Neural Network Static Library

Классы:

- matrix - матрица чисел типа double:

  size_t rows - число строк
  size_t cols - число столбцов

  matrix transpose() - возвращает матрицу, полученную из данной транспонированием
  void random_fill(double mean, double sigma) - заполняет матрицу случайными числами из нормального распределения
  void resize(size_t rows, size_t cols) - изменяет форму матрицы
  vector<double> mult_by_v(vector v) - умножает матрицу на вектор, возвращает результат в виде вектора.

- layer - служебный класс, описывает нейронный слой:

  size_t layer_size - число нейронов в слое
  size_t prevlayer_size - число нейронов в предыдущем слое
  matrix weights - матрица весов связей
  vector<double> bias - вектор сдвигов

  void store(string filename) - сохраняет веса и сдвиги в файл
  void load(string filename) - загружает веса и сдвиги из файла

- network - нейросеть:

  vector<layer> layers - нейронные слои
  function<double(double)> activation - функция активации нейрона
  function<double(double)> d_activation - производная активации
  cost_function* cost_f - указатель на функцию потерь

  network(create_instructor ci) - создаёт нейросеть по указанным параметрам
  void sgd_step(sgd_instructor si) - шаг градиентного спуска
  void unwrapped_sgd_learn(learn_instructor& li) - “сырой” градиентный спуск
  void sgd_learn(learn_instructor& li) - градиентный спуск (оболочка)
  vector<double> feedforward(vector<double> input) - пропускает input через сеть, возвращает активации последнего слоя
  void save(string filename, bool logs) - сохраняет веса нейросети в файл
  void load(string filename, int layer_num, bool logs) - загружает веса нейросети из файла
  double batch_test(vector<vector> inp, vector<vector> ans, function comparator) - тестирует нейросеть: inp - вектор входных данных, ans - вектор соответствующих ответов, comparator - функция определения ответа. Возвращает долю правильных ответов.

- cost_function (абстрактный) - функция потерь:

vector<double> lastlayer_delta(vector pred, vector ans, vector inp, function d_act) - возвращает ошибку последнего слоя.
double error(vector pred, vector ans) - возвращает ошибку предсказания

(Унаследованы две функции потерь: MSE и кросс-энтропия)

Инструкторы: (созданы на основе struct)

- create_instructor - параметры создания нейросети:
  int perception_neurons - количество входных нейронов
  int output_neurons - количество выходных нейронов
  function<double(double)> activation - функция активации нейрона
  function<double(double)> d_activation - производная активации
  cost_function* cf - функция потерь
  vector<int> hidden_layers - вектор с числом нейронов в скрытых слоях. (Например {100, 50} - два скрытых слоя: на 100 и 50 нейронов)

- learn_instructor - параметры обучения нейросети:
  int batch_size - размер партии с обучающими примерами
  int epoch_count - количество эпох обучения
  int thread_number - число потоков для обучения
  int upd_frequency - частота обновления скорости обучения
  int gen_seed - зерно для генератора случайных чисел
  int patience - терпимое число эпох без улучшения результатов (необходимо при флаге only_best_score = true)

  double learning_rate - скорость обучения
  double L2_lambda - коэффициент регуляризации
  double mult_factor - на сколько умножать скорость обучения при её обновлении

  vector<vector<double>>* train_input - обучающие данные
  vector<vector<double>> train_output - обучающие ответы
  vector<vector<double>> validate_input - валидационные данные
  vector<vector<double>> validate_output - валидационные ответы
  function<bool(vector<double>, vector<double>)> comparator - функция для сравнения ответов (нужно для валидации)

  mutex* mtx - мьютекс для работы в нескольких потоках

- sgd_instructor - параметры шага градиентного спуска (служебный обьект):
  int batch_size - размер партии с обучающими примерами
  double learning_rate - скорость обучения
  double L2_lambda - параметр регуляризации
  vector<double> input - входные данные
  vector<double> output - выходные данные (ответ)
  mutex* mtx - мьютекс для многопоточного случая

- gridsearch_params - параметры гридсёрча:
  vector<double> learning_rate_variations - вариации скорости обучения
  vector<double> l2_lambda_variations - вариации параметра регуляризации
  vector<double> mult_factor_variations - вариации уменьшения скорости обучения

  vector<vector<double>>* train_input - обучающие данные
  vector<vector<double>> train_output - обучающие ответы
  vector<vector<double>> valid_input - валидационные данные
  vector<vector<double>> valid_output - валидационные ответы

  create_instructor ci - параметры для нейросетей
  function<bool(vector<double>, vector<double>)> comparator - функция для сравнения ответов.

Дополнительные функции:

- double scalar_product(vector v1, vector v2) - скалярное произведение
- double random_number(double mean, double sigma) - генерация случайного числа из нормального распределения с средним *mean* и отклонением *sigma*
- double sigmoid(double val) - сигма-функция
- double d_sigmoid(double val) - производная сигмы.
- vector<double> random_vector(double mean, double sigma, int size) - генерирует вектор случайных чисел размера size
- vector<double> apply(vector v, function f) - создаёт новый вектор, применяя f к v поэлементно
- vector<double> hadamard_product(vector v1, vector v2) - произведение Адамара (поэлементное произведение векторов)
- Перегружены операторы + и - для векторов, перегружен оператор * для вектора и числа типа double.