Н.Г.Аксак

ПАРАЛЛЕЛЬНАЯ НЕЙРООБРАБОТКА БОЛЬШИХ ДАННЫХ В РАСПРЕДЕЛЕННОЙ СРЕДЕ НА ОСНОВЕ MAPREDUCE

Аннотация. *Предложены методы ускоренного обучения и функционирования многослойной нейронной сети прямого распространения в компьютерных кластерах с различными топологиями передачи данных. Разработанные MapReduce модели нейрообработки больших данных позволили равномерно планировать вычислительную нагрузку.*

***Ключевые слова*:** *модель MapReduce, параллельная реализация, многослойная нейронная сеть, облачные вычисления.*

Abstract. *Methods for speed up training and functioning of the multilayered feedforward neural network in computer clusters with different virtual topologies are developed. The balanced planning of processing power is executed by developed MAPREDUCE model of neuroprocessing big data.*

Keywords: *model MapReduce, Parallel implementation, Multi-layer neural network, Cloud computing.*

Введение и постановка задачи. С быстрым развитием сетевых технологий и облачных вычислений современная информация вступает в эру больших данных. Реальные данные (аудио и видео форматы, структуры генома, базы данных изображений лица, веб-страницы и т.д.) обычно характеризуются большим множеством показателей, свойств, атрибутов и имеют огромные объемы. Существующие технологии искусственного интеллекта сталкиваются с проблемой обработки больших данных. Например, при решении задач обработки и анализа изображений (контроль топологии печатных плат, диагностика, робототехника, аэрокосмические съемки, цифровая картография, системы безопасности, медицина [1,2] и др.) на больших наборах данных очень трудно осуществить предъявляемые требования из-за отсутствия приемлемой скорости вычислений и недостаточных объемов памяти. Нейронные сети, например, при решении задач визуализации многомерных данных [3] при малом размере обучающей выборки способны решать такие задачи эффективно и быстро. Однако, с увеличением размерности входных сигналов структура нейронной сети становится более сложной, что в свою очередь ведет к более длительному времени обучения, к замедлению скорости сходимости. В то же

время, нейронная сеть является высокопроизводительным вычислителем, поэтому алгоритмы, предназначенные для работы на такой сети, являются параллельными.

Для решения трудоемких задач активно используются параллельные и распределенные технологии – облачные вычисления, которые как вычислительная парадигма предлагают:

– инфраструктуру как услугу IaaS (Infrastructure as a Service) – вычислительная инфраструктура (серверы, хранилища данных, сети, операционные системы), которая предоставляется клиентам для разворачивания и запуска собственных программных решений);

– платформу как услугу PaaS (Platform as a Service) – полноценная среда разработки и развертывания в облаке с ресурсами, которые позволяют предоставлять любые приложения, от простых облачных приложений до продвинутых облачных приложений промышленного класса;

– программное обеспечение как услугу SaaS (software as a service) – технология продажи и использования программных продуктов, предполагающая, что заказчикам предоставляется доступ к ПО через всемирную паутину.

С обработкой и созданием больших данных тесно связана модель программирования MapReduce, разработанная компанией Google. MapReduce обеспечивает надежную, масштабируемую и отказоустойчивую вычислительную среду для хранения и обработки массивных наборов данных. В то же время возникает необходимость в обосновании выбора технологии MapReduce среди других решений для обработки больших данных. Основными критериями выбора являются размерность данных, масштабируемость системы, тип вычислений, возможность выполнения задач автоматически с целью уменьшения профессиональных требований к специалисту. Однако, производительность этой модели ограничена жесткой конфигурационной стратегией. На практике для простой работы в MapReduce большое количество параметров настройки должно быть задано конечными пользователями, которые часто сталкиваются с проблемами производительности, что может привести к потере производительности.

Вычислительная модель MapReduce в рамках пардигмы map/reduce может быть описана основными двумя фазами:

1. Map:(*key1*,*value1*)–>(*key2*,*value2*) – применяется к каждой входной паре ключ\значение (*key1*,*value1)* и выводит список промежуточных пар

ключ\значение (*key2,value2).* Здесь также можно выделить этап *Partition/Combain*. Целью этапа *Partition* (разделение) является распределение промежуточных результатов, полученных на этапе Map по Reduce-заданиям (*key2', reduces_count*)*–> (reduces_id),* где *reduces_count* – количество узлов, на которых запускается операция свертки, *reduces_id* – идентификатор целевого узла. Основной цель этого этапа является балансировка нагрузки. Некорректно реализованная функция *Partition* может привести к неравномерному распределению данных между reduce-узлами.

2. Reduce:(*key2,value2*)*–>value3* – применяется ко всем промежуточным значениям, связанным с одним и ключом, и создает список выходных значений.

Программисты определяют логику приложения с использованием этих двух примитивов. Следует отметить, что выполнение функции Map осуществляется независимо и параллельно на разных вычислителях кластера.

Наиболее популярными реализациями модели MapReduce являются Hadoop, Mars и Phoenix ++.

Hadoop – проект с открытым исходным кодом, находящийся под управлением Apache Software Foundation, используется для надежных, масштабируемых и распределенных вычислений, но может также применяться и как хранилище файлов общего назначения, способное вместить петабайты данных. Mars использует возможности GPU на графическом процессоре NVIDIA G80.

В проекте Phoenix модель MapReduce применяется на компьютерах с общей памятью.

Таким образом, возникает необходимость разработки методов и моделей обработки больших данных нейронными сетями, позволяющих равномерно планировать нагрузку вычислительного кластера.

Применение программной модели MapReduce Hadoop для нейрообработки больших данных позволило существенно ускорить вычисления с уменьшением объема передаваемой между узлами информации, а также эффективно балансировать нагрузкой вычислительных ресурсов с различными топологиями передачи данных.

Многослойная нейронная сеть с обратным распространением ошибки.

Нейронная сеть с обратным распространением является многослойной сетью прямой передачи (MLP), которая обучает входные данные с использованием механизма обратного распространения ошибок. Она стала

одной из наиболее широко используемых нейронных сетей. Во время распространения ошибки сеть продолжает настраивать свои параметры до тех пор, пока не адаптируется ко всем входным экземплярам. Типичная MLP, состоящая из произвольного количества входов показана на рисунке 1.

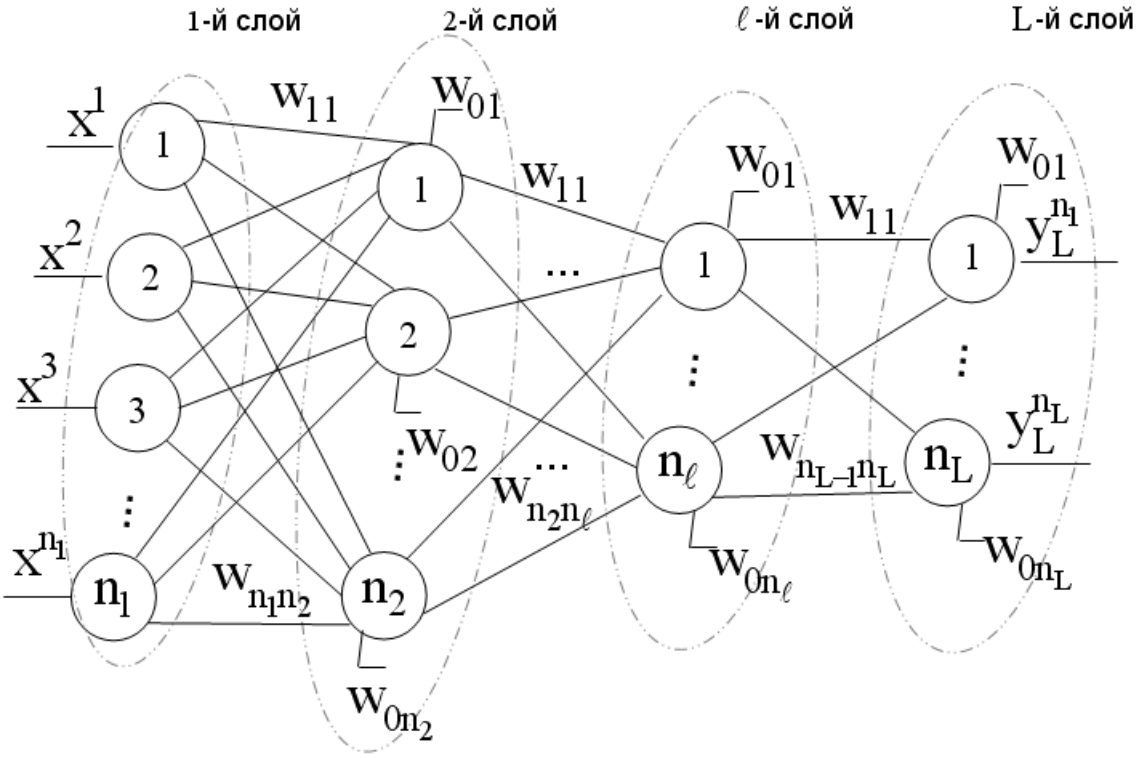

Рис. 1. Архитектура многослойной нейронной сети прямого распространения

В общем виде MLP задается архитектурой $n_1 - n_2 - ... - n_L$, где n_1 - количество нейронов во входном слое, n_ℓ - количество нейронов в скрытых слоях, n_L - количество нейронов в выходном слое. Обучающая выборка $\{(X(1), D(1)), (X(2), D(2)), ..., (X(K), D(K))\}$ состоит из K примеров $X(k) = \left[x^1(k), ..., x^{n_L}(k)\right]^T$ и соответствующих им значений целевого признака $D(k) = \left[d^1(k), ..., d^{n_L}(k)\right]^T$, $k = \overline{1,K}$. Для обработки обучающей и тестовой выборок требуется U эпох обучения.

В общем виде метод обратного распространения ошибки включает следующие выражения для пакетного режима обучения.

Вектор выходов нейронов $Y_\ell(k) = \left[y_\ell^1(k), ..., y_\ell^{n_\ell}(k)\right]^T$ слоев $\ell = \overline{1,L}$

вычисляется в соответствии с

$$Y_\ell(k) = f\left(W_\ell(k)Y_{\ell-1}(k) + W_0^T(k)\right) =$$
$$= f\left(\begin{bmatrix} w_{11}(k) & w_{21}(k) & \dots & w_{n_{\ell-1}1}(k) \\ w_{12}(k) & w_{22}(k) & \dots & w_{n_{\ell-1}2}(k) \\ \dots & \dots & \dots & \dots \\ w_{1n_\ell}(k) & w_{2n_\ell}(k) & \dots & w_{n_{\ell-1}n_\ell}(k) \end{bmatrix} \times \begin{bmatrix} y_{\ell-1}^1(k) \\ y_{\ell-1}^2(k) \\ \dots \\ y_{\ell-1}^{n_{\ell-1}}(k) \end{bmatrix} + \begin{bmatrix} w_{01}(k) \\ w_{02}(k) \\ \dots \\ w_{0n_\ell}(k) \end{bmatrix}\right). \quad (1)$$

Причем $Y_1(k) = X(k)$, k – текущий пример обучающей выборки, $f(\bullet)$ – активационная функция, $W_\ell(k) = (w_{ij}(k))_{i=o,j=1}^{n_{\ell-1},n_\ell}$ – матрица весовых коэффициентов слоев $\ell = \overline{2,L}$, в которой элементы нулевой строки $W_0(k) = \left[w_{01}(k), w_{02}(k), ..., w_{0n_\ell}(k)\right]$ матрицы $W_\ell(k)$ представляют собой смещения соответствующего слоя.

Вектор ошибок реакции нейронов $\Sigma_\ell(k) = \left[\delta_\ell^1(k), ..., \delta_\ell^{n_\ell}(k)\right]^T$ определяется для выходного L-го слоя

$$\Sigma_L(k) = Y_L(k)\left(1 - Y_L(k)\right)\left(D(k) - Y_L(k)\right) \quad (2)$$

и для скрытых слоев $\ell = \overline{L-1,2}$

$$\Sigma_L(k) = Y_\ell(k)\left(1 - Y_\ell(k)\right)\Sigma_{\ell+1}(k)W_{\ell+1}(k) =$$
$$= \begin{bmatrix} y_\ell^1(k) \\ y_\ell^2(k) \\ \dots \\ y_\ell^{n_\ell}(k) \end{bmatrix}\left(1 - \begin{bmatrix} y_\ell^1(k) \\ y_\ell^2(k) \\ \dots \\ y_\ell^{n_\ell}(k) \end{bmatrix}\right) \times \begin{bmatrix} w_{11}(k) & w_{12}(k) & \dots & w_{1n_{\ell+1}}(k) \\ w_{21}(k) & w_{22}(k) & \dots & w_{2n_{\ell+1}}(k) \\ \dots & \dots & \dots & \dots \\ w_{n_\ell 1}(k) & w_{n_\ell 2}(k) & \dots & w_{n_\ell n_{\ell+1}}(k) \end{bmatrix}\begin{bmatrix} \delta_{\ell+1}^1(k) \\ \delta_{\ell+1}^2(k) \\ \dots \\ \delta_{\ell+1}^{n_{\ell+1}}(k) \end{bmatrix}, \quad (3)$$

где роль единицы играет единичный вектор $1 = \left[1,1,...,1\right]^T$.

Настройка весовых коэффициентов слоев $\ell = \overline{2,L}$ нейронов $j = \overline{1,n_\ell}$ осуществляется в соответствии

$$w_{ij}(k+1) = \eta\delta_\ell^j(k)y_\ell^j(k) + \alpha w_{ij}(k), \quad (4)$$

где η – параметр скорости обучения, α – коэффициент инерционности.

Определение текущей среднеквадратичной ошибки обучения E осуществляется на основе

$$E = \frac{1}{2}\sum_{k=1}^{K}\left(D(k) - Y_L(k)\right)^2. \quad (5)$$

Определение ошибки обобщения во время тестирования нейронных сетей осуществляется в соответствии с

$$E_g = \frac{1}{K}\sum_{t=1}^{T}\Delta_t, \quad (6)$$

где Δ_t определяется на основе

$$\Delta_t = \begin{cases} 1, \text{ если } D(t) \neq y(t), \\ 0, \text{ если } D(t) = y(t). \end{cases}$$

Данный метод предложен в различных вариациях, например, с использованием равномерного критерий качества обучения или адаптивного упрощения обучающей выборки, а также в пространстве вейвлет-преобразования.

Параллельная реализация многослойной нейронной сети на основе модели MapReduce Hadoop.

При разработке эффективных параллельных и распределенных процедур обучения и функционирования MLP для решения трудоемких задач необходимо учесть большое число взаимосвязанных данных:

- исходные параметры: структура MLP, объемы обучающей и тестовой выборок, значение количества эпох обучения;

- аппаратные характеристики распределенной вычислительной среды: количество вычислителей, физическая топология среды, пропускная способность, латентность;

- алгоритмические характеристики: метод декомпозиции процедуры, характер информационных взаимодействий (топология передач данных).

Будем использовать кластер Hadoop, который имеет свою распределенную файловую систему HDFS (Hadoop Distributed File System), обеспечивающую высокоскоростной доступ к данным приложения с целью управления данными. В кластере Hadoop есть один узел имен (Namenode) и

несколько узлов данных (Datanodes) для выполнения заданий. Namenode управляет метаданными кластера, Datanodes является фактическим узлом обработки. Функции карты (mappers) и функции сжатия (редукторы) выполняются на узлах данных. Когда задание отправляется в кластер Hadoop, входные данные делятся на небольшие куски одинакового размера и сохраняются в HDFS.

Для сокращения времени обучения и функционирования МНС, а также для эффективного использования вычислительных ресурсов необходима динамическая корректировка формируемого состава подзадач: при малом числе вычислителей объем подзадач увеличивается или, наоборот, уменьшается в противном случае.

Для равномерной загрузки машин кластера разобъем нейроны каждого слоя MLP на группы для параллельной обработки с учетом кластерной архитектуры – «звезда» и «полносвязный граф» [4]. Для каждой топологии разработаем модель параллельной реализации многослойной нейронной сети на основе MapReduce (MRMLP).

Метод равномерного распределения нейросетевой обработки данных в вычислительной среде с топологией «звезда»

Каждый обучающий пример и информация, определяемая форматом (7), сохраняются в одном файле HDFS. В каталоге HDFS может располагаться любое количество файлов. Для каждого обучающего примера выполняются соотношения (7) - (14), которые будут выполняться до тех пор, пока не будет достигнута желаемая точность в соответствии с (5) - (6).

При первом запуске функции Map (Mapping process 1) инициализируется одна запись из файла HDFS в следующем формате

$$\left\langle \mathrm{KeyM1_h}, \mathrm{X(k)}, \mathrm{W_{\ell_h}(k)}, \mathrm{D_h(k)} \right\rangle, \tag{7}$$

где $\mathrm{KeyM1_h}$ – представляет собой h-ый запуск функции Map, $\mathrm{h} = \overline{0, \mathrm{P}-1}$;

$\mathrm{W_{\ell_h}(k)}$ – подматрицы весов размерностью $\mathrm{n_{\ell-1}} \times \left|\mathrm{g_h}\right|$ матрицы

$\mathrm{W_\ell(k)} = \left\lfloor \mathrm{W_{\ell_1}(k), W_{\ell_2}(k), ..., W_{\ell_{p-1}}(k)} \right\rfloor$ $(\ell = \overline{0, \mathrm{L}-1})$;

$\mathrm{D_h(k)}$ – группы значений целевого признака $\mathrm{D(k)} = \left\lceil \mathrm{D_1(k), D_2(k), ..., D_{p-1}(k)} \right\rceil^{\mathrm{T}}$.

На выходе этой функции вычисляются $\mathrm{g_h}$-е выходы нейронов с использованием соотношения (1)

$$\left\langle \mathrm{KeyM1_h}, \mathrm{Y_{\ell_h}(k)}, \left\{ \mathrm{W_{\ell_h}(k)}, \mathrm{D_h(k)} \right\} \right\rangle, \tag{8}$$

где $\mathrm{Y_{\ell_h}(k)}$ – группы наборов нейронов $\mathrm{Y_\ell(k)} = \left[\mathrm{Y_{\ell_1}(k)}, \mathrm{Y_{\ell_2}(k)}, ..., \mathrm{Y_{\ell_{p-1}}(k)} \right]^{\mathrm{T}}$, $(\ell = \overline{2, \mathrm{L}})$;

$$\mathrm{g_h} = \begin{cases} (\mathrm{h}-1)\left(\left[\mathrm{n_\ell} / \mathrm{P}-1 \right] + 1 \right) + \varphi, \ \varphi = \overline{1, \left(\left[\mathrm{n_\ell} / \mathrm{P}-1 \right] + 1 \right)}, & \text{если } 1 \le \mathrm{h} \le \mathrm{b}; \\ (\mathrm{P}-1-\mathrm{h_d})\left(\left[\mathrm{n_\ell} / \mathrm{P}-1 \right] + 1 \right) + (\mathrm{h}-\mathrm{P}-1+\mathrm{b}-1)\left[\mathrm{n_\ell} / \mathrm{P}-1 \right] + \varphi, \ \varphi = \overline{1, \left[\mathrm{n_\ell} / \mathrm{P}-1 \right]}, & \text{если } \mathrm{b} \le \mathrm{h} \le \mathrm{P}-1; \end{cases} \tag{9}$$

$\mathrm{g_h}$ – возрастающая последовательность, элементами которой являются номера нейронов $\ell = \overline{2, \mathrm{L}}$ слоев, которые обрабатываются h-м процессором;
$b = n_\ell \bmod(P-1)$ – номер вычислителя, после которого для $h_d = P-1-b$ компьютеров, начиная с $(b+1)$-го процессора, количество обрабатываемых нейронов уменьшается на один для равномерной загрузки.

Функция Reduce (Reducing process 2), связанная с $KeyM1_h$, собирает все результаты $Y_{\ell_h}(k)$ $(h = \overline{0, P-1})$

$$\left\langle \mathrm{KeyR2_0}, \mathrm{Y_\ell(k)}, \left\{ \mathrm{Y_{\ell_h}(k)} \right\} \right\rangle. \tag{10}$$

Следующий запуск фукции Map (Mapping process 3) формирует локальные ошибки для $\mathrm{g_h}$-х наборов нейронов выходного слоя в соответствии с (2)

$$\left\langle \mathrm{KeyM3_h}, \Sigma_{\mathrm{L_h}}(\mathrm{k}), \left\{ \mathrm{Y_\ell(k)}, \mathrm{D_h(k)} \right\} \right\rangle, \tag{11}$$

где $\Sigma_{\mathrm{L_h}}(\mathrm{k})$ – группы ошибок реакции нейронов $\Sigma_{\mathrm{L}}(\mathrm{k}) = \left[\Sigma_{\mathrm{L_1}}(\mathrm{k}), \Sigma_{\mathrm{L_2}}(\mathrm{k}), ..., \Sigma_{\mathrm{L_{p-1}}}(\mathrm{k}) \right]^{\mathrm{T}}$.

И соответственно функция Reduce (Reducing process 4) формирует общий вектор $\Sigma_{\mathrm{L}}(\mathrm{k})$

$$\left\langle \mathrm{KeyR4_0}, \Sigma_{\mathrm{L}}(\mathrm{k}), \left\{ \Sigma_{\mathrm{L_h}}(\mathrm{k}) \right\} \right\rangle. \tag{12}$$

На основе (3) функция Map (Mapping process 5) определяет локальные

ошибки $\Sigma_{\ell_h}(k)$ g_h-х наборов нейронов скрытых слоев $\ell = \overline{L-1,2}$ для создания векторов $\Sigma_\ell(k)$

$$\left\langle KeyM5_h, \Sigma_{\ell_h}(k), \left\{ Y_\ell(k), \Sigma_{\ell+1}(k), W_{\ell+1}(k) \right\} \right\rangle, \tag{13}$$

где $\Sigma_\ell(k) = \left[\Sigma_{\ell_1}(k), \Sigma_{\ell_2}(k), ..., \Sigma_{\ell_{p-1}}(k) \right]^T$, $(h = \overline{1,P-1})$.

Далее функция Reduce (Reducing process 6) формирует общий вектор $\Sigma_\ell(k)$

$$\left\langle KeyR6_0, \Sigma_\ell(k), \left\{ \Sigma_{\ell_h}(k) \right\} \right\rangle. \tag{14}$$

Корректировка элементов подматриц $W_{\ell_h}(k+1)$ размерностью $n_{\ell-1} \times |g_h|$ матриц $W_\ell(k+1)$ слоев $\ell = \overline{2,L}$ в соответствии с (4) осуществляется функцией Map (Mapping process 7)

$$\left\langle KeyM7_h, W_{\ell_h}(k+1), \left\{ Y_\ell(k), W_\ell(k) \right\} \right\rangle. \tag{15}$$

Сборка матриц $W_\ell(k+1)$ осуществляется функцией Reduce (Reducing process 8)

$$\left\langle KeyR8_0, W_\ell(k+1), \left\{ W_{\ell_h}(k+1) \right\} \right\rangle. \tag{16}$$

Вычисление среднеквадратичной ошибки обучения E осуществляется на основе (5).

На рис. 2. приведены примеры распределения наборов нейронов в соответствии с (9) для ускоренного обучения нейронной сети размером 5-4-3 с топологией передачи данных «звезда» (P = 4).

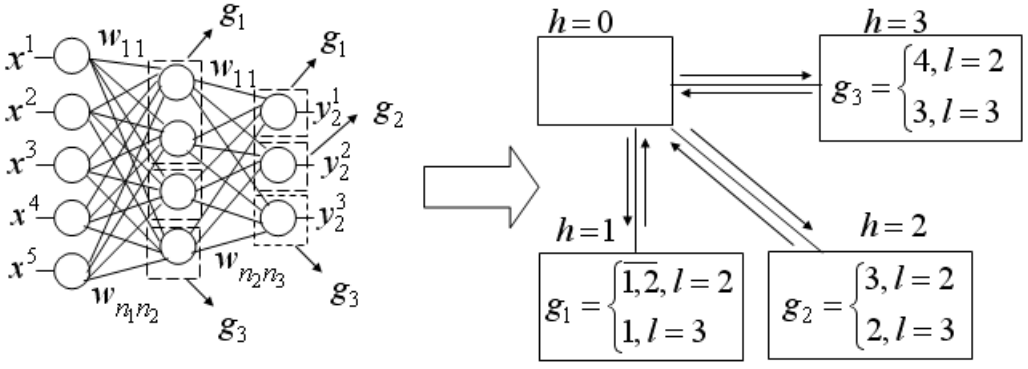

Рис. 2. Пример отображения g_h-х наборов нейронов MLP на топологию передач данных «звезда»

Графовая модель нейрообработки данных на основе MapReduce представлена на рис. 3.

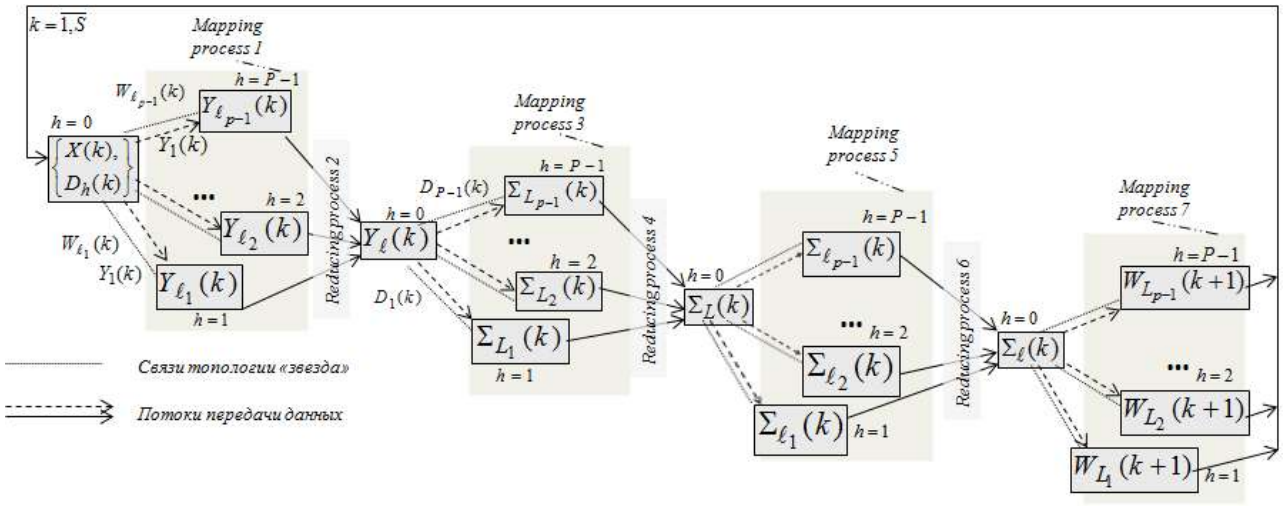

Рис. 3 Модель MapReduce параллельной нейрообработки данных в распределенной среде с топологией «звезда»

Вершинам графа соответствуют независимые параллельные операции, а направленным дугам – передачи данных между вычислителями (результаты операций Map передаются как аргументы операциям Reduce).

Метод равномерного распределения нейросетевой обработки данных в вычислительной среде с топологией «полносвязный граф».

Каждый обучающий пример и информация, определяемая форматом (17), сохраняются в одном файле HDFS. Для каждого обучающего примера выполняются соотношения (17) - (22) до тех пор, пока не будет достигнута желаемая точность в соответствии с (5) - (6).

Первый запуск фукции Map (Mapping process 1) инициализирует одну запись из файла HDFS в следующем формате

$$\left\langle \mathrm{KeyM1_h}, \mathrm{X(k)}, \mathrm{W_{\ell_h}(k)}, \mathrm{D(k)} \right\rangle, \tag{17}$$

где $\mathrm{KeyM1_h}$ – представляет собой h-ый запуск функции Map, $\mathrm{h} = \overline{0, \mathrm{P}-1}$;

$\mathrm{W_{\ell_h}(k)}$ – подматрицы весов размерностью $\mathrm{n_{\ell-1}} \times \left|\mathrm{r_h}\right|$ матриц $\mathrm{W_\ell(k)} = \left\lfloor \mathrm{W_{\ell_0}(k), W_{\ell_1}(k), ..., W_{\ell_{p-1}}(k)} \right\rfloor$ $(\ell = \overline{2, \mathrm{L}})$ для текущего k-го примера обучающей выборки значений целевого признака

$D(k)=\left[D_0(k),D_1(k),...,D_{p-1}(k)\right]^T$, определяемых в соответствии с наборами нейронов в возрастающей последовательности r_h (18), элементами которой являются номера нейронов $\ell=\overline{2,L}$ слоев, которые обрабатываются h-м процессором;

$$r_h=\begin{cases} h\left(\left[n_\ell/P\right]+1\right)+\varphi,\ \ \varphi=\overline{1,\left(\left[n_\ell/P\right]+1\right)}, & \text{если } 0\le h\le b; \\ \left(P-h_d\right)\left(\left[n_\ell/P\right]+1\right)+\left(h-P+b\right)\left[n_\ell/P\right]+\varphi,\ \ \varphi=\overline{1,\left[n_\ell/P\right]}, & \text{если } b\le h\le P-1; \end{cases} \quad (18)$$

$b=n_\ell \bmod(P)$ – номер вычислителя, после которого для $h_d=P-b$ компьютеров, начиная с $(b+1)$-го процессора, количество обрабатываемых нейронов уменьшается на один для равномерной загрузки.

В результате работы функции будут вычислены r_h-е выходы нейронов с использованием соотношения (1)

$$\left\langle KeyM1_h, Y_{\ell_h}(k), \left\{W_{\ell_h}(k), D_h(k)\right\}\right\rangle, \quad (19)$$

где $Y_{\ell_h}(k)$ – группы наборов нейронов $Y_\ell(k)=\left[Y_{\ell_0}(k),Y_{\ell_1}(k),...,Y_{\ell_{p-1}}(k)\right]^T$, $\ell=\overline{2,L}$, $h=\overline{0,P-1}$.

Функции Reduce (Reducing process 2_h $(h=\overline{0,P-1})$) параллельно определяют частичные значения локальных ошибок $\Sigma_L(k)=\left[\Sigma_{L_0}(k),\Sigma_{L_1}(k),...,\Sigma_{L_{p-1}}(k)\right]^T$ для r_h-х наборов нейронов выходного слоя в соответствии (2)

$$\left\langle \begin{Bmatrix} KeyR2_0, \Sigma_{L_0}(k), \left\{Y_{\ell_h}, D_h(k)\right\}, \\ KeyR2_1, \Sigma_{L_1}(k), \left\{Y_{\ell_h}, D_h(k)\right\}, \\ ..., \\ KeyR2_{P-1}, \Sigma_{L_{p-1}}(k), \left\{Y_{\ell_h}, D_h(k)\right\} \end{Bmatrix} \right\rangle. \quad (20)$$

Следующий вызов функций Reduce (Reducing process 3_h $(h=\overline{0,P-1})$)

вычисляет на основе (3) локальные ошибки $\Sigma_{\ell}(k)=\left[\Sigma_{\ell_0}(k),\Sigma_{\ell_1}(k),...,\Sigma_{\ell_{p-1}}(k)\right]^T$ для r_h -х наборов нейронов скрытых слоев

$$\left\langle\begin{Bmatrix}\mathrm{KeyR3}_0,\Sigma_{\ell_0}(k),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\},\\ \mathrm{KeyR3}_1,\Sigma_{\ell_1}(k),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\},\\ ...,\\ \mathrm{KeyR3}_{P-1},\Sigma_{\ell_{P-1}}(k),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\}\end{Bmatrix}\right\rangle. \quad (21)$$

Функции Reduce (Reducing process 4_h $(h=\overline{0,P-1})$) формирует элементы подматриц $W_{\ell_h}(k+1)$

$$\left\langle\begin{Bmatrix}\mathrm{KeyR4}_0,W_{\ell_0}(k+1),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\},\\ \mathrm{KeyR4}_1,W_{\ell_1}(k+1),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\},\\ ...,\\ \mathrm{KeyR4}_{P-1},W_{\ell_{P-1}}(k+1),\{\Sigma_{L_h}(k),Y_{\ell_h},\Sigma_{\ell+1_h}(k),W_{\ell+1}(k)\}\end{Bmatrix}\right\rangle. \quad (22)$$

На рис. 4. приведены примеры распределения наборов нейронов в соответствии с (18) для ускоренного обучения нейронной сети размером 5-4-3 с топологией передачи данных «полносвязный граф» (P = 4).

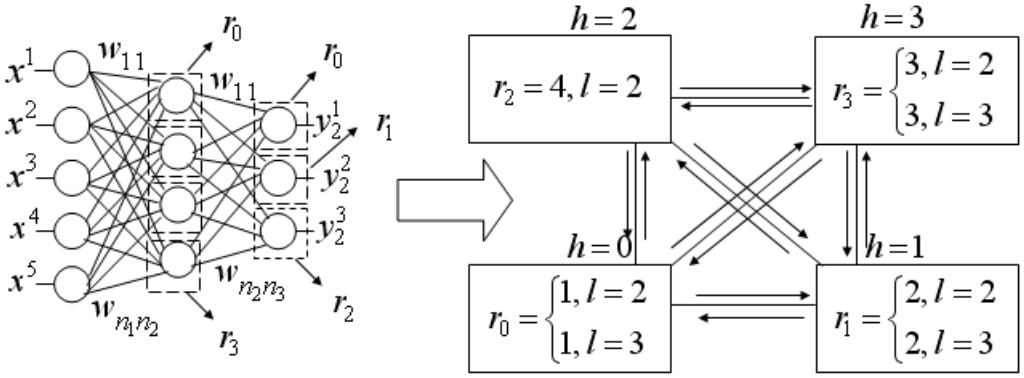

Рис. 4. Пример отображения r_h -х наборов нейронов на топологию передач данных «полносвязный граф»

Графовая MapReduce модель нейрообработки данных представлена на рис. 5

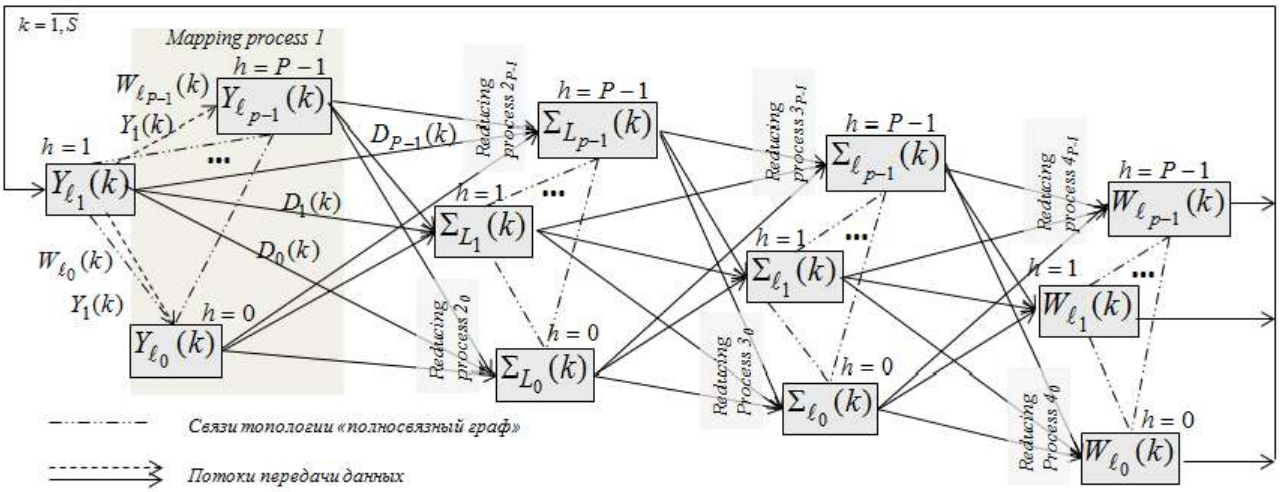

Рис. 5. Модель MapReduce параллельной нейрообработки данных в распределенной среде с топологией «полносвязный граф»

Как видно из рис. 2 и рис. 4, метод равномерного распределения нейросетевой обработки данных поэтапно и равномерно загружает вычислители работой путем автоматического распределения обрабатываемых данных в распределенной вычислительной среде с топологиями передачи данных «звезда» и «полносвязный граф». Отличия методов состоит в объеме вычислительных работ, приходящихся на один вычислитель, а также в количестве и нумерации рабочих вычислителей.

Моделирование.

Для оценки эффективности алгоритмов создан экспериментальный кластер Hadoop (версия 2.7.5), состоящий из 5 компьютеров (Intel Core 2 Quad CPU Q8200 @2.33GHz, ОС Microsoft Windows 10) с физической топологией «звезда» с: 4 узла являются Datanodes, один – Namenode. Задачи управлялись менеджером YARN. При установке серверов кластера выдвигаются очень жесткие требования к памяти и к жесткому диску. Именно с этим было связано ограничение на размер обрабатываемых данных.

Тестирование проведено для решения задачи классификации рукопечатных символов многослойной нейронной сетью. Выбор структур многослойных нейронных сетей осуществлялся экспериментально, как компромисс между достижением значения среднеквадратичной ошибки обучения ($E < 0{,}0001$) и максимально возможным уменьшением значения

ошибки обобщения. Ошибка обобщения при сравнении требуемого значения и класса, определенного посредством MLP, составила 0,23% примеров тестовой выборки в соответствии с (6), что является хорошим показателем.

Результаты проведенных экспериментов, в которых реализованы параллельные процедуры обучения MRMLP на топологиях передачи данных «звезда» и «полносвязный граф» с различными объемами исходных данных приведены в табл. 1.

Таблица 1

Время выполнения процедуры параллельного обучения MRMLP в распределенной среде на различных топологиях передачи данных

Время выполнения последовательной реализации (сек)	Реализация на кластере	
	Топология передачи данных	Время (сек)
Объем обучающей выборки 650000		
923	«звезда»	219
	«полносвязный граф»	226
Объем обучающей выборки 1050000		
3613	«звезда»	759
	«полносвязный граф»	892
Объем обучающей выборки 2460000		
5431	«звезда»	925
	«полносвязный граф»	1052

Применение процедуры параллельного обучения MLP на основе MapReduce позволило сократить время вычислений при увеличении объемов обучающей и тестовой выборок. Равномерное разделение долей параллельного кода между вычислителями значительно повысило эффективность процедуры.

Определено, что большее количество передач данных осуществляется при использовании топологии «полносвязный граф». Минимальное количество передач достигается при применении топологии передачи данных «звезда», из чего следует, что при увеличении структуры MLP и объемов обучающей выборки для самого быстрого решения задач целесообразнее применять процедуру обучения MRMLP с топологией передачи данных, совпадающей с физической.

Оценивание вычислительной сложности метода MRMLP подтвердило [5], что предложенный метод имеет меньшую на один порядок

вычислительную сложность и меньшие временные затраты по сравнению с последовательными вычислениями (табл. 2).

Таблица 2

Сравнительный анализ последовательной и параллельной нейросетевой обработки данных на основе MapReduce

Обучение MLP	Функционирование MLP
Последовательный метод	
$O\left(U\times\left(M\times\sum_{m=L-1}^{2}n_{m-1}n_m n_{n+1}+T\times\sum_{m=L-1}^{L}n_{m-1}n_m\right)\right)$	$O\left(\sum_{m_1=2}^{L}n_{m-1}n_m\right)$
Метод MRMLP с разными топологиями передачи данных	
$O(U\times(M\times(\sum_{\substack{m_1=1\\ r^{-1}(m_1)\neq 1\\ r^{-1}(m_1)\neq L}}^{J}n_{r^{-1}(m_1)-1}n_{r^{-1}(m_1)+1}+n_L n_{L-1})+T\times\sum_{\substack{m_1=1\\ r^{-1}(m_1)\neq 1\\ r^{-1}(m_1)\neq L}}^{J}n_{r^{-1}(m_1)-1}))$	$O\left(\sum_{\substack{m_1=1\\ r^{-1}(m_1)\neq 1\\ r^{-1}(m_1)\neq L}}^{J}n_{r^{-1}(m_1)-1}\right)$

В таблице 2 используются следующие обозначения: J – номер элемента в перестановке $n_{r^{-1}}$, значения которого является граничным для условий $P<O\left(n_{r^{-1}(j+1)}\right)$ и $P\geq O\left(n_{r^{-1}(j)}\right)$, $j\in\{2,...,L-1\}$, M – объем обучающей выборки K; T – количество примеров в тестовой выборке; U – количество эпох обучения

Таким образом, вычислительная сложность метода последовательного функционирования МНС является функцией второго порядка, а вычислительная сложность метода последовательного обучения МНС – функцией пятого порядка, что указывает на высокую трудоемкость. Вычислительная сложность метода равномерного распределения нейросетевой обработки данных с топологиями сети передачи данных «звезда» и «полносвязный граф» имеет один и тот же порядок, поскольку при параллелизме на уровне реализации функций нейронов общее количество независимых параллельных скалярных операций совпадает. Предложенный метод при сравнении с последовательным аналогом имеет меньшую на один порядок вычислительную сложность и меньшие временные затраты.

Заключение.

Разработан метод обработки больших данных нейронными сетями, основанный на динамическом перераспределении работ между вычислителями кластера. Предложенный метод позволяет равномерно планировать нагрузку вычислительного кластера с топологией передачи данных «звезда» и «полносвязный граф».

Применение программной модели MapReduce Hadoop для нейрообработки больших данных позволило адаптировать топологию передачи данных, соответствующую архитектуре нейронной сети, на вычислительный класте, что сокращает объемы передаваемой информации между узлами информации, а также повышает быстродействие при решении сложных задач.

Литература

1. Axak N. G. Development of multi-agent system of neural network diagnostics and remote monitoring of patient. /N. G. Axak //Eastern-European Journal of Enterprise Technologies. – 2016. - 4/9 (82) – P. 4-11.

2. Axak N.G. Development of the hand gesture recognition system on the basis of clonal selection model ./N.G. Axak , O.U. Barkovskaya , H.S. Ivashchenko //СИСТЕМИ ОБРОБКИ ІНФОРМАЦІЇ ЗБІРНИК НАУКОВИХ ПРАЦЬ Випуск 3 (149). - 2017, . – P.76-80

3. Shklovets A. V. Visualization of High Dimensional Data Using Two Dimensional Self Organizing Piecewise Smooth Kohonen Maps /A. V. Shklovets and N. G. Axak //ISSN 1060 992X, Optical Memory and Neural Networks (Information Optics), 2012, Vol. 21, No. 4, pp. 227–232.

4. Аксак Н.Г. Процедура параллельного обучения многослойной нейронной сети. Топология передачи данных «звезда» / Н.Г. Аксак, А.Ю. Лебёдкина, О.В. Хоменко // Науковий вісник Чернівецького національного університету імені Юрія Федьковича. Серія: Комп’ютерні системи та компоненти. – Чернівці: ЧНУ, 2010. – Том 1, випуск 2. – С. 95-103.

5. Аксак Н.Г. Метод равномерного распределения параллельных операций ускоренного обучения многослойной нейронной сети с различными топологиями передач данных / Н.Г. Аксак, А.Ю. Лебёдкина // Збірник наукових праць «Системи управління, навігації та зв’язку. – Київ: ДЦ «Центральний науково-дослідний інститут навігації і управління», 2011. – Випуск 2(18). – С. 66-73.