



中南大學

CENTRAL SOUTH UNIVERSITY

黄品溢个人学习笔记

Title _____ FGRMER 论文精读

Tips _____ Learning Notes From 2022.06.05

学生姓名 _____ 黄品溢

指导教师 _____ 王磊

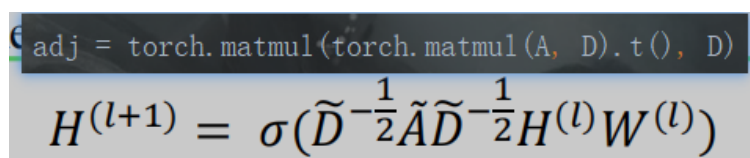
学 院 _____ 计算机学院

目 录

第一章 学习过程中的疑问

1.1 关于 GCN 传播公式实现的代码

由矩阵 \tilde{A} 的求取方法可以知道，在矩阵 \tilde{A} 中， $P_{ij} \neq P_{ji}$ ，即矩阵 \tilde{A} 并不是对称矩阵。那么根据代码来看，矩阵运算 $(\tilde{A}D)^T D = (D^T \tilde{A}^T) D = D \tilde{A}^T D$ 是否与原公式不符？



$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)})$$

图 1.1 上方为官方实现代码，下方为 GCN 传播公式

1.2 关于 nn.Dropout 和 nn.ReLU

nn.Dropout 和 nn.ReLU 是可训练的吗？当所有 ReLU 和 Dropout 层参数均相同时，为什么不能整个模型都采用同一个 nn.Dropout 和 nn.ReLU 层，而要分开写？

或者，能不能在需要使用的时候，直接从 torch.nn 里进行调用？

```
self.relu1 = nn.ReLU()
self.relu2 = nn.ReLU()
self.relu3 = nn.ReLU()
self.relu4 = nn.ReLU()
self.relu5 = nn.ReLU()

self.dropout1 = nn.Dropout(dropout)
self.dropout2 = nn.Dropout(dropout)
```

1.3 关于神经网络最后的 log_softmax

既然最后同样是使用了 torch.data.max 函数来找输出每一行中的最大值，那么最后的这个 log_softmax 步骤是否可以省略？

```
1 x = torch.cat([part1, part2], dim = 1)
2 x = self.layer_norm1(x)
3 x = self.linear5(x)
4 return F.log_softmax(x, dim=1)
```

1.4 关于 \tilde{A} 矩阵

在 GCN 的计算公式中：

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}) \quad (1.1)$$

按道理来说 \tilde{A} 应该是一个不可训练的矩阵，即应该是一个固定的矩阵，为什么还要使用 `Parameter()` 转换为可训练的矩阵？

是因为之后有个 `detach()` 阻断了传播，实际上并不会训练吗？

```
1  # 这里生成了  $A = A + In$ 
2  _adj = gen_A(9, adj_file)
3  self.A = Parameter(torch.from_numpy(_adj).float())
4  # 这里生成了归一化后的  $D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ 
5  adj = gen_adj(self.A).detach()
```

1.5 关于归一化的问题

在代码中，既使用了 `BatchNorm` 也使用了 `LayerNorm`，这两个归一化函数应该怎样选择？即何时使用哪一种归一化该如何决定？

第二章 Introduction 总体概述

2.1 关于微表情识别方法的三大类

- 基于 LBP 的微表情识别方法：在微表情帧之间的变化中，其更多的是面部肌肉引起的几何变换。而由于 LBP 更多注重于纹理信息，纹理信息是十分容易受到外部因素干扰的，如环境照明变化、人种等，因此 LBP 算法通常都不是微表情识别的最好方法。
- 基于光流的 (*Optical-Flow-Based*) 方法：LBP 特征通常被称为低级特征，而光流则被称为更高级的特征。在深度学习方法被引入 MER 之前，通常都是采用光流进行特征提取，而现在则主要是使用神经网络，如 CNN、LSTM、GCN 等，或他们的变体及组合进行特征提取。
- 其他新方法，主要都是基于深度学习。

在视频动作增强中，本文使用了基于转换学习的 MagNet。经过该深度学习网络增强后的形状表示是几何特征，因此可以作为更深层特征提取和学习的输入。

而经过这一次几何特征增强后，这一方法在当时取得了最好的识别率，这也证明了增强的几何特征对 MER 有着十足的帮助。

2.2 关于点学习 (Node Learning) 部分

由于微表情通常都是面部的微小变化，因此将整个面部照片直接输入进神经网络是十分粗鲁的，由此需要提取一些特定大小的 patches 作为输入。

由于之后的边学习中使用到了 Encoder，因此每个 patch 需要转换成 1×49 的形状。为了在一定程度上保留特征的空间信息，减少因形状拉伸导致的垂直空间信息损失，需要进行一个“点学习”。

点学习的过程主要步骤如下：

1. 首先，从增强后的形状特征中，截取 30 个 patches，每个 patches 的大小为 7×7 ，具体方法为：在人脸的 66 个特征点中，找出 30 个与眉毛和嘴巴相关的特征点，在每个特征点周围取相邻的 7×7 像素区域。这些 $30 \times 7 \times 7$ 的数据就被称为 面部图。
2. 然后，将 30 个 patches 视为 30 个 channels，将其输入 Depthwise CNN 进行学习。在此过程中，要保持每个 patch 的形状不变，即仍为 7×7 ；

2.3 关于 AU 部分

部分论文中并没有考虑到面部表情可以通过动作单元 AU 进行编码的机制，因此考虑 AU 对 MER 的影响有着极大的提升空间。

不同的表情类别对应着不同的 AU 组合，因此 AU 中包含的信息对 MER 有帮助。

2.4 关于融合 Fusion 的部分

在本文中，设计了两条通道进行工作：

- 使用神经网络和 Encoder 对特征进行提取和联系学习的通道；
- 对 AU 信息进行学习的通道，这一部分通过 GCN 进行学习。

最后，两个通道的信息通过一种特定的方式进行融合。

第三章 论文中提出的方法

在论文中提出了“一个通道学习面部图表示，另一个通道学习一个动作单元矩阵，并提出了一个双通道融合的新机制并用于微表情识别”。

一方面，仅采用了每个表情的 Onset Frame 和 Apex Frame 输入 MagNet，来从中间层提取增强后的形状特征。然后提取 $30 \times 7 \times 7$ 的图表示，并在图表示中进行点学习，之后使用 Encoder 进行图表示学习。

另一方面，在所有 36 个 AU 中，只有 9 个 AU 与眉毛和嘴巴区域相关，因此将这 9 个区域通过词嵌入和 GCN 进行学习，并得到 AU 特征矩阵。

最后，使用一个融合机制，将双通道信息融合，并用于微表情分类识别。

3.1 点学习 (Node Learning)

其大致步骤如下：

- 首先使用一个深度卷积 (*Depthwise Convolution*) 来学习空间特征；
- 之后将每个 patches 拉伸，经过卷积后，这一步中带来的垂直空间信息损失，在一定程度上得到了缩减；
- 将拉伸后的 patches 丢入 Encoder 中学习相互的关联；
- 最后分别经过两次全连接层、两次激活层和一次 Dropout 层进行学习，增强神经网络的表示能力。

第四章 关于代码

4.1 Mydataset.py

这一部分是关于数据的读取。在这一部分中，需要思考如下几个问题来辅助重写：

- 构造的 `MyDataset` 最终需要使用 `DataLoader` 进行数据提取，因此需要继承自什么类？
- 关于这篇论文，出于测试，需要一些什么参数？
- 在构造这个类别时，至少需要自己重载几个函数？分别需要什么参数？实现什么功能？
- 在读取原始数据的时候，需要使用什么包读取？怎样读取？语法是什么？
- 数据怎样被组织？怎样被保存？

4.2 Transformer.py

这篇论文中只使用到了 Encoder 部分，来获取各节点之间的关系，称为边学习。由此理论上来说，只需要编写 Encoder 部分即可。

根据以下 Encoder 的结构图，仔细观察数据的流通方式和模型结构。

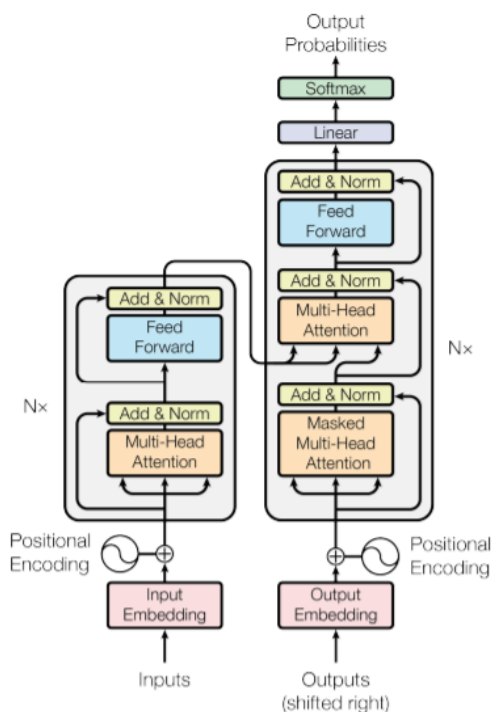


图 4.1 Transformer 结构图

思考如下几个问题来辅助重写：

- Encoder 部分需要几个参数？分别代表了什么含义？

- Encoder 中有几个部分？需要定义几个类？
- MultiHeadAttention 中有几个部分？需要定义几个层？需要什么参数？
- FeedForward 中有几个层？需要什么参数？
- 关于 Encoder 的关键算式，需要怎样实现？需要什么参数？

4.3 Networks.py

在这一文件中，写的是使用模型的代码。思考如下问题辅助编写：

- NetworkModel 模型需要几个输入参数？每个参数代表了什么含义？
- NetworkModel 中包含了几个部分？各自如何实现？
- 需要用到几次归一化？需要的参数分别是多少？
- 关于面部特征学习通道：
 - 按照 Linear \rightarrow ReLU \rightarrow Dropout 的顺序来写代码；
 - 每一次线性变换或卷积变化之后，都需要一次 ReLU，而是否需要 Dropout 需要视情况而定；
 - 深度卷积之后需要使用一次批归一化和 ReLU；
 - 注意何时需要转换形状
- 关于 AU 部分：
 - 怎样实现 GCN 传播公式？各个变量怎样得到？
 - 每个变量代表了什么含义？怎样实现？
 - 在这一部分使用了什么激活函数？参数是多少？
 - 怎样初始化参数？有几处？
- 关于融合部分：
 - 怎样实现融合？
 - 融合以后怎样输出？

4.4 utils.py

在这个文件中，实现了两个函数，其分别对应于求 GCN 中的两个矩阵： \tilde{A} 和 $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ 。

4.5 FGRMER_run.py

在这个文件中，实现了各个模块的运行和流程的控制。

注意在过程中需要将模型转移到 GPU 上，并使用测试模式。