Обзор

Программа NeuronNetwork реализует простую нейронную сеть прямого распространения, используя предоставленный класс Neuron. Он включает в себя функции построения сети, прямого распространения, прогнозирования, расчета ошибок, обратного распространения для корректировки веса и файлового ввода-вывода для сохранения и загрузки сети. Сеть поддерживает переменные размеры слоев и инициализацию случайного веса.

Структура класса

class NeuronNetwork {

public:

int layers;

int\* size;

Neuron\*\* neurons;

double\*\*\* weights;

NeuronNetwork(int n, int\* p);

~NeuronNetwork();

// ...

};

Конструктор (NeuronNetwork(int n, int\* p))

Конструктор инициализирует нейронную сеть указанным количеством слоев n и массивом p, представляющим количество нейронов в каждом слое. Он выделяет память для нейронов и весов, устанавливает значения смещения и инициализирует веса случайными значениями.

Деструктор (~NeuronNetwork())

Деструктор освобождает выделенную для нейронной сети память.

Конструкция сети

- `SetInput(double\* p)`: устанавливает входные значения для входного слоя.

Прямое распространение

- `ForwardFeed()`: выполняет прямое распространение через нейронную сеть.

Прогноз

- `Predict() -> int`: прогнозирует выходные данные, находя индекс нейрона с максимальным значением активации в выходном слое.

Ошибка расчета

- `ErrorCounter(double\* ra) -> double`: вычисляет ошибку в выходном слое для оценки правильности прогнозов.

Обратное распространение ошибки

- `BackPropagation(double\* ra, double ls)`: применяет обратное распространение ошибки для обновления весов на основе вычисленных ошибок.

Файловый ввод-вывод

- `SaveNetwork(const char \*filename)`: сохраняет веса и смещения сети в файл на диске.

- `LoadNetwork(const char \*filename)`: загружает веса и смещения сети из файла на диске.

Функция Softmax

- `SoftMax() -> double\*`: вычисляет функцию softmax для выходного слоя, обеспечивая распределение вероятностей.

Использование

Программа обеспечивает гибкую и модульную реализацию нейронной сети. Пользователи могут создавать экземпляры класса NeuronNetwork, настраивать сетевую архитектуру с помощью конструктора, устанавливать входные значения, выполнять прямое распространение, прогнозировать выходные данные, вычислять ошибки, применять обратное распространение ошибки и сохранять/загружать веса и смещения сети.

Примечание. Обязательно включите необходимые файлы заголовков, особенно файл Neuron.h, для правильной компиляции программы.