

一、变分自编码器生成MNIST 手写数字（结合代码描述实现步骤以及提交下面要求提交的结果）

推荐使用高斯分布随机初始化模型参数，可以避免一部分模式坍塌问题。

1、模型架构：

① 编码器（全连接层）：

输入图片维度：784 (28×28)

输出层维度（ReLU）400

② 生成均值（全连接层）：

输入层维度：400

输出层维度：20

③ 生成标准差（全连接层）：

输入层维度：400

输出层维度：20

④ 使用均值和标准差生成隐变量 z

⑤ 解码器（全连接层）：

输入维度：20

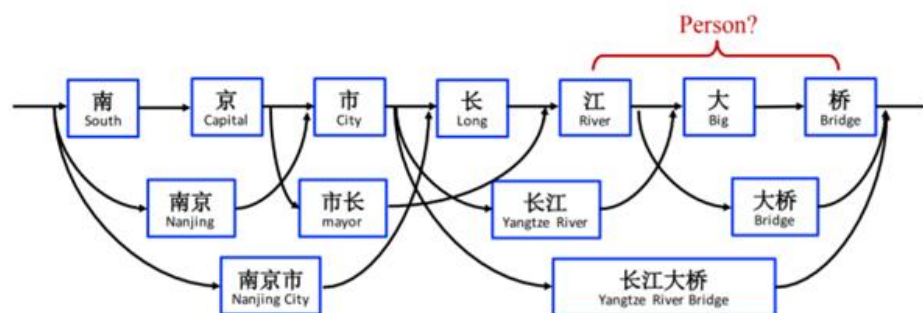
隐藏层维度（ReLU）400

输出层维度（Sigmoid）784

训练完网络，需要提交重构损失和KL散度的随迭代次数的变化图，以及10 张生成的手写数字图片。

二、使用Transformer解决命名实体识别 (Named Entity Recognition) 任务

1. 任务：命名实体识别 (Named Entity Recognition, 简称NER) 是自然语言处理领域的基础任务之一，是指识别文本中具有特定意义的实体，主要包括人名、地名、机构名、专有名词等。下图举了一个NER的例子，对人类来说识别出“南京市”和“长江大桥”是比较简单的任务，但是对模型来说却有可能识别出错误的实体。



2. 模型：近年来，以Transformer为基础的深度学习模型在自然语言处理和视觉领域盛行。此次作业旨在熟悉Transformer的原理及调用。推荐使用python库transformers来载入以及训练一个transformers模型。具体的模型采用bert-base-cased作为编码器，全连接层用于分类。

3. 数据集：CoNLL2003

CoNLL2003共包含4种实体类别，分别是location（地点名），organization（组织名），person（人名）和 miscellaneous（杂项）。此外，不属于任何实体类别的单词应该被标注为0（其他）。以下为示例：

示例输入：Japan began the defence of their Asian Cup title with a lucky 2-1 win against Syria in a Group C championship match on Friday

真实标签：B-location 0 0 0 0 0 B-misc I-misc 0 0 0 0 0 0 0 B-location 0 0 0 0 0 0 0 0

说明：实体类别前的B-/I-表示Begin和Inside，实体的第一个词应该以B-开头，实体后面的词应该以I-开头。例如Asian Cup的Asian标注为B-misc，Cup则标注为I-misc。

4. 任务说明：

(1) 阅读提供的代码，补充TODO位置的代码；

```
main.py 5 x
main.py > BertTagger > __init__

13 class BertTagger(nn.Module):
14     def __init__(self, hidden_dim, output_dim, model_name):
15         super(BertTagger, self).__init__()
16         # TODO:
17         # (1) 利用AutoConfig.from_pretrained定义config
18         # (2) 利用AutoModelWithLMHead.from_pretrained定义模型，注意要传入刚才的config
19         # (3) 定义一个线性层用于分类预测
20         # 提示：参考文档https://huggingface.co/bert-base-cased
21         config =
22         self.bert_model =
23         self.classifier =
24     def forward(self, X):
25         # TODO:
26         # (1) 把X输入bert_model得到hidden_states;
27         # (2) 提取其中属于最后一个transformer layer的hidden_states作为最终特征;
28         # (3) 最后把特征输入线性层完成预测。
29         # 提示：需要用到output_hidden_states参数，并参考以下文档
30         # https://huggingface.co/docs/transformers/v4.21.2/en/model\_doc/bert#transformers.BertLMHeadModel
31         features =
32         logits =
33         return logits
34
```

(2) 利用main.py训练一个NER模型，记录并可视化训练过程，比如loss，f1；

(3) 训练结束后，利用predict.py载入保存的模型，并输入自定义的例子进行预测，分析模型的输出结果；

(4) 回答思考题：① CoNLL2003有4种实体类型，为什么输出的维度是9；②为什么需要额外加线性层用于预测，而不用transformers模型原有自带的线性层；③在predict.py当中，tokenizer的作用是什么；④在predict.py当中，mask的作用是什么。

(5) 扩展（选做）：说明现有模型不足之处并改进模型，展示改进模型的性能。比如，增加CRF层。