

基于深度学习数学字符化的手写输入法的实现

王静雅

安徽信息工程学院 安徽芜湖 241199

摘 要: 深度学习是机器学习的一个分支, 在图像识别、语音识别、自然语言处理等领域已经取得了巨大的成功, 其发展为人工智能的发展带来了巨大的推动力。论文从学习工作的实际需求出发, 研究实现了一个基于深度学习的数学字符化的手写输入法并详细论述了整个输入法程序实现和 GUI 编程的过程。

关键词: 卷积神经网络; 生成对抗网络; GUI 编程

引言

学习、工作中我们常常有编辑数学公式的需求, 而许多编辑工具效果都差强人意, 所以出于方便 LaTeX 数学公式编辑的需求, 此设计诞生了

1 程序结构与结构实现

1.1 输入法程序结构

1. 文本输入框, 用以盛载 LaTeX 数学公式文本。
2. 数学符号键盘 (常用符号、希腊字母、分数微分、根式角标、极限对数、三角函数、积分运算、大型运算、括号取整、数组矩阵), 手写数学字符识别模块, 用以编辑 LaTeX 数学公式。

3. LaTeX 数学公式渲染显示界面, 用以实时渲染输入文本框的 LaTeX 数学公式。

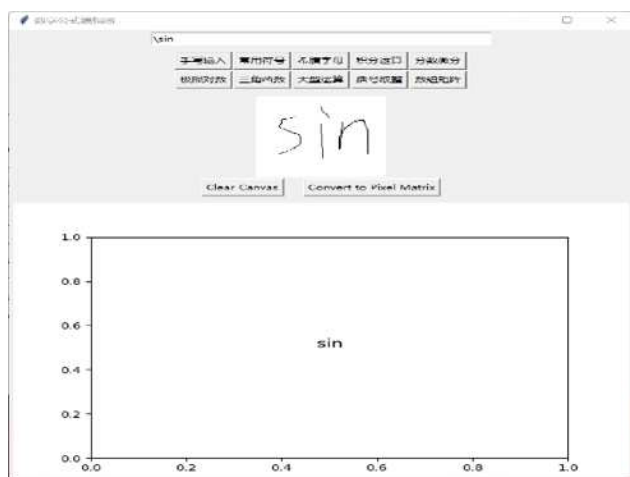


图 2-1 图形界面展示

1.2 结构实现

1.2.1 整体结构实现

使用 python 中的库 tkinter 创建 GUI 可视化界面。将文本输入框、键盘切换按键、各个数学符号键盘按键、LaTeX 数学公式渲染显示模块分别放到 tkinter 的 Frame 容器中, 在 Frame 容器中进行单独排版。将 Frame 容器自上而下放在 Tk 主容器中, 各个数学符号键盘放在同一层, 共 4 层^[1]。

1.2.2 具体结构实现

(1) 键盘切换实现

正按下不同键盘切换按键时触发的事件为: 显示对应数学符号键盘, 隐藏其他数学符号键盘。

(2) 实时渲染实现

将文本输入框 tk.Entry() 与 tk.StringVar() 创建的字符串类型变量绑定, 字符串类型变量的值随文本输入框中内容的改变而改变, 进而触发 LaTeX 数学公式渲染显示模块, 实现实时渲染^[2]。

(3) 手写输入键盘的实现

用 tk.Canvas() 创建绘图区域类, 将创建的绘图区域类与 <Button-1> 鼠标左键按下事件、<B1-Motion> 鼠标左键拖动事件、<ButtonRelease-1> 鼠标左键释放事件绑定, <Button-1> 事件触发时, 将鼠标在绘图区域内的位置坐标 (event.x,event.y) 赋值给 (prev_x,prev_y), <B1-Motion> 事件触发时, 将鼠标拖动位移后的位置坐标 (event.x,event.y) 赋值给 (x,y), 通过坐标 (prev_x,prev_y) 和坐标 (x,y), 使用 tk.Canvas().create_line() 在画布上绘制直线, 之后将坐标 (x,y) 赋值给 (prev_x,prev_y), 鼠标拖动的过程中, 位置坐标不断变化, 由此实现绘图区域内的图形绘制。

使用 PIL 模块中的 `Image.new()` 方法 `create` 一个新的图像对象, 使用 PIL 模块中的 `ImageDraw.Draw()` 类在给定的图像上创建一个绘图对象, 通过坐标 $(prev_x/a, prev_y/b)$ 和坐标 $(x/a, y/b)$, 使用 `ImageDraw.Draw().line` 在创建的图像对象上绘制直线, 由此实现在刚刚创建的图像上的图形绘制。(a 、 b 用于将绘图区域内的坐标映射到图像对象内的坐标。)

提取所创建图像的像素矩阵到 `np.array()`, 将调整后的像素矩阵输入到神经网络模型, 将预测结果对应的字符输入到文本框。

(4) LaTeX 数学公式渲染显示模块的实现

`FigureCanvasTkAgg` 用于在 Tkinter 中显示 Matplotlib 图形的后端。创建一个 `FigureCanvasTkAgg` 对象, 将 Matplotlib 图形嵌入到 Tkinter 界面中, 在 Tkinter 窗口中显示 Matplotlib 图形。

2 数据集

2.1 数据集分析

训练神经网络模型实现手写数学字符的识别, 需要许多各种各样数学字符的手写图像数据, 构建带标签的图像数据集。为了得到足够的图像数据, 论文中使用的数据集综合了两个现有手写数学字符图像的数据集。分别是 `Handwrittenmathsymbolsdataset` 和 `TheHASYv2dataset`。前者由 `45x45jpg` 图像文件组成, 共包含 82 个类别共 375970 张手写数学字符图像, 包括基本的希腊字母符号, 如 α 、 ϕ 和 θ , 小写英文字母, 数学运算符, 集合运算符, 基本的数学函数, 如 $\log$ 、 $\lim$, 数学符号, 如 $\int$ 、 $\sum$ 、等, 由 CROHME 数据集经过解析、提取和修改而来。后者由 `32x32png` 图像文件组成, 图像中仅有黑白像素, 共包括 369 个类别共 168233 张手写数学字符图像, 包括拉丁字符、阿拉伯数字、32 种不同类型的箭头、分形、括号等。图 4-1 展示了使用 matplotlib 库中的 `imshow` 函数来分别显示 `Handwritten math symbols dataset` 和 `The HASYv2 dataset` 中的图像的示例。

2.2 数据集构建

通过双线性插值方法将 3232 的图像等比例放大为 4545, 因为 3232 的图像中仅有黑白像素, 通过双线性插值法将其等比例放大时, 图像会失去其二值图像特征, 然后设置一个参数, 将参数值以上的像素值设为 255, 参数值以下的值设为 0, 就可以加粗图像中的字符轨迹, 然后通过膨胀

算法将图像中的字符轨迹变细, 尽可能地使 3232 图像接近 4545 的图像。

选择特定的数学符号类别构建数据集, 在已有的图像当中有的类别数量多, 有的类别数量却很少, 当某个类别的图片在 `Handwrittenmathsymbolsdataset` 和 `TheHASYv2dataset` 中的数量加起来都依然很少时, 通过对抗生成网络生成一定数量的图片进行数据集的扩充^[3]。

3 实践过程

3.1 生成对抗网络的构建和模型训练

3.1.1 网络的构建

(1) 生成器的构建

生成器第一层为全连接层, 其中包含 $5 \times 5 \times 128 = 3200$ 个神经元, 将输入的长度为 100 随机噪声向量映射到一个长度为 3200 的向量。第二层将第一层的输出变换为一个大小为 $(5, 5, 128)$ 的张量, 这个张量可以被看作是一个 5×5 的 128 通道的特征图。第三层为转置卷积层, 对由第一层输出变换而来的大小为 $(5 \times 5 \times 128)$ 张量进行转置卷积操作, 从而将其尺寸放大, 并增加特征图的深度, 这一层有 64 个 $(5 \times 5 \times 128)$ 的卷积核, 卷积步幅为 3, 采用 `samepadding`, 激活函数为 `relu`。第四层为转置卷积层, 有 32 个 $(5 \times 5 \times 64)$ 的卷积核, 卷积步幅为 3, 采用 `samepadding`, 激活函数为 `relu`。第五层为转置卷积层, 有 1 个 $(3 \times 3 \times 32)$ 的卷积核, 卷积步幅为 1, 采用 `samepadding`, 这次输出的特征图深度为 1, 因为生成的是单通道的图像(灰度图像), 这一层的激活函数为 `tanh`, 这一层激活值的大小在 -1 到 1 之间, 与灰度图像的像素值范围相匹配^[4]。

(2) 判别器的构建

判别器第一层是第一个卷积层, 包含 32 个的卷积核, 卷积步幅为 2, 采用 `samepadding`, 第一层的输入是大小为的张量, 表示灰度图像, 添加 `LeakyReLU` 激活函数, 它允许负值通过, 防止 `relu` 激活函数可能导致的神经元“死亡”。第二层是第一个池化层, 为最大池化, 池化窗口为, 池化操作的步幅为 2, 采用 `samepadding`。第三层是第二个卷积层, 包含 64 个的卷积核, 步幅为 2, 采用 `samepadding`, 添加 `LeakyReLU` 激活函数。第四层是第二个池化层, 为最大池化, 池化窗口为, 步幅为 2, 采用 `samepadding`。第五层将第四层的输出展开成向量, 连接到下一层。第六层是全连接层, 具有 1024 个神经元, 使用 `relu` 激活函数, 这一层使

用 Dropout, Dropout 参数为 0.5。第七层是全连接层, 只有 1 个神经元, 使用 sigmoid 激活函数, 激活值大小在 0 和 1 之间。这个全连接层的输出表示输入数据是真实数据的概率。

(3) 将生成器和判别器连接起来构成整体网络

将生成器和关闭训练功能的判别器连接起来构成整体网络, 使用二元交叉熵作为损失函数, 使用 AdaptiveMomentEstimation 优化算法, 参数设为默认值。

3.1.2 模型训练

判别器训练: 在数据集也就是真实图像中随机选取数量为 $n_{imi}-batch$ 大小的真实图片, 用过 (像素值-127.5)/127.5, 将其归一化到 [-1,1], 生成相同数量的噪声向量输入生成器生成相同数量的生成图像, 分别用真实图像和生成图像训练判别器, 分别计算损失函数, 将真实图像的损失和生成图像的损失取平均作为最终的判别器损失函数。

3.2 卷积神经网络的构建和模型训练

3.2.1 卷积神经网络的构建

网络整体结构为卷积层 → 池化层 → 卷积层 → 池化层 → 展平层 → 全连接层 → 全连接层。

网络第一层为第一个卷积层, 输入大小为 28 的张量, 包含 32 个的滤波器, 步幅为 1, 采用 samepadding, 使用 relu 激活函数。输入数据经过第一个卷积层后的大小变为 28。该层共 832 个参数。

网络第二层为第一个池化层, 为最大池化, 池化窗口为 2, 池化操作的步幅为 2, 采用 samepadding, 即当输入大小在减去池化窗口大小后不能被池化步幅整除, 那么池化层会在特征图的边缘进行填充, 确保特征图的边缘信息得到保留。数据经过第一个池化层后大小变为 14。

网络第三层为第二个卷积层, 输入大小为 14 的张量, 包含 64 个的滤波器, 步幅为 1, 采用 samepadding, 使用 relu 激活函数, 数据经过这个卷积层后大小变为 14。该层共 51264 个参数。

网络第四层为第二个池化层, 为最大池化, 池化窗口为 2, 步幅为 2, 采用 samepadding, 数据经过第二个池化层后大小变为 7。

网络第五层将第四层输出的张量展开成长度为 9216 的向量。

网络第六层为第一个全连接层, 输入数据为长度为 9216 的向量, 具有 1024 个神经元, 使用 relu 激活函数, 这

一层使用 Dropout, Dropout 参数为 0.5。该层共 9438208 个参数。

网络第七层为第二个全连接层, 该全连接层有 32 个神经元, 使用 softmax 激活函数。该层共 32800 个参数。

整个网络共有 9523104 个参数, 使用分类交叉熵作为损失函数, 在参数迭代的过程中采用 AdaptiveMomentEstimation 优化算法。

3.2.2 模型训练

(1) 数据集导入

导入训练集和测试集, 将训练集和测试集的格式转换为 (-1, 45, 45, 1) / 255, -1 会自动匹配合适的格式, 图像像素值为 0 到 255, /255 对训练集和测试集中图像的像素矩阵进行归一化, 将矩阵内元素的值的范围归一化到 0-1 之间。将训练集和测试集标签转为 onehot 格式。

(2) 模型训练和评估

将 batch_size 的值设为 64, 训练 10 个 epoch。图 5-7 展示了训练过程中损失函数和准确率的变化, 把训练集和测试集分别输入未经训练的神经网络, 得到的损失函数的值和准确率分别为 [3.18977, 0.04529], [3.18902, 0.04535], 图像展示了训练 5 个 epoch 之后训练集和测试集上的损失函数和准确率变化。

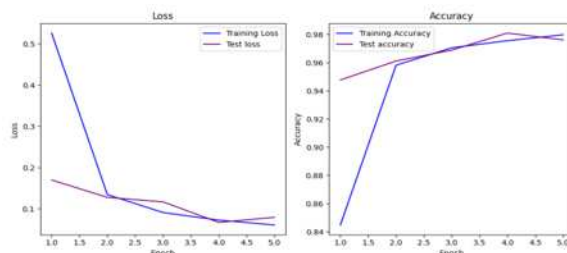


图 3-1 损失函数和准确率

4 总结与展望

通过对神经网络的训练和优化, 模型实现了对常见数学字符的识别, 将其应用到数学公式编辑器中的手写输入, 取得了不错的效果。论文验证了使用生成对抗网络生成图像扩充数据集的可行性, 实现了基于深度学习的数学字符化的手写输入。模型只实现了对单个手写数学字符的识别, 虽然能识别例如 sin、lim 的数学函数名, 但识别这些连续字母的含义的原理与识别单个数学符号的原理并无区别。

参考文献:

[1] 王发新. 实时手写数学符号识别研究与应用 [D]. 电

子科技大学 ,2022.

[2] 方定邦 . 基于卷积神经网络的手写数学公式字符识别的算法研究 [D]. 华侨大学 ,2020.

[3] 王博 . 用于手写数字识别的改进混合式生成对抗

LeNet-5 卷积神经网络 [D]. 广西大学 ,2020.

[4] 刘雨涵 . 生成对抗网络中的优化算法研究 [D]. 中国石油大学 (北京),2020.