ПАРАЛЛЕЛЬНОЕ ОБУЧЕНИЕ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ НА ОСНОВЕ СИСТЕМЫ ГРИД-ВЫЧИСЛЕНИЙ ANTHILL

Нгуен Занг Зуи Чыонг, А.А. Краснощёков

Ключевые слова: *распределенные вычисления, грид-вычисления, искусственные нейронные сети, параллельное обучение нейронных сетей, алгоритм обратного распространения ошибки, алгоритмы обучения нейронных сетей, слабосвязанные задачи, python, Django.*

В данной работе описан процесс проектирования, реализации и внедрения системы распределенного обучения нейронных сетей на основе алгоритма learning-by-block для нейронных сетей прямого распространения. Система базируется на созданной ранее open-source платформе для проведения grid-вычислений Anthill. Применение концепции grid-вычислений, возможность управления через web-интерфес, свободного выбора библиотек для моделирования нейронных сетей позволило создать систему, обладающую высокой степенью гибкости и простоты использования.

I. Введение

В последние годы распределенные вычисления открыли новые пути приложениям, требующим больших вычислительных мощностей. На сегодняшний день, в связи с ростом объёмов данных, можно говорить о практически повсеместном использовании распределенных вычислений в научной среде [1]. Но в то же время, применение распределенных вычислений для обучения искусственных нейронных сетей (ИНС) является относительно новой задачей и было мало исследовано [2, 3].

Ещё в 40-ых годах прошлого века достижения нейробиологии позволили создать первую искусственную нейронную сеть, которая имитировала деятельность человеческого мозга. Но только через несколько десятилетий, вместе с появлением современных компьютеров и адекватного программного обеспечения стала возможной разработка сложных приложении в области ИНС. С этого момента теория нейронных сетей стала одним из наиболее перспективных направлений научных исследований. Этому способствовала сама природа параллельных вычислений и практически доказанная возможность адаптивного обучения нейронных сетей.

С увеличением сложности задач решаемых современной наукой требуются всё большие вычислительные мощности. Это требует серьёзных инвестиций в модернизацию вычислительных систем. Но в действительности уже имеющиеся ресурсы расходуются довольно расточительно и компьютеры в научных организациях работают только на максимум 10% своей мощности, серверы на 30% [3]. При рациональном использовании уже имеющихся ресурсов можно осуществлять существенные объёмы вычислений.

Технология grid-вычислений призвана решить задачу эффективного использования ресурсов, принадлежащих к одной или нескольким организациям в глобальном масштабе. Grid-вычисления, подтвердили свое место в науке такими проектами, как: Boinc, EGEE, AntHill.

Таким образом, растущая потребность в вычислительных мощностях и высокая степень интеграции распределённых вычислительных ресурсов делают создание платформы для обучения нейронных сетей приоритетной задачей. В данной работе изучены подходы к построению и параллельному обучению ИНС, преимущества и недостатки каждой отдельной архитектуры. Исходя из этого, была создана распределенная вычислительная платформа для ресурсоёмкого обучения нейронных сетей DisANN (Distributed Artificial Neural Network).

II. Основные концепции DisANN

В настоящее время распределенные вычислительные системы мало задействованы для решения задач с применением нейронных сетей. Проведя исследование существующих технологии распределенного обучения нейронных сетей [4,5,6] были сформулированы основные концепции для построения и функционирования программной платформы удовлетворяющей актуальным потребностям в моделировании нейронных сетей. Использование свободных мощностей, волонтёрские вычисления, стандартные интерфейсов и протоколы, масштабируемость, высокие требования к качеству услуг являются одними из стандартных подходов при создании grid-систем. В рамках реализации механизма параллельного обучения нейронных сетей были приняты следующие концепции:

- Минималистичный подход к распределению нейронных сетей

Для решения широкого круга задач необходимо распределение не самой нейронной сети, а только обучающей выборки. Массив обучающих векторов разделяется на несколько блоков, а блоки в свою очередь, распределяются между вычислительными узлами. Таким образом, обучение ИНС распараллеливается в рамках одной эпохи. Процесс обучения относится к классу слабосвязанных задач.

- Свободный выбор архитектуры ИНС

Архитектуры ИНС крайне разнообразны и для их применения к конкретной задачи зачастую требуется не только изменение параметров, но иногда и изменение самой модели. Поэтому система должна поддерживать модульную интеграцию программных библиотек реализующих различные типы нейронных сетей.

- Потери данных

Так как ИНС по принципу своей работы являются устойчивыми к ошибкам, то потери блоков не критичны. Более того, для избежания узких мест в момент синхронизации в конце каждой эпохи, обучение может проводиться с потерями. Для синхронизации и перехода к следующей необходимо и достаточно определённого процента использованных для обучения блоков или заданного объёма времени процессора. В рамках каждой эпохи потери в обучении компенсируются взвешенным значением последнего успешного результата:

$$w_{ij}^{k+1} = w_{ij}^{k} + 0.5^{k-g} \nabla w_{ij}^{g} \quad (1)$$

где i,j – индексы нейронов, k – номер текущей эпохи, g – номер последней успешной эпохи

- Высокая скорость вычислений

Для разработки системы требуется простой и популярный в научной среде кроссплатформенный язык программирования. Исходный код программы должен быть минималистичен и лёгок в управлении и развитии. Руководствуясь этим, основным языком для разработки системы был выбран Python. Код на Python несёт обобщающую функцию, а все критические задачи были оптимизированы. Реализованы механизмы кэширования, оптимизированы запросы к базе данных, обмен изменениями весов ИНС происходит в бинарном формате, поддерживаются многопроцессорных системы.

III. Разработка DisANN

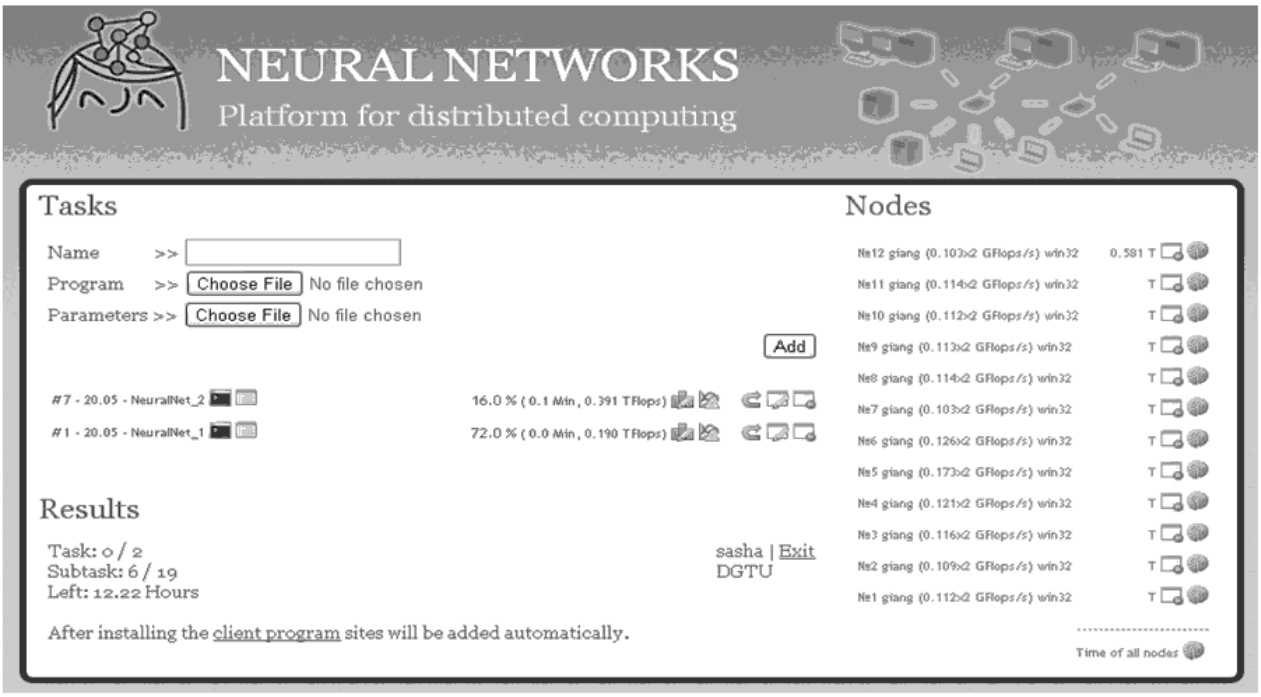

Рис. 1. Web-интерфейс системы

Руководствуясь концепциями, описанными в предыдущей главе, была создана программная платформа «DisANN» с открытым исходным кодом [4]. Система была реализована в виде клиент-серверной модели, управляемой посредством web-интерфейса созданного на основе Django (Рис 1.).

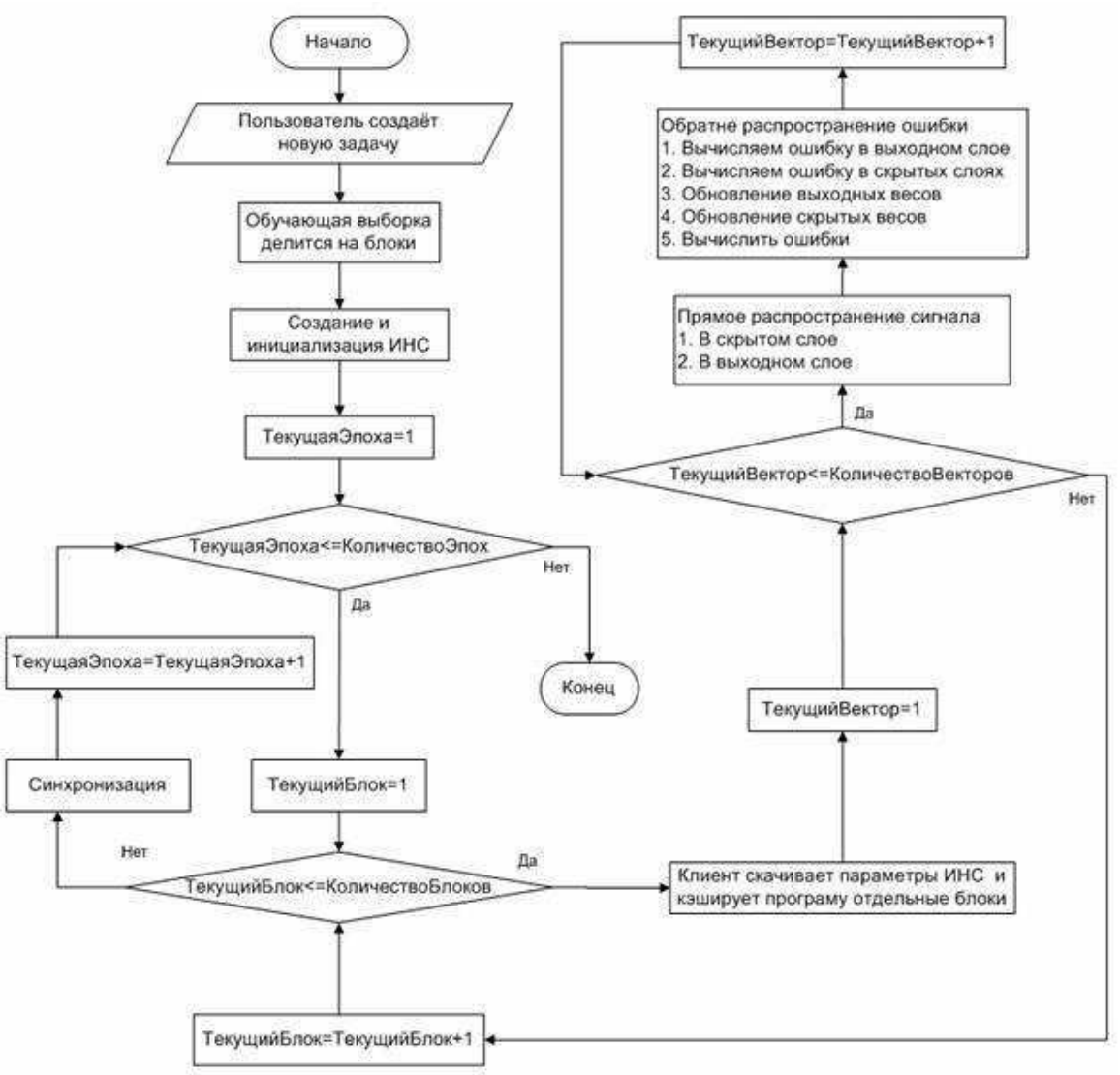

Рис. 2. Алгоритм функционирования системы

Чтобы проиллюстрировать работу системы рассмотрим пример решения задачи. Пользователь должен войти в web-интерфейс и загрузить файл с описанием структуры ИНС и файл с обучающей выборкой.

Далее, обучающая выборка разбивается на заданное количество блоков и эти блоки вместе со структурой нейронной сети распределяются по вычислительным узлам.

На каждом узле проходит обучение сети на основе полученного блока и на сервер возвращаются бинарные изменения весов. В конце каждой эпохи сервер обновляет модель ИНС и снова рассылает её клиентам (Рис 2).

Введём следующие обозначения:

Ni – размер входного слоя, *Nh* – размер скрытого слоя, *No* – размер выходного слоя,

N – скорость, *M* – момент, *Ai* – входной вектор *(a1, a2, …, ani)*,

Ti – вектор выходных данных *(t1, t2, …, tno)*

Прямое распространение сигнала:

1. В скрытом слое

$ah_j = f(\sum_{i=1}^{ni} ai_i * wi_{ij}), j = \overline{1, nh-1}$, *ah* - выходной вектор скрытого слоя

2. В выходном слое

$ao_k = f(\sum_{i=1}^{nh} ah_j * wo_{jk}), k = \overline{1, no}$, ao - выходной вектор сети

Обратное распространение ошибки:

1. Вычисляем ошибку в выходном слое

$bo_k = f(ao_k) * (t_k - ao_k), k = \overline{1, no}$, где *bo* - значение ошибки, используемой для изменения выходных весов

2. Вычисляем ошибку в скрытых слоях

$bh_j = f(ah_j) * (\sum_{k=1}^{no} bo_k * wo_{jk}), j = \overline{1, nh}$, где *bh* - значение ошибки, используемой для изменения скрытых весов

3. Обновление выходных весов

$wo_{jk} = wo_{jk} + n * bo_k * ah_j + m * \nabla w_{jk}$

$\nabla w_{jk} = bo_k * ah_j, j = \overline{1, nh}, k = \overline{1, no}$ где *wo* - вес соединяющий скрытый нейрон *j* с *k* в выходном слое сети

4. Обновление скрытых весов

$wh_{ij} = wh_{ij} + n * bh_j * ai_i + m * \nabla w_{ij}$

$\nabla w_{ij} = bh_j * ai_i, i = \overline{1, ni}, j = \overline{1, nh}$, где *wh*: вес соединяющий входной нейрон *j* с *k* в скрытом слое сети

5. Вычисляем ошибки

$E = \sum_{k=1}^{no} \frac{1}{2} (t_k - ao_k)^2$, где *E* - суммарная ошибка (для одной эпохи)

IV. Результаты испытаний

Было протестировано несколько обучающих выборок. Эффективность распределения обучения может быть наглядно показана рисунке 3. Обучающая выборка составила 10000 векторов, вычисления производились на одном и 10 компьютерах соответственно. Одна и та же нейронная сеть в случае обучения на одном компьютере не позволяет добиться желаемого значения ошибки (слева), в то же время на 10 компьютерах сеть обучается в полной мере (справа).

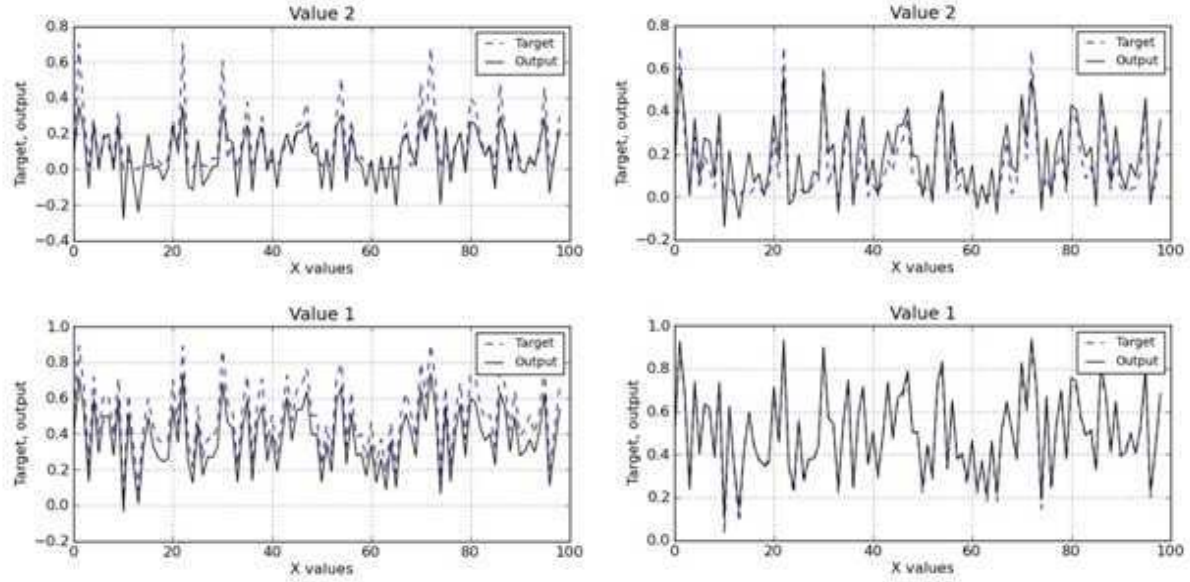

Рис. 3. Результаты обучения сети. Вверху - начало обучения, внизу — результат обучения в течении 10 сек.. Слева — результаты для 1 компьютера, справа — для 10 компьютеров. Пунктирная линия — желаемое значение, сплошная — выход ИНС.

Следует отметить, что в среде с потерями данных функция ошибки заведомо не будет гладкой (рис.4)

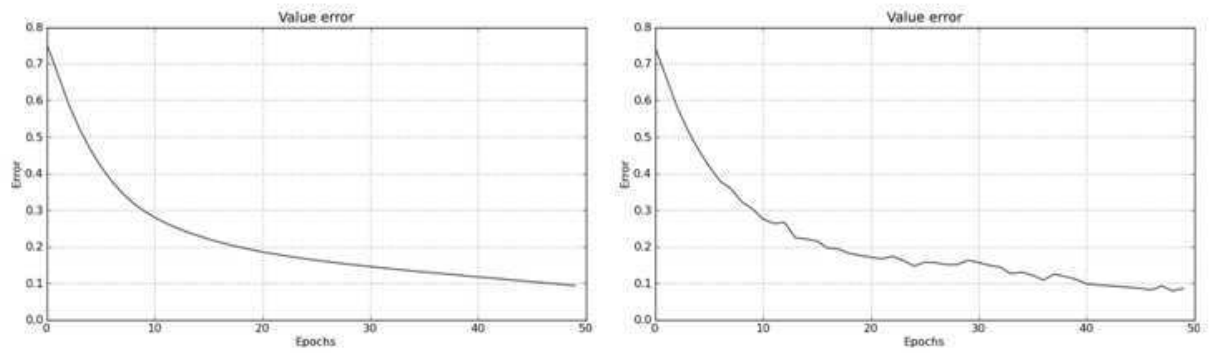

Рис. 4. График среднеквадратичной (MSE) ошибки в зависимости от количества эпох обучения ИНС.

При распределении заданий между вычислительными узлами небольшой процент мощности тратиться на обеспечение функционирования системы (Рис 5.). Время отправки / получения данных будет больше, но итоговое время обучения сети - во много раз меньше, чем в случае обучения на одной машине.

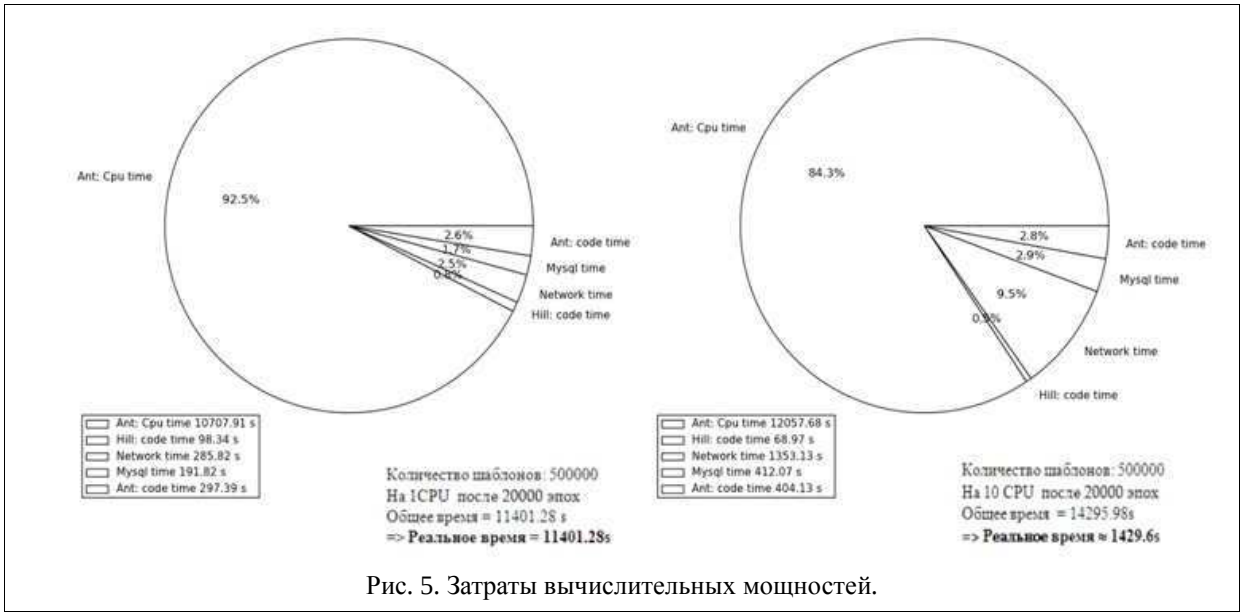

Рис. 5. Затраты вычислительных мощностей.

V. Практическое применение

В настоящее время система DisANN используется для решения для обратных задач механики твёрдого тела на кафедре «Информационные Технологии» Донского государственного технического университета. Система показала высокую эффективность при обучении сетей размером до 10000 нейронов (Рис 6.).

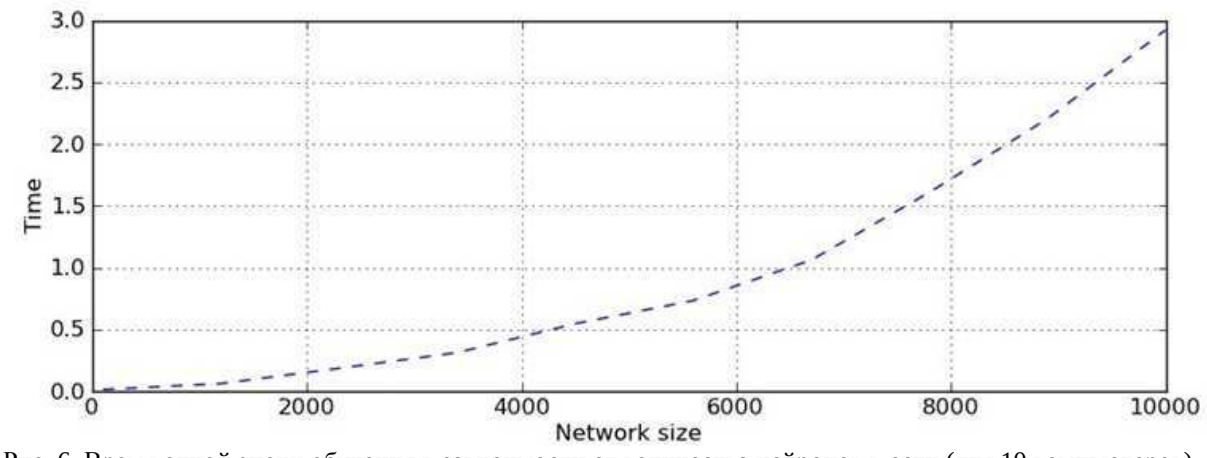

Рис. 6. Время одной эпохи обучения в зависимости от количества нейронов в сети (для 10 компьютеров).

V. Выводы

В результате проведённого исследования была успешно разработана программная платформа, которая позволяет реализовать распределённое обучение нейронных сетей. Пользователи могут подключать свои собственные модели нейронных сетей. Программа была оптимизирована с точки зрения производительности, функциональности и надёжности.

Система может быть легко применена для создания импровизированного дата-центра и обеспечения исследователей расчётами, при минимальных временных затратах на развёртывание и функционирование.

Авторы выражают сердечную благодарность за руководство над проведением исследования и разработки д.т.н., проф. Соболю Борису Вдадимировичу.

Работа выполнена при частичной финансовой поддержке РФФИ, проект № 10-08-00839.

ЛИТЕРАТУРА:

1. Distributed computing // URL: http://en.wikipedia.org/wiki/Distributed_computing
2. *Simon Haykin*, Neural Network a comprehensive foundation(2nd edition), Prentice Hall, 842 pages, 1998
3. *N. Sundararajan, P. Saratchandran*, Parallel architectures for Artificial Neural Networks, Wiley-IEEE Computer Society Press, 412 pages, 1998
4. GitHub // URL: http://github.com/s74/disann, Дата обращения 21.04.2012.
5. Distributed computing on internet // URL: http://fgouget.free.fr/distributed/index-en.shtml
6. *Lisandro Daniel Dalcin*, Techniques for High-Performance distributed computing in computational fluid mechanics, CIMEC Document Repository, 102 pages, 2008.