Автоматическое построение оптимальной структуры сетей глубокого обучения для задач классификации временных рядов

Роман Прилепский

Московский Физико-Технический Институт,
Сколковский Институт Науки и Технологий

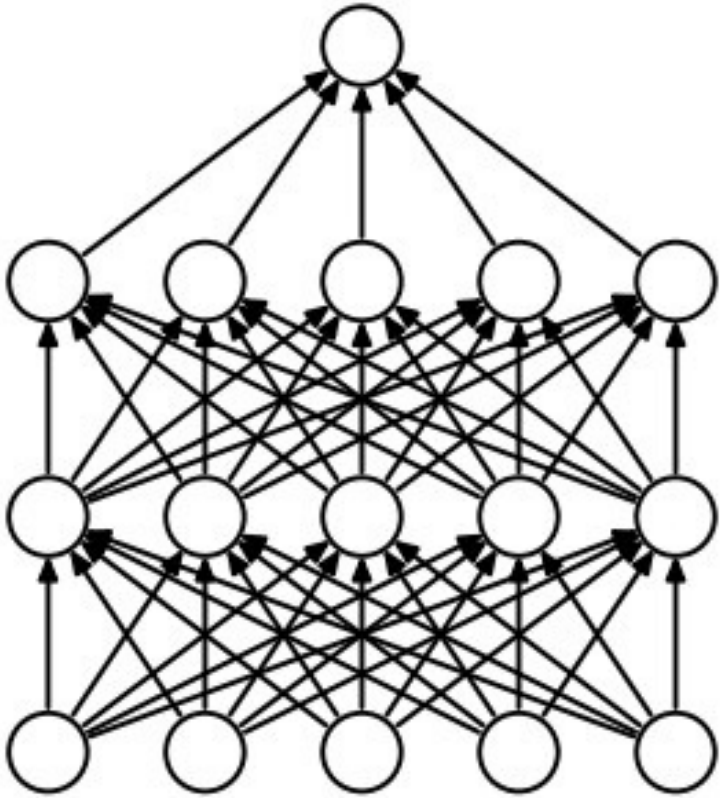
Научный руководитель: В.В. Стрижов
Д.ф.-м.н., ведущий научный сотрудник ВЦ РАН

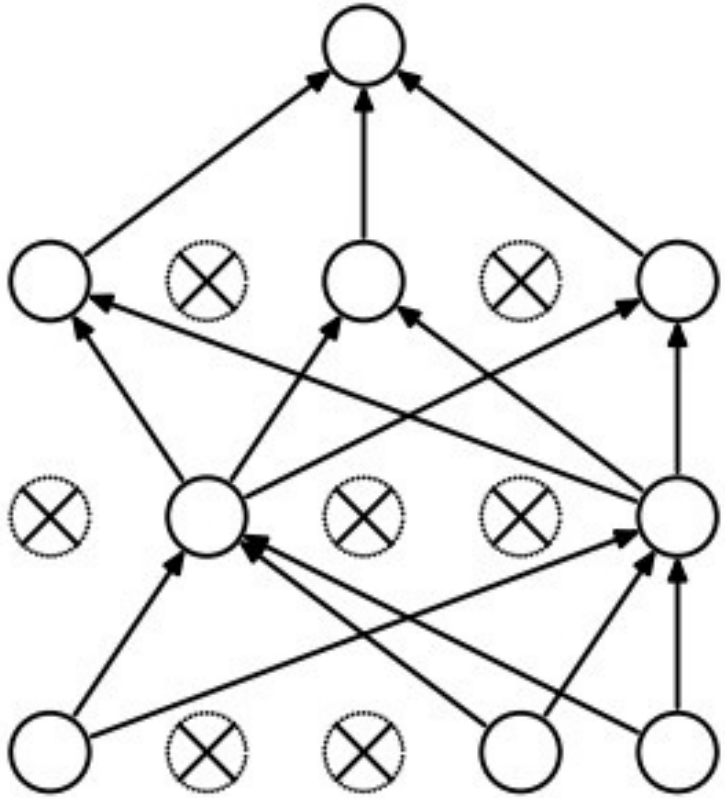
Москва,
2015.

Цель работы

- **Цели:**
  1. Разработать алгоритм автоматического построения оптимальной структуры сетей глубокого обучения для задач классификации временных рядов
  2. Создать техническую систему на базе графических ускорителей для автоматического построения сетей глубокого обучения.
- **Решаемая проблема:** распознавание физической активности человека по временным рядам, полученным с трёхосного акселерометра смартфона.
- **Методы:** комбинация генетического алгоритма, стохастического отбора признаков и Optimal Brain Surgeon в сравнении с Optimal Brain Damage.

Автоматическое построение сетей глубокого обучения с оптимальной структурой





На рисунке представлен результат прореживания нейронной сети без негативного влияния на точность классификации.

1. **Автоматическое построение сетей глубокого обучения:**
   - Liu C., et al., *Pruning Deep Neural Networks by Optimal Brain Damage*, 2014.
   - Krizhevsky A., et al., *ImageNet Classification with Deep Convolutional Neural Networks*, 2012.
   - Hassibi B., et al., *Second Order Derivatives for Network Pruning: Optimal Brain Surgeon*, 1993.
   - LeCun Y., et al., *Optimal Brain Damage*, 1990.

2. **Техническая часть:**
   - Jia Y., et al., *Caffe: Convolutional Architecture for Fast Feature Embedding*, 2014.
   - Bastien F., et al., *Theano: new features and speed improvements*, 2012.

Постановка задачи отбора признаков I

$\mathfrak{D} = \{(\mathbf{x}_i, \mathbf{t}_i)\}, i \in \mathcal{I} = \{1 \dots m\}$ – исходная выборка.
$\mathbf{t}_i \in \{0, 1\}^z$, где z – количество классов.
$\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m], \mathbf{X} \in \mathbb{R}^{m \times n}$ – матрица плана.
$\mathbf{y} = [\mathbf{t}_1, \dots, \mathbf{t}_m]^\mathsf{T}$ – целевой вектор.

Моделью классификации является суперпозиция функций

$$\mathbf{f}(\mathbf{X}, \mathbf{w}) = \boldsymbol{\mu}_\mathbf{1}(\boldsymbol{\mu}_\mathbf{2}(\dots \boldsymbol{\mu}_\mathbf{K}(\mathbf{X}))) = \mathbf{y},$$

где $\boldsymbol{\mu}_\mathbf{k}$, $k \in \{1, \dots, K\}$ – модель класса нейронная сеть.
Пример $\boldsymbol{\mu}_\mathbf{k}$ – двухслойная нейронная сеть:

$$\mathbf{a}(\mathbf{x}) = \mathbf{W}_2^\mathsf{T}\mathbf{tanh}(\mathbf{W}_1^\mathsf{T}\mathbf{x}),$$

$$\mathbf{p}(\mathbf{x}) = \frac{\exp(\mathbf{a}(\mathbf{x}))}{\sum_j \exp(a_j(\mathbf{x}))}.$$

где $\mathbf{w} = \mathrm{vec}(\mathbf{W}_1^\mathsf{T} | \mathbf{W}_2^\mathsf{T}) \sim N(\mathbf{w_0}, \mathbf{A}^{-1})$ – вектор параметров.

Постановка задачи отбора признаков II

Функция ошибки:

$$S(\mathbf{w}|\mathcal{K}) = -\sum_{i\in\mathcal{K}}\sum_{\xi=1}^{z} t_{i\xi}\ln\bigl(p_\xi(\mathbf{x}_i,\mathbf{w})\bigr),$$

максимизирующая логарифм правдоподобия случайной величины $\mathbf{y}$ и заданную на разбиении выборки $\mathfrak{D}$, определенной некоторым множеством индексов $\mathcal{K}\subseteq\mathcal{I}$, $\mathbf{t}_i = [t_{i1},\ldots,t_{i\xi},\ldots,t_{iz}]^\mathsf{T}$.

Оптимальный вектор параметров модели $\mathbf{f}_\mathcal{A}$ – вектор $\hat{\mathbf{w}}_\mathcal{A}$, который является решением следующей задачи оптимизации:

$$\hat{\mathbf{w}}_\mathcal{A} = \operatorname*{argmin}_{\mathbf{w}_\mathcal{A}\in\mathbb{R}^k} S(\mathbf{w}_\mathcal{A}|\mathcal{L}).$$

Снижение размерности пространства параметров

Наращивание сети

Стратегии наращивания:

- Изменить количество нейронов
- Изменить связность
- Ассиметрия
- Соединение нейронов в одном слое
- Пропуск слоёв
- Обратная связь
- Добавление входов смещения
- Наращивание нейронов внутри слоев

Методы наращивания:

- Генетические алгоритмы
- Эволюционные алгоритмы
- Алгоритм имитации отжига

Прореживание сети

Стратегии прореживания:

- Optimal Brain Damage
- Optimal Brain Surgeon

Несколько существующих правил выбора количества нейронов в скрытом слое:

- Weka — $N_h = \frac{N+M}{2}$
- Neuralware — $N_h = \frac{T}{5\times(M+N)}$
- Barron — $N_h = \sqrt{\frac{T}{N \log T}}$
- Masters — $N_h = \sqrt{N \times M}$

N – размерность входных данных;
N_h – количество нейронов в скрытом слое;
M – размерность выходных данных;
T – количество доступных обучающих векторов.

Optimal Brain Surgeon

$$\delta S = \left(\frac{\partial S}{\partial \mathbf{w}}\right)^T \cdot \delta \mathbf{w} + \frac{1}{2}\delta \mathbf{w}^T \cdot H \cdot \delta \mathbf{w} + O(\|\delta \mathbf{w}\|^3)$$

Мы решаем проблему минимизации

$$\delta S = \frac{1}{2}\delta \mathbf{w}^T \cdot H \cdot \delta \mathbf{w} \to \min,\ \mathbf{e}_q^T \cdot \delta \mathbf{w} + w_q = 0$$

1. Обучить сравнительно большую сеть до минимальной ошибки S.
2. Вычислить H^{-1} – обратную матрицу Гессе.
3. Найти q, который дает наименьший $L = \frac{{w_q}^2}{2[H^{-1}]_{qq}}$.

   Если изменение ошибки много меньше S, то q^{th} нейрон должен быть удален, дальше – шаг 4; иначе – шаг 5 (переобучить сеть).
4. Используя q из шага 3, обновляем *все* веса и переходим к шагу 2:

$$\delta \mathbf{w} = -\frac{\omega_q}{[H^{-1}]_{qq}} H^{-1} e_q$$

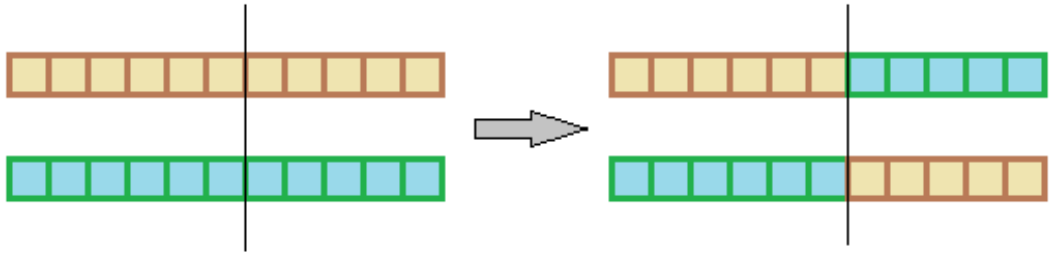
Стохастический отбор признаков I

$\mathcal{F} = \{\mathbf{a}_i\}$ – популяция нейронных сетей,

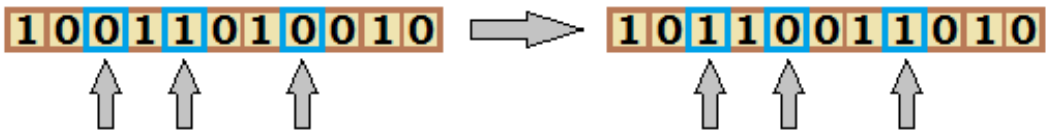
$$\begin{cases} \mathbf{a}_j = 1, & \text{если } j \in \mathcal{A},\ \mathcal{A} \text{ множество активных нейронов;} \\ \mathbf{a}_j = 0, & \text{иначе.} \end{cases}$$

1. Выбрать $\mathcal{F}' \in \mathcal{F}$. Каждый $\mathbf{a}_j$ может быть выбран с вероятностью
$$p_j = \frac{\exp -\frac{S_j}{S_{max}}}{\sum_{i=1}^{N} \exp \frac{-S_i}{S_{max}}}.$$
$\mathcal{F}' = \{\mathbf{a}_1^\mathsf{T}, \ldots, \mathbf{a}_P^\mathsf{T}\}$, P – четное число.
2. Множество $\mathcal{F}'$ случайным образом делится на пары $\{\mathbf{a}_i^\mathsf{T}, \mathbf{a}_j^\mathsf{T}\}$.
3. С каждой парой $\{\mathbf{a}_i^\mathsf{T}, \mathbf{a}_j^\mathsf{T}\}$ проводятся операции скрещивания и мутации:

Стохастический отбор признаков II



Генетический алгоритм – Скрещивание



Генетический алгоритм – Мутация

Полученное множество векторов обозначим $\mathcal{F}'' = \{\mathbf{a}_i''\}_{i=1}^{P}$

4. Для каждого $\mathcal{F}''$ оценивается значение $\mathbf{w}$ и S.
5. При стабилизации $\mathcal{F}'$ алгоритм останавливается. Иначе, процедура начинается с шага 1.

Критерии оценки и функция ошибки

- **Многоклассовая логарифмическая функция потерь**

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log(p_{ij}),$$

N – количество наблюдений,

M – количество меток классов,

y_{ij} – 1, если наблюдение i – из класса j и 0, если наоборот.

p_{ij} - предсказанная вероятность, что наблюдение i принадлежит классу j.

- **Точность (Precision) и Полнота (Recall)**

$$P = \frac{tp}{tp + fp},$$

$$R = \frac{tp}{tp + fn}$$

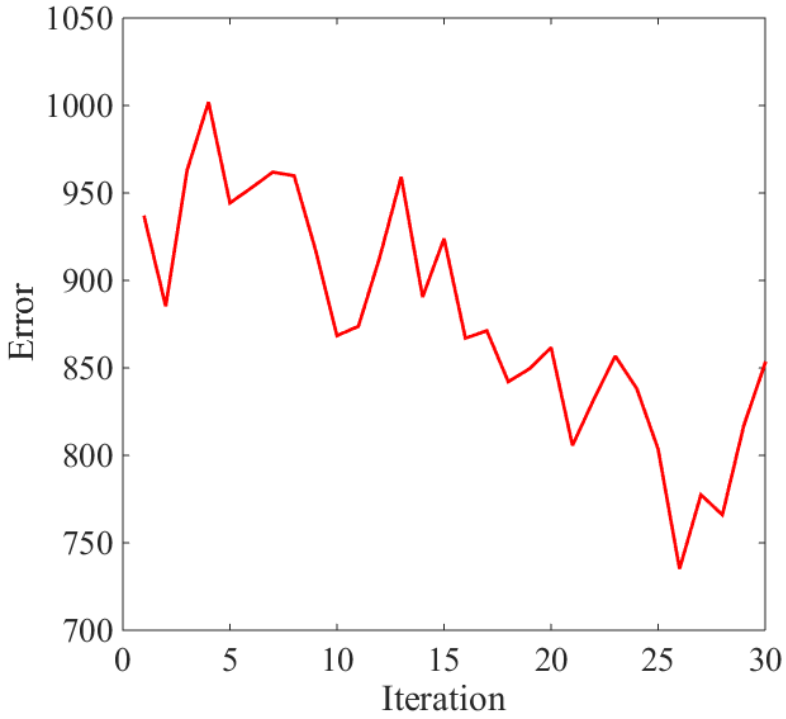
tp, fp, fn - количество истинно-положительных, ложно-положительных и ложно-отрицательных классификаций соответственно.

Вычислительный эксперимент

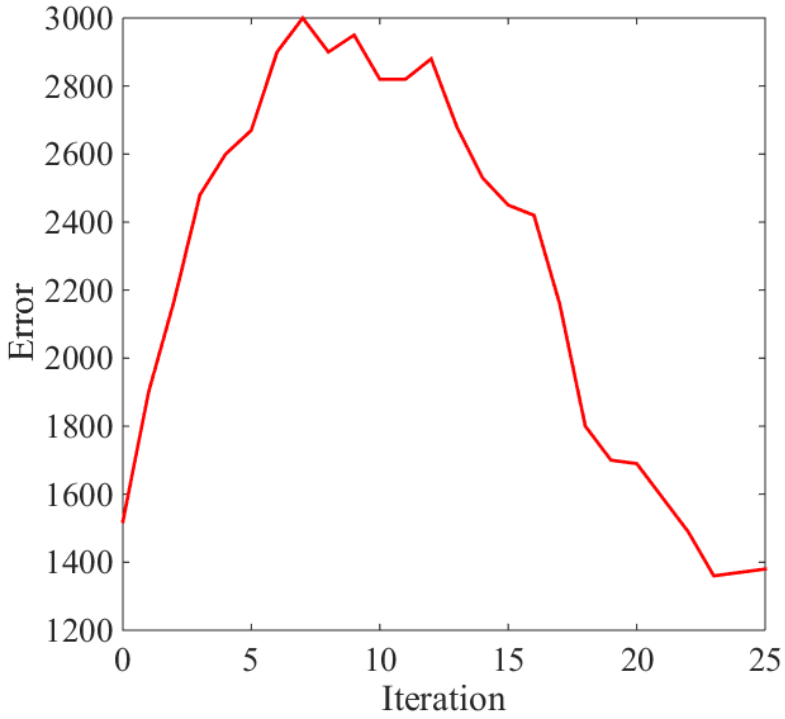
- **Цель**: Проверить точность классификации предложенного метода в задаче распознавания физической активности человека по временным рядам, полученным с трёхосного акселерометра смартфона.
- **Данные**: Обучающая и тестовая выборка опубликованы на веб-сайте Wireless Sensor Data Mining Lab.

Всего элементов:	1,098,207
Всего атрибутов:	6
Распределение по классам:	
Ходьба:	424,400 (38.6 %)
Бег:	342,177 (31.2 %)
Вверх по лестнице:	122,869 (11.2 %)
Вниз по лестнице:	100,427 (9.1 %)
Сидение:	59,939 (5.5 %)
Стояние:	48,395 (4.4 %)

Результаты – Экспериментальная модель в Matlab

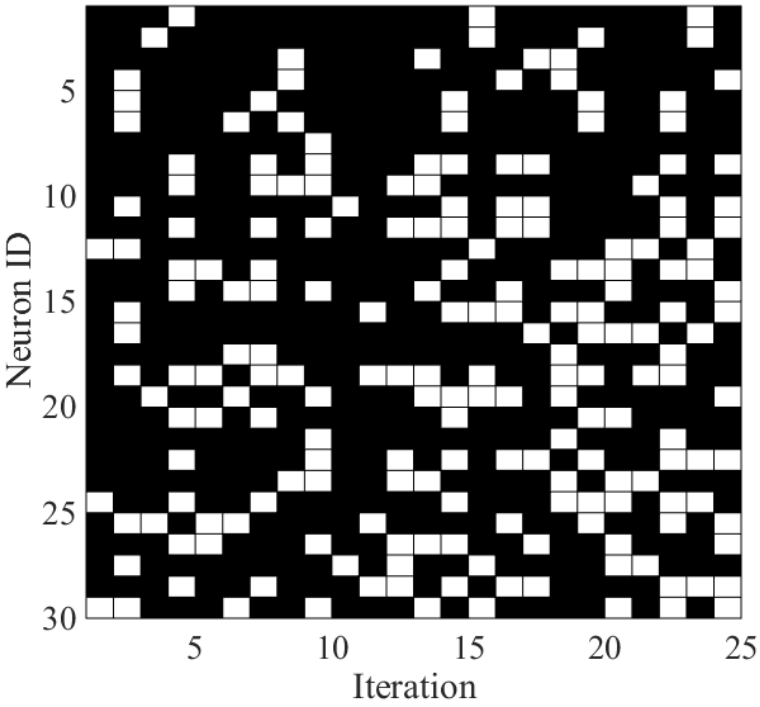


Предложенная стратегия

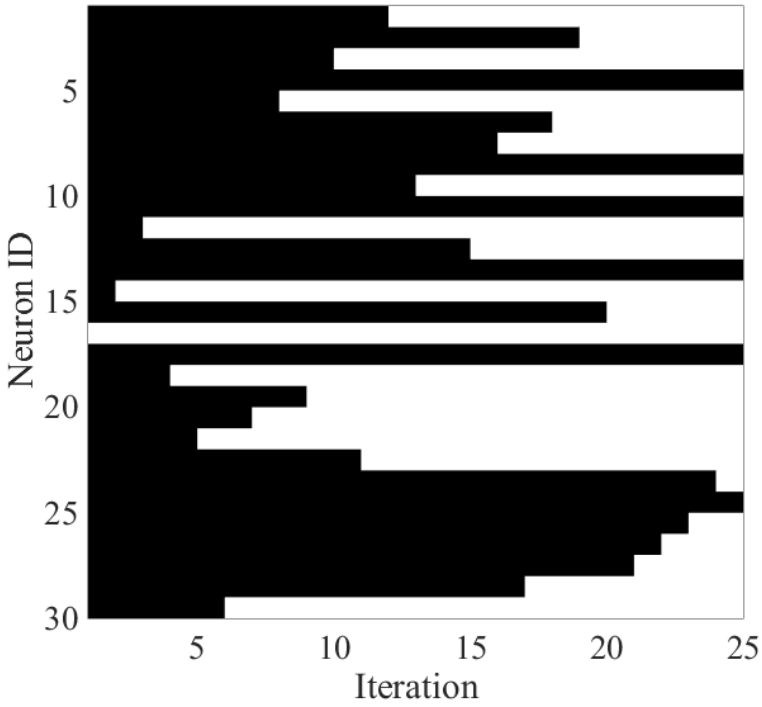


Optimal Brain Damage

Результаты – Визуализация изменений структуры сети



Предложенная стратегия



Optimal Brain Damage

Параметры наиболее точной модели

Алгоритм	**OBD**		**GA+OBS**	
Метрика	P,%	R,%	P,%	R,%
Бег	84,1	85,3	87,9	86,5
Ходьба	95,6	96,7	97,7	97,1
Вверх по лестнице	53,2	47,4	56,4	52,0
Вниз по лестнице	47,2	47,0	41,2	50,9
Сидение	93,1	91,9	91,1	90,1
Стояние	92,6	91,4	93,1	92,4

Выбор ПО для решения проблемы

Фреймворк	Основной язык	Связки	Лицензия
Caffe	C++/**CUDA**	Python/MATLAB	BSD
Theano	**CUDA**	Python	BSD
Torch7	C/**CUDA**	LuaJIT	BSD
ArrayFire	C++/**CUDA**	C++	BSD 3.0 Clause
Cuda-CNN	C++/**CUDA**	MATLAB	BSD
cuda-convnet2	C++/**CUDA**	Python	Apache 2.0 License
cuDNN	**CUDA**	Python/MATLAB	BSD
CXXNET	C++/**CUDA**	C++	Apache 2.0 License
Deeplearning4j	**CUDA**	Java	Apache 2.0 License
nnForge	**CUDA**	C++	Apache 2.0 License
Decaf	Python	Python	BSD
DeepLearnToolbox		Python/MATLAB	BSD
deepmat		MATLAB	GNU GPL
Deepnet		MATLAB	BSD
Groundhog	Python	Python	BSD 3.0 Clause
Lasagne	Python	Python	MIT License
OverFeat	Lua	C++, Python	-
PyBrain	Python	Python	BSD
Pylearn2	Python	Python	BSD 3.0 Clause
RNNLIB	C++	Python	GNU GPL v.3.0

Конфигурация технической GPU-системы

Параметры сервера

Атрибут	**Значение**
Семейство	GPU машины
Тип	g2.2xlarge
Количество vCPU	8
Память (ГБ)	15
Дисковое подсистема (ГБ)	1 x 60 (SSD)
Быстродействие сети	Высокое
ОС	Ubuntu 14.04.2
Python	Python 2.7.6
Theano	0.7.0

Результаты – Theano-модель классификации

Параметры модели:

- 6-слойный персептрон с 10 нейронами в каждом скрытом слое (43-10-10-10-10-10-6).
- Стохастический градиентный спуск "mini-batch" (200 элементов в каждом).
- Скорость обучения — 0.1.
- L1 регуляризация — 0.01.
- L2 регуляризация — 0.0001.
- Количество эпох — 1000.

Результаты эксперимента:

- Общее время: 27 секунд
- Наилучшая точность (тестовая выборка): 84.1 %
- Наилучшая точность (контрольная выборка): 86.3 %

Результаты работы

1. Разработан алгоритм автоматического построения оптимальной структуры сетей глубокого обучения для задач классификации временных рядов.
2. Предложенное решение превосходит модель OBD по точности при равном количестве нейронов.
3. Создан прототип технической системы на базе графических ускорителей для автоматического построения сетей глубого обучения.
4. Полученные результаты показывают, что использование Theano серьезно повышает скорость процесса обучения сети.
5. Поэтому рекомендуется дальнейшая разработка Theano-модели классификации.