

论文读后感以及改进思路

背景知识

识别化合物蛋白相互作用(CPI)是药物发现和化学基因组学研究中的一项关键任务，但三维(3D)结构缺失的蛋白质占潜在生物靶点的很大一部分，这就需要研究仅利用蛋白质序列信息预测CPI的方法。

生物学靶点 (英语： Biological target) 是指位于生物体内，能够被其他物质 (配体、 药物等) 识别或结合的结构。比如，一些微生物的趋化性 (chemotaxis) 系统中的甲基 趋化受体蛋白 (methyl-accepting chemotaxis protein) 能够被其生存环境中的一些分子 (葡萄糖等) 结合，进而调节微生物的鞭毛运动。常见的药物靶点包括蛋白质、核酸 和离子通道等。

背景知识

然而，它们的性能，特别是它们在外部数据上的泛化能力，往往受到**数据不平衡**的影响，这归因于缺乏经验证的**非活性（负面）样本**。

计算方法

1. 基于结构——通过从分子对接模拟生成的**三维复合物模型**中计算基于物理化学的分数来评估CPI，当目标蛋白质的可靠三维结构可用时，可以在没有已知相互作用的先验信息的情况下应用对接方法。
2. 基于结构无关（基于化学基因组学）——化学信息学方法，**利用已知CPI的先验信息来预测未知的相互作用。当先验信息涵盖足够的药理空间时，它们可以以相对较低的计算成本准确预测CPI。**

已知CPI数据库

ChEMBL、BindingDB、PubChem、DrugBank和PDBbind

最新进展

CPI数据库的迅速增长，加速了使用机器学习（ML）算法开发各种基于结构无关方法。

这些ML方法在统一的框架中考虑化合物信息、蛋白质信息以及它们之间的相互作用。最近，使用深度学习（DL）技术的CPI预测模型，如**卷积神经网络（CNN）**、**图卷积网络（GCN）**和**Transformer算法**，显著提高了预测性能和可解释性。这些模型可以在学习相互作用的端到端学习过程中提取化合物和蛋白质的特征表示。

传统方法

为了弥补负样本的不足，先前的研究采用了随机配对方法（以下简称随机负样本）。该方法从尚未经实验证实为相互作用的化合物-蛋白质对中随机生成负样本。然而，提取的样本可能包含潜在的活性（正面）样本，从而降低了可信度。

文章提出的方法

作者提出了一种新的自我训练方法，**有效增加负样本**。这种方法被应用于**基于图的CPI预测模型**，并成功地提高了模型的性能，包括泛化能力。

自我训练方法的工作流程

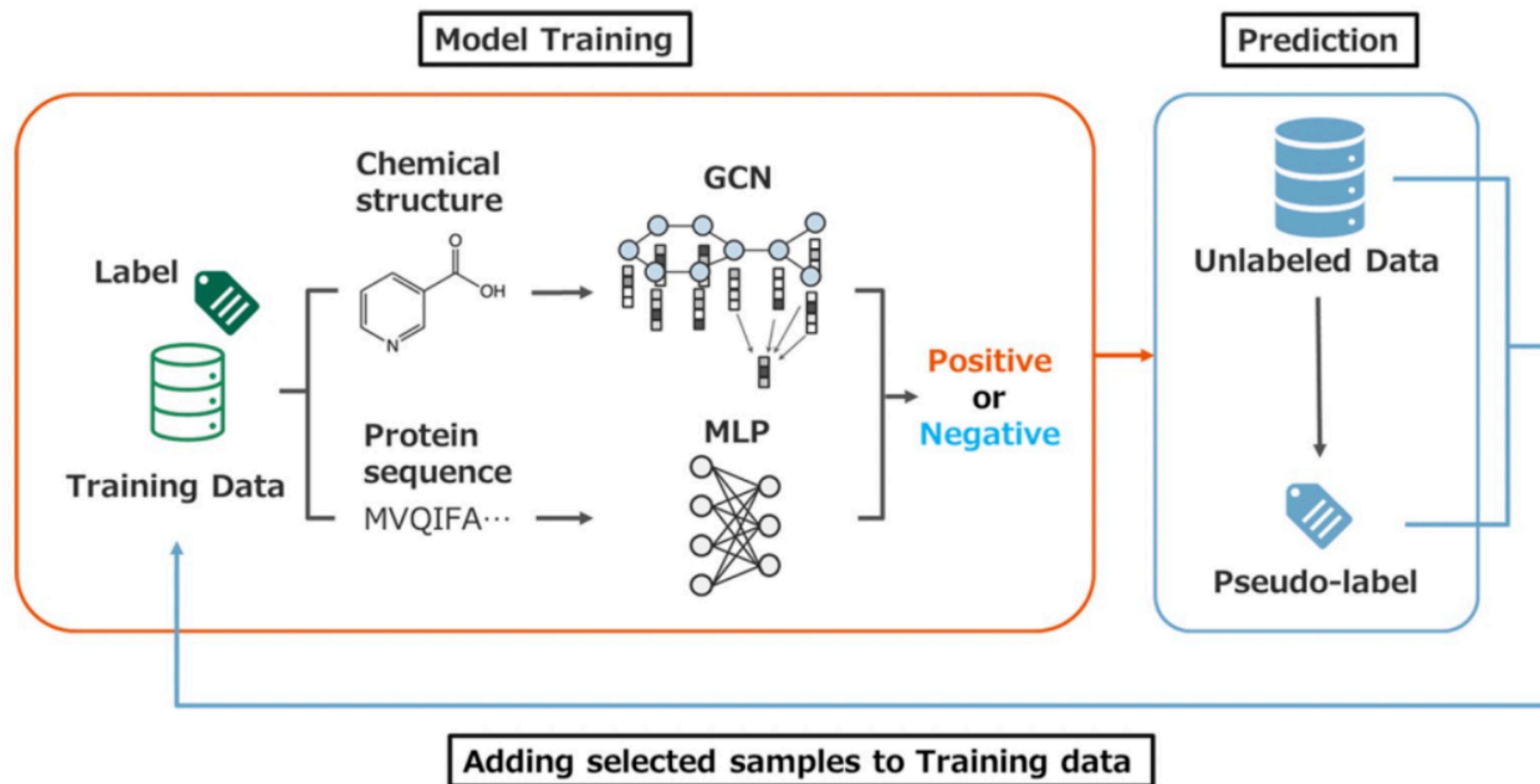


图 1

文章提出的方法

自我训练始于使用标记数据训练一个教师模型。教师模型用于通过对未标记数据进行预测来生成伪标签。在将选定的伪标记数据添加到训练数据后，使用更新后的数据训练一个学生模型。这个过程是通过使用学生模型作为下一个教师模型来迭代的。图1说明了作者提出的自我训练方法的工作流程，并且实施了以下步骤。第一步，使用最小化二元交叉熵损失进行训练，得到教师模型。第二步，利用教师模型 f 为未标记的数据集生成伪负标签。如果对于某个数据样本，满足 $\phi < f(x) < 0.5$ ，那么该样本被认为是一个伪负样本。在这里， $\phi \in [0, 0.5)$ 是一个阈值参数。第三步，将伪标记的负样本被添加到标记数据中。第四步利用新数据训练学生模型。第五步，将学生模型视为新的教师模型，重复如上过程。

模型效果评估

ROC-AUC PR-AUC

通过将自我训练方法应用于不平衡数据，正负样本的数量逐步逼近，并且每个蛋白质中正样本的比例在最终迭代时收敛到0.5。为了评估目前构建模型的泛化能力，作者比较了模型在外部数据集上的性能，其中BioPrint适用于GPCR家族，Davis适用于激酶家族。**在这里采用了PR-AUC分数作为主要指标，该指标更适用于评估在由负样本主导的不平衡数据集上的模型性能。**

	ROC-AUC (Std ^a)	PR-AUC (Std)	ROC-AUC (Std)	PR-AUC (Std)
baseline	0.9139 (0.0083)	0.9962 (0.0005)	0.9175 (0.0035)	0.9946 (0.0004)
weighted loss	0.9182 (0.0079)	0.9965 (0.0005)	0.9149 (0.0089)	0.9945 (0.0006)
random undersampling	0.9026 (0.0046)	0.9958 (0.0002)	0.8982 (0.0066)	0.9933 (0.0006)
random negative	0.9034 (0.0063)	0.9961 (0.0004)	0.9053 (0.0012)	0.9940 (0.0002)
similarity controlled	0.9234 (0.0059)	0.9969 (0.0002)	0.9166 (0.0060)	0.9947 (0.0005)
self-training (ours)	0.9336 (0.0035)	0.9974 (0.0003)	0.9336 (0.0021)	0.9960 (0.0002)

^aStandard deviation (Std) was calculated from the results of 5-fold

结论

对伪标记样本确定的分数阈值进行分析表明，通过使用经验证的、基于预测值的可信样本，并优先添加靠近边界的信息丰富样本，可以实现改进的泛化能力。

简单来说：这篇文章的思想与半监督学习十分相似，在无标签数据比有标签数据多的情况下，以有监督学习为起点，不断预测伪标签，不断重复，直至正负标签比例达到平衡。

前置知识

最小化二元交叉熵

$$H(P, Q) = -P(1)\log(Q(1)) - P(0)\log(Q(0))$$

其中， $P(1)$ 和 $P(0)$ 是真实标签的概率， $Q(1)$ 和 $Q(0)$ 是预测标签的概率。

- 1.公式简单：二元交叉熵的公式相对简单，只涉及到两个类别的概率和预测概率，因此计算量相对较小。
- 2.对称性：二元交叉熵对于真实标签和预测标签是对称的，即真实标签为1时，预测标签可以为1或0，真实标签为0时，预测标签也可以为1或0。这种对称性使得二元交叉熵在处理二分类问题时更加自然和合理。
- 3.稳定性：二元交叉熵对于样本的错误分类具有一定的稳定性。即使是在样本数据存在噪声或者异常值的情况下，二元交叉熵也能够相对稳定地计算出样本之间的相似度。
- 4.适合处理不平衡数据：在二分类问题中，经常会遇到类别不平衡的情况，即其中一个类别的样本数量远大于另一个类别。二元交叉熵能够考虑到这种情况，通过调整每个类别的权重，使得模型能够更好地处理不平衡数据。
- 5.易于优化：二元交叉熵的优化相对容易，因为它只涉及到两个类别的概率和预测概率。这使得二元交叉熵在训练过程中能够更快地收敛，并且得到更优的模型表现。

前置知识

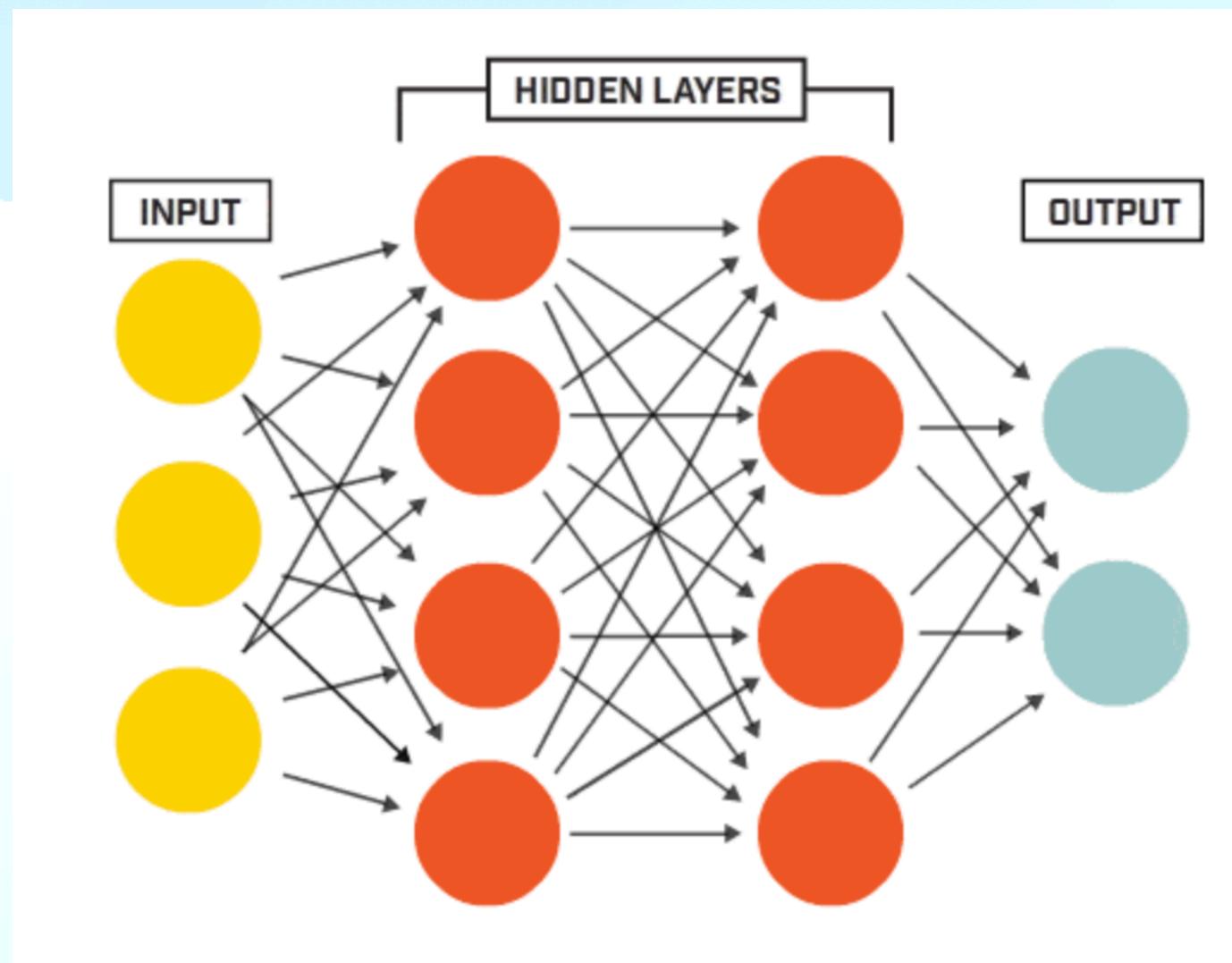
MLP

最简单且原汁原味的神经网络则是多层感知器（Multi-Layer Perception，MLP），神经网络的变种有很多，误差反向传播（Back Propagation，BP）神经网络、概率神经网络、卷积神经网络（Convolutional Neural Network，CNN-适用于图像识别）、时间递归神经网络（Long short-term Memory Network，LSTM-适用于语音识别）

前置知识

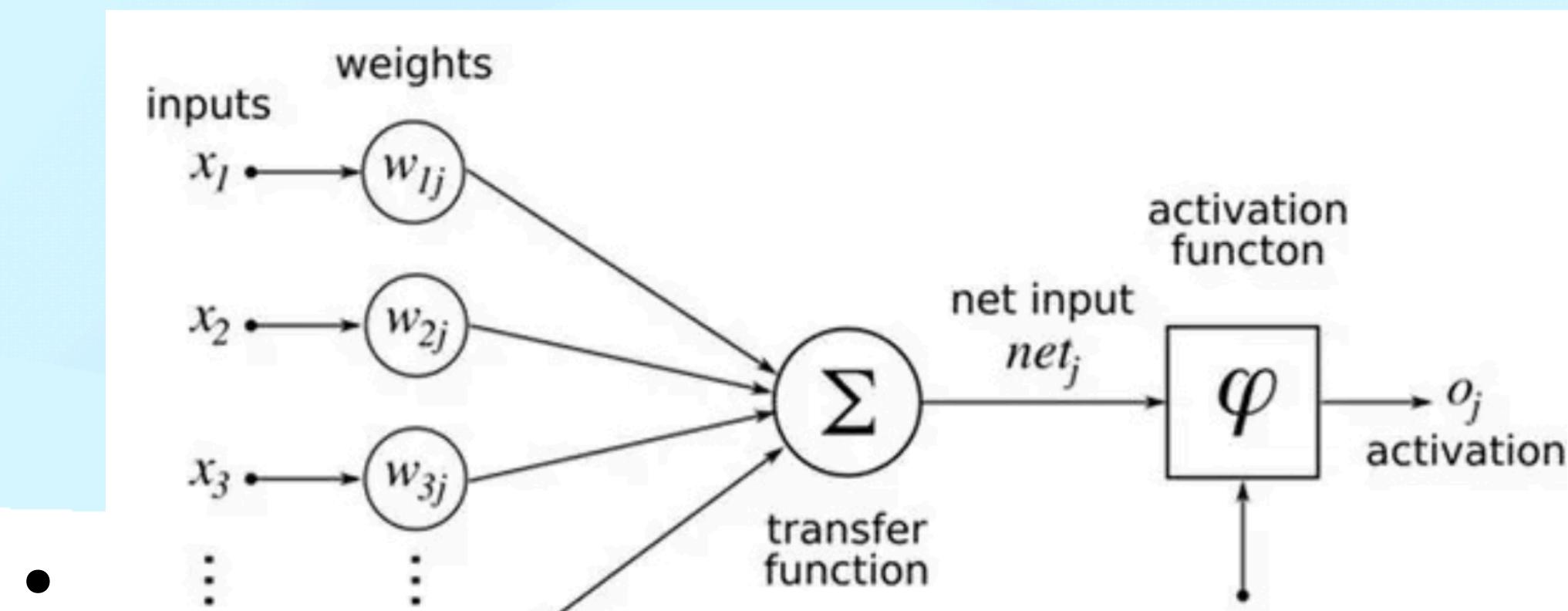
MLP

我们基于生物神经元模型可得到多层感知器MLP的基本结构，最典型的MLP包括包括三层：输入层、隐层和输出层，MLP神经网络不同层之间是全连接的（全连接的意思就是：上一层的任何一个神经元与下一层的所有神经元都有连接）。



前置知识

MLP



由此可知，神经网络主要有三个基本要素：权重、偏置和激活函数

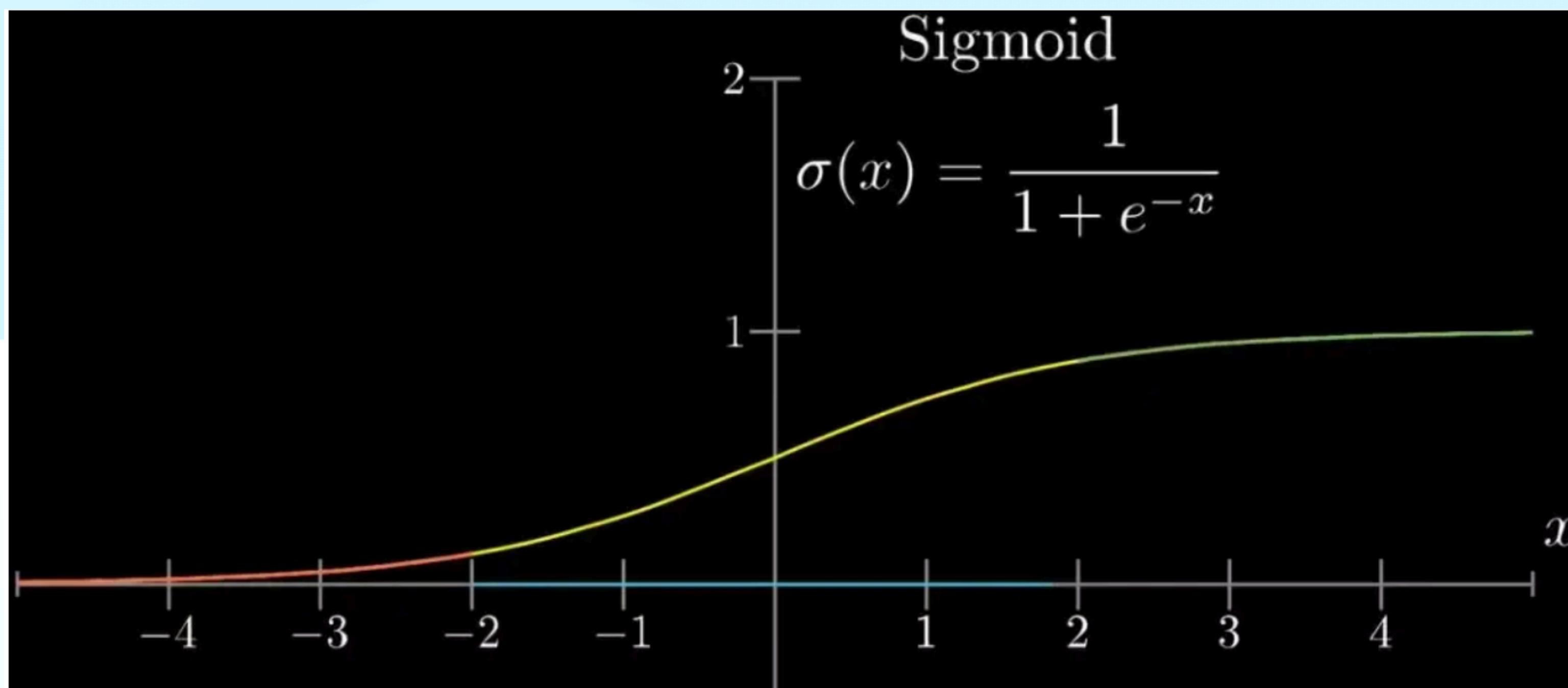
权重：神经元之间的连接强度由权重表示，权重的大小表示可能性的大小

偏置：偏置的设置是为了正确分类样本，是模型中一个重要的参数，即保证通过输入算出的输出值不能随便激活。

激活函数：起非线性映射的作用，其可将神经元的输出幅度限制在一定范围内，一般限制在 $(-1 \sim 1)$ 或 $(0 \sim 1)$ 之间。最常用的激活函数是Sigmoid函数，其可将 $(-\infty, +\infty)$ 的数映射到 $(0 \sim 1)$ 的范围内。

前置知识

MLP

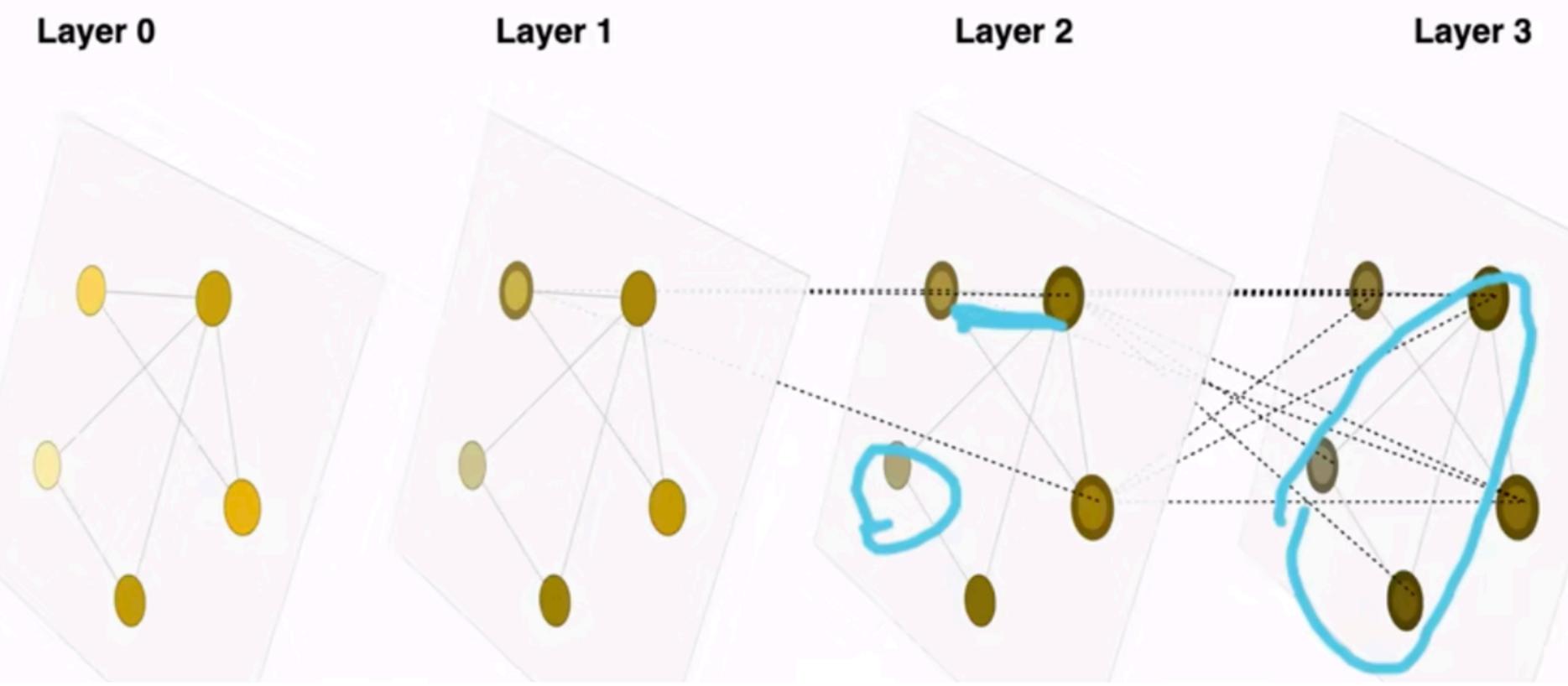


前置知识

GCN

A Gentle Introduction to Graph Neural Networks

Neural networks have been adapted to leverage the structure and properties of graphs. We explore the components needed for building a graph neural network - and motivate the design choices behind them.



每一层的顶点是由它上一层的邻居得来的，如果网络结构足够深，很有可能一个节点是能够处理到整个比较大范围的图里面的节点信息

前置知识

GCN

Images as graphs

We typically think of images as rectangular grids with image channels, representing them as arrays (e.g., 244x244x3 floats). Another way to think of images is as graphs with regular structure, where each pixel represents a node and is connected via an edge to adjacent pixels. Each non-border pixel has exactly 8 neighbors, and the information stored at each node is a 3-dimensional vector representing the RGB value of the pixel.

A way of visualizing the connectivity of a graph is through its *adjacency matrix*. We order the nodes, in this case each of 25 pixels in a simple 5x5 image of a smiley face, and fill a matrix of $n_{nodes} \times n_{nodes}$ with an entry if two nodes share an edge. Note that each of these three representations below are different views of the same piece of data.

图片怎么表示为图?

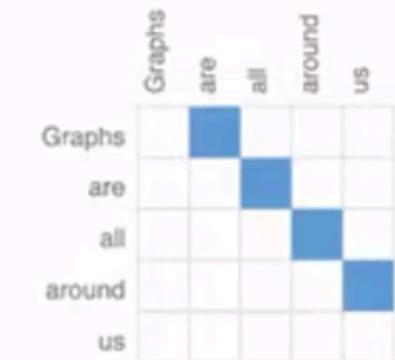
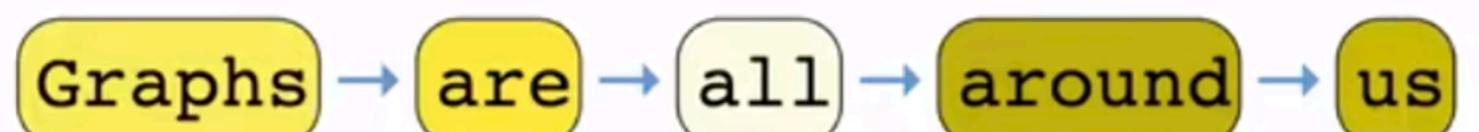
每一个像素是一个点，如果一个像素与之前的像素是连接关系的话，像素之间就连条边

前置知识

GCN

Text as graphs

We can digitize text by associating indices to each character, word, or token, and representing text as a sequence of these indices. This creates a simple directed graph, where each character or index is a node and is connected via an edge to the node that follows it.



- Edit the text above to see how the graph representation changes.

把每一个词表示成一个顶点，一个词和下一个词之间有一条有向边

前置知识

GCN

图层面任务 graph-level task
对整个图进行识别

顶点层面任务 node-level task
对一个顶点的属性判断

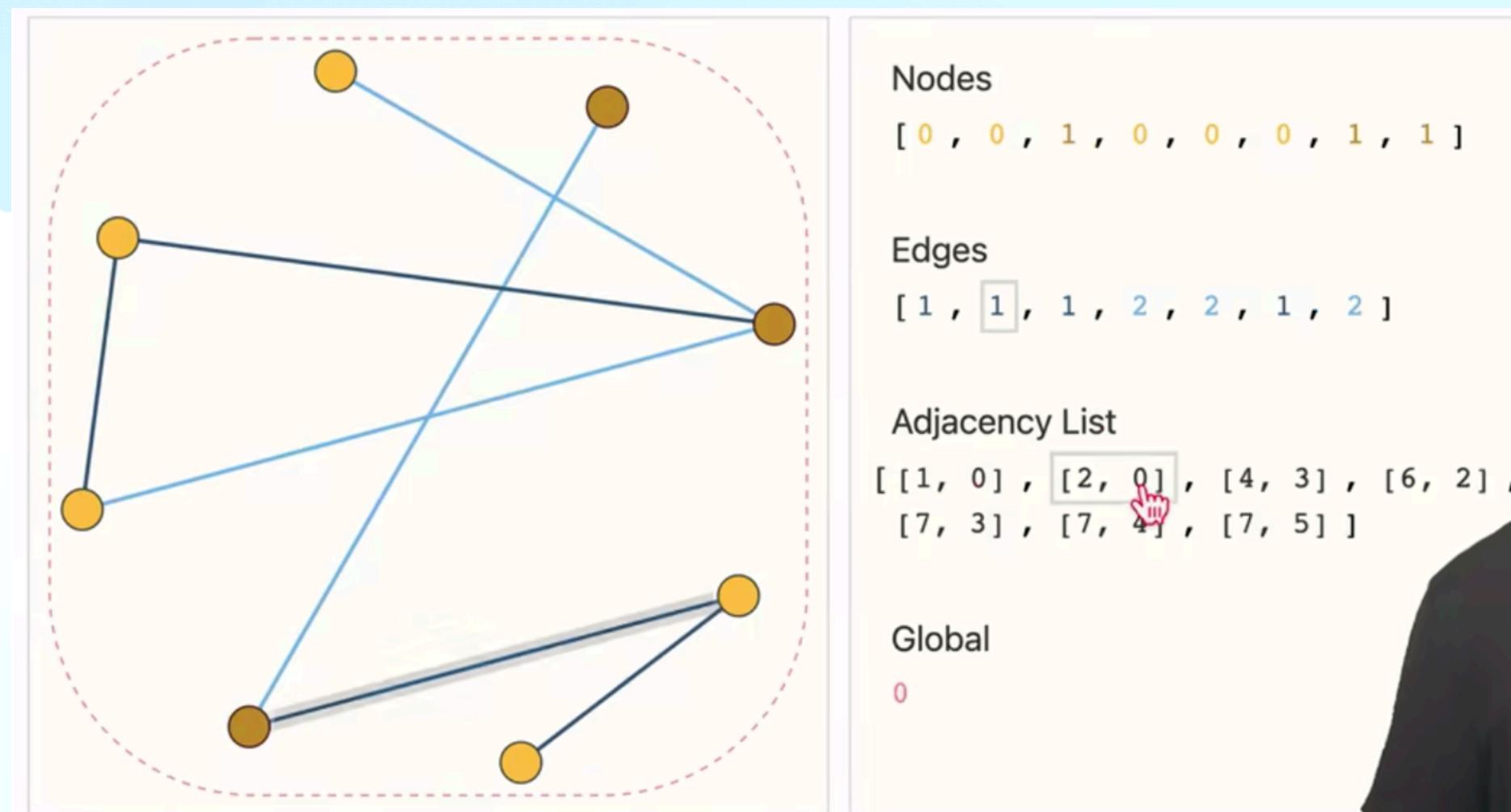
边级别任务 edge-level task
对一个边的属性判断

前置知识

GCN

机器学习做预测：nodes, edges, global-context, connectivity(连接性，邻接矩阵稀疏问题，行列顺序交换不会影响它，放进神经网络后代表的都应该是同一张图)

维护邻接列表



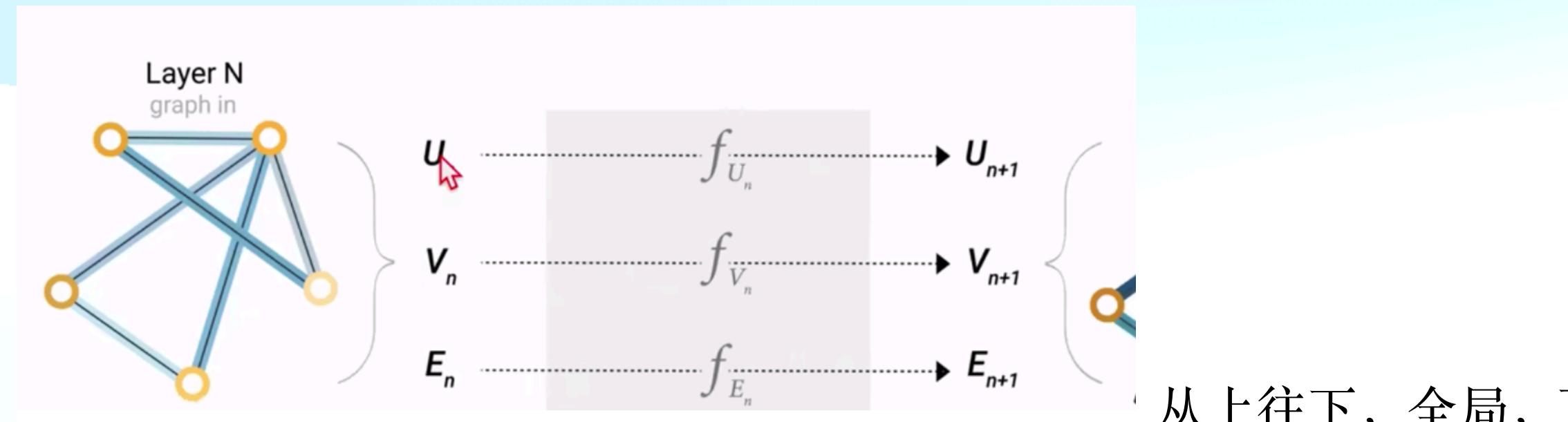
Adjacency List长度和边数一样，第*i*个项表示的是第*i*条边，它连接的哪两个节点，存储上只有把边和所有的属性存下来，所以在存储上是高效的，另外对于顺序无关，可以把边的顺序打乱，只要把邻接列表的顺序相应的变了就行了，顶点顺序打乱，只需要把邻接列表里面的数字做相应的更新就行了

前置知识

GCN

GNN对图上所有的属性，包括了顶点，边，全局的上下文，进行的一个可以优化的变换，这个变换是能够保持住图的对称信息的，所谓的对称信息就是说我把这些顶点进行另外一个排序之后，整个结果是不会变的

信息传递的神经网络，输入是一个图，输出也是一个图，会对那些属性就是你的顶点，边和全局的那些向量进行变换，但是不会去改变这个图的连接性，边是连接那些顶点的这个信息在进入图神经网络后是不会被改变的



从上往下，全局，顶点，边

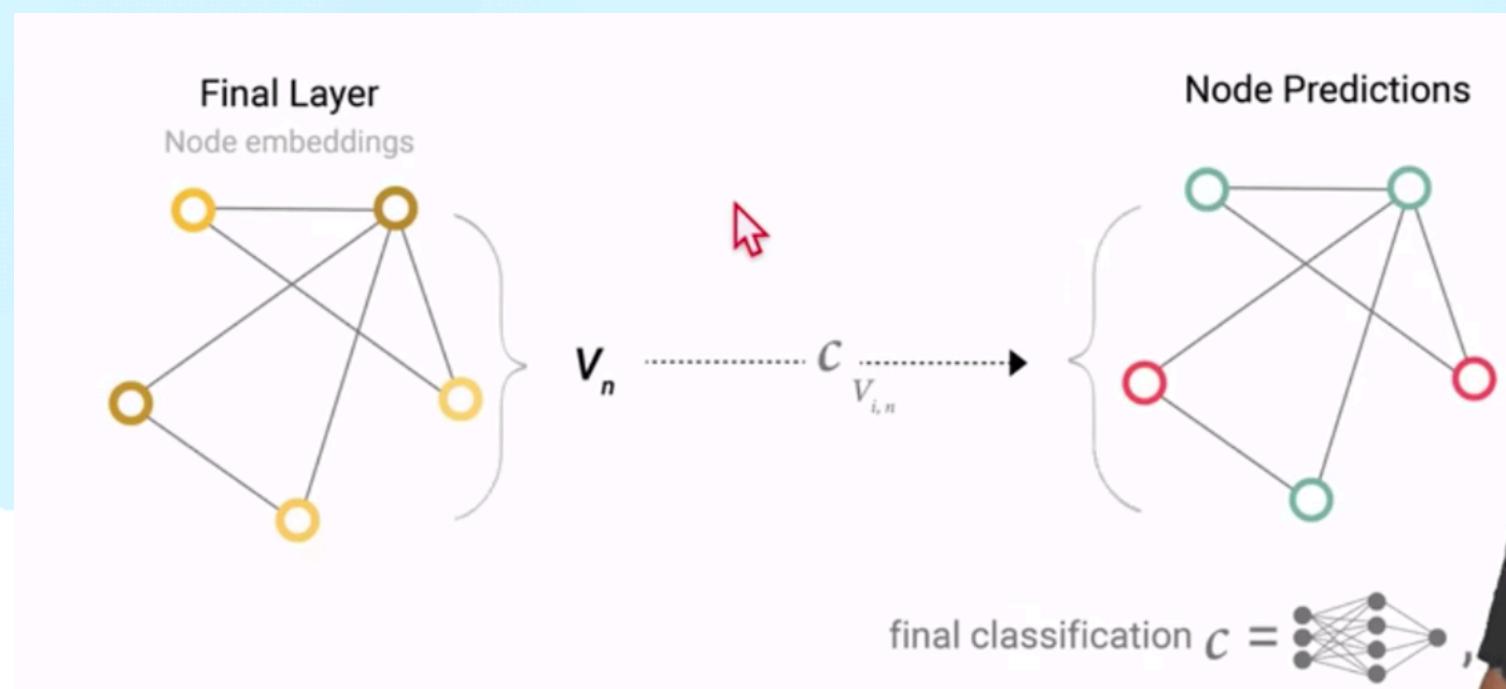
最简单的图神经网络，对于顶点向量，边向量，全局向量，分别构造一个MLP，就是多层感知机，MLP输入大小和输出大小是一样的，取决于输入的向量，这三个mlp就组成了一个GNN的层，输入是一个图（最左边），输出还是一个图，属性进行变换但不改变图的结构

因为mlp是对每一个向量独自作用的，它不会考虑所有的连接信息，所以不管对整个顶点做任何排序也好，都不会改变结果

前置知识

GCN

最后一层的输出，怎么得到预测值，首先看最简单的情况

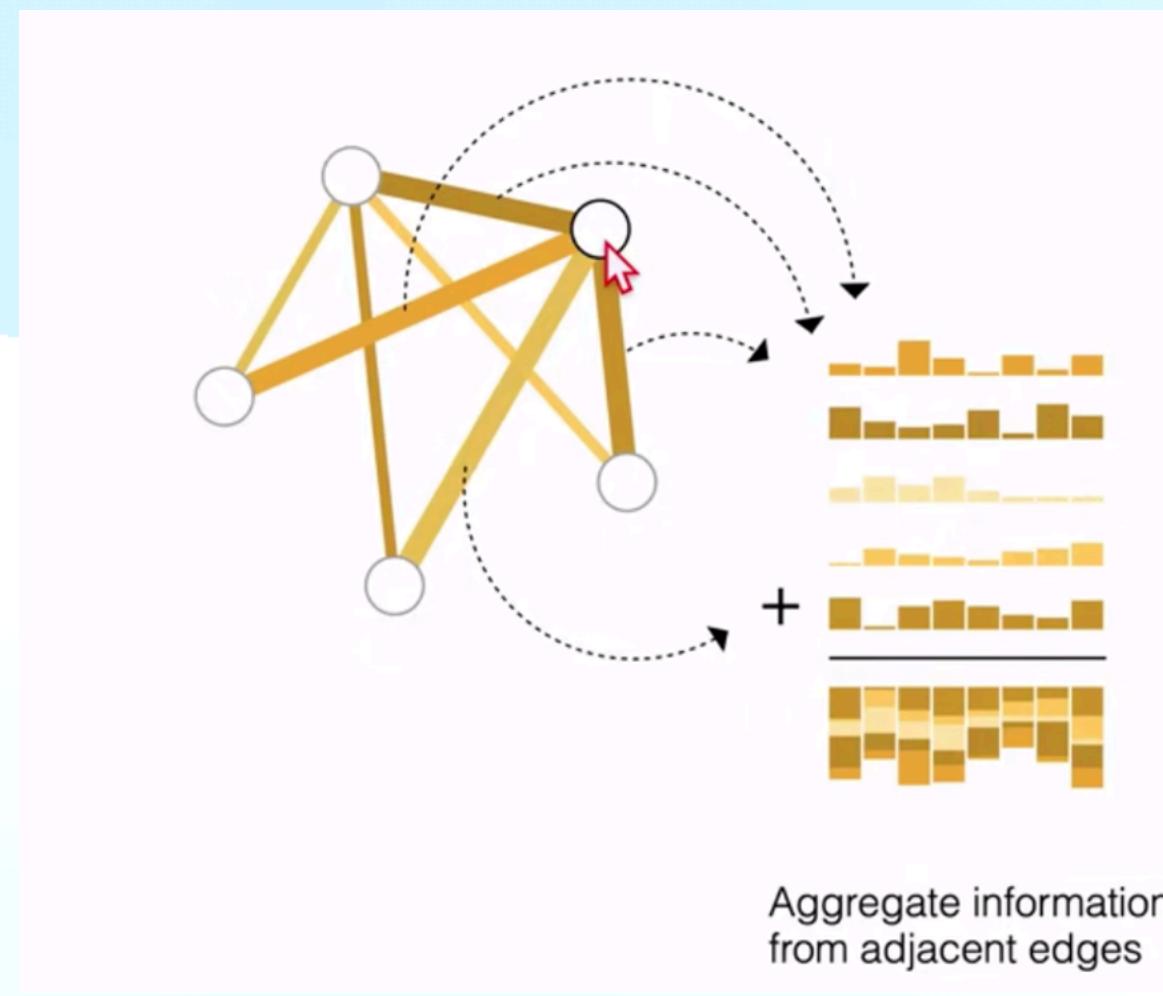


比如二分类问题，每一个顶点已经有了它的向量表示，在后面加一个输出的度为2的全连接层，然后再加一个softmax就得到了，但如果是做n类的话，就做一个输出大小是n的一个连接层，然后再加一个softmax就会得到多类输出，做回归也是一样的就是单输出就行了

前置知识

GCN

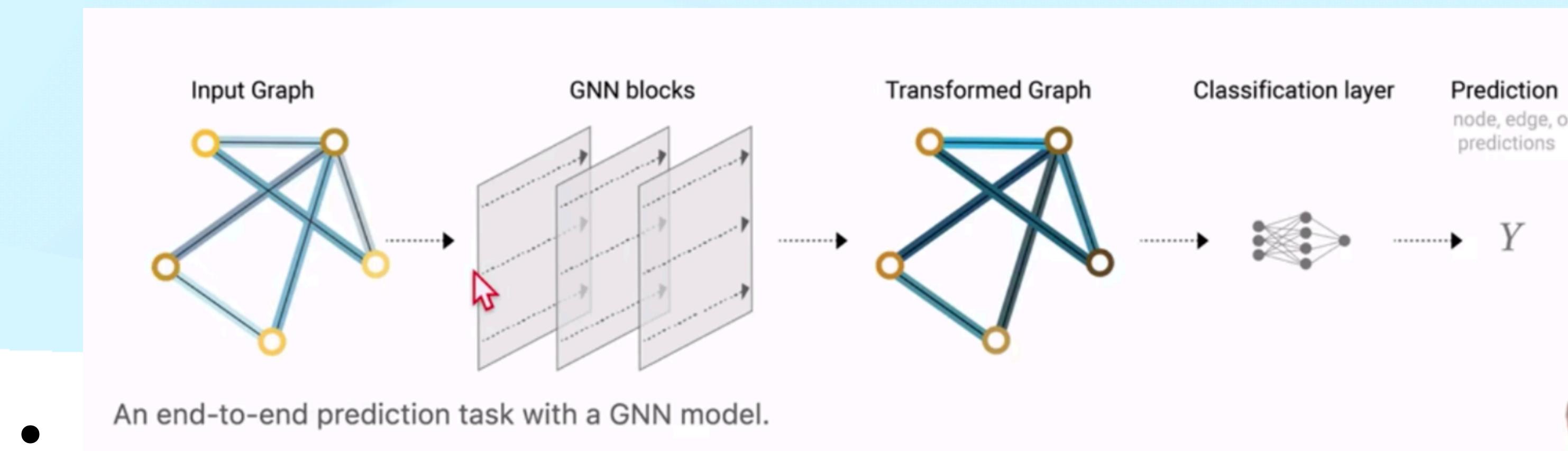
对一个顶点做预测，但是顶点并没有它的向量怎么办？这个时候我们会用到一个技术叫pooling，与cnn里面的pooling是一样的



可以把跟这个点连接的那些边的向量拿出来，同样把全局的向量拿出来，就会拿出五个向量，五个向量加起来就会得到代表这一个点的向量

前置知识

GCN



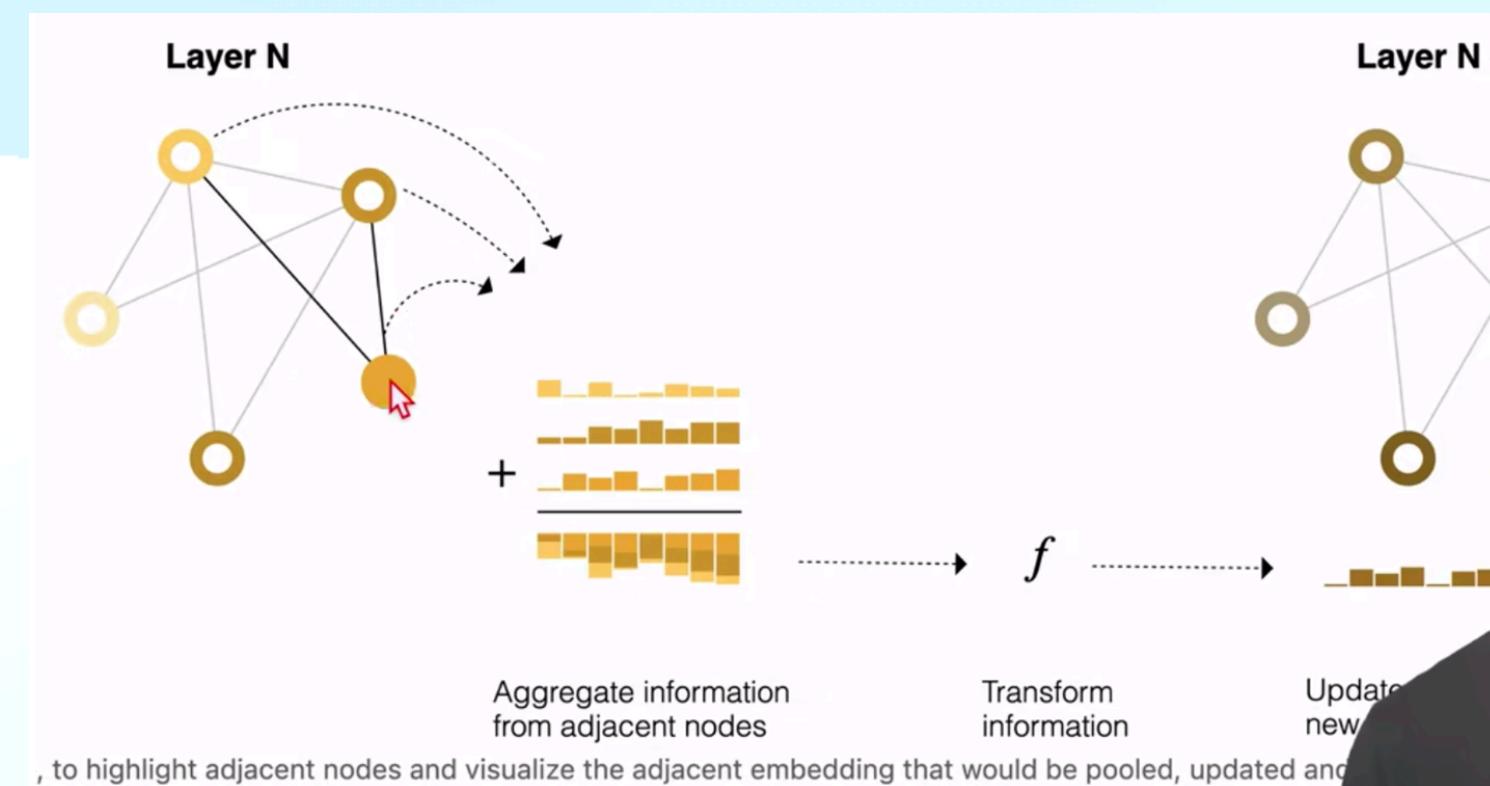
首先进入一系列的GNN层，每一个层里面就是有三个MLP, 对应的是三种不同的属性，逐一过去最后的输出会得到一个保持结构的输出，但是里面所有的属性已经发生了变化，最后要对哪一个属性做预测的话，就添加合适的输出层，如果缺失信息的话，加入合适的汇聚层，就可以完成预测

前置知识

GCN

局限性：在GNN blocks这一块的时候并没有对它使用图的结构信息，对每个属性做变换的时候，它就是每个属性进入相应的MLP，并没有说这个顶点是跟哪些边相连的，或跟哪些顶点相连的

改进：信息传递

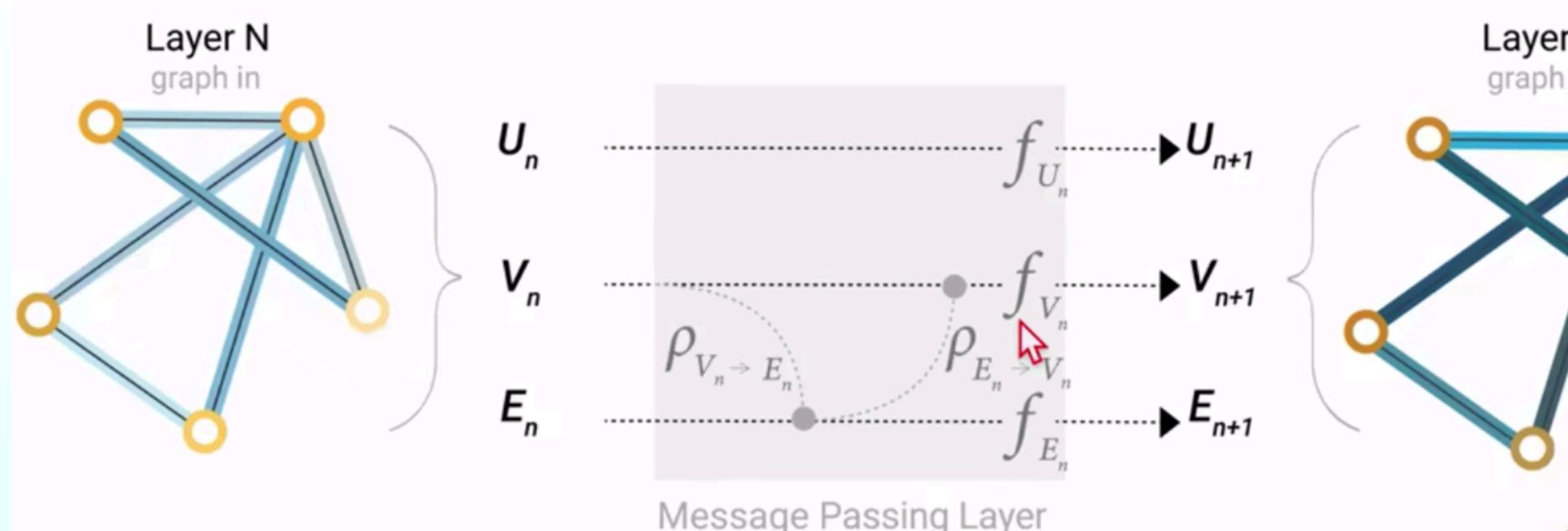
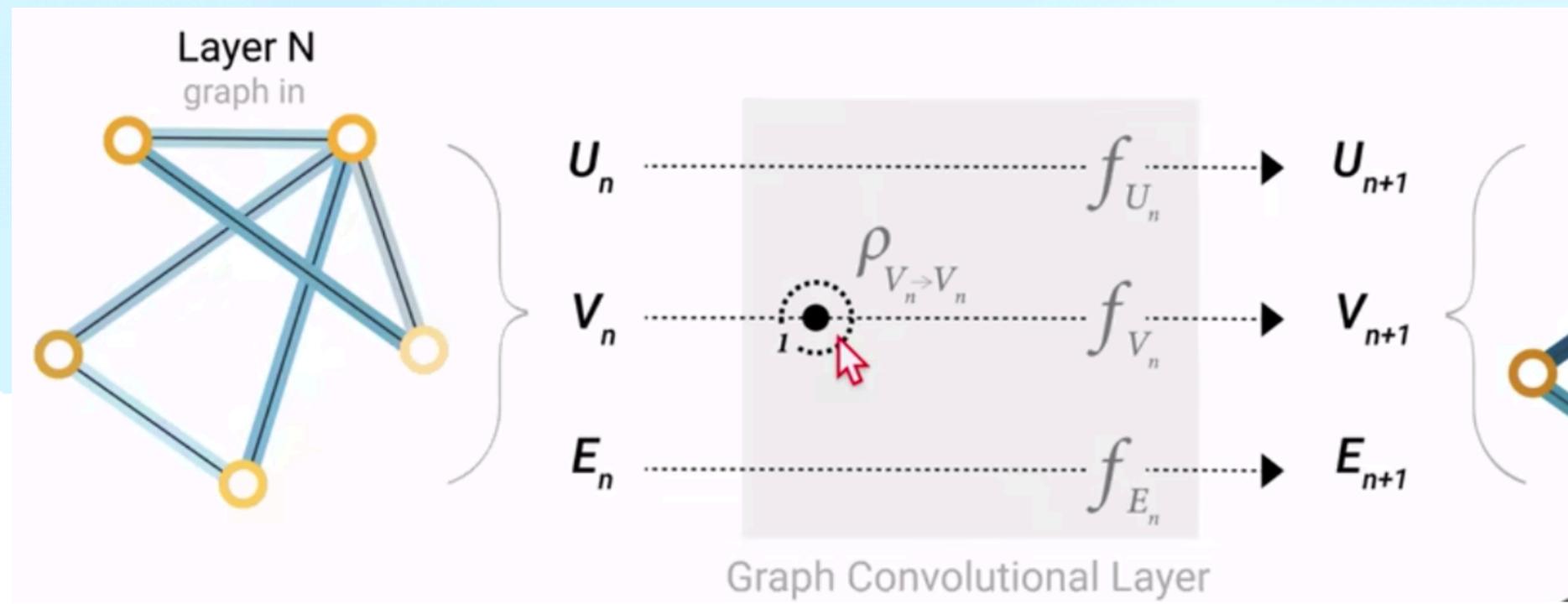


假设要对这一个顶点的向量进行更新，之前的做法是把顶点的向量拿过来然后进入到 f , 就是MLP, 直接变换得到更新的向量，但在信息传递里做了一些额外的事情，把顶点的向量和顶点邻居的两个顶点的向量都加在一起得到一个汇聚的向量，把汇聚的向量再进入MLP就会得到这个顶点的向量的更新

前置知识

GCN

就算每一层只考虑一个顶点和它的邻居的关系，但是如果有很多层放在一起的话，比如说最后一层的节点，会把图里面很多的邻居的信息全部汇聚过来，所以就完成了整个图的比较长距离的一个信息的传递（有点像attention机制）



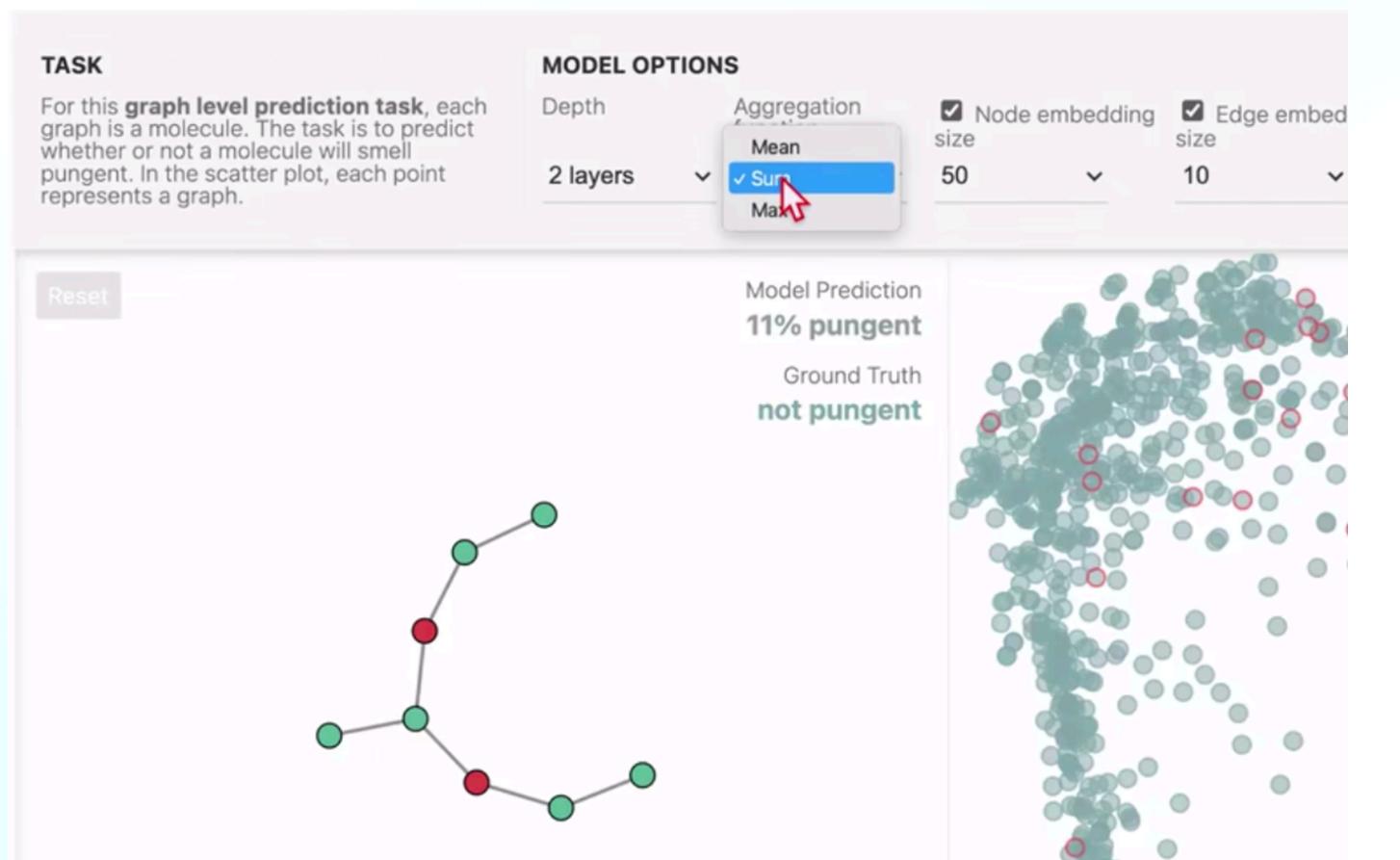
交替更新，同时顶点汇聚到边，然后边汇聚到顶点，然后汇聚过之后，先别加，进行双向传递，两边的信息都可以保留

前置知识

GCN

GNN对超参数敏感

1. 层数
2. embedding大小
3. 汇聚操作
4. 信息的传递



前置知识

GCN

采样

由于信息传递，假设有很多层的时候，最后一层的一个顶点，就算每一层里面只看它的一近邻，但最后这个顶点，因为有很多层消息传递，所以最后顶点其实能看到的是一个很大的图，假设这个图联通性足够，最后这个顶点可能看到的是整个图的信息，在计算梯度的时候，要把中间变量存下来，这样导致计算可能是无法承受的，所以需要对图进行采样，每次取一个小图出来，在小图上面做信息的汇聚，这样在计算梯度的时候，只需要在小图上计算

1. 随机采样
2. 随机游走
3. 结合1和2
4. 取一个点，一近邻，二近邻，三近邻，然后往前走 k 步把它做一个宽度遍历，然后把一个子图拿出来

论文算法存在的问题

1. 其中一个弊端是伪负标签的质量无法得到保证，可能会引入噪音，影响模型的性能。
2. 另一个弊端是由于模型的自我训练过程是基于当前模型的预测结果，因此错误的预测可能会被错误地放入训练数据中，导致模型进一步偏离真实标签。

改进思路

引入更加可靠的伪标签生成方法，例如使用**集成模型**的预测结果作为伪标签，或者通过模型的不确定性来筛选伪标签。例如，可以使用**集成模型的预测结果作为伪标签，或者通过模型的不确定性来筛选伪标签**。模型的不确定性是指模型对于给定输入的输出的置信度或可靠性度量。在自我训练中，可以使用模型的不确定性来判断哪些预测结果应该被视为可靠的伪标签。比如，**可以设定一个阈值，只将模型预测结果的不确定性低于该阈值的样本作为伪标签**。

使用**半监督学习方法**，结合标记数据和伪标签数据进行训练，以提高模型的性能和泛化能力。

对生成的伪标签进行进一步的筛选和验证，例如使用一致性训练方法或者主动学习方法来减少噪音和错误标签的影响。一致性训练方法是指利用未标记数据进行训练的方法，通过鼓励模型在不同的输入下产生相似的输出来增强模型的泛化能力。常见的一致性训练方法包括**自监督学习、生成对抗网络**等。通过使用一致性训练方法，可以减少伪标签带来的噪音和错误标签的影响，提高模型的性能。

改进思路

随机降采样+集成，在不平衡比较高时需要较多的基学习器来达到较好的效果。注意Boosting容易被噪声影响，Bagging方法是真正的万金油，增加基学习器数量效果一般不会下降。高级降采样+集成，也可以尝试，运行会慢并且效果不能保证比随机方法好。高级过采样+集成，同上，数据规模大且不平衡程度高情况下，训练样本数量爆炸。尤其是集成方法还要训练好多个基学习器。BalanceCascade，信息利用效率高，只用很少的基学习器就能达到较好的效果，但对噪声不鲁棒。

改进思路总结

1. GAN生成数据样本，生成大量负样本
2. 利用bagging算法预测伪标签，并行计算加速训练过程

改进思路总结

GAN

GAN的原理基于对抗性训练的思想，由两个网络组成：**生成器**和**判别器**。生成器尝试生成看起来可信的数据，而判别器则尝试将生成器生成的看起来可信但实际上不可信的数据与真实数据进行**对比**。生成器在开始训练的几次中会生成明显不可信的数据，但随着训练的进行，生成器逐渐具备了能够产生骗过判别器的数据的能力。最终，生成器生成的假数据与真实数据难以区分，使得判别器无法判断出数据究竟是真实的还是生成的。这种对抗性训练过程可以使得生成器生成的假数据尽可能地接近真实数据。

在训练过程中，**生成网络G的目标就是尽量生成真实的图片去欺骗判别网络D**。而D的目标就是尽量把G生成的图片和真实的图片分别开来。这样，G和D构成了一个动态的“**博弈过程**”。

改进思路总结

GAN

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))].$$

•

- 整个式子由两项构成。x表示真实图片，z表示输入G网络的噪声，而G(z)表示G网络生成的图片。
- D(x)表示D网络判断真实图片是否真实的概率（因为x就是真实的，所以对于D来说，这个值越接近1越好）。而D(G(z))是D网络判断G生成的图片的是否真实的概率。
- G的目的：上面提到过，D(G(z))是D网络判断G生成的图片是否真实的概率，G应该希望自己生成的图片“越接近真实越好”。也就是说，G希望D(G(z))尽可能得大，这时V(D, G)会变小。因此我们看到式子的最前面的记号是min_G。
- D的目的：D的能力越强，D(x)应该越大，D(G(x))应该越小。这时V(D, G)会变大。因此式子对于D来说是求最大(max_D)

改进思路总结

GAN

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by **ascend**ing its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log (1 - D(G(\mathbf{z}^{(i)}))) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by **descend**ing its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(\mathbf{z}^{(i)}))).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

随机梯度下降法训练D和G

第一步我们训练D，D是希望 $V(G, D)$ 越大越好，所以是**加上梯度**(ascending)。第二步训练G时， $V(G, D)$ 越小越好，所以是**减去梯度**(descending)。整个训练过程交替进行。

改进思路总结

Boosting与Bagging比较

Boosting和Bagging都是集成学习中常用的技术，以下是它们的一些常见机器学习算法：

1. Boosting算法：

- AdaBoost：通过将多个弱分类器组合在一起，根据分类器的错误率来调整其权重，最终得到一个强分类器。
- Gradient Boosting：通过梯度下降法来优化损失函数，每次迭代都试图纠正之前所有预测的错误。
- XGBoost：一个高效的梯度提升库，使用C++编写，具有高效、灵活和便携等特点。

1. Bagging算法：

- Random Forest：通过构建多个决策树，然后以投票的方式进行分类或回归。
- Gradient Boosting Decision Tree：通过梯度提升算法与决策树算法相结合，构建出多个决策树，然后以投票的方式进行分类或回归。
- AdaBoost Decision Tree：通过AdaBoost算法与决策树算法相结合，构建出多个决策树，然后以投票的方式进行分类或回归。

这些算法在机器学习中被广泛使用，可以用于分类、回归和异常检测等问题。

改进思路总结

Boosting与Bagging比较

Boosting和Bagging都是集成学习中常用的技术，但它们在以下五个方面存在一些差异：

1. 样本选择：

- Bagging：在训练集中进行**有放回的随机抽样**，从原始集中选出的各轮训练集之间是**独立的**。
- Boosting：每一轮的训练集不变，只是训练集中每个样例在分类器中的权重发生变化，而权值是根据上一轮的分类结果进行调整的。

1. 样本权重：

- Bagging：**所有样本的权重相等**。
- Boosting：根据错误率不断调整样例的权值，错误率越大则权重越大。

1. 预测函数：

- Bagging：**所有预测函数的权重相等**。
- Boosting：每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重。

1. 并行计算：

- Bagging：**各个预测函数可以并行生成**。
- Boosting：各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果。

1. 适用范围：

- Bagging适用于**分类和回归问题**，而Boosting主要适用于分类问题。

总的来说，Boosting和Bagging在样本选择、样本权重、预测函数、并行计算和适用范围等方面存在明显的差异。选择使用哪种技术取决于具体的问题和应用场景。