MIT新冠病毒药物准确性预测问题报告

摘要

我们小组在本任务下尝试了所有的药物分子表示学习的方法。不仅取得高达88%的auc-roc结果，还广泛尝试了所有论文中见到的药物分子处理方法，同时对数据不平衡问题做出尝试。

首先对三类手工提取的化学特征进行多种经典分类方法，其中在分子描述符和mol2vec的特征过随机森林的方法中取得0.85的roc和0.44的prc；

其次对smiles字符串进行了四种循环网络结构的尝试，最高得到得到0.64的roc；

最后图网络尝试不同网络结构，mpnn和gcn分别得到0.88和0.87的roc；

另外还对数据不平衡问题尝试了多种采样方式以希望解决数据不平衡的问题。

经典方法

基于分子描述符的分类方法

分子描述符是化学计量学的重要工具，是指分子在某一方面性质的度量，既可以是分子的物理化学性质，也可以是根据分子结构通过各种算法推导出来的数值指标。它包括基于分子图论、各种理论或实验光谱数据（如紫外光谱）、分子组成（如氢键供体数、化学键数）、理化性质（如酯水分布系数）描述符、分子场描述符以及分子形状描述符等。

分子描述符可以说是本问题最为重要及有效的特征，相比于复杂的网络结构，我们发现使用更好的分子描述符特征可以对结果带来更大的增长。

来自化学信息学的专用软件Rdkit提供了方便的处理smiles字符串并提取分子描述符的函数，使用起来十分方便，可以提取smiles字符串299个分子描述符特征特征。

```
def get_fps(mol):
    calc=MoleculeDescriptors.MolecularDescriptorCalculator([x[0] for x in
Descriptors._descList])
    ds = np.asarray(calc.CalcDescriptors(mol))
    arr=Fingerprinter.FingerprintMol(mol)[0]
    return np.append(arr,ds)
```

使用提取的分子描述符提取特征后，使用随机森林和多层感知机分类器进行大肠杆菌的抗药性任务十折验证。其中随机森林属于集成学习 中的 Bagging方法，通过多个决策树中多数结果确定最终结果，这种集成方式可以使得分类结果更优秀。而多层感知机是一种前向结构的人工神经网络，经过超参调参，使用relu作为激活层，adam优化方式，两层隐藏层的网络结构，隐藏层的单元数定义为100和10。尽管在这个任务中，随机森林的效果和多层感知机相差无几，但是随机森林的训练时间远小于多层感知机。

	随机森林	**多层感知机**
roc	81%	79%
prc	45%	44%

基于分子指纹的分类方法

化学指纹识别是一种将绘制的分子转换为0和1位的流的方法。比较常见的有摩根指纹，是一种圆形指纹，对所有可能的距离目标分子中的给定原子半径以内的所有可能原子和其连接形式进行编码，得到的0，1数字流。

目前在化学信息学中比较流行使用的有ECFP4指纹，ECFP6指纹，RDKFP指纹。其中ECFP4指纹，既扩展连通性指纹，是最为流行使用的，其生成方法是依赖于摩根算法，在rdkit已经被集成。通过getmorganfingerprint模块可以方便地生成。

```
  data['fingerprint']=data['mol'].apply(lambda m:
AllChem.GetMorganFingerprintAsBitVect(m, radius=2, nBits=2048))
```

在此为了探究各个类型指纹在本问题中的表示能力，我将对比ECFP6，ECFP4和RDKFP三个指纹在随即森林和多层感知机下的表现能力。其中ECFP4的探查半径定义为2，ECFP6的探寻半径定义为3，三种指纹的位数都限定为2048。其在大肠杆菌的分类任务中十折平均效果如下表。

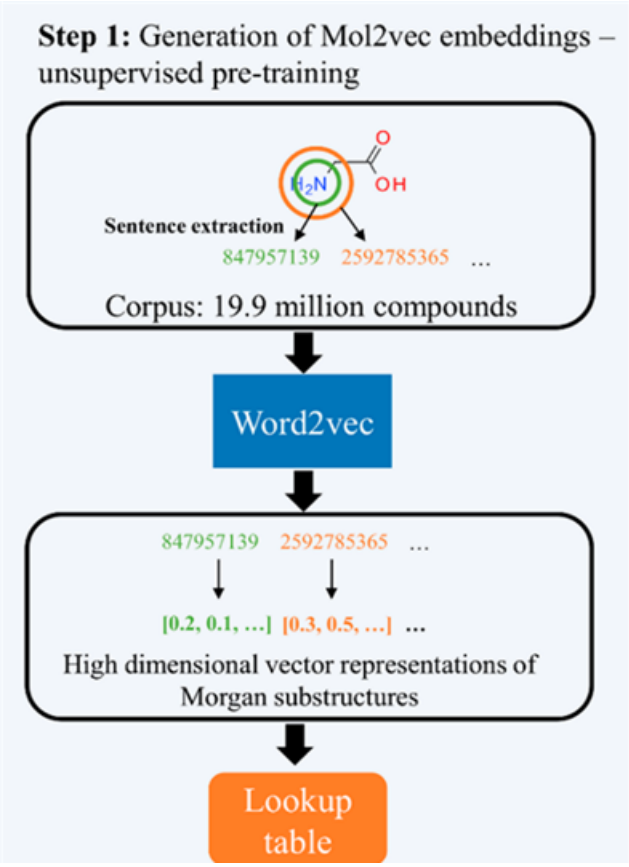
roc	随机森林	多层感知机
ECFP4	77.5%	69.4%
ECFP6	76.8%	71.1%
RDKFP	77.4%	72.1%

prc	随机森林	多层感知机
ECFP4	41.6%	33.0%
ECFP6	39.5%	31.5%
RDKFP	39.7%	39.8%

在这个任务的表现中可以看到，ECFP4在roc和prc的表现上都是最为优秀的。而随机森林和多层感知机相比，随机森林会更好一些。但是我们可以从分子指纹的长度中看到问题，那就是分子指纹共有2048位，其维度太高，所以可以再进行降维处理。但是需要注意的是在mit的公开数据排行版上的第三名就是采用的我这里的随机森林和多个指纹融合的方法，所以可能这种化学学科专用的指纹特征反而比用网络提取出来的特征更为有效。

基于mol2vec的分类方法

在上一节中我们说到分子指纹的纬度过高问题，类比自然语言处理中的词与句子的关系，在分子表示中我们将分子的指纹认为成词汇，将分子认为成句子，即可将自然语言处理中的word2vec方法引用到化学信息学中。



在Mol2vec: Unsupervised Machine Learning Approach with Chemical Intuition这篇文章中，作者通过运用word2vec算法在分子上，就可以将高维的分子指纹降维到300维的词嵌入向量中。该模型在通过CBOW或Skip-gram在数量非常庞大的ZINC分子数据库进行训练，最终获得专门运用在分子表示上的embedding方法—mol2vec。

我们在这里运用mol2vec方法，将smiles式中每个分子视为句子，每个分子指纹（指纹可以看作分子的亚结构）视为单词，经过mol2vec后，我们得到了每个单词的词向量表示，而我们认为每个分子都是其子结构词向量的和，因此每个分子的所有子结构的向量表示既分子的表示。

每个分子被表示成300维的向量，我们再通过上几节用过的随机森林和多层感知机对大肠杆菌的任务进行处理，得到十折上的平均roc和prc。

	随机森林	**多层感知机**
roc	82.3%	71.4%
prc	45.6%	37.2%

在这里可以看到相比上一节的指纹表示，我们这里特征维度更低的mol2vec的roc结果反而会更好，可以说mol2vec保留了指纹的主要信息。

分子描述符、mol2vec、分子指纹特征混合

为了使特征工程的表示尽可能涵盖更多的信息，我尝试了将上述的三种特征进行了拼接，有效的提高了roc的结果。

我们可以看到分子描述符主要表示的是分子作为整体的特征，既分子作为一个整体呈现的整体性质，并不涉及原子级别的特征。而mol2vec和分子指纹都是通过原子和其邻居原子的连接结构形式进行特征提取的，可以说是和分子描述符不相重合的表示特征。而mol2vec和我选用的ECFP4其实在特征表示上也并不重叠，这是由于mol2vec是基于半径为1产生的指纹，可以说和半径为4的ECFP4并不重叠。因此将

这三个特征进行拼接既可以提高结果。

通过表格里的结果我们得到了我们小组做出的最优秀结果，同时方法却是非常简单的一种。和我们之后的图网络和循环网络来说，我们的优秀结果得益于化学专家为我们提供的非常可靠的表示特征。由于特征工程的工作做得充分，在简单的随机森林就可以得到优秀结果。

如果特征还不够充分的地方就在于我们最终的特征止步于分子的亚结构上（分子指纹），而对于单个原子还有原子化学键的信息是没有涉及到的，因此我们在之后的图网络中格外对原子和化学键特征进行了补充，希望能做出更好的结果。

	随机森林
roc	85%
prc	44%

循环神经网络

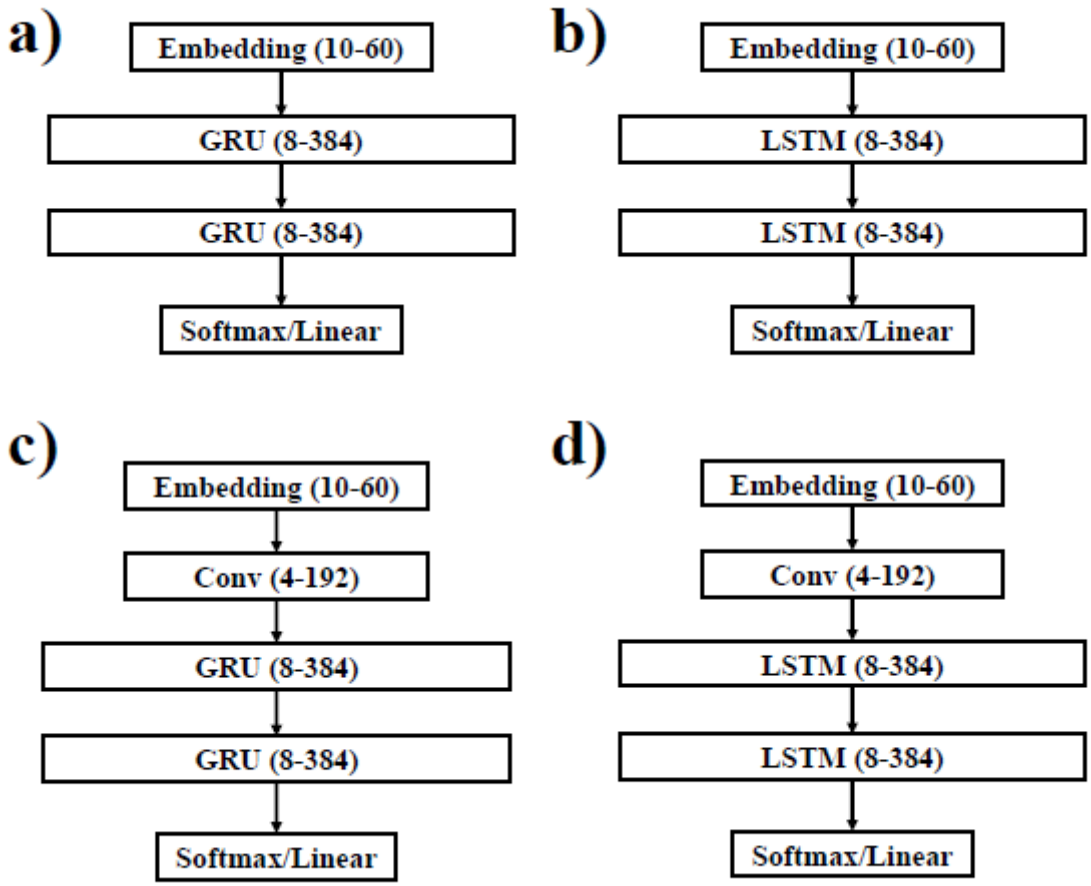
循环神经网络

参考SMILES2Vec: An Interpretable General-Purpose Deep Neural Network for Predicting Chemical Properties 文章中的做法，认为将字符串smiles转换为向量的方法可以视为预测其物理化学性质的方法。而处理字符串的方法，我们可以尝试循环神经网络的方法。

其实在图网络运用于化学分子问题前，cv和nlp的方法都被运用在化学分子的性质预测上，其中cv的方法被用于处理分子图，nlp的方法被用于处理smiles串。我们可以看到这里的神经网络的优点相比于经典方法，是不再需要专家的经验，而完全依靠网络就可以出结果。

但是这里的主要问题在于，是否smiles字符串就可以表示出分子的全部特性呢？我们知道对于一个结构复杂的有环分子，从不同的位置断链都可以形成不同的smiles表达式，但是实际上这些分子是一个物质，但是smiles表达式就已经发生了巨大的变换。这个问题导致一个分子会有多重smiles串表达形式。同时我们数据中的分子数据偏少，其实训练rnn并不充足。

在尝试使用循环网络的方法上，我尝试了四种网络结构。其整体结果是不如经典方法那么出色的。整体的网络框架按照seq2seq框架走，调整使用不用rnn层，也同时尝试加入一维的卷积层。



a类网络使用普通的bedding加两个双向gru层的结构。

```
model.add(Embedding(num_words+1, 50, input_length=input_length))
model.add(Bidirectional(LSTM(128, return_sequences=True)))
model.add(Bidirectional(LSTM(128)))
model.add(Dense(128, activation='relu'))
model.add(Dropout(dropout_rate))
model.add(Dense(output_dim, activation='sigmoid'))
```

b类网络采用普通embedding加两个双向lstm层的结构。

```
 model.add(Embedding(num_words+1, 50, input_length=input_length))
 model.add(Bidirectional(GRU(128, return_sequences=True)))
 model.add(Bidirectional(GRU(128)))
 model.add(Dense(128, activation='relu'))
 model.add(Dropout(dropout_rate))
 model.add(Dense(output_dim, activation='sigmoid'))
```

c类网络采用在gru前面加上一层一维卷积层的结构。

```
model.add(Embedding(num_words+1, 50, input_length=input_length))
model.add(Conv1D(192,3,activation='relu'))
model.add(Bidirectional(GRU(224, return_sequences=True)))
model.add(Bidirectional(GRU(384)))
model.add(Dense(128, activation='relu'))
model.add(Dropout(dropout_rate))
model.add(Dense(output_dim, activation='sigmoid'))
```

d类网络在lstm层前加入一层一维卷积层的结构。代码类似不再赘述。

最后我另外参考有的论文采用的两层一维卷积层的方法，虽然不是循环网络，但是在这个问题上却比循环网络有效。

```
model.add(Embedding(num_words+1, 50, input_length=input_length))
model.add(Conv1D(192, 10, activation='relu'))
model.add(BatchNormalization())
model.add(Conv1D(192, 3, activation='relu'))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(dropout_rate))
model.add(Dense(output_dim, activation='sigmoid'))
```

下表中是循环网络的结果，可以从结果中看到其实aucroc的水平在0.6右浮动，即使虽好也只有0.64，同时prc较低，和我们在前面使用特征提取和经典方法中的结果不能相提并论。但是循环网络并不需要人手工提取特征，这是其优点所在，但是这种提取特征的能力相比于图网络相差较远。

	aucprc	**aucroc**
bigru+bigru	0.08	0.58
bilstm+bilstm	0.08	0.60
cnn+gru	0.11	0.62
1Dcnn+1Dcnn	0.23	0.65

不平衡问题优化

采样

基于数据不平衡的问题，我尝试通过不同的采样方式来优化这个问题。

常用的用来解决数据不平衡的采样方式有三种，分别是过采样，欠采样和混合采样。

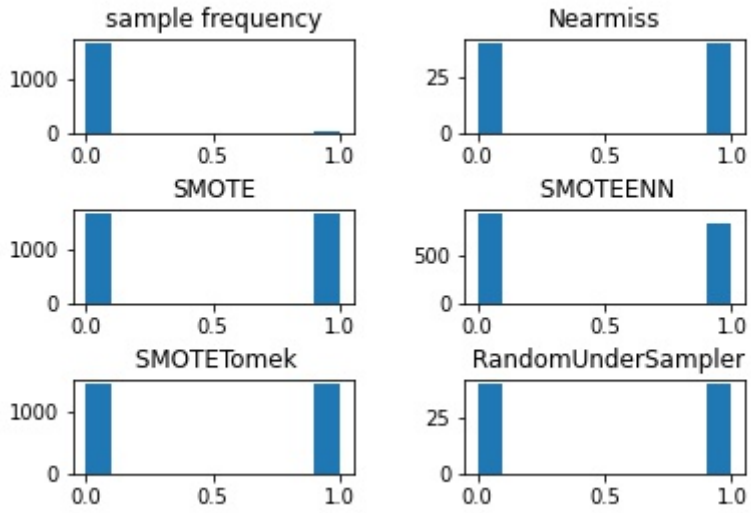
过采样是通过对少数样本进行额外采样的方法，最为常用的是smote方法。smote方法是通过少数样本之间进行插值来产生额外样本，具体说是通过对少数样本i进行k近邻法，求出离i距离最近的k个少数样本。

但是smote方法的两个问题就是:1.如果选取的少数样本周围也都是少数样本，则新合成的样本不会提供太多有用信息 2.如果选取的少数类样本周围都是多数样本，则这类样本可能是噪音，则新合成的样本与周围多数类样本大部分重叠，导致分类困难。

欠采样是通过对多数样本剔除，但是这种方法的缺点是可能包含一些重要信息，导致学习出来的模型效果不好。我尝试了多种欠采样的模型。

1. 随机欠采是随机对多数样本进行剔除，显而易见的缺点是会丢失信息。
2. Nearmiss是一种从多数类样本中选取最具有代表性进行训练的方法，采取一些规则来对样本进行类型划分。

另一种方式通过**过采样和欠采样**的结合。通过先过采样在欠采样，从而处理掉重叠样本。通过图片中的显示可以看到经过smote过采样之后的样本少数样本增多，而仍有重叠现象，然后经过混合采样后，无论使用smote+enn还是smote+tomek都使得多数样本和少数样本的重叠减小。下图是通过不同方式后的正负样本的分布，可以看到样本的分布经过采样后发生了均衡，但是欠采样的样本数剩余过少，而过采样又补充了非常多的少数样本，而混合采样则数目相对均衡，但是仍样本数量仍减少许多。



我desciptor和mol2vec和特征混合过随机森林的基础上使用**欠采样**，原本结果为82%。但是得到的aucroc结果普遍下滑，这是因为欠采样对重要信息丢失的原因。另外auc-prc下降的非常严重。

欠采样	**auc-prc**	**auc-roc**
未处理	40%	82%
ClusterCentroids	4.1%	52.8%
RandomUnderSampler	32.0%	76.0%
NearMiss	9.1%	61.4%
InstanceHardnessThreshold	9.6%	76.4%
CondensedNearestNeighbour	40.0%	79.6%

我对desciptor和mol2vec的随机森林结果进行过采样。使用smote方法，结果仍然是比未处理的结果差。

过采样	**auc-prc**	**auc-roc**
未处理	40%	82%
smote	35%	75.6%

我对desciptor和mol2vec的结果进行了**混合采样**。结果仍然是下降。

混合采样	**auc-prc**	**auc-roc**
未处理	40%	82%
smote-enn	36.4%	72.7%

经过我对多种采样方式的尝试，结果并没有变好，而是普遍下降。但这个结果是和之后的分类方法相关，在图网络中采取的采样方法就会使得结果变好。在经典方法中可以理解为，欠采样丢失了太多信息，而过采样又增加了太多噪声，混合才养仍丢失了部分信息，所以三者的采样效果均变差。

因为以上的结果经过采样后均变差，我又对分类器进行变换，尝试使用xgboost做分类器，但是采样后的效果仍然十分不好。通过采样后的样本的分配图中可以看到，欠采样导致了大量的样本的丢失，而过采样引入的噪声对分类结果引入的噪声是巨大的，所以任何一种采样结果的都使得原始未采样结果的下降。

	auc-prc	auc-roc
xgboost未处理	0.43	0.81
SMOTE	0.23	0.72
ADASYN	0.27	0.70
NearMiss	0.09	0.64
SMOTE+ENN	0.241	0.69
SMOTE+Tomek	0.207	0.71

集成

因为观察到即使使用相同特征，在不同方法下，每折的表现也会有较大的差异。所以尝试使用集成的方法，希望能让roc更高。因为之前已经用过的一些boosting的分类器比如xgboost和随机森林，在这里我们主要尝试bagging的方法。我尝试了投票法和线性混合，线性混合的结果下降非常多。所以只简单展示下投票法的结果。

我使用了mlp和随机森林和xgboost三个分类器在所有特征混合上进行了实验。尽管我们这里投票的结果可以说是从统计意义上和最高结果随机森林没有差异。所以这里的集成也没有太大效果。

	auc-roc	auc-prc
投票法	0.83	0.46

图网络方法

数据预处理

图神经网络中，除了网络结构对模型的有效性会产生很大的影响，图的节点、边信息作为输入同样重要。在此，我们主要讨论数据预处理中对边以及节点的特征提取工作。

开源架构**Deepchem**为我们提供了从分子中提取原子以及边特征的有效方法：

```
node_feature = dc.feat.graph_features.atom_features()
bond_feature = dc.feat.graph_features.bond_features()
```

文章*Analyzing Learned Molecular Representations for Property Prediction*中，作者使用了如下特征作为原子和化学键特征。

Table 1. Atom Features[a]

feature	description	size
atom type	type of atom (ex. C, N, O), by atomic number	100
# bonds	number of bonds the atom is involved in	6
formal charge	integer electronic charge assigned to atom	5
chirality	unspecified, tetrahedral CW/CCW, or other	4
# Hs	number of bonded hydrogen atoms	5
hybridization	sp, sp2, sp3, sp3d, or sp3d2	5
aromaticity	whether this atom is part of an aromatic system	1
atomic mass	mass of the atom, divided by 100	1

[a]All features are one-hot encodings except for atomic mass, which is a real number scaled to be on the same order of magnitude.

Table 2. Bond Features[a]

feature	description	size
bond type	single, double, triple, or aromatic	4
conjugated	whether the bond is conjugated	1
in ring	whether the bond is part of a ring	1
stereo	none, any, E/Z or cis/trans	6

[a]All features are one-hot encodings.

类似地，使用deepchem的特征抽取函数，我们可以得到：

节点（原子）特征

性质	描述	长度
Symbol	atom type 原子类型	44
Degree	原子包含的边数，与其相连的化学键数	10
Valence	原子化合价	6
Formal Charge	形式电荷，该原子被分配到的电荷数	5
Radical Electronics	活跃电子数	4
Hybridization	杂化轨道，包括：sp, sp2, sp3, sp3d, or sp3d2	5
Aromaticity	是否为芳香族元素	1

以上特征使用one-hot编码的，编码后特征总长度为`75`。为了进一步提升网络模型的能力，我们考虑添加新特征。在传统方法中，化学分子描述符是非常有效的化学分子特征表示方式，因此，我们选择原子对特定分子描述符的贡献来刻画其在分子中的性质：

性质	Contrib	长度
原子电荷	`_GasteigerCharge`,`_GasteigerHCharge`	2
MolLogP	`_CalcCrippenContribs`	1
分子近似表面积	`_CalcLabuteASAContribs`	1
TPSA	`_CalcTPSAContribs`	1

在添加新特征后，训练时出现了loss变成NaN的问题，考虑一般情况，有以下几种可能性：

1. 没有归一化处理
2. 添加正则化处理（Batch Normalization）
3. 学习率太大（降低学习率）。

但这些方法在实验中都没有获得成效。真正的原因是新特征引入了脏数据：部分原子不存在`_GasteigerCharge`,`_GasteigerHCharge`这两种特征，导致输入特征中出现NaN，网络自然会出现错误。

删除原子电荷特征后，考虑图神经网络训练时间长，我们仅选取一折的数据来看新的特征方法是否有效，网络模型选用gcn，可以发现，添加新特征后，网络的预测能力并没有提升，泛化能力也没有增强：

节点特征向量长度	**auc-poc**	**auc-roc**
75	0.1251	0.9531
79	0.1382	0.7182

这可能是由于`deepchem`提供的方法中，所有的特征都被编码为`one-hot`的表示，而新增的分子性质是连续实数，这会导致输入特征有所偏斜。尽管可以对输入作归一化/正则化处理，仍没有直接使用`one-hot`编码更有效。

边（化学键）特征

所有的边特征都适用`one-hot`编码

性质	**描述**	**长度**
Bond Type	单键，双键还是三键	3
Conjugated	是否共轭	1
InRing	是否为成环键中的一部分	1
Stereo	化学键立体性质	1

数据采样

可以知道，我们所使用的数据集由严重的数据倾斜问题，为了优化现阶段的数据，在图网络中，我们同样使用采样的方法进行处理。过采样、欠采样以及混合采样同样适用于图神经网络模型。但由于神经网络训练需要大量数据样本作为支持，欠采样方法将大量减少数据样本数量。因此，我们主要使用过采样的方法，看是否对网络有显著的影响，由于图网络训练慢，我们仅选取一折的数据来看采样是否有效，网络模型皆选用gcn：

采样方法	**auc-prc**	**auc-roc**
未采样	0.0509	0.5465
简单随机过采样	0.2315	0.8388
smote	0.1787	0.8940

可以看到，采样方法并没有对图神经网路作出大幅改善，但在可以一定地提升auc-roc值以及auc-prc值，因此，后文中输入我们的网络的数据都已经做出过采样预处理。

图网络模型

信息传递网络（Message Passing Neural Networks, MPNN) 是由Goggle提出的一种图神经网络通用框架。经验性地，在化学分子性能预测上有较为优秀的表现。因此，在使用图网络的方法中，我们优先使用MPNN来实现分子性能预测的任务。

MPNN原理

MPNN的传播机制中包含两个阶段：信息传递阶段（message passing）以及 读出（read out）阶段。

信息传递阶段

顾名思义，在这一阶段中，信息将在图网络中进行扩散传播。对于网络中的节点v，有如下的公式：

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}) \tag{1}$$
$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

m_v^{t+1}代表节点`v`在`t+1`步中收到的消息，$N(v)$代表节点`v`的邻居节点，h_v^t, h_w^t为节点在第`t`步的特征（即为网络中的隐藏神经元），该特征的初始输入为数据预处理后提取的节点特征，e_{vw}则是节点 v 之间的 w 的边特征向量。可以理解为，节点 v 在第`t`步从邻居节点处得到了新的信息，M_t为消息更新函数。

收到新消息，需要对节点进行更新：U_t为节点更新函数，得到新的节点状态h_v^{t+1}。每一次消息传递需要有`t`步，可以认为消息传递网络有`t`层。

读出阶段

在这一阶段中，网络讲根据各节点神经元的状态输出最终的预测结果，有如下的公式：

$$y = R(h_v^T | v \in G) \tag{2}$$

读出函数要求输出结果与输入节点的排序不相关，也就是说网络对于同构的输入图应该有相同的输出。

在实验中，通过改变消息更新函数，节点更新函数，读出函数，我们试用了两种不同的MPNN，如下：

MPNN-1

消息传递阶段

- 消息更新函数 + 节点更新函数：两层线性回归函数`nn.Linear` + 激活函数`selu`
- 先将邻居节点输入特征h_w^t作线性变化，使用函数`lin0`，输出记为：$h_w^{t'}$；边特征e_{vw}经过线性层`lin1`转换为e'_{vw}。$h_w^{t'}$ 与h_v^t，边特征e'_{vw}一起过一层线性层`lin2`。最后使用激活函数`selu`得到该步消息传递的输出h_v^{t+1}，输出将作为下一步消息传递的节点处是状态

读出阶段

- 读出函数：一层线性层`nn.Linear` + 激活层（`selu`） + 线性层（线性相加）+ 激活层（`tanh`）+ 线性层
- 首先将各个节点的最新特征读出，作线性变化，过线性层`lin3`，使用激活函数`selu`得到输出h'_v，随后将图中所有节点特征相加，由激活函数`tanh`得到输出向量，最后由线性层`lin4`将该向量转换到相应的输出维度上，得到y。

网络结构

Layer	Input	Output
Lin0	n_feature_dim	n_h_feature_dim_1
Lin1	e_feature_dim	e_h_feature_dim
Lin2	n_h_feature_dim + n_feature_dim + e_h_feature_dim	n_feature_dim
Lin3	2 * n_feature_dim	n_h_feature_dim_2
Lin4	n_h_feature_dim_2	1

```
self.lin0 = {0: nn.Linear(self.n_feature_dim, self.n_h_feature_dim),
        1: nn.Linear(self.n_feature_dim, self.n_h_feature_dim),
        2: nn.Linear(self.n_feature_dim, self.n_h_feature_dim)}
self.lin1 = nn.Linear(self.e_feature_dim, self.e_h_feature_dim)
self.lin2 = {
            0: nn.Linear(self.n_h_feature_dim + self.n_feature_dim +
self.e_h_feature_dim, self.e_h_feature_dim),
        1: nn.Linear(self.n_h_feature_dim + self.n_feature_dim +
self.e_h_feature_dim, self.e_h_feature_dim),
        2: nn.Linear(self.n_h_feature_dim + self.n_feature_dim +
self.e_h_feature_dim, self.e_h_feature_dim)}
self.lin3 = nn.Linear(2 * self.n_feature_dim, self.n_out_dim)
self.lin4 = nn.Linear(self.n_out_dim, 1)
```

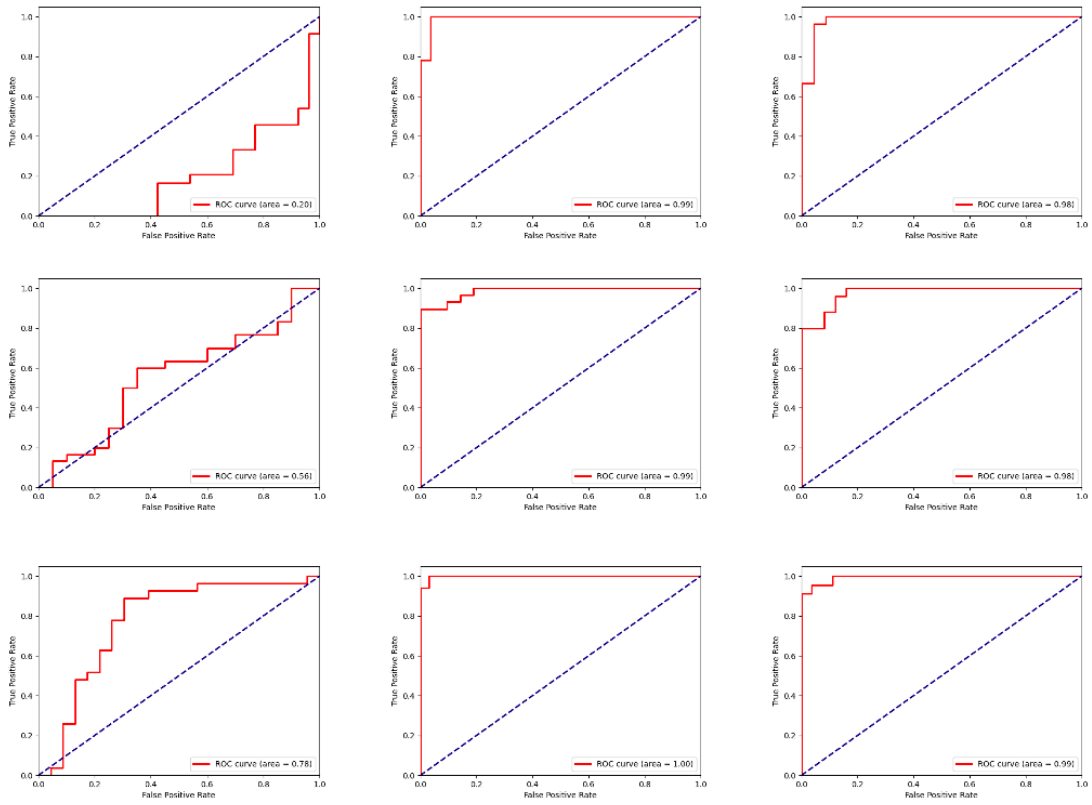
训练结果

虽然该网络没有使用复杂的读出函数以及消息更新函数，但在抽取节点特征维度为`75`，边特征为`6`，传递消息共有`5`步时，`batch_size`为`50`，`epoch`为`150`已经有较好的结果：

	ROC_AUC	PRC_AUC
fold 0	0.9202	0.4338
fold 1	0.8388	0.2392
fold 2	0.7109	0.0488
fold 3	0.9009	0.5669
fold 4	0.9781	0.2315
fold 5	0.7162	0.0268
fold 6	0.9082	0.0250
fold 7	0.9607	0.4145
fold 8	0.9447	0.1380
fold 9	0.9592	0.2648
mean	0.8838	0.2383

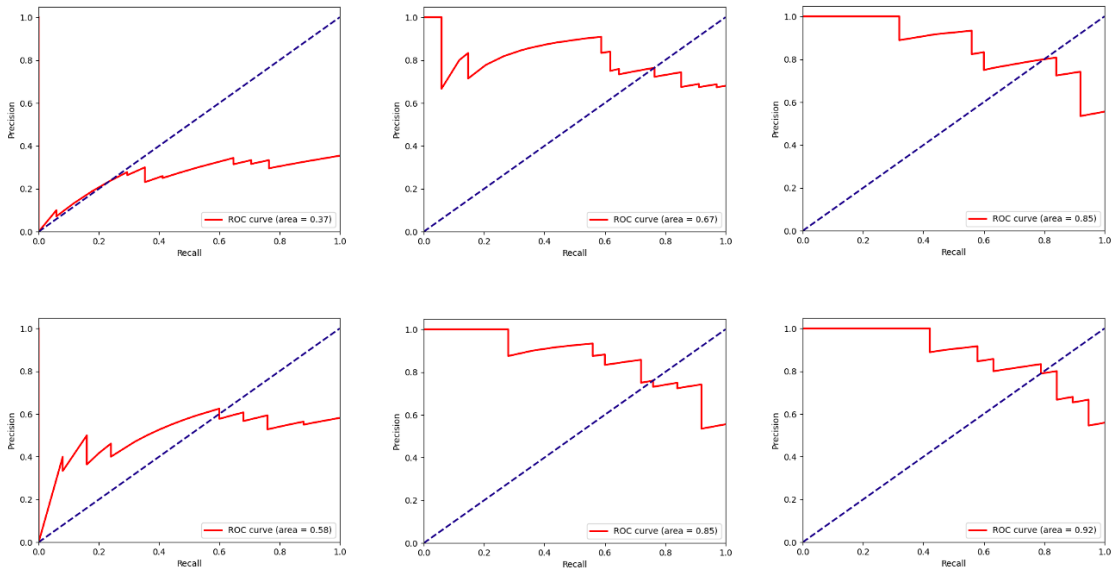
MPNN在roc值上取得了较好的结果，泛化能力也较强，可以在测试集上得到平均0.8838的`roc_auc`，但在`prc`的表现上却不尽如人意。我们知道roc曲线在正负样本分布变化时保持不变。而prc则更可以表现出在倾斜数据集上模型的能力。由于在本数据集中，正例样本要远远小于负例样本，低roc反映出：分类器将大量的负例错判为正例。从训练过程可知，经过多轮迭代，`roc_auc`和`prc_auc`值都可以在训练数据中中达到接近1的值。因此，模型有可能出现了过拟合的行为，泛化能力较差。

ROC曲线



上图为在fold `5,6,7` 训练过程中的ROC曲线变化图，其中第一列数据为第一次迭代时的ROC曲线，中间一列为第100此迭代时的结果，第三列为最后一次迭代的结果。在实验中，随着epoch数增加，loss下降，roc值趋于上升。但并非一直处于上升趋势。可见图中三折，第三列数据中的roc都要略逊于中间一列。

PRC曲线



上图为在fold `0,1` 中训练过程中PRC曲线变化图，其中第一列数据为第一次迭代时的PRC曲线，中间一列为第100此迭代时的结果，第三列为最后一次迭代的结果。在实验中，随着epoch数增加，loss下降，PRC值趋于上升。

MPNN-2

参考AWS开源图神经网络框架DGL，DGL的实现依据为Google文章 Neural Message Passing for Quantum Chemistry，即门控神经网络GG-NN

消息传递阶段

- 线性层 + 激活层（`relu`）+ 卷积层 + GRU
- 首先将输入的全部节点特征$\{h_{v_1}^t, h_{v_1}^t, \ldots, h_{v_n}^t\}$过一层线性层（`lin0`）和激活层（`relu`），得到输出$v' = \{h_{v_1}^{t'}, h_{v_2}^{t'}, \ldots, h_{v_n}^{t'}\}$

  （输出为$n \cdot n_hidden_dim$维的），将输出的特征向量和边特征，初始节点特征一起过卷积层（卷积层中包含对边特征的处理）以及激活层`relu`，得到输出u'，将u', v'转换到一个维度上，并输入门控层GRU，后得到输出`out`，步骤2将重复`t`步。

读出阶段

- set2set + 线性层 + 激活层（`relu`） + 线性层
- set2set是一种池化方法，将消息传递层的输出特征向量压缩得到对图网络的描述向量。随后经过线性层`lin1`和激活函数`relu`，并通过线性层`lin2`将预测结果转化到相应的维度

网络结构

Layer	input	Output
Lin0	node_input_dim (batch_size * node_feature_dim)	n_hidden_dim
conv	n_hidden_dim + e_input_dim	n_hidden_dim
gru	n_hidden_dim	n_hidden_dim
Lin1	2 * n_hidden_dim	n_hidden_dim
Lin2	n_hidden_dim	1

```
self.num_step_message_passing = num_step_message_passing
self.lin0 = nn.Linear(node_input_dim, node_hidden_dim)
edge_network = nn.Sequential(
            nn.Linear(edge_input_dim, edge_hidden_dim), nn.ReLU(),
            nn.Linear(edge_hidden_dim, node_hidden_dim * node_hidden_dim))
self.conv = NNConv(in_feats=node_hidden_dim,
                   out_feats=node_hidden_dim,
                   edge_func=edge_network,
                   aggregator_type='sum')
self.gru = nn.GRU(node_hidden_dim, node_hidden_dim)
self.set2set = Set2Set(node_hidden_dim, num_step_set2set, num_layer_set2set)
self.lin1 = nn.Linear(2 * node_hidden_dim, node_hidden_dim)
self.lin2 = nn.Linear(node_hidden_dim, output_dim)
```

训练结果

虽然`GG-NN`有更强的网络结构，但在这个任务的处理上，并没有表现出超过简单MPNN的能力，在抽取节点特征维度为`75`，边特征为`6`，传递消息共有`5`步时，`batch_size`为`50`，`epoch`为`100`时结果如下：

	ROC_AUC	PRC_AUC
fold 0	0.5020	0.0236
fold 1	0.4483	0.4133
fold 2	0.6830	0.0129
fold 3	0.5997	0.0236
fold 4	0.6808	0.0105
fold 5	0.5849	0.1093
fold 6	0.5750	0.0101
fold 7	0.5465	0.0509
fold 8	0.6808	0.0105
fold 9	0.5465	0.2286
mean	0.5848	0.0907

在训练中，GG-NN虽然有大幅度收敛，但收敛趋势依旧不稳定，roc_auc和prc_auc相比另两种模型，都要更差。值得注意的是，虽然在训练中，roc值不断下降，prc值却始终维持较低的值。

考虑以下可能：

1. 网络的结构导致收敛能力有限：改变各隐藏层维度，消息传播层，set2set层步长、维度，模型依旧没有更好的结果；
2. 网络的输入有误，缺少正则化，归一化：添加对输入的正则化/归一化处理，结果没有改善。

MPNN-2在实验的不断挑参数中并一直没有取得理想的结果，因此在MPNN模型，我们以MPNN-1作为我们的选择。

GCN

图卷积神经网络（GCN）也是一种常用的图网络结构，我们也想知道，MPNN相比GCN，在分子性能预测任务上是否有更好的能力。实验中，我们试用的GCN结构如下为：`1层embedding层 + n图层卷积层 + 1层全连接层`。

网络结构

Layer	input	Output
Embedding	node_input_dim	node_feature_dim
Conv	node_feature_dim	node_feature_dim
Fc	node_feature_dim	1

训练结果

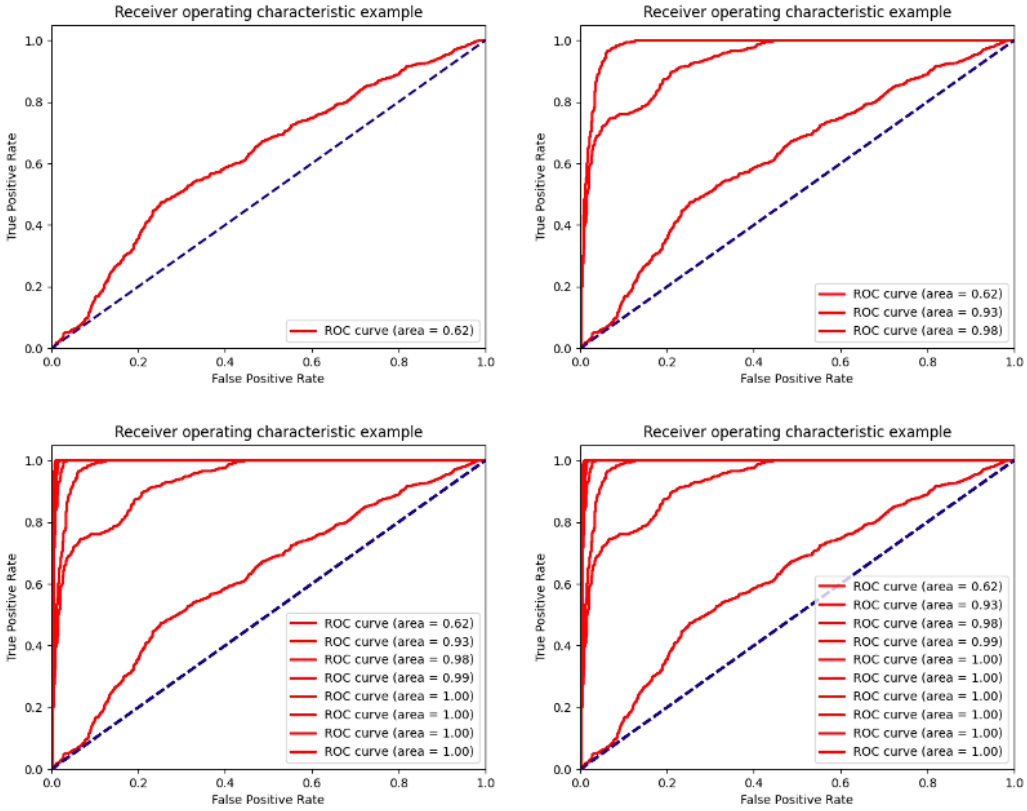
与MPNN不同，这里使用的GCN网络仅仅使用节点的特征以及图的邻接矩阵作为输入。在抽取节点特征维度为`75`，网络深度为`5`层时，`batch_size`为`50`，`epoch`为`100`时结果如下：

	ROC_AUC	PRC_AUC
fold 0	0.9367	0.7106
fold 1	0.9035	0.5185
fold 2	0.8718	0.5914
fold 3	0.5644	0.0282
fold 4	0.9308	0.8439
fold 5	0.6869	0.4133
fold 6	0.8502	0.0156
fold 7	0.9810	0.7269
fold 8	0.9531	0.1251
fold 9	0.9429	0.8149
mean	0.8754	0.4788

可以看到，GCN的训练中，`roc`虽然在大多数折中可以取得不错的表现，但依旧有比较差的数据（如fold3）。并且有实验过程可以知道，在表现比较大的数据，如fold3，fold5中，在训练集上依旧有非常可观的结果（>0.9），这是由于模型的泛化能力不佳造成的。虽然GCN的`roc`表现略逊于MPNN-1，但它在`prc`上有更强的表现，这说明GCN模型在应对我们的偏斜数据集时，有更强的能力

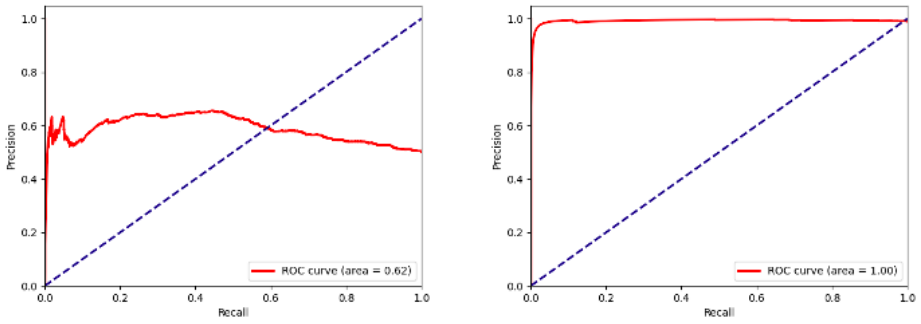
没有化学键特征信息，仅使用简单的网络结构，依旧可以得到不错的结果。这或许是由于化学键信息对于化学性质预测的贡献有限，或者复杂的网络在表达信息上不够有效导致的。

ROC曲线



以上roc为针对`fold 0`数据叠加的曲线，可以发现，随着epoch增加，roc曲线的面积区域增大，甚至可以达到1.0。

PRC曲线



以`fold 0`为例，左图为第一次迭代的prc曲线，右图为最后一次迭代的prc曲线。在训练中，训练集上的prc值可以快速收敛，并有显著的提高。

结论

在药物预测任务中，我们尝试了经典方法，循环网络方法和图网络方法，并在不平衡问题进行了采样和集成的优化，实现了现有的药物预测问题解决方案的大部分方案。

在经典方法中主要依赖于特征工程，因为化学分子学已经沉淀了非常成熟的分子特征，对分子的描述符和分子指纹的提取就可以是使即使简单的分类器，也能达到0.80以上的roc结果，和0.40左右的prc效果，且训练时间非常的短。同时借用自然语言处理中的word2vec的词向量方法，借用训练好的mol2vec模型，在分子特征基础上加入词向量特征，使得最高结果roc可达0.85，同時prc可以达到0.44，是非常不错的效果。因此我们在图网路中也引入了4个分子描述符作为特征。

而使用循环网络方法，尽管是我们看到smiles串的第一想法，但是却效果不佳，同时训练时间偏长，所以单纯的循环网络可能并不适合药物预测任务。

图神经网络MPNN和GCN都表现出了一定的预测能力。其中，GCN的roc可达到0.8754，prc可达到0.4788，MPNN在roc上表现更优，可达到0.8838，但prc值不佳，只有0.2383。尽管图神经网络效果可观，但训练时间长，且由于数据量小，数据分布存在倾斜，模型可能出现过拟合问题，在泛化能力上仍有待加强。

同时我们尝试使用采样方法对数据不平衡的问题进行改善，尽管多种采样方法在随机森林和xgboost的分类过程中使结果降低，但是对于图网络却提升了结果，所以采样方法的有效性会根据分类方法而易。

代码结构

1. 特征工程的代码包括采样和集成的结果在 classicclassifier.ipynb中
2. **循环神经网络**中代码和结果在 rnn.ipynb 中
3. **图网络**

- gcn.py 存放图卷积神经网络代码
- preprocessing.py 存放读取数据以及预处理代码
- mpnn_1.py 存放MPNN-1代码
- mpnn_2.py 存放MPNN-2代码

分工

- 前期讨论，review由两人共同完成，并确定做分工
- 王润 18300720003
    - 经典方法、循环神经网络方法，采样实验、报告
- 田睿 173000750084
    - 图网络方法实验、报告