NeuralBase - нейросеть за 5 минут

Библиотека компонентов , предназначена для программной реализации нейронных сетей. В качестве примера, созданы компоненты реализующие две нейросетевые парадигмы: рекуррентную нейронную сеть, в нашем случае – это сеть Хопфилда и многослойную нейронную сеть обучаемую по алгоритму обратного распространения ошибки (back propagation).
Основным назначением библиотеки является интеграция нейронных сетей в информационные системы, для расширения аналитических возможностей систем. Реализация нейронных сетей в виде компонентов, наличие открытого кода позволяет легко встраивать в другие программы. Объектно-ориентированное исполнение придает особую гибкость, достаточно переписать пару методов и вы можете получить компонент, оптимизированный под ваши задачи.

Иерархия классов

Существует три базовых класса TNeuron, TLayer, TNeuralNet. Все остальные являются производными от них. На рис.1 приведена иерархия классов, сплошными линиями показано наследование (стрелкой указан потомок), пунктирными в каких классах они используются.

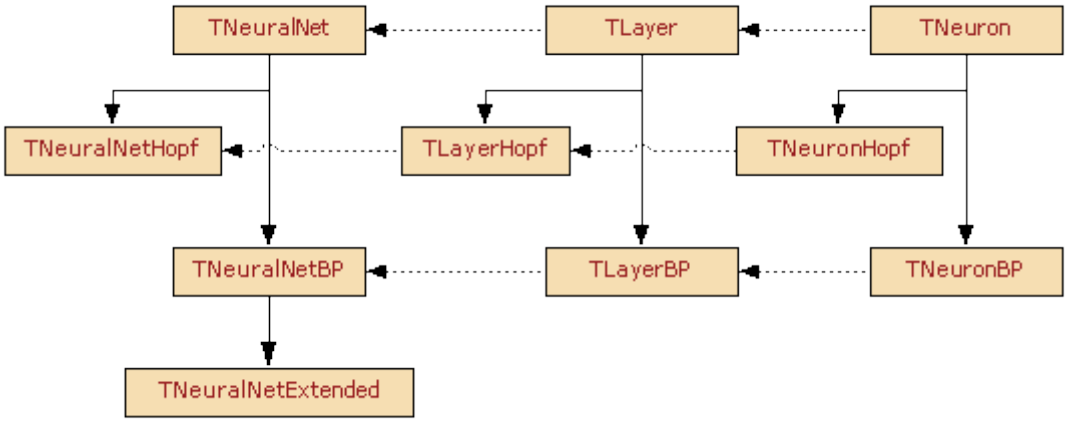

Рис.1. Иерархия классов.

TNeuron является базовым классом для нейронов, несет всю основную функциональность, имеет индексированное свойство Weights, представляющее собой весовые коэффициенты (синапсы), свойство Output, которое является выходом нейрона (результатом вычислений) и сумматор, роль которого, выполняет метод ComputeOut.
TNeuronHopf, потомок TNeuron, реализует нейрон используемый в нейронной сети Хопфилда, единственным отличием от базового класса, является использования активационной функции в перекрытом методе ComputeOut.
Следующим порожденным классом, является **TNeuronBP** служащий для программной реализации многослойных нейронных сетей. Аббревиатура BP в имени класса не должна вводить вас в заблуждение, что нейрон этого типа используется исключительно в сетях обучаемых по алгоритму обратного распространения, этим, мы лишний раз хотели подчеркнуть, что в нашем случае нейронная сеть обучается по этому алгоритму.
Переписан метод ComputeOut, использующий теперь нелинейную активационную функцию, которая реализована в виде индексированного свойства процедурного типа OnActivationF. Кроме того, добавлены два важных свойства, Delta – содержащая локальную ошибку и индексированное свойство PrevUpdate – содержащее величину коррекции весовых коэффициентов на предыдущем шаге обучения сети.
Основным назначением базового класса **TLayer** и его потомков **TLayerHopf** и **TLayerBP** является объединение нейронов в слой, для упрощения работы с нейронами.
Компонент **TNeuralNet** базовый компонент для всех видов нейронных сетей. TNeuralNet обеспечивает необходимую функциональность производных компонентов. Этот компонент поддерживает методы для работы со слоями сети (AddLayer, DeleteLayer) и методы для манипуляций с исходными данными (AddPattern, DeletePattern, ResetPatterns). Метод Init служит для построения нейронной сети. Большинство методов объявленных в разделе **public** в базовом компоненте и его потомках – виртуальные, что позволяет легко перекрывать их.
Компонент TNeuralNetHopf реализует нейронную сеть Хопфилда.

Дополнительно включены следующие методы: InitWeights – запоминает предъявленные образцы в матрице образов и метод Calc – вычисляет выход сети Хопфилда.
Компонент **TNeuralNetBP** реализует многослойную нейронную сеть обучаемую по .

Дополнительно включены следующие методы: Compute – вычисляет выход нейронной сети, используется после обучения сети; TeachOffLine – обучает нейронную сеть. Компонент позволяет в режиме design-time, в окне Object Inspector, конструировать нейронную сеть добавляя или удаляя слои и нейроны в сети. Для этого используется редактор свойств NeuronsInLayer, имеющий следующий вид:

Рис.2. Редактор свойства NeuronsInLayer – конструирование сети.

Совместимость с Neural Network Wizard

Следующим компонентом является **TNeuralNetExtented** порожденный от TNeuralNetBP, который обеспечивает полную совместимость с Neural Network Wizard. Дополнительно включены следующие методы: для записи (LoadPhase1, LoadPhase2, LoadPhase4, LoadNetwork) и чтения (SavePhase1, SavePhase2, SavePhase4, SaveNetwork) обученной нейронной сети в формате *.nnw; LoadDataFrom – загружает данные из текстового файла, а также метод NormalizeData нормализации входных и выходных данных; Train – для обучения нейронной сети; ComputeUnPrepData – для вычисления выхода сети, используется в том случае, если у вас входные значения ненормализованы.

Компонент позволяет в режиме design-time, в окне Object Inspector, выбирать нужные поля, а также задавать тип нормализации полей. Для этих целей используется редактор свойств, имеющий следующий вид:

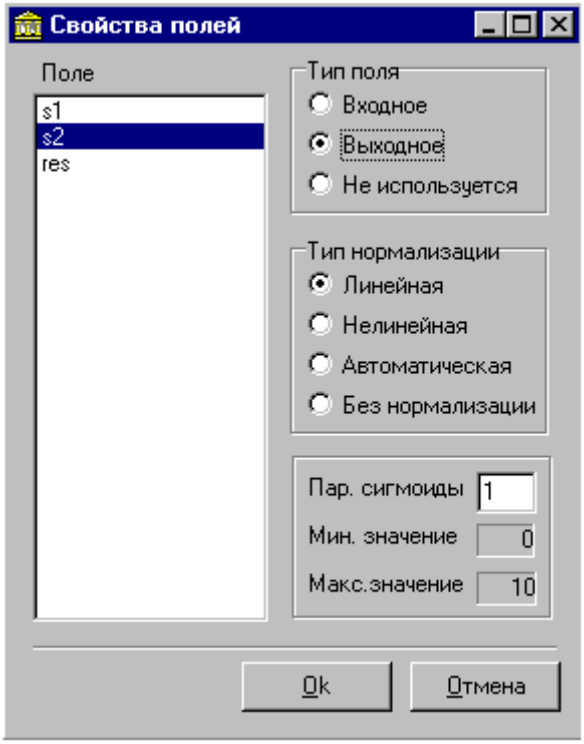

Рис.3. Редактор свойства – слои.

Компонент TNeuralNetExtented один из самых мощных в библиотеке NeuralBase. Используя этот компонент, практически за считанные минуты можете получить готовое полнофункциональное приложение.

Демонстрационные программы

В качестве демонстрационных примеров приведены три программы, показывающие возможности предложенных компонентов.

Программа **Recognition** используя компонент TNeuralNetHopf , реализует нейронную сеть Хопфилда. Программа решает задачу распознавания образов. На вход сети подается некий образ, возможно искаженный или неполный и нейронная сеть восстанавливает образ, т.е. относит предъявляемый образ к одному из хранимых сетью образов, либо в случае неудачи, выдает новый образ, иногда называемый "химерой".

Программа **XOR_Problem**, реализует функцию "исключающее или", которая является стандартным тестом, после знаменитой работы Минского и Пейперта "Перцептроны". В основе программы лежит компонент TNeuralNetBP.

Программа **EasyNNW**, использующая компонент TNeuralNetExtented представляет собой аналог программы Neural Network Wizard, единственным отличием от NNW является несколько "облегченный" интерфейс. Данный пример показывает, насколько легко и быстро, создаются программы реализующие нейронные сети с достаточно хорошей функциональностью на основе библиотеки компонентов NeuralBase.