



**综合拓展报告**

**变分自编码器的演变由来及其**

**在因果推断中的应用**

学 院 计算机学院

专 业 软件工程

年级班别 2018级（5）班

学 号 3118005164

学生姓名 杨泽勤

指导教师 刘鹏

2021年 10 月

**变分自编码器的演变由来及其在因果推断中的应用**

**杨泽勤**

**计算机学院**

# 摘 要

在机器学习领域，密度估计是一类无监督学习问题，其任务是根据数据集来估计概率密度函数。然而有的时候直接对概率密度函数建模是非常困难的，因此我们通常通过引入隐变量来简化模型。如果要建模含有隐变量的分布，传统方法是使用EM算法，当E步的后验概率无法计算的时候，通常引入变分推断，得到变分EM算法，再利用神经网络对分布进行建模，就可以得到变分自编码器。本文的第一部分工作就是基于变分推断的视角，从EM算法出发，一步步导出变分自编码器。

近年来，基于观察数据估计因果效应已成为一个更具有吸引力的研究课题。潜在结果框架是目前应用最为广泛的因果推断框架之一，该框架要回答的核心问题是一个反事实估计的问题。然而，现有的大部分在潜在结果框架下进行因果推断都是在满足SUTVA、ignorability和positivity三个假设的前提下提出的。在实践中，第二个假设往往很难成立。本文的第二部分工作就是以变分自编码器为基础，当存在未观察到的混杂因素的时候，利用代理变量恢复混杂并估计因果效应。

**关键词：**EM算法，变分推断，变分自编码器，因果效应估计

# Abstract

In the field of machine learning, density estimation is a type of unsupervised learning problem. Its task is to estimate the probability density function based on the dataset. However, sometimes it is very difficult to directly model the probability density function , so we usually simplify the model by introducing a hidden variable . If we want to model the distribution with hidden variables, the traditional method is to use the EM algorithm. When the posterior probability of E step cannot be calculated, the variational inference is usually introduced to obtain the variational EM algorithm, and when the neural network is used to model the distribution, we can get the variational autoencoder. The first part of the work is based on the perspective of variational inference, starting from the EM algorithm, step by step derivation of the variational autoencoder.

In recent years, estimating causal effects based on observational data has become a more attractive research topic. The potential outcome framework is currently one of the most widely used causal inference frameworks. The core question that the framework needs to answer is a counterfactual estimation problem. However, most of the existing causal inferences under the framework of potential outcomes are proposed under the premise of satisfying the three assumptions : SUTVA, ignorability and positivity. In practice, the second assumption is often difficult to establish. The second part of this paper is based on the variational autoencoder, use proxy variables to restore confounding and estimate causal effects when there exists unobserved confounding factors.

**Keywords:** EM algorithm, variational inference, variational autoencoder, causal effect estimation

**目 录**

[1设计任务 5](#_Toc86699873)

[1.1 从EM到变分自编码器的演变 5](#_Toc86699874)

[1.2 变分自编码器在因果推断中的应用 5](#_Toc86699875)

[2综合拓展的时间计划安排 7](#_Toc86699876)

[3网络详细设计 8](#_Toc86699877)

[3.1 变分自编码器的网络设计 8](#_Toc86699878)

[3.2 因果效应变分自编码器的网络设计 8](#_Toc86699879)

[4代码实现 11](#_Toc86699880)

[4.1 变分自编码器实现 11](#_Toc86699881)

[4.2 因果效应变分自编码器代码实现 13](#_Toc86699882)

[5模型测试及相关问题分析 22](#_Toc86699883)

[5.1 变分自编码器 22](#_Toc86699884)

[5.2 因果效应变分自编码器 24](#_Toc86699885)

[6总结与展望 25](#_Toc86699886)

[7参考文献 26](#_Toc86699887)

[8致 谢 27](#_Toc86699888)

# 1设计任务

## 1.1 从EM到变分自编码器的演变

在机器学习中，密度估计是一类无监督学习问题：给定一组数据，假设它们都是独立地从相同的概率密度函数中产生的。密度估计的任务就是根据数据集来估计其概率密度函数。举个例子，在手写体数字图像的密度估计中，我们将图像表示为一个随机变量，其中每一维表示一个像素值。假设手写体数字图像服从一个未知分布，希望通过一些观测样本来估计该分布。然而手写体数字图像的不同像素之间存在复杂的依赖关系，很难用一个图模型来描述其依赖关系，所以直接对建模是非常困难的。因此我们通常通过引入隐变量来简化模型，如此密度估计问题可以转换为估计变量的两个局部条件概率和。一般为了简化模型，假设隐变量的先验分布是标准高斯分布且隐变量的每一维之间是独立的，在这个假设下先验分布不存在参数，因此密度估计的重点是估计条件分布。

如果要建模含有隐变量的分布，一般利用EM算法来进行密度估计。在EM算法中，需要估计条件分布和后验分布。问题是，当后验分布无法计算的时候，如何利用变分推断求取近似后验分布？当这两个分布比较复杂的时候，如何利用神经网络来进行建模？对于第一个问题，答案是变分EM算法；对于第二个问题，答案是变分自编码器。因此，此综合拓展的第一个任务就是理清楚从EM到变分EM，再到变分自编码器的演变过程，主要是对变分自编码器VAE进行代码实现。

## 1.2 变分自编码器在因果推断中的应用

相比于传统通过进行随机对照试验来估计因果效应，如何基于观察数据估计因果效应已成为一个更具有吸引力的研究课题。潜在结果框架是目前应用最为广泛的因果推断框架之一，该框架要回答的核心问题是一个反事实估计的问题，例如我们想回答“如果这个病人当初接受了另外一种药物治疗，他会有不同的结果吗？” 回答这些反事实的问题是具有挑战性的，有两个原因：一是我们只观察到了事实结果，而不是如果他们选择了不同的治疗方案，可能发生的反事实结果；二是在观察数据中，治疗策略通常不是随机分配的，这可能导致治疗组的协变量分布相较于对照组发生了分布偏移，我们称这种情况为存在混杂。

得益于机器学习、深度学习领域的快速发展，各种基于观测数据的因果效应估计方法已经涌现出来。首先，针对于反事实估计问题，借助诸如决策树、集成方法、深度神经网络等强大的模型方法，我们可以更准确地估计反事实结果；其次，机器学习方法也为处理混杂提供了一个新的视角：受益于深度表征学习方法的发展，我们可以通过学习所有协变量的平衡表征来调整混杂变量，使得在学习到的平衡表征的条件下，治疗分配策略独立于混杂。

然而，现有的大部分基于观察数据的因果效应估计方法都是在满足SUTVA、ignorability和positivity三个假设的前提下提出的。然而在实践中，针对于第二个假设，我们往往是无法保证的。那么当存在未观察到的混杂因素的时候，如何基于变分自编码器，利用代理变量恢复混杂并估计因果效应？这就是综合拓展的第二个任务，这份报告主要是对因果效应变分自编码器CEVAE进行代码实现。

报告的结构如下：第二节给出专项设计的时间计划安排；第三节给出VAE和CEVAE网络的具体设计，包括网络的层数、隐层的维度、激活函数等；第四节给出VAE和CEVAE的代码实现；第五节给出VAE和CEVAE的训练结果、在这个过程中发现的一些问题及其分析；第六节是总结。

# 2综合拓展的时间计划安排

**表2.1 时间计划安排表**

|  |  |
| --- | --- |
| 时间 | 任务 |
| 10月10号~10月14号 | 理解VAE的网络设计思路 |
| 10月15号~10月18号 | 理解CEVAE的网络设计思路 |
| 10月19号~10月20号 | VAE的代码编写 |
| 10月21号~10月23号 | CEVAE的代码编写 |
| 10月25号~10月27号 | 调节CEVAE的超参数，  尽量复现论文结果 |
| 10月28号~10月30号 | 报告撰写 |

# 3网络详细设计

## 3.1 变分自编码器的网络设计

变分自编码器的网络结构如下图所示：

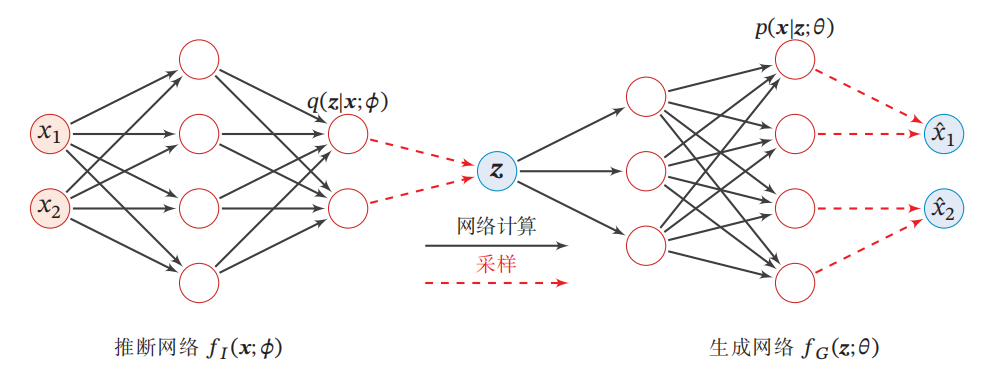


图3.1 变分自编码器网络结构图

推断网络利用神经网络来估计变分分布，具体地，搭建三层神经网络。第一层是卷积层，卷积核的kernel\_size=4，stride=2，padding=1，通道数由1变成64；第二层是卷积层，卷积核的kernel\_size=4，stride=2，padding=1，通道数由64变成128；第三层有两个全连接层，分别输出隐变量分布的均值和方差，两个全连接层的in\_features=128\*7\*7，out\_features=2。

生成网络利用神经网络来估计概率分布，具体地，搭建三层神经网络，第一层是全连接层，in\_features=2，out\_features=128\*7\*7；第二层是反卷积层，卷积核的kernel\_size=4，stride=2，padding=1，通道数由128变成64；第三层是反卷积层，卷积核的kernel\_size=4，stride=2，padding=1，通道数由64变成1。

## 3.2 因果效应变分自编码器的网络设计

因果效应变分自编码器的网络结构如下图所示：

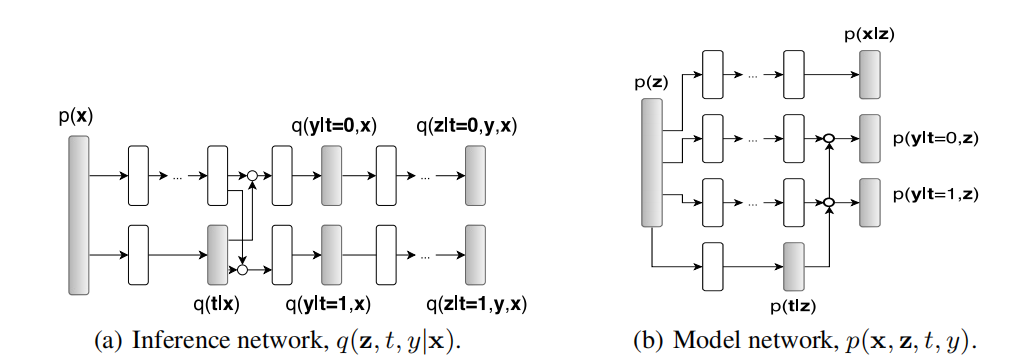


图3.2 因果效应变分自编码器网络结构图

推断网络由三个神经网络组成，分别是, ，，具体地：

由三个全连接层组成，第一层全连接层的in\_features=25, out\_features=64,第二个全连接层的in\_features=64,out\_features=64,第三个全连接层的in\_features=64,out\_features=1；

由四个全连接层组成，前两个全连接层是共用的，后两个分别代表t=1和t=0的分支。具体地，第一层全连接层的in\_features=25, out\_features=64,第二层全连接层的in\_features=64,out\_features=64,第三层全连接层表示t=0的分支，in\_features=64,out\_features=1，第四层全连接层表示t=1的分支，in\_features=64,out\_features=1；

由六个全连接层组成，前两个全连接层是共用的，第三个全连接层输出t=0的分支的均值，第四个全连接层输出t=1的分支的均值，第五个全连接层输出t=0的分支的方差，第五个全连接层输出t=1的分支的方差。具体地,第一层全连接层的in\_features=25,out\_features=64，第二层全连接层的in\_features=64, out\_features=64, 第三层全连接层的in\_features=64, out\_features=1,第四层全连接层in\_features=64, out\_features=1, 第五层全连接层的in\_features=64, out\_features=1, 第六层全连接层的in\_features=64, out\_features=1。

同样地，生成网络也由三个神经网络组成，分别是, 和，具体地：

由五个神经网络组成，前两个全连接层是共用的，第三个全连接层表示离散输出，第四个全连接层表示连续输出的均值，第五个全连接层表示连续输出的方差。具体地，第一层全连接层的in\_features=25, out\_features=64,第二层全连接层的in\_features=64,out\_features=64, 第三层全连接层的in\_features=64,out\_features=20,第四层全连接层的in\_features=64,out\_features=25,第五层全连接层的in\_features=64,out\_features=25；

由三个神经网络组成，第一层全连接个的in\_features=25, out\_features=25,第二个全连接层的in\_features=64,out\_features=64,第三个全连接层的in\_features=64,out\_features=1；

由六个神经网络组成，前三个神经网络对应t=0的分支，分别代表输入、隐层和均值，后三个神经网络对应t=1的分支,分出代表输入、隐层和均值。具体地,第一层全连接层in\_features=25,out\_features=64, 第二层全连接层的in\_features=64,out\_features=64, 第三层全连接层的in\_features=64,out\_features=1,第四层全连接层的in\_features=25,out\_features=64,第五层全连接层的in\_features=64,out\_features=64,第六层全连接层的in\_features=64,out\_features=1。

# 4代码实现

## 4.1 变分自编码器实现

数据加载函数的实现如图4.1所示。读取的是MNIST数据集，读取数据集后，使用transforms.ToTensor()方法对像素值进行了归一化。

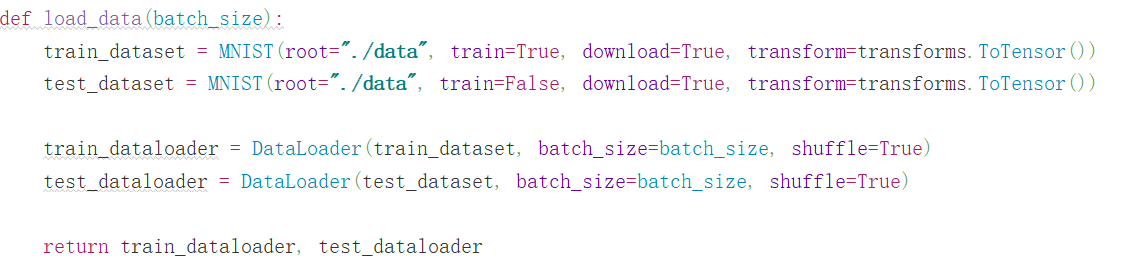


图4.1 数据加载代码实现

变分自编码器，即VAE类的实现如图4.2所示。在\_\_init\_\_()方法中encoder对应推断网络，decoder对应生成网络；forward()方法定义前向传播的过程，输入数据先经过encoder得到隐变量的均值和方差，通过重参数化技巧采样得到隐变量，最后隐变量通过decoder重构输入数据；reparameterize()方法实现的是重参数化过程。

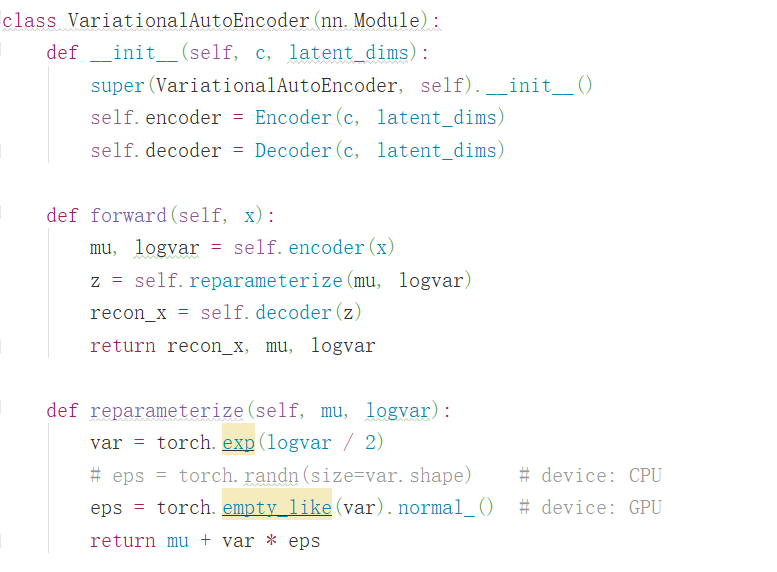


图4.2 类变分自编码器的代码实现

具体地，Encoder和Decoder类的代码分别如图4.3和4.4所示。Encoder将输入数据经过两层卷积层之后，再分别经过各自的全连接层得到隐变量对应的均值和方差；Decoder的前向传播过程恰好与Encoder相反，先将采样得到的隐变量经过一层全连接层，再通过两层反卷积层。



图4.3 类Encoder的代码实现

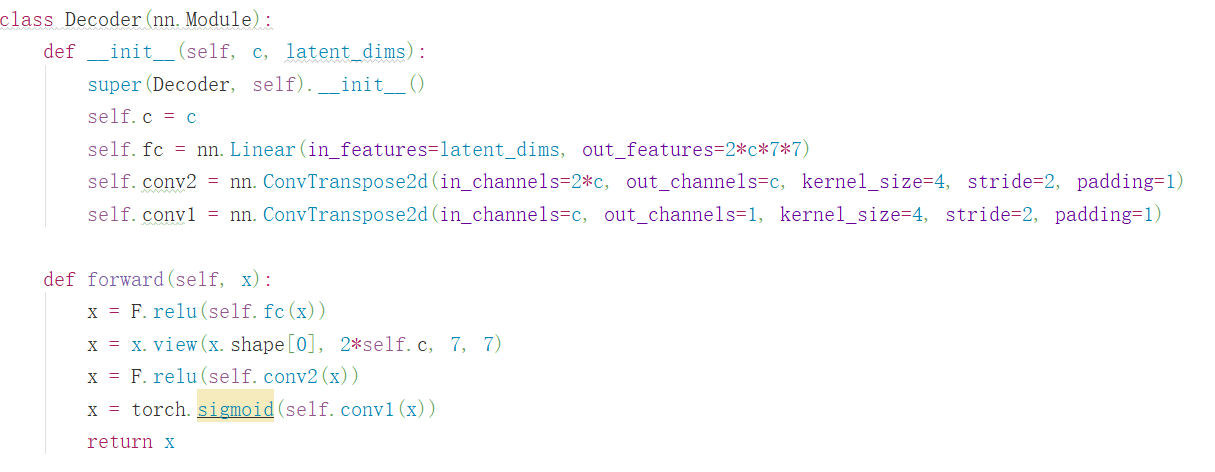


图4.4 类Decoder的代码实现

损失函数的实现如图4.5所示。损失函数由重构损失和KL损失构成，这里重构损失采用的是BCE损失函数，KL损失的计算在假设隐变量空间服从高斯分布后可进行花化简。

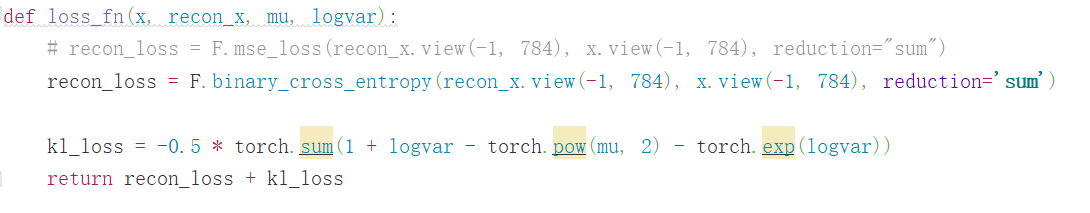


图4.5 损失函数的代码实现

模型训练的函数实现如图4.6所示。在每一轮epoch的循环中，从train\_dataloader中加载一批新数据，输入VAE进行前向传播后得到重构的数据，隐变量的和方法，基于上述结果计算损失后，进行反向传播，从而实现神经网络的参数更新。



图4.6 模型训练的代码实现

## 4.2 因果效应变分自编码器代码实现

这里采用IHDP数据集，该数据集是根据婴儿健康与发展计划进行的随机对照实验生成的，其个体对象是低出生体重早产儿。一共有25个协变量，包括儿童及其母亲的各个方面的数据；在treatment组，为婴儿提供了高质量的儿童看护和经常的专家家教；衡量的结果是婴儿的认知测试得分。该数据集一共有30列，前5列分别是治疗策略，事实结果，反事实结果以及无噪声的潜在结果，后25列是协变量。

将数据加载封装成IHDP类，实现如图4.7所示。\_\_init\_\_()方法中的self.binfeats和self.contfeats分别是离散型协变量和连续型协变量的索引；\_\_iter\_\_()方法是读取数据集；get\_train\_valid\_test()方法的作用是划分训练集、测试集和验证集。

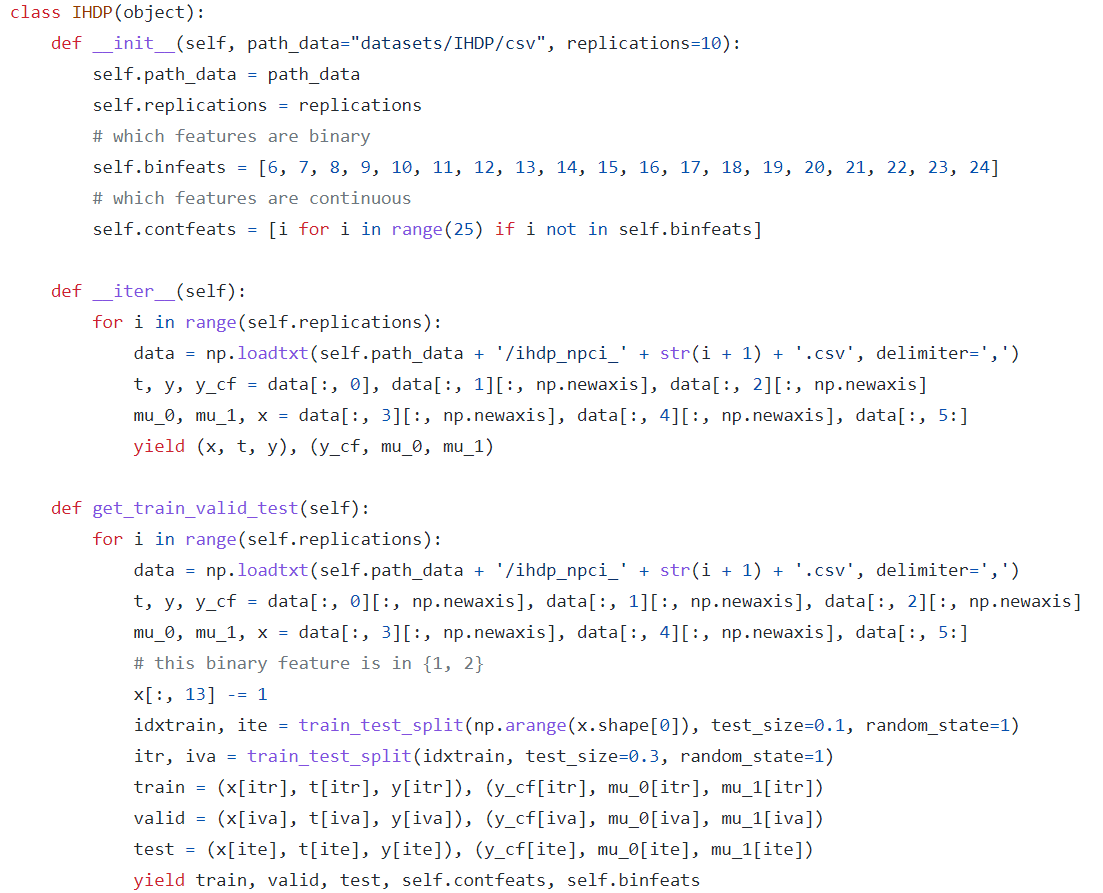


图4.7 数据加载类的代码实现

CEVAE的推断网络包括q\_t\_x，q\_y\_xt以及q\_z\_tyx三部分。具体地：

q\_t\_x的代码实现如图4.8所示，该网络基于输入的协变量x预测二元治疗策略treatment，前向传播过程是将x经过两层全连接层，这两层的激活函数是elu函数，前两层全连接层的输出进入第三层全连接层，第三层的激活函数是sigmoid，经过sigmoid函数映射之后的最后将sigmoid函数输出的概率作为伯努利分布的参数。

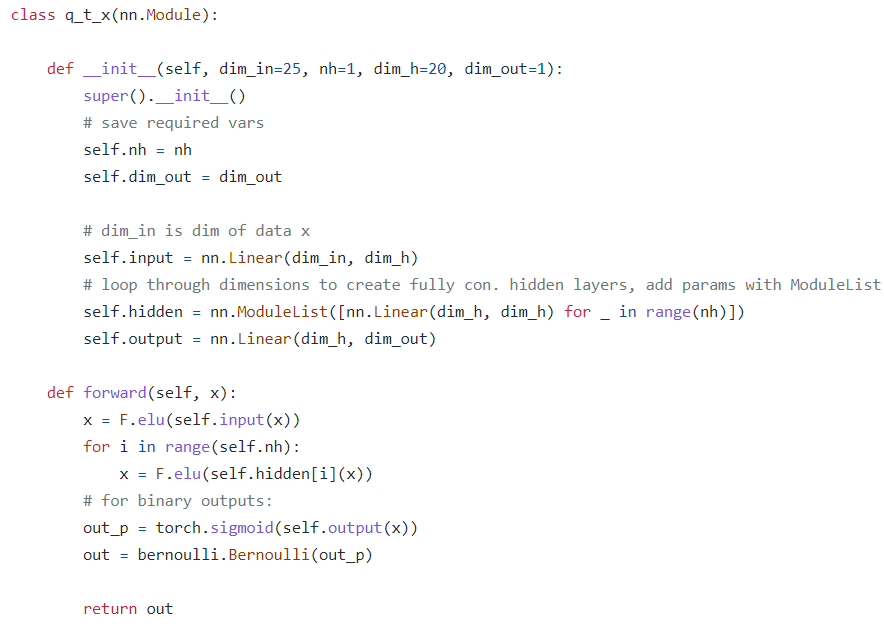


图4.8 q\_t\_x的代码实现

q\_y\_xt的代码实现如图4.9和4.10所示。该网络基于输入的协变量x和治疗策略t预测输出y，前向传播过程是将x经过4层全连接层，每层全连接层的激活函数都是elu函数；之后将输出通过self.mu\_t0和self.mu\_t1分别得到t=0和t=1时对应的输出mu\_t0和mu\_t1；最后如果当前个体处于治疗组，则将mu\_t1作为y所服从的正态分布的均值，反之将mu\_t0作为该正态分布的均值。

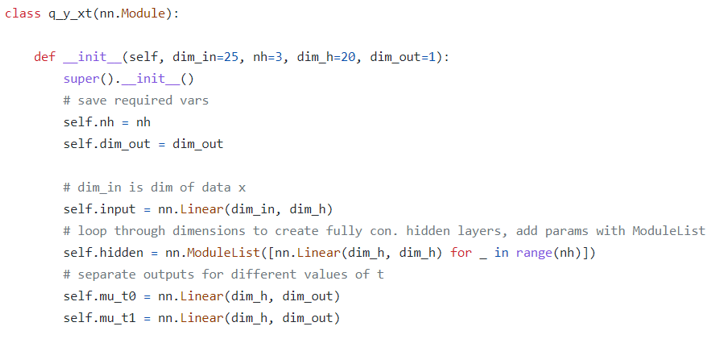


图4.9 q\_y\_xt的\_\_init\_\_部分代码实现

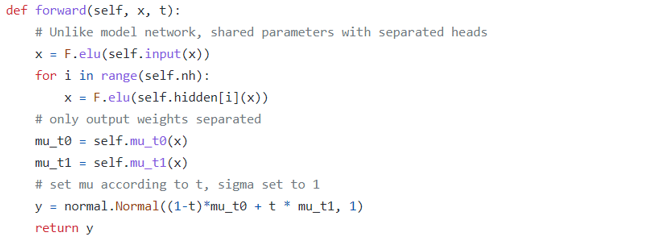


图4.10 q\_y\_xt的forward部分代码实现

q\_z\_tyx的代码实现如图4.11和4.12所示。该网络基于输入的协变量x、治疗策略t和预测输出y预测隐变量z，前向传播过程是将xy经过4层全连接层，每层全连接层的激活函数都是elu函数；之后将输出通过self.mu\_t0和self.mu\_t1分别得到t=0和t=1时对应的输出mu\_t0和mu\_t1，表示隐变量z所服从的高斯分布的均值；同时将前四层全连接层的输出通过self.sigma\_t0和self.sigma\_t1分别得到t=0和t=1时对应的输出sigma\_t0和sigma\_t1，表示隐变量z所服从的高斯分布的方差；最后基于该个体所接受的治疗策略选择对应的均值和方法。

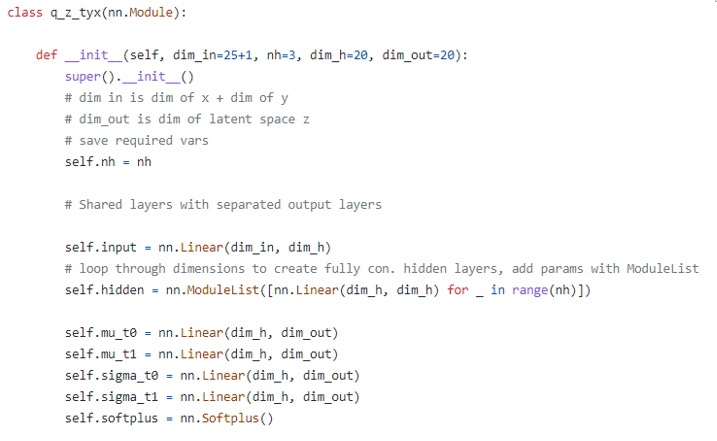


图4.11 q\_z\_tyx的\_\_init\_\_部分代码实现

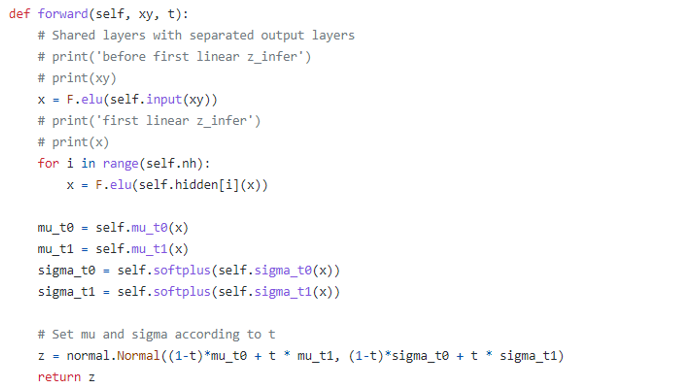


图4.12 q\_z\_tyx的forward部分代码实现

CEVAE的生成网络部分包括p\_x\_z，p\_t\_z以及p\_y\_zt三部分。具体地：

p\_x\_z的代码实现如图4.13和4.14所示。该网络基于推断网络生成的隐变量z去恢复出协变量x；前向传播过程是将z经过三层全连接层得到新的z，每层全连接层的激活函数都是elu函数；之后将新z经过一层全连接层并通过sigmoid激活函数，得到伯努利分布的参数，基于伯努利分布采样即可恢复离散型协变量；同理得到高斯分布的均值和方差，基于高斯分布采样即可恢复连续型协变量。

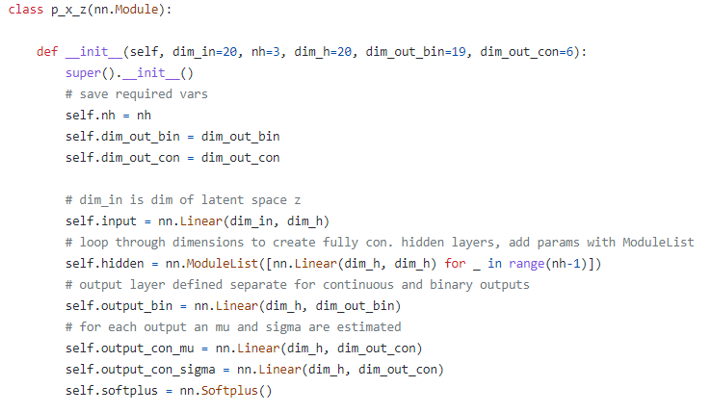


图4.13 p\_x\_z的\_\_init\_\_部分代码实现

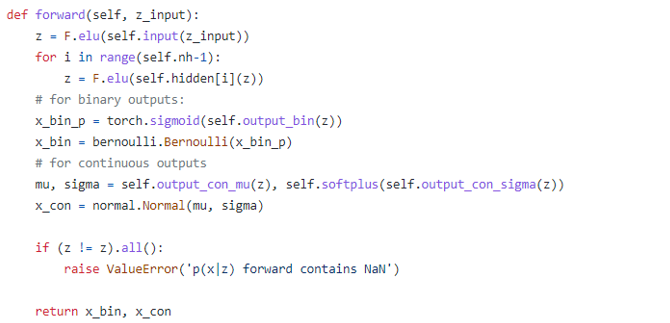


图4.14 p\_x\_z的forward部分代码实现

p\_t\_z的代码实现如图4.15所示，该网络基于推断网络生成的隐变量z去恢复出治疗策略t；前向传播过程是将z经过两层全连接层得到新的z，每层全连接层的激活函数都是elu函数；之后将新z经过一层全连接层并通过sigmoid激活函数得到伯努利分布的参数，基于伯努利分布采样即可恢复治疗策略t。

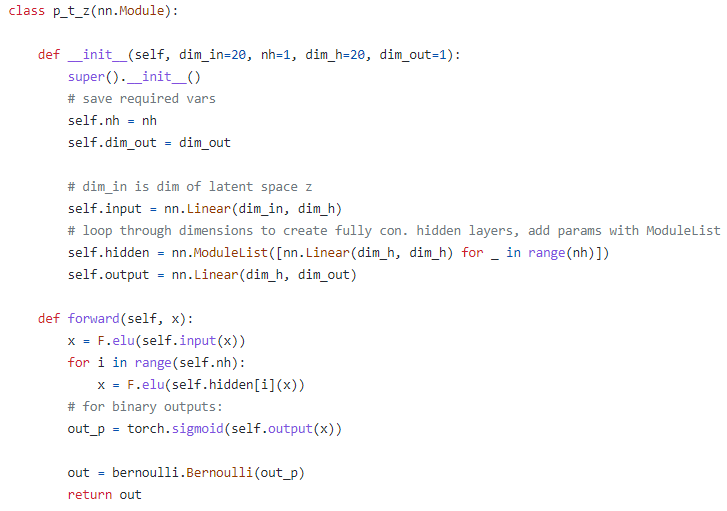


图4.15 p\_t\_z的代码实现

p\_y\_zt的实现如图4.16所示。该网络基于推断网络生成的隐变量z去恢复出输出y；前向传播过程是将z经过四层全连接层得到新的x\_t0，每层全连接层的激活函数都是elu函数；之后将x\_t0经过一层全连接层并通过elu激活函数得到当treatment=0时，y所服从的高斯分布的均值；同理可得treatment=1时，y所服从的高斯分布的均值。

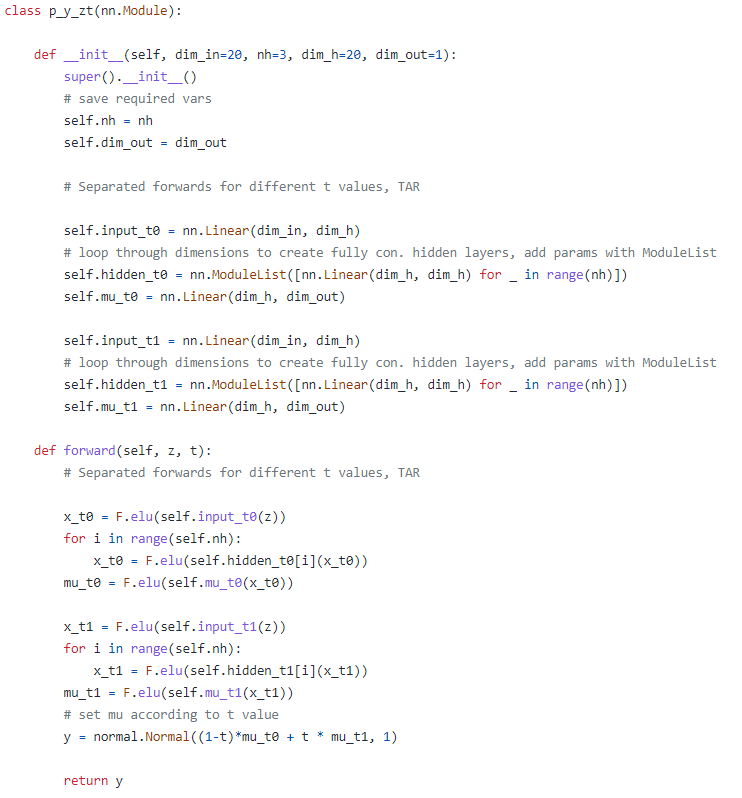


图4.16 p\_y\_zt的代码实现

CEVAE的损失函数由重构误差reconstruction loss、正则化误差regularization loss(KL散度)和辅助误差auxiliary loss三部分组成。具体地：

reconstruction loss的计算公式为，代码实现如图4.17所示：

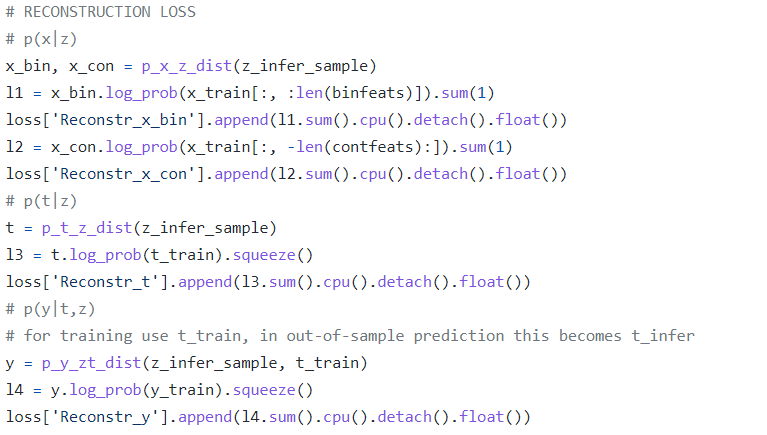


图4.17 reconstruction loss的计算

regularization loss的计算公式为，代码实现如图4.18所示：

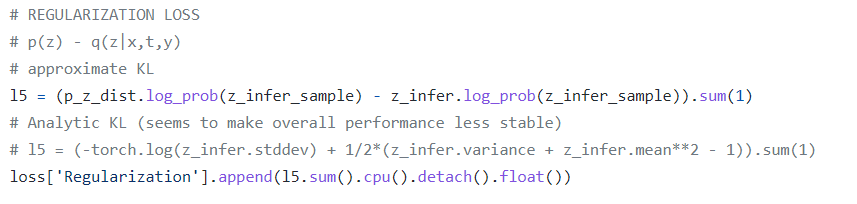


图4.18 regularization loss的计算

auxiliary loss的计算公式为，代码实现如图4.19所示：

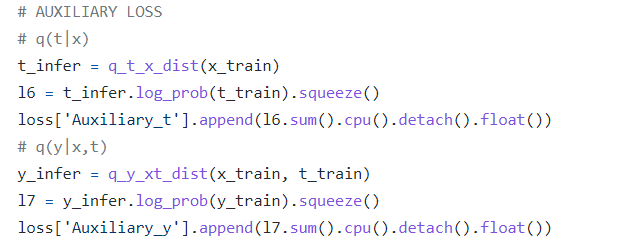


图4.19 auxiliary loss的计算

在每轮epoch中，利用q\_t\_x，q\_y\_xt，q\_z\_tyx，p\_x\_z，p\_t\_z以及p\_y\_zt六个网络的前向传播结果计算得到reconstruction loss，regularization loss(KL散度)和auxiliary loss，再求和得到total loss，之后基于total loss反向传播更新神经网络的参数。这部分的代码如图4.20所示：

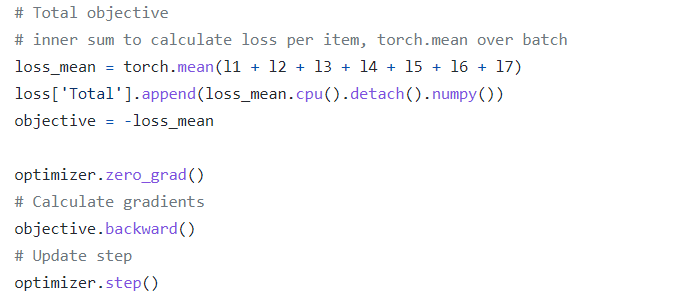


图4.20 total loss的计算

# 5模型测试及相关问题分析

## 5.1 变分自编码器

我使用MNIST数据集来测试实现的VAE的效果。具体地，在训练过程中，设置batch\_size=128；epochs=20；隐变量z的维度是2；优化方式采用学习率为0.001的Adam。

当重构误差的损失函数使用BCE 损失函数的时候，在二维空间，将x的范围设置在[-1.5, 1.5]并切分成20段；将y的范围设置在[-1.5, 1.5]并切分成20段。如此，在这400个空间内基于VAE生成数字图像的结果如图5.1所示：

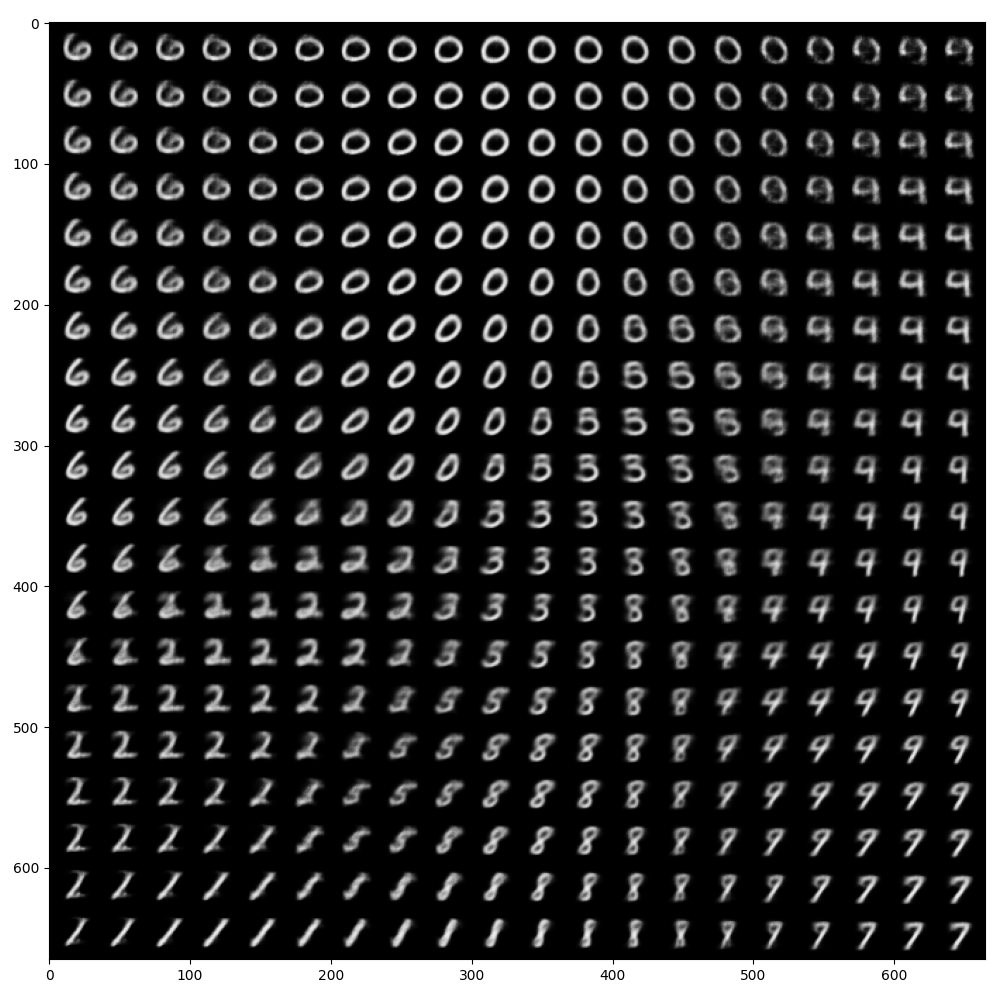


图5.1 基于BCE损失函数训练的模型结果

在实现VAE的过程中，我发现无论是论文作者的官方实现还是网上的众多版本，对于MNIST数据集，使用的都是BCE损失函数。一开始我对这一点感到十分疑惑：我们的目标是对每个像素点进行重构，每个像素点是取值范围是0到255，即使做了归一化，这也不是一个二分类的问题，为什么要使用BCE损失函数，而不是MSE损失函数呢？

于是我又用MSE损失函数作为重构误差的损失函数，其他操作不变，即在二维空间，将x的范围设置在[-1.5, 1.5]并切分成20段；将y的范围设置在[-1.5, 1.5]并切分成20段。如此，在这400个空间内基于VAE生成数字图像的结果如图5.2所示：

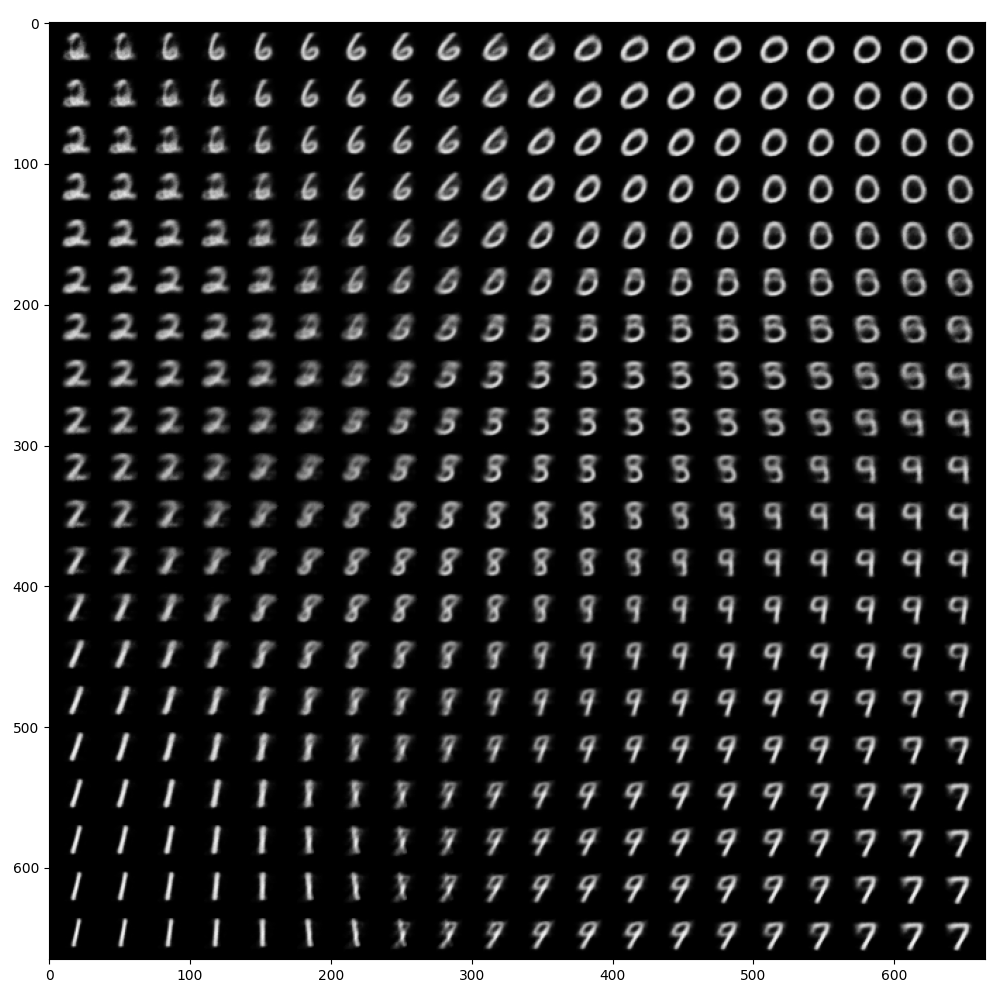


图5.2 基于MSE损失函数训练的模型结果

通过这两张图可以发现，其实无论是使用BCE损失函数还是MSE损失函数，生成模型的能力都是差不多的（可能是MNIST数据集太过于简单）。不同数字的图像的过渡都是比较圆润的，符合VAE的目的。至于为什么网上的实现都使用BCE损失函数，猜测只是将像素归一化到0到1的区间后，当作软分类的问题来看待而已。

## 5.2 因果效应变分自编码器

我使用IHDP数据集来测试实现的CEVAE的效果。具体地，在训练过程中，设置batch\_size=100；epochs=100；隐变量z的维度是20；隐层的维度是64；随着迭代次数的进行，对所有参数设置一个衰减项；优化方式采用学习率为0.01，设置了exponential decay schedule的Adamax；通过验证集的下界使用了early stopping策略。

训练的结果与原论文的结果如表5.1所示：

**表5.1 复现结果与原论文对比**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 论文结果 | 2.6 | 0.46 |
| 模型结果 | 5.8 | 4.5 |

明显发现我们自己实现的模型与原论文的结果是存在较大差异的，导致这一现象的原因可能有两个：

1. 超参数设置的原因。原论文并没有公开全部的超参数设置，因此部分超参数只能根据自己的经验去设置，而CEVAE模型又对超参数十分敏感，所以超参数的设置不当可能是一个原因；

2. 框架的原因。原论文使用的是Tensorflow框架去实现，而我自己实现的时候使用的是Pytorch框架，虽然这两个框架都是深度学习常用的框架，但是它们具体的网络模块的底层实现是存在差异的，这也可能是导致复现结果与原文差距较大的另一个原因。

# 6总结与展望

总的来说，此次综合拓展计主要分为两部分：第一部分主要是动手实现变分自编码器VAE；第二部分主要是动手实现因果效应变分自编码器CEVAE。

但仔细学习了CEVAE[5]后可以发现，其实这个模型是在已有模型的基础上稍作修改，并不复杂。但是这个工作的贡献在于作者针对于鲜有人关注的问题和难点，对问题进行形式化定义之后，仅在现有模型的基础上稍作调整，就能够很好地解决了难题。这给我的启示是：与其对现有模型盲目地进行缝缝补补，不如脚踏实地的做研究。先好好发现一个问题，接着对该问题进行形式化地定义，将其抽象成一个本质上的数学问题，最后根据实际面临的问题对现有工具进行修改去解决问题。

后续的工作方向是针对于因果效应估计领域，先进行理论上的突破，再应用到实际问题上。例如打破了假设的因果效应估计，此次专项设计涉及到的是变分自编码这个工具，但是针对具体的数据类型，能够有更加匹配的模型可以更好地解决这个问题？以打破SUTVA假设为例，本质上就是个体之间加入了“干扰”：如果是在网络数据，要解决的难点其实就是如何聚合一阶邻居、二阶邻居甚至更高阶的邻居的“干扰”信息，而这种网络中信息聚合的方式，能够基于图神经网络来完成？因为它们的目标是一致的；如果是在时序数据，循环神经网络、状态空间模型等模型能够更好地聚合“干扰”信息？这些都是待探索的事情。而一旦理论上有了进展，再落地迁移到诸如推荐系统等实际应用中就是非常容易的事情了。

# 7参考文献

[1] Ruocheng Guo, Lu Cheng, Jundong Li, P. Richard Hahn, Huan Liu. A Survey of Learning Causality with Data: Problems and Methods. ACM Computing Surveys(2020).

[2] Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, Aidong Zhang. A Survey on Causal Inference. arXiv preprint arXiv:2002.02770, 2020.

[3] David M. Blei, Alp Kucukelbir, Jon D. McAuliffe. Variational Inference: A Review for Statisticians. Journal of the American Statistical Association(2017).

[4] D. P. Kingma and M. Welling. Auto-encoding variational bayes. International Conference on Learning Representations. In ICLR'14.

[5] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. In NIPS’17.

[6] Vineeth Rakesh, Ruocheng Guo, Raha Moraffah, Nitin Agarwal, Huan Liu. Linked Causal Variational Autoencoder for Inferring Paired Spillover Effects. In CIKM’18.

[7] M. Kuroki and J. Pearl. Measurement bias and effect restoration in causal inference. Technical report, DTIC Document, 2011.

[8] J. Pearl. On measurement bias in causal inference. arXiv preprint arXiv:1203.3504, 2012.

[9] U.Shalit, F.Johansson, and D.Sontag. Estimating individual treatment effect: generalization bounds and algorithms. ArXiv e-prints, June 2016.

# 8致 谢

本设计是在我的指导教师刘鹏老师的亲切关怀和悉心指导下完成的。他严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我。从题目的选择到最终完成，刘鹏老师都始终给予我细心的指导和不懈的支持。

感谢刘鹏老师在课题研究初期给予的帮助，在刘鹏老师的帮助下课题才得以很好的展开，有了一个很好的开端！在本设计完成过程中，本人得到了刘鹏老师和许多师兄的热心帮助，本人向他们表示深深的谢意！