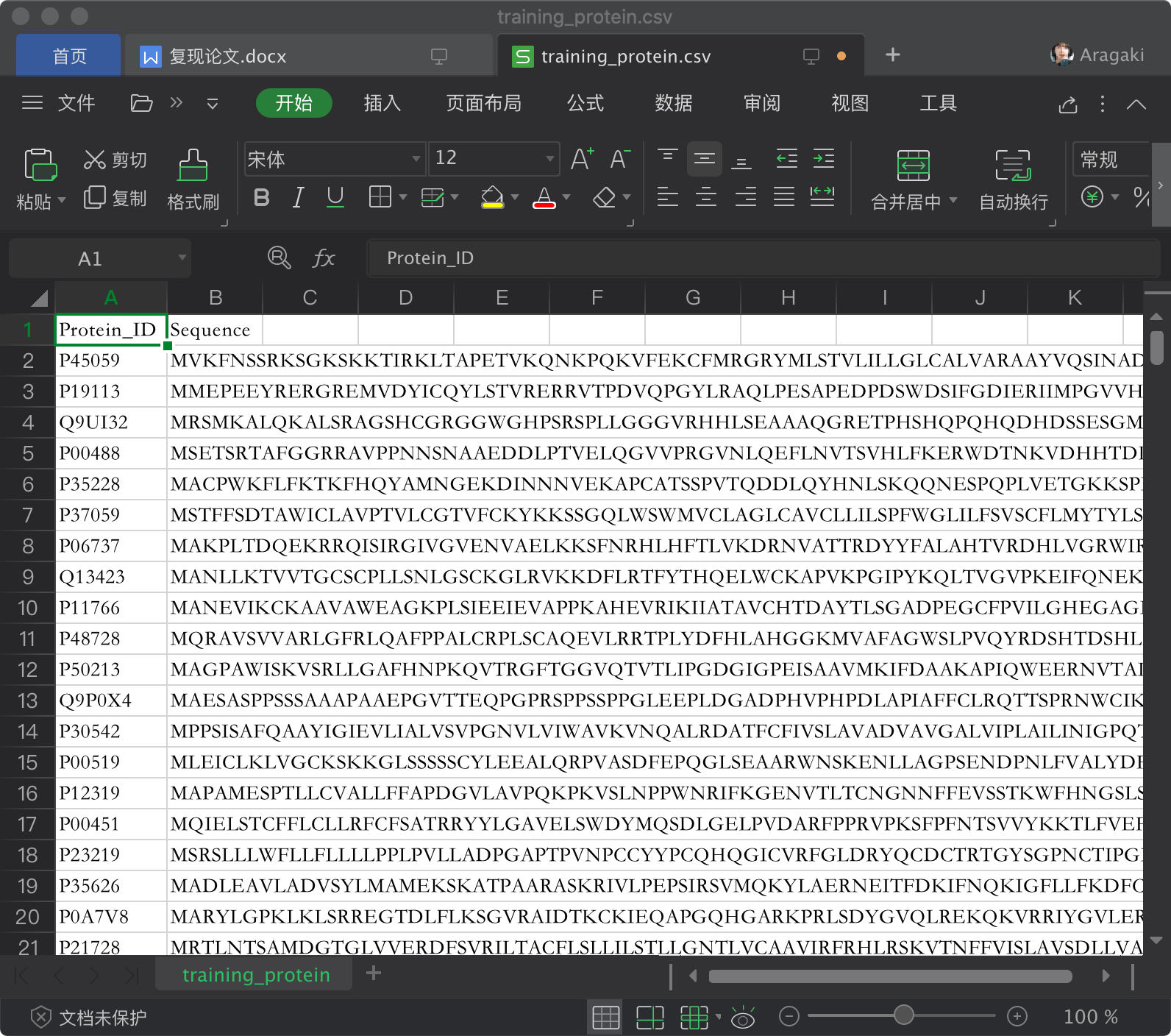




1. 我们的数据集——our\_dataset
2. **构造protein文件。**

由于作者提供的样例文件中包含大量用于与其他做比较的模型，和一些冗余属性，所以我们对属性字段进行了删减。参考作者的网络蛋白质输入为氨基酸序列，最终，我们的protein文件包含了protein\_ID和Sequence两个字段。

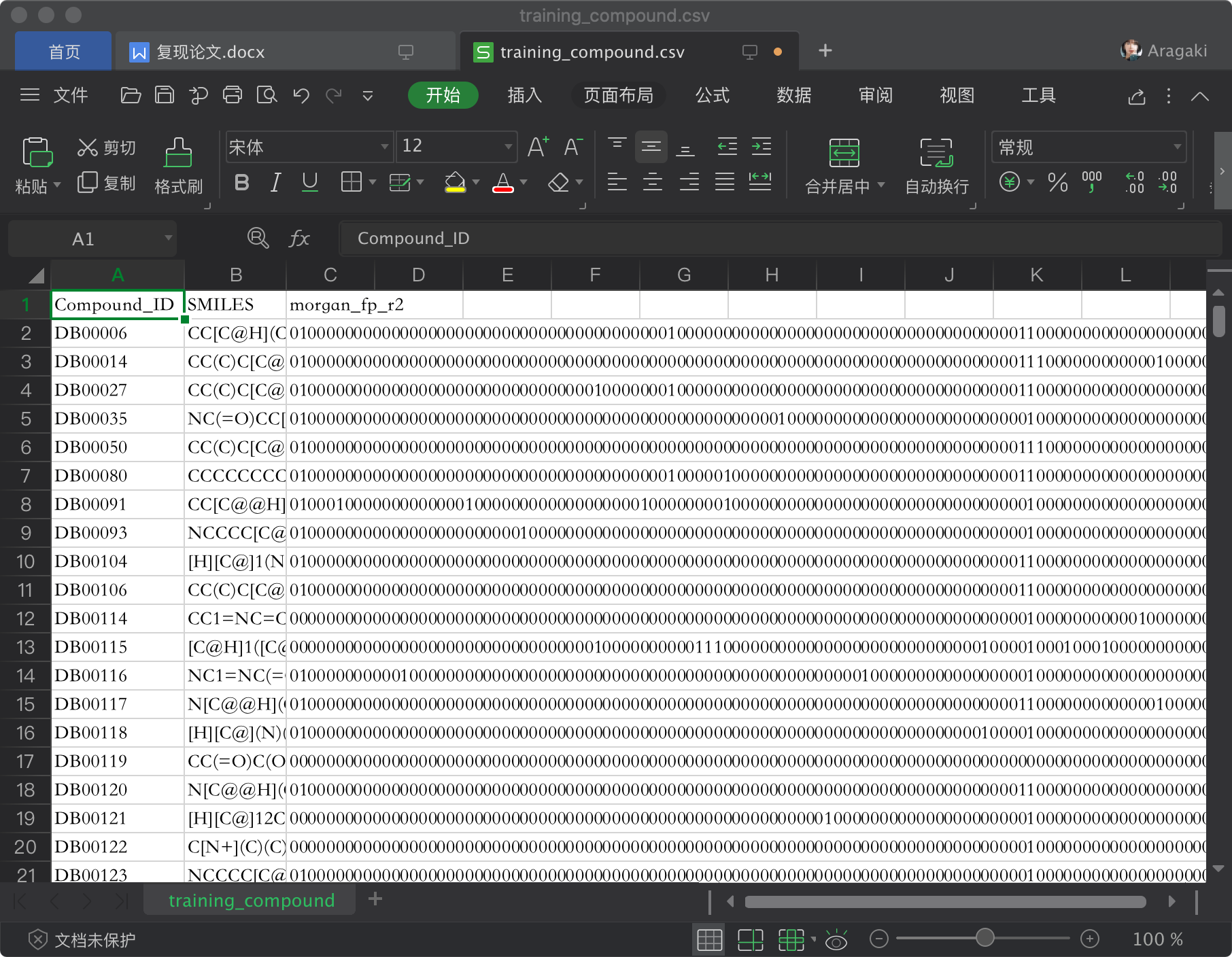
首先参考[https://github.com/zhanglu-cst/Drug-Target-Interaction。](https://github.com/zhanglu-cst/Drug-Target-Interaction/tree/master/data_process_to_fingerprint)中的代码，从target\_sequence中解析出<protein\_id, sequence>形式的json字符串，再利用python将json字符串转化为protein.csv文件。最终数据集如图所示，共有4769条数据。



1. **构造compound文件。**

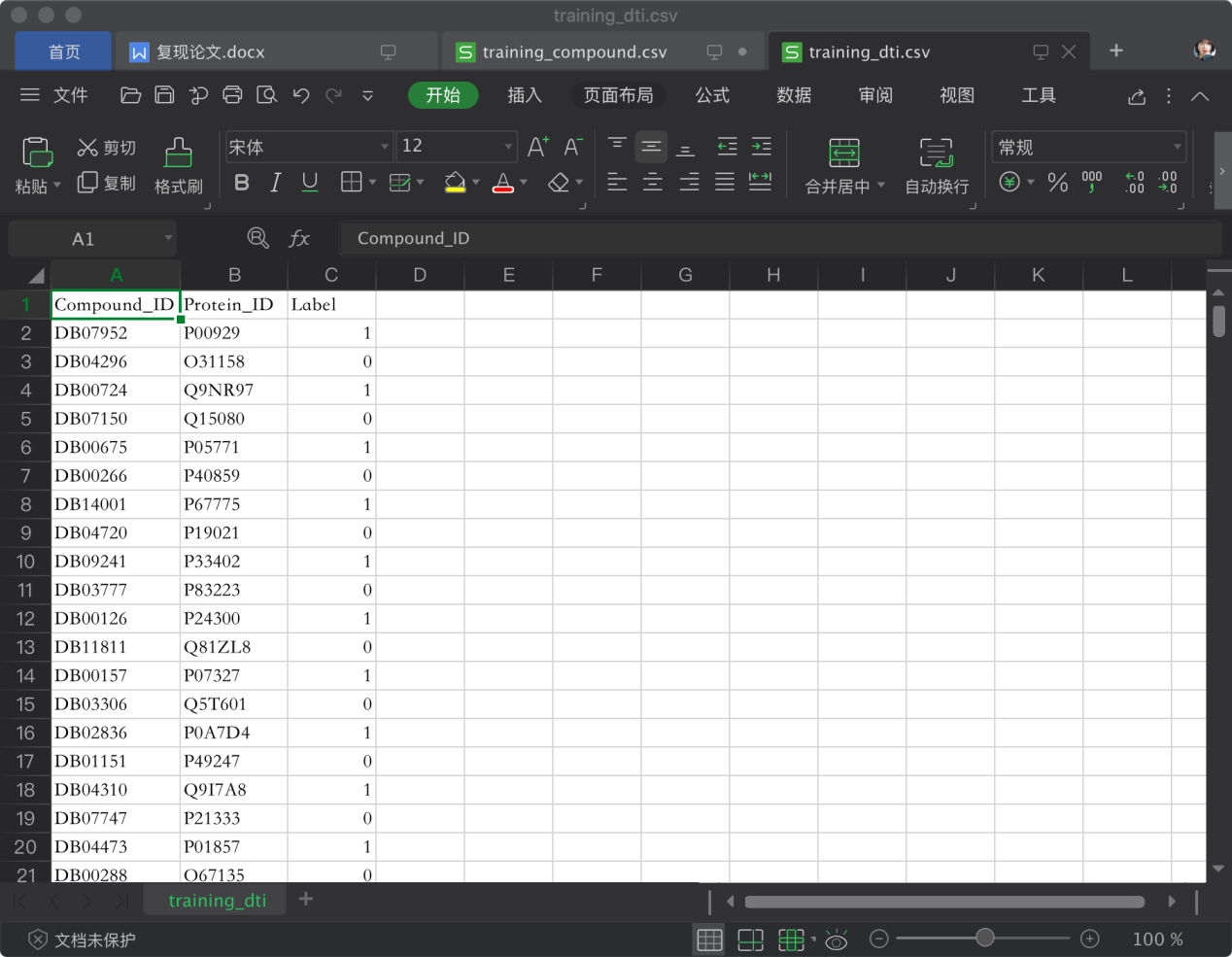
和protein文件一样，作者在compound中引入了大量属性来复现其他人的模型以作比较。在本文模型中，作者输入网络的是化合物描述符是半径为2(n=2)的摩根指纹。故我们的数据集只保留compound\_id、smiles、morgan\_r2三个属性。事实上，smiles仅作为中间过度，目的是计算摩根指纹。

数据源为drugbank的complete database，该文件为xml文件。首先从xml文件中解析出<drug\_id, smiles>形式的json字符串，再将json字符串转化为.csv文件。此时文件中仅有compound\_id、smiles两个属性。我们调用python的rdkit模块，利用其中的Allchem.GetMorganFingerprint()函数，将化合物的smiles转化为摩根指纹。最终得到的compound文件如下，共有9582条数据。



1. **构造dti文件。**

首先，从drugbank的target\_sequence中解析出<drug\_id, protein\_id>形式的药物-靶标数据，并全部标注为1。由于drugbank中target\_sequence文件均为正例，故需要通过drug与无法结合的protein进行随机组合构造反例。为了保证反例构造的合理与平均，我们直接采用https://github.com/zhanglu-cst/Drug-Target-Interaction中构造的反例，并保留drugbank中的所有正例。构造后的dti文件如下，我们按照6:2:2的比例分为训练集、验证集和测试集。



1. 程序的描述

1、总体组成

这个DTI预测模型程序主要由以下三部分组成：

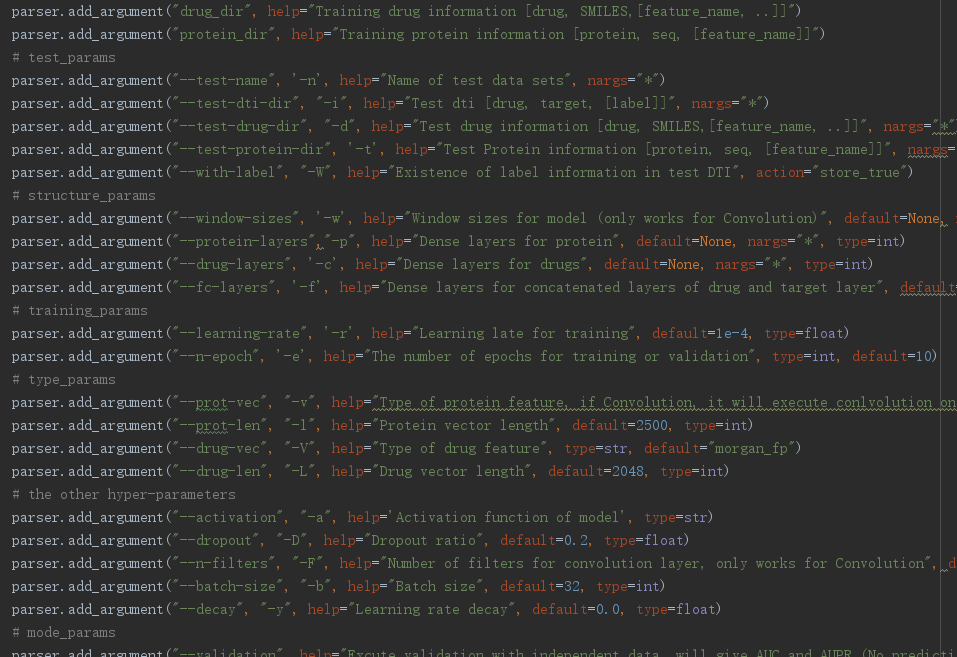
(1) 数据输入与处理，包括蛋白质和药物各自特征序列的处理；

(2) 深度卷积网络搭建，输入为两者特征，输出为DTI为0或1；

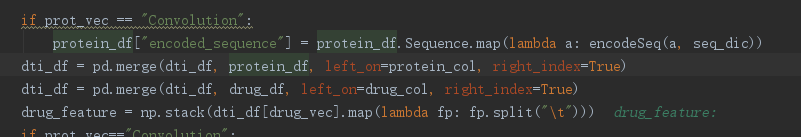
(3) 网络训练与验证。

2、数据输入与处理阐释

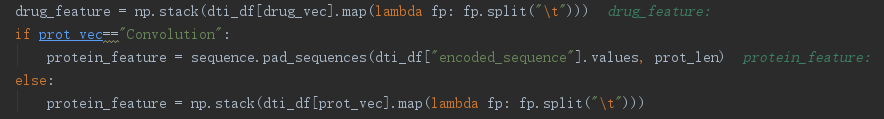
首先，程序使用命令行输入训练样本、测试样本的路径，各种超参数（学习率，dropout率，蛋白质最大长度，药物morgan fingerprint长度等



其次，程序从compound，protein，dti三个表中读取出关键数据列，由于dti表的两个外键分别是compound和protein表中主键（药物ID和蛋白质ID），需要将三个表做join操作，拿到蛋白质特征、药物特征、dti标签，作为神经网络的三个输入输出参数来源。



拿到数据来源之后，对数据来源做进一步处理：将药物的morgan fingerprint r2打包成数据张量，将蛋白分子的字母序列转化为数字序列，并在前面填上padding的0值使得每个蛋白质特征序列长度相同。



1. 神经网络构建阐释

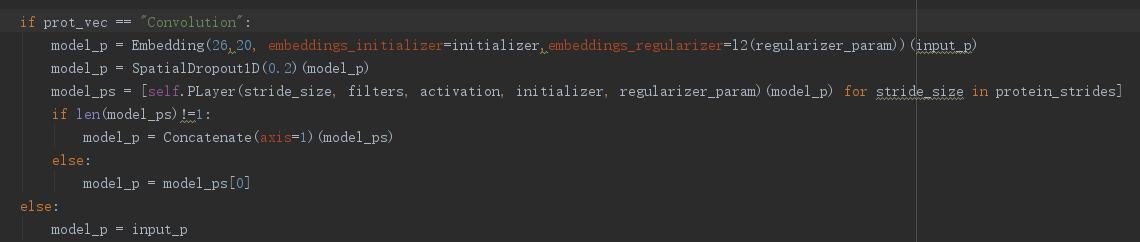
keras构建神经网络输入：输入为蛋白序列向量、药物fp特征向量。

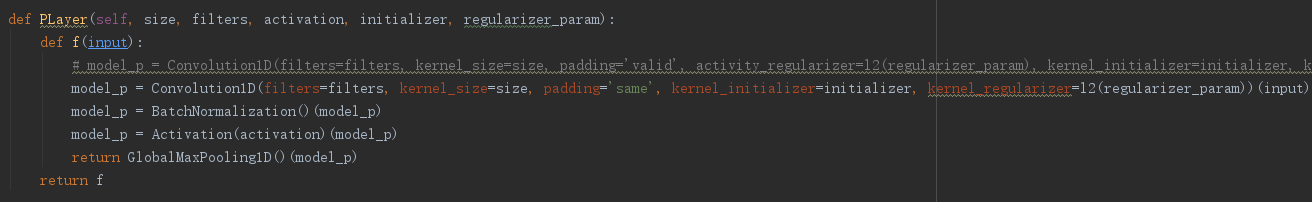


药物隐藏特征提取部分的网络被分为两层，每一层为全连接，使用BN归一化。

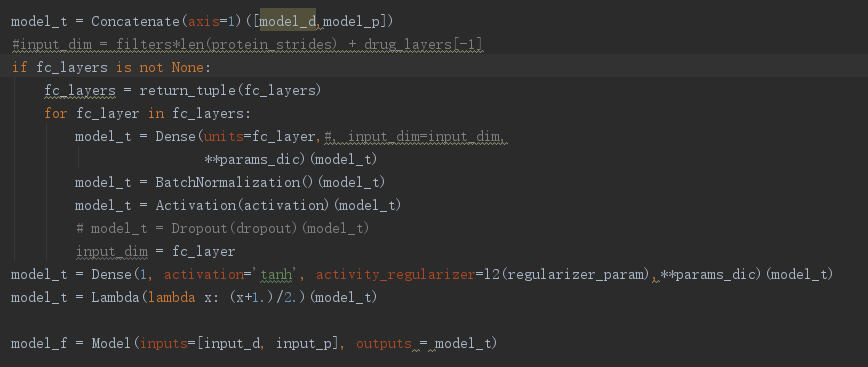


蛋白隐藏特征：输入使用词向量embedding后，并行放入多层一维卷积网络中，使用不同的stride值，最后将这些网络输出拼接。



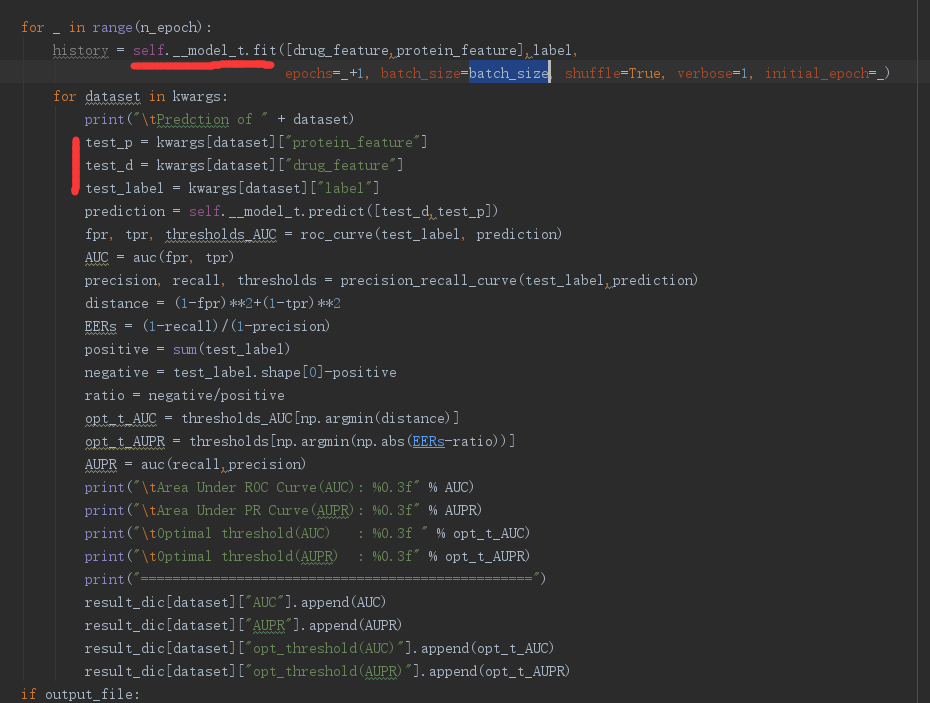


将获得的蛋白质特征和药物特征拼接到一起，做几个全连接层得到单值输出，输出使用tanh激活函数并把值域调整到[0, 1]，该值即神经网络预测的对应dti的label。



1. 神经网络训练方法

该模型使用keras进行训练，每个batch为32条数据。使用test集合中的数据做模型检验，并打出准确率趋势图等。也可以选择使用验证集，每个epoch做一次验证，或者使用训练集做K折交叉验证。



四、运算的结果及展示

在展示运算结果之前我们需要做几点说明：

(1) 原论文中采用的结合了多个生物数据库(eg:drugbank,kegg,pubchem等)，由于时间关系，我们所有的数据均来自drugbank，由于数据没有原论文采用的多，故复现性能会略微下降。

(2) 原论文有相当一部分工作在于复现前人的工作，并将自己的模型与前人的模型做性能对比，在本项目中我们对作者提出的药物-靶标预测深度卷积网络模型进行了复现，对前人的工作没有复现。因此，我们的数据集相对于原论文作者的数据集做了一定的删减。

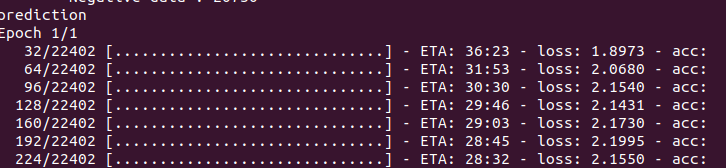
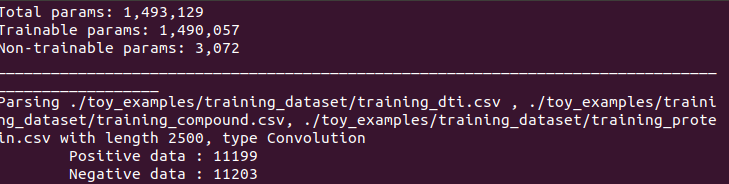
(3) 本次复现的环境为Ubuntu18.04、python3.6、tensorflow1.15.0、keras2.3.1。硬件配置为：CPU：Intel Core i5 9600k、内存：16GB。一轮epoch大约用时35分钟，同样由于硬件限制和时间关系，仅训练模型40个epoch，这也可能带来一定的性能损失。

(4) 本次复现中训练集共有9582种化合物、4769种靶标蛋白，11199个正样本，11203个负样本。验证集有7001个样本用于优化超参数。测试集有4647个正样本和3352个负样本，其中靶标蛋白和药物分子均取自训练集，但药物-靶标关系为全新关系。

下面开始展示结果。

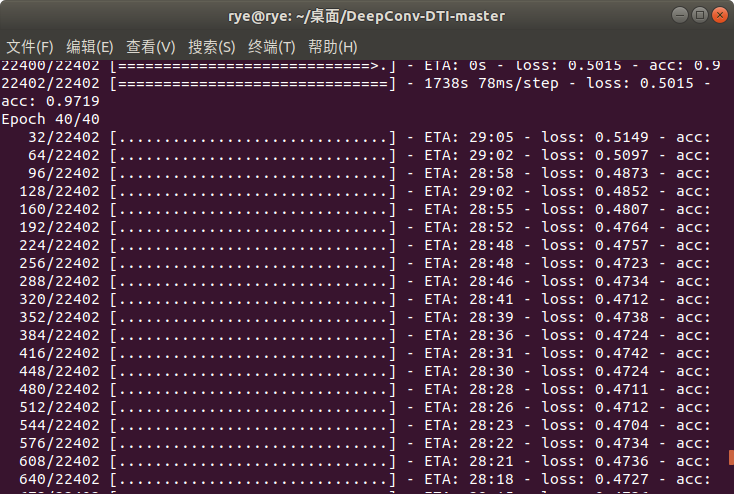
1. 训练过程
2. **开始训练**

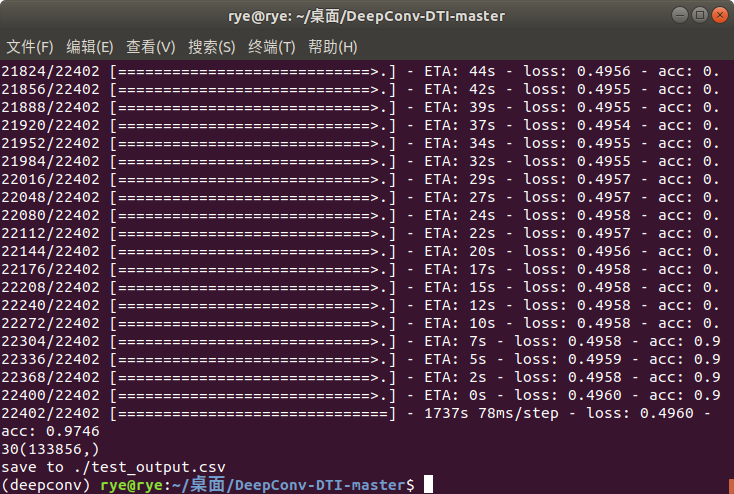
开始训练，可以看到网络训练参数个数为1490057，每个epoch预计完成时间为36:23



1. **结束训练**

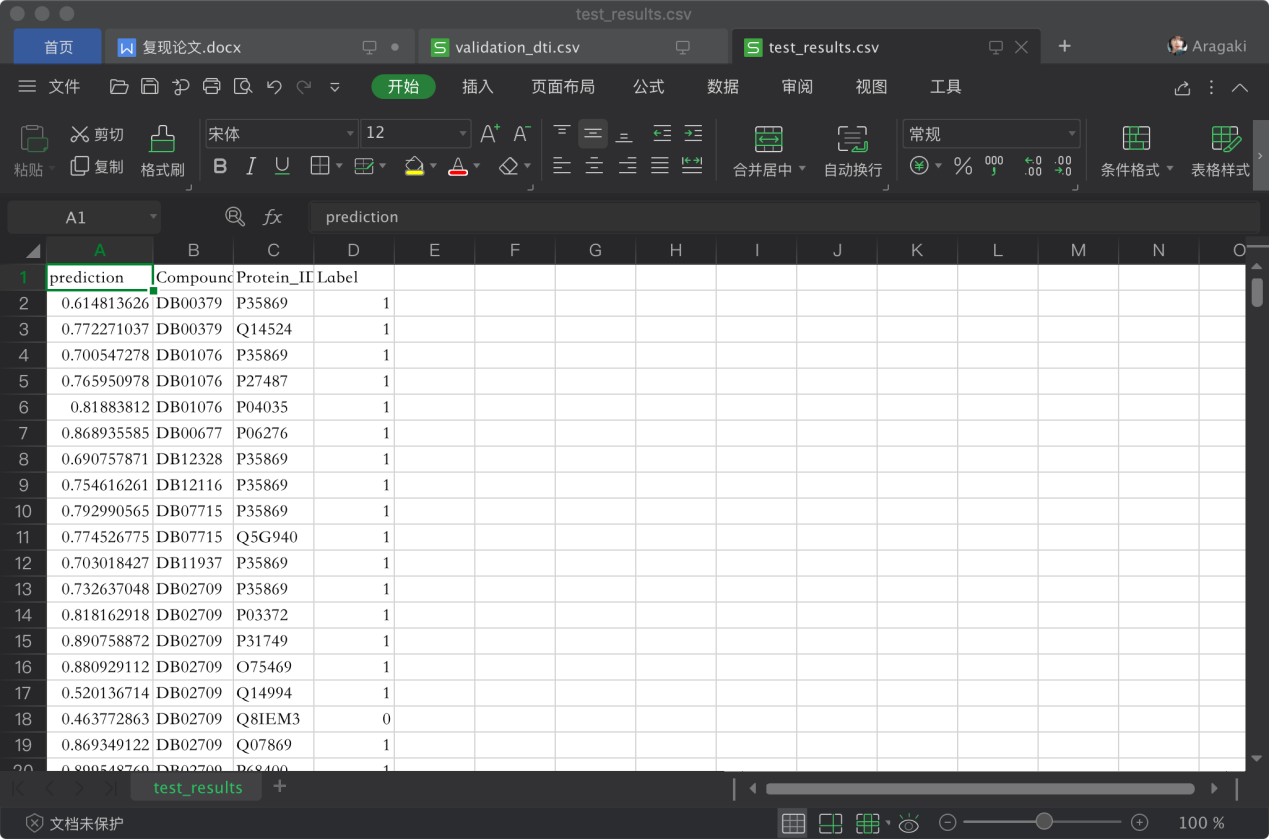
如图，第一张为最后一个epoch(epoch 40)开始的情况，此时loss为0.5015，acc为0.9719。第二张为训练完成时的情况，loss为0.4960，acc为0.9746。如果继续训练，应该还可以继续优化loss和模型。



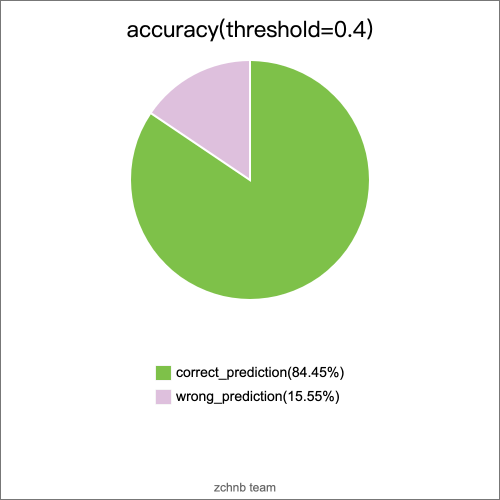


2、预测结果

预测得到的test\_results如图所示，第一列为模型预测为正例的得分(score)，后三列为相应的<compound, protein, label>药物靶标关系，最后一列为真实label。

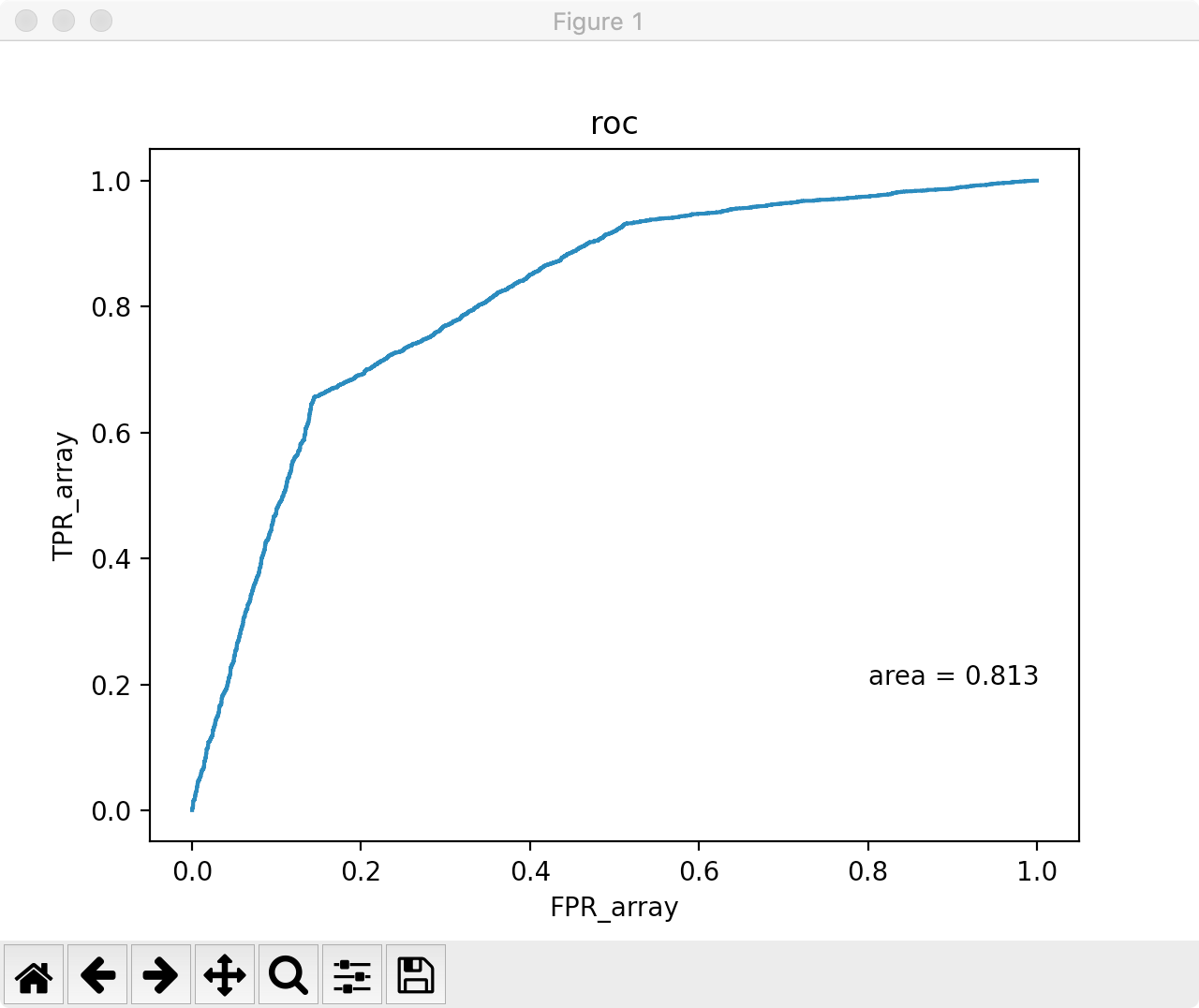
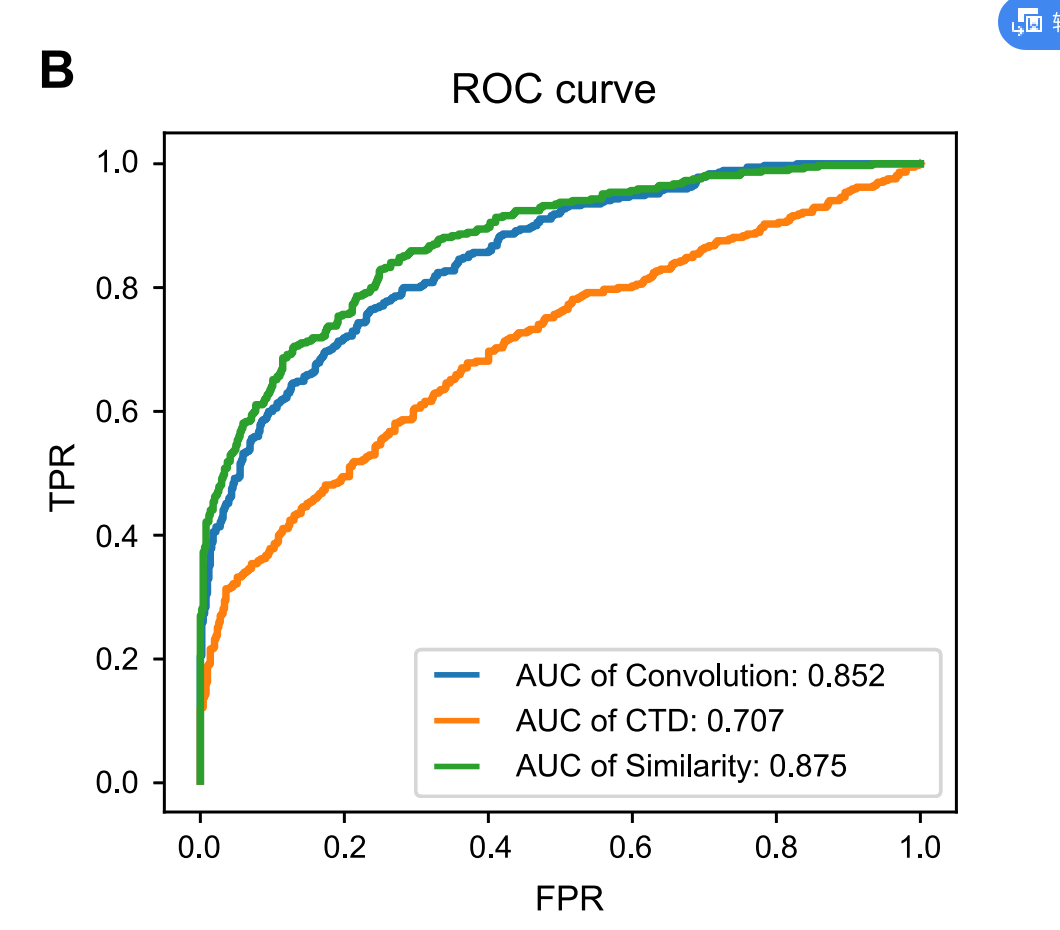


我们对预测的准确度(accuracy)做了可视化，取阈值(threshold)为0.4，即预测得分大于0.4的判断为正例，预测结果小于0.4的判定为负例。在8000条测试集，共预测正确6756例，acc为84.45%，低于训练集的acc。图示如下：

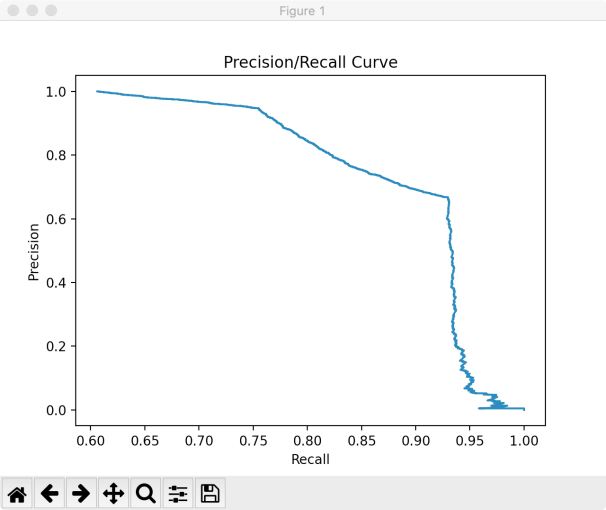
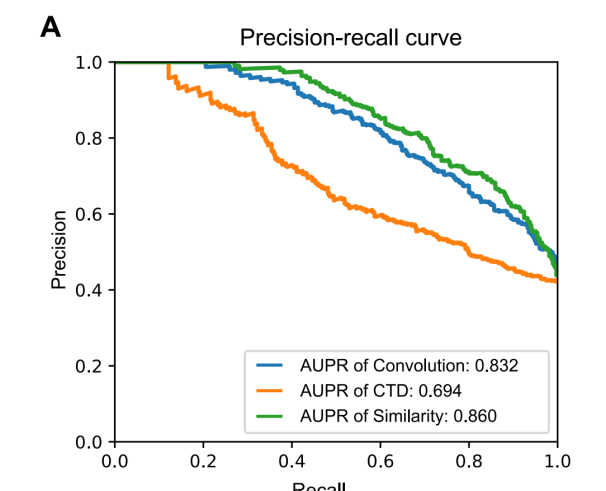


3、和原论文的预测结果比较

原论文对模型预测结果的precision-recall曲线和roc曲线做了可视化，我们也对复现结果的两条曲线进行可视化并与原论文比较，结果如下：



上图中左侧绿色线条为作者论文中模型的roc曲线，右侧为我们复现的结果。我们没有对曲线进行平滑操作。可以看到尽管测试集数目更少且来自统一数据库，我们roc围成的面积也不如原论文的好。这说明我们复现的模型在查准率和误判率两个指标上没有完全复现原论文的结果。



Precision-recall曲线的表现有些不尽如人意。左侧绿色线条为作者论文的结果，右侧为我们复现的结果。曲线不如原论文平滑，且当召回率大于0.95时，性能不如原论文。

4、讨论

本次复现论文时间较为仓促，故没有能百分之百复现出原论文的效果。对于上述结果中展示的测试集上acc的丢失、roc和pr曲线与原论文的性能差异，主要可以归结为以下几点原因：

1. 数据集的选择没有完全按照原论文的叙述的方式选取。作者在训练阶段融合了drugbank，kegg和IUPHAR三个数据集上的数据，训练集的大小大约在我们的两倍左右，这可能造成性能的损失。
2. 由于硬件的限制，导致训练时间过长，仅仅训练40个epoch就需要连续运行24小时左右，这导致了没有足够的资源来进行更充分和完备的训练，也可能造成性能的损失。

总体来说，从阅读文献到寻找数据，从配置环境到运行网络，再到最后的运行结果分析，还是让我们体会到了从论文到实实在在的模型的转变过程，也为我们今后的科研之路打下了基础。

1. 参考文献

在论文复现过程中，感谢下列博主/githuber带给我们的帮助，当然还有免费为我们提供数据的drugbank。

1、<https://www.jianshu.com/p/33212dbdface>

2、<https://github.com/Jhird/Deep-Learning-For-Drug-Protein-Interactions>

3、<https://github.com/hanzheng0730/Prediction-of-Drug-Protein-Interaction>

4、[https://github.com/MiraYin/DSPS-CNN/](https://github.com/MiraYin/DSPS-CNN/tree/master/src)

5、<https://www.zealseeker.com/archives/ecfp-introduction/>

6、<https://github.com/zhanglu-cst/Drug-Target-Interaction/>

7、<https://www.drugbank.ca/>