**BP手写识别的计算器**

组长： 1152703 方志晗

组员： 1152689 薛 蒙

**1.项目简介**

主要分为两个模块:计算模块和手写识别模块；计算模块执行计算，通过点击输入或者手写输入可以执行多项式计算。手写模块主要用BP神网算法实现手写识别，可以自动学习，自动记错，通过算法优化，实现用户记错功能且保证网络的稳定性。

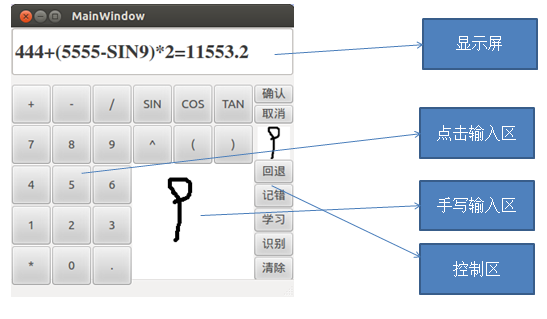
**2.功能介绍**

**计算模块：**

1.可实现+，-，\*，/，^，三角函数等基本运算，

2.实现带括号的混合运算；

3.回退，清除运算等；

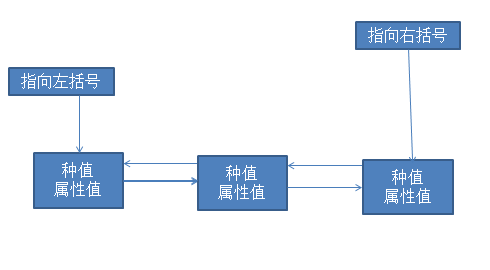


**实现：**

1. 仿编译器算法实现混合运算

算法简介： 词法分析结链(包含种植和属性值)🡪 对链分析出括号🡪利用括号对应算法实现计算

未用传统栈式算法，双向指针结链



**手写模块**

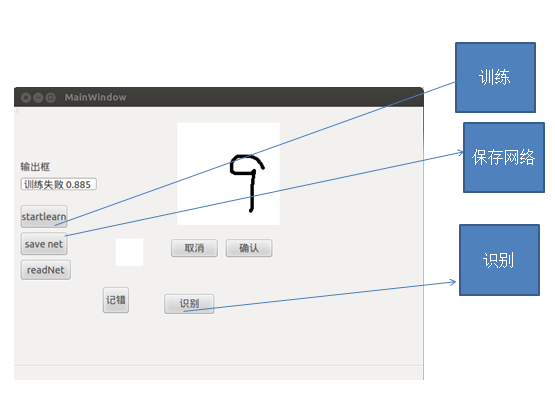
**功能**

1. 手写识别功能

2. 识别数字，部分字母，括号，尖括号等，满足计算器输入

3. 自动纠错功能

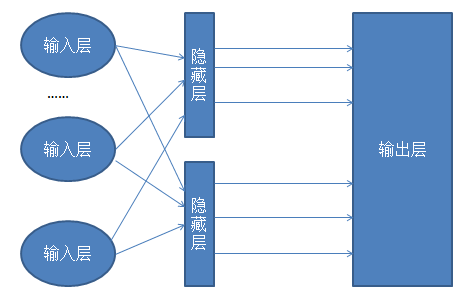
4. 学习功能



**记错功能**

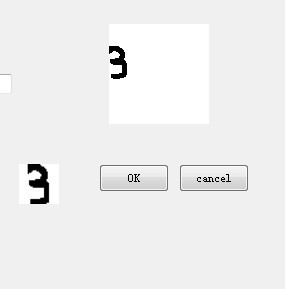


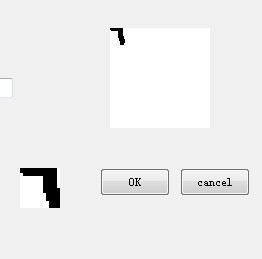
**神网算法**



**手写识别算法：**

我们设定的手写板的大小是150\*150。由于手写的特殊性，用户写的字符会出现在手写板的任何位置，那么为了提高输入数据的一致性，或者说能够让字符出现在大致相同的位置，这就有必要对输入数据进行一定的预处理。





可以看到，最终数字被放在了一张40\*40大小的图片中，并且位置相应大致居中，图像的预处理工作完成。

神经网络：

Bp神经网络是一种按误差逆传播算法训练的多层前馈网络，是目前应用最广泛的神经网络模型之一。因为它的算法步骤简洁明了，我们决定采用bp神经网络来作为我们的学习算法。

一般bp网络选择3层结构，1层输入层，1层中间层，以及一层输出层。但是经过我们的实验，发现当中间层为2层时，网络学习速率为0.2时，网络工作的最好。所以我们最终的网络模型是4层结构。至于网络的具体实现细节，这里不再赘述。（详细代码在network.h和network.cpp中）

输入样本：

由于之前已经完成了输入图像的预处理工作，所以我们的输入样本便是40\*40大小的png图片。1600个像素点，我们分割为100个单位，每个单位16个像素点，分别计算每单位中黑色像素点的密度，作为100个输入数据。

输出数据：

我们一共定义了23个字符，如果选择0、1代表的话，只需要5位即可。但是考虑到以后的拓展性，我们决定将输出数据的范围扩大到6位，即最大可识别的字符数为64个。

样本准备与学习算法：

由于bp网络自身的缺陷和性质，训练样本的选择和学习的方式就显得尤为重要。首先，网络训练的样本应该做到最大化的打散输入。如果某一个样本的重复率过高，可能造成网络最终的收敛达不到我们预期的效果，同时也要考虑网络陷入局部最小点的问题。

在开发中，我们一共遇到了样本输入的几种问题：

1. 有的样本数量很多，有的样本数量很少（比如1，样本数本身不需要很多），那么在选择输入的时候，仅仅选择随机方式就可能造成某一个字符的训练次数要高于另外一种。即使是保证图片输入的不重复性，仍然在样本的数量上存在不均衡。
2. 无法充分的利用所做样本。一开始由于样本数量较少，所以我们可以直接选择随机数的方式来决定该输入那个样本。但是最后为了提高网络识别的宽度，我们大大加大了样本的数量，而且每个字符的样本数量也存在非常大的差距。要是保证每次输入样本字符的差异性，并且相同字符样本的差异性，就存在一定的困难。
3. 如果选择用户自定义输入学习的话，由于用户可以无限的添加同一个字符样本，这样会造成样本的数量悬殊。这样如果没有一个稳定的学习输入算法的话，最坏的结果是bp网络崩溃。

解决办法：

Vector<vector<string>>

首先随机选择一个输入字符，然后在这个字符的所有样本中选择一张样本输入，然后删除这个样本的索引。如果某一链的样本删除到0时，立刻重新补全该立链。然后这样解决的方案就是可以充分的利用每一个样本，并且不受样本数量不同的影响，还能够保证数据足够的杂乱性。

**总结**

**虽然我们做了各种各样的优化，但是基于bp自身的局限性以及每个人的手写输入风格的不统一，对于新用户的手写风格还不适应，只能够保证用户的正规标准输入识别率保持在90%以上，但是一旦出现具有强烈的个人风格的手写输入或者极端的测试输入，效果还是不理想，我们提供的解决办法是网络的自动错误学习，但是这样的话用户体验是一个问题。**