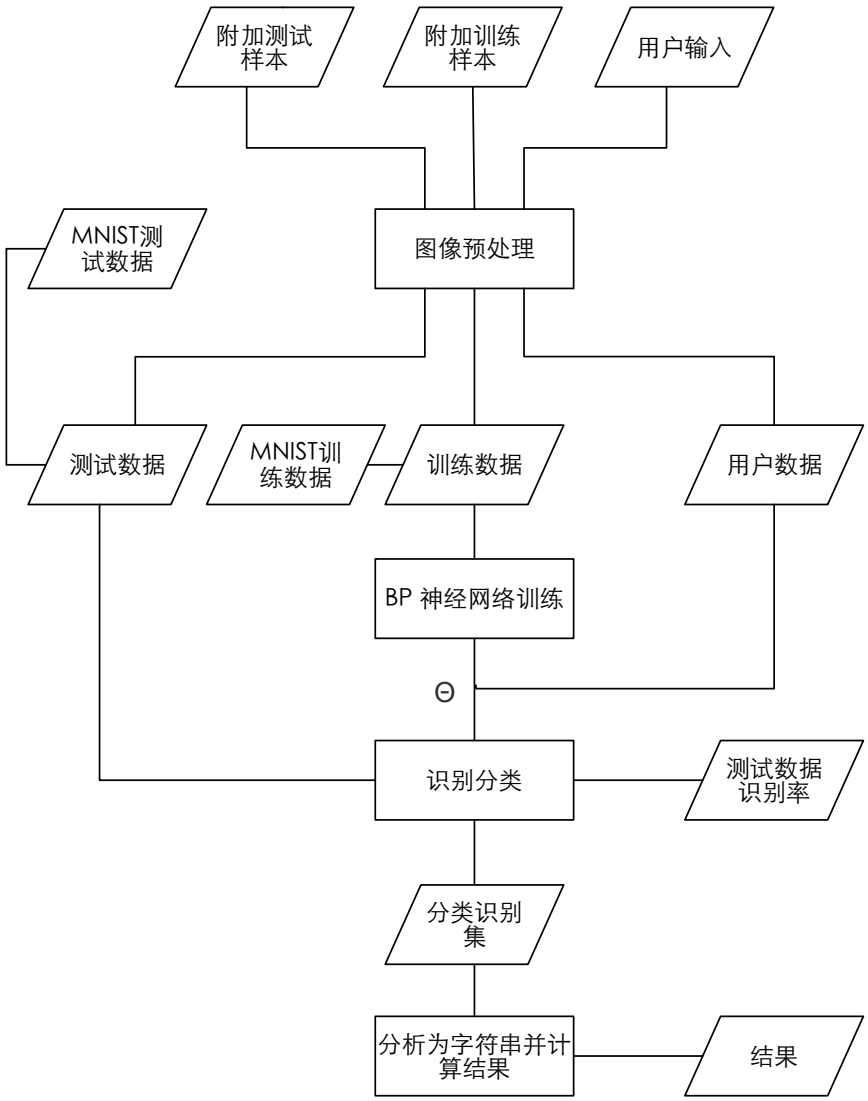
基于 BP 网络的简单手写表达式识别与求值系统

刘畅 李辉 朱晨旭 刘建南
(信电工程学院 2013 级计算机科学与技术 2 班)

摘要：本程序采用 Matlab 和 C 混编，实现了简单手写数学表达式的识别和求值功能，支持带括号的加、减、乘、除和幂运算。识别过程由 Matlab 程序分为三个步骤完成：图像预处理，特征提取，分类识别。表达式识别结果以字符串形式传入 C 程序中，求值过程由 C 程序采用算符优先算法完成。

主流程图：



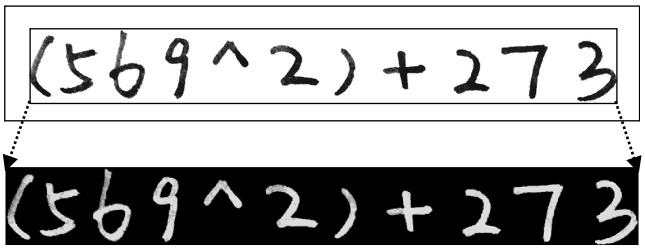
一、识别系统

1.1 图像获取

程序接受 JPG 格式图像作为输入，图像可通过扫描仪或数码相机获得。

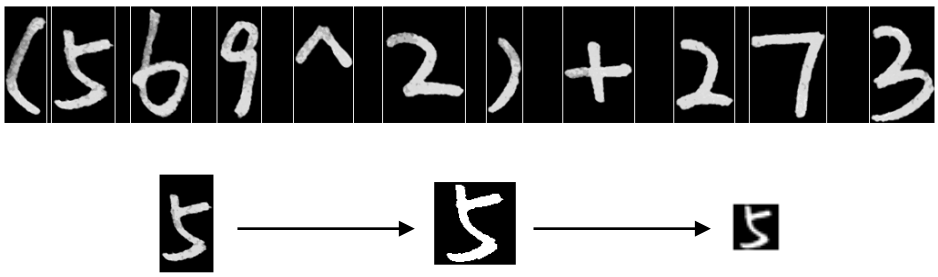
1.2 预处理

图像转为灰度图像后反色处理，并进行中值滤波以平滑图像，然后进行整体边界限定，截取表达式区域图像。



横向提取并拼接有效行，纵向提取并分割有效字符。

提取每个分割区域为独立图像，进行边界限定，计算灰度阈值并转为二值图像，最后大小归一化图像：填充至正方形并统一为 20 * 20 的图像，以供训练或识别。



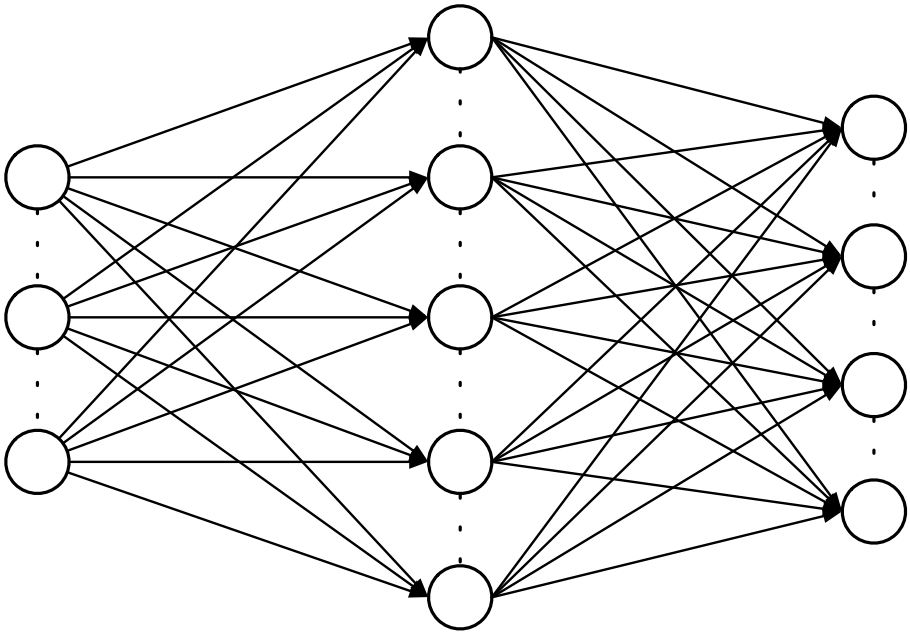
1.3 特征提取

对每个上一步得到的字符图像，将其像素矩阵转化为一个 1 * 400 的特征向量。

1.4 机器学习

识别前需要训练分类器，本系统采用 BP (Back Propagation) 神经网络作为分类器，包含一个隐藏层共 80 个隐藏单元，激活函数 (activation function) 采用 sigmoid 函数。数字训练数据一部分来源于 MNIST Database (http://yann.lecun.com/exdb/mnist/)，另一部分数字训练数据和符号训练数据来源于小组成员手写扫描图像，共十七类字符。测试数据来源同上。

开始阶段先对权重矩阵进行随机初始化 (Random Initialization) 以打破对称性。训练阶段对每个样本的特征向量，先通过正向传播 (Forward Propagation) 计算各层结果，再反向计算各层残差以计算代价函数 (cost function) 在当前权重矩阵处的梯度；自学习过程采用梯度下降法不断更新权重矩阵以获得代价函数的局部最优解，得到相应的权重矩阵。测试阶段计算测试集识别率和交叉验证集识别率。通过选择不同的正则项 (regularization) 多次训练网络，选取识别率最优的权重矩阵。



输入层: $x^{(i)} \in R^{400}$　　隐藏层: $a_2^{(i)} \in R^{80}$　　输出层: $h_\Theta(x^{(i)}) \in R^{17}$

训练集: $\{(x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})\}$　　$a_2^{(i)} = \frac{1}{1+e^{-x^{(i)}\Theta^{(1)T}}}$　　$h_\Theta(x^{(i)}) = \frac{1}{1+e^{-a_2^{(i)}\Theta^{(2)T}}}$

所用神经网络模型简化图

1.5 识别

在识别过程中，用分类器识别待识别的特征向量。通过 Matlab engine 与 C 交互，在 C 中通过 mxArray *engGetVariable(Engine *, const char *) 函数获取识别结果并通过 void dataAnalyze(const mxArray *, char **str) 函数分析识别结果为字符串并存入 str 中。

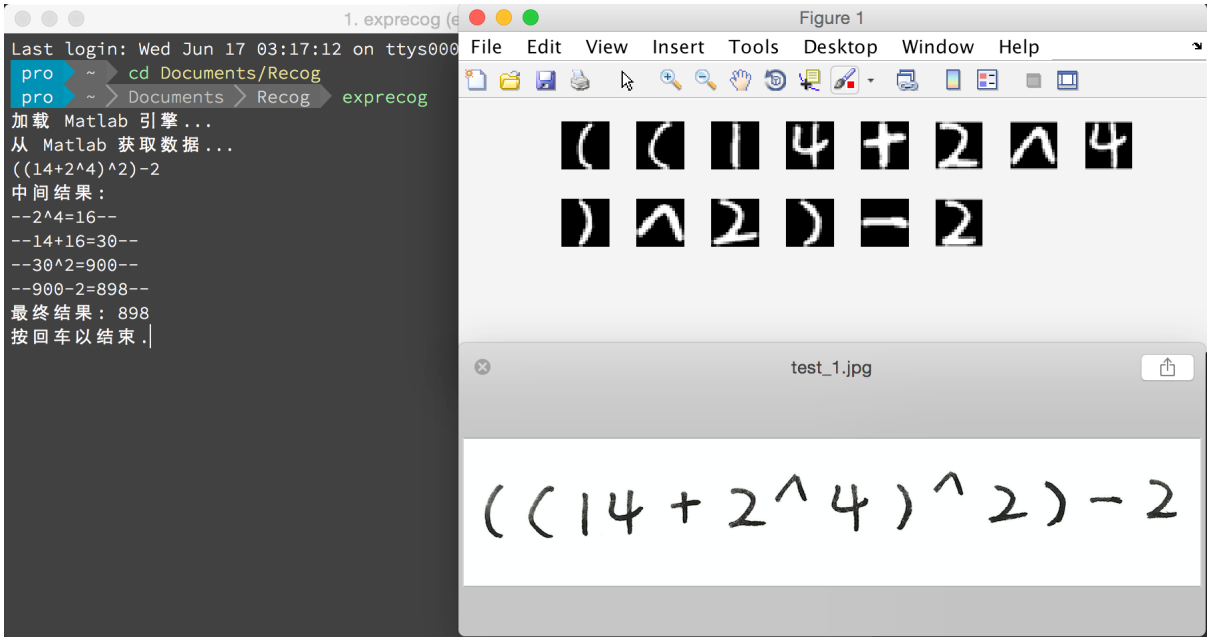
二、计算表达式的值

计算部分采用算符优先算法，使用两个工作栈，分别存储操作数和运算符，并将运算符间的优先关系事先存入表中。优先关系表如下：

```
/*                    |              运算符 b              |   */
/*                    | + | - | * | / | ^ | ( | ) |\0 |   // 运算符 a */
char table[8][8] = {{'>','>','<','<','<','<','>','>'},  // +
                    {'>','>','<','<','<','<','>','>'},  // -
                    {'>','>','>','>','<','<','>','>'},  // *
                    {'>','>','>','>','<','<','>','>'},  // /
                    {'<','<','<','<','<','<','>','>'},  // ^
                    {'<','<','<','<','<','<','=','>'},  // (
                    {'>','>','>','>','>','e','>','e'},  // )
                    {'<','<','<','<','<','<','e','='}}; // \0
```

通过 int getUnit(char **str) 函数从 str 中每次读取一个操作数或运算符。读到操作数时立即放入操作数栈中，读到运算符则和运算符栈的栈顶运算符比较优先关系后作相应操作，直至整个表达式求值完毕。

三、运行结果



四、总结与不足

1. 程序由 Matlab 和 C 混编，无法脱离 Matlab 运行，未来可改用 OpenCV 做图像预处理和识别过程。

2. 部分字符识别率不高，未来可增加训练数据量，并改进模型，如改用卷积神经网络。

3. 图像预处理时分割算法不够好，无法分割靠得太近的字符，未来可改用基于连通域提取的图像分割算法。

4. int getUnit(char **str) 函数用到了几个全局变量，代码逻辑较为复杂，未来应改善代码逻辑并减少全局变量引用。

5. 目前能处理的表达式和运算符都比较简单，未来可增强。

五、杂项

1. 在读取 MNIST Database 的 idx 格式的训练数据和测试数据时用到的函数 readMNIST 由 Siddharth Hegde 编写并开源发布在 MathWorks File Exchange 中。

http://www.mathworks.com/matlabcentral/fileexchange/27675-read-digits-and-labels-from-mnist-database

2. 在自动化学习过程中采用 fmincg 函数替代 fminunc 函数以求解 cost function 最小值，该函数由 Carl Edward Rasmussen 编写并提供在 mlclass-ex3 中。

https://www.coursera.org/course/ml

参考文献：

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.

[2] Sneha Sharma, Dr. Roxanne Canosa. "Extraction of Text Regions in Natural Images". 2006.

[3] Mark Allen Weiss. 数据结构与算法分析: C语言描述. 北京: 机械工业出版社. 2004.