|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

***Метод повышения производительности рабочей***

***станции за счет вычислительного кластера***

Студентка ИУ7-41м **Новоженов В.А.**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР  **Ковтушенко А.П.**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **Строганов Ю.В.**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

# **АННОТАЦИЯ**

**выпускной квалификационной работы**

Новоженова Владимира Алексеевича

на тему: «Метод повышения производительности рабочей станции за счет вычислительного кластера»

Объем ВКР: количество страниц 80, иллюстраций 25, таблиц 6, использованных источников 30, приложений 1.

Ключевые слова: повышение производительности, кластерные вычислительные системы, нейронные сети, распределенные алгоритмы, параллельные вычисления.

Объект исследования: вычислительные кластеры и методы повышения производительности при их использовании.

Цель работы: создание метода повышения производительности в задаче обучения нейронных сетей с использованием вычислительного кластера.

В ходе работы был создан метод повышения производительности вычислений, которые необходимо провести на рабочей станции. На рабочей станции развернута нейронная сеть, распознающая пользователей по клавиатурному почерку. В следствии, так называемого «concept drifting», появляется необходимость в дообучении нейронной сети. Дообучение нейронной сети является вычислительно сложной задачей, требующей много ресурсов и выполняющей большое число математических операций. Вследствие этого, не следует нагружать ресурсы рабочей станции задачей дообучения нейронной сети, а отправить задачу на вычислительный кластер. Само дообучение нейронной сети также следует реализовать в соответствии со спецификой кластера, а для повышения скорости получения результата и исключения дублей заданий, отправляемых на кластер следует использовать особую систему контроля заданий.

Результатом работы является сам метод повышения производительности при помощи высокопроизводительного кластера. Сам метод применим для обучения любого класса нейронных сетей, созданных на основе фреймворков TensorFlow и Pytorch.

Новизна работы состоит в использовании системы отслеживания заданий обучения нейронных сетей, отправляемых на вычислительный кластер для вычисления параллельно и распределенно.

Прогнозом развития метода является аппроксимация метода и системы контроля заданий под любой класс вычислительных математически-сложных задач, которые следует отправлять на выполнение на кластер и получать результат вычисления в другой среде, отличной от кластерной (например на некотором сервере или рабочей станции).

Автор ВКР «\_\_\_\_\_\_\_\_\_\_\_\_\_» «Новоженов Владимир Алексеевич»

**СОДЕРЖАНИЕ**

[**АННОТАЦИЯ** 2](#_Toc11867411)

[**ВВЕДЕНИЕ** 4](#_Toc11867412)

[**1 АНАЛИТИЧЕСКИЙ РАЗДЕЛ** 6](#_Toc11867413)

[**1.1 Известные методы верификации подписи с использованием нейронных сетей** 8](#_Toc11867414)

[**1.2 Метод разблокировки смартфона Face ID** 16](#_Toc11867415)

[**1.3 Выводы** 19](#_Toc11867416)

[**2 КОНСТРУКТОРСКИЙ РАЗДЕЛ** 20](#_Toc11867417)

[**2.1 Архитектура алгоритма распознавания рукописного текста** 20](#_Toc11867418)

[**2.2 Математическая модель метода распознавания рукописного текста** 23](#_Toc11867419)

[**2.3 Состав ансамбля нейронных сетей** 39](#_Toc11867420)

[**2.4 Описание обучающей и тестовой выборок** 40](#_Toc11867421)

[**3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ** 43](#_Toc11867422)

[**3.1 Детали подготовки данных для обучения и тестирования системы** 43](#_Toc11867423)

[**3.2 Оценка качества работы метода** 48](#_Toc11867424)

[**3.3 Пути развития предложенного метода** 54](#_Toc11867425)

[**3.4 Выводы** 56](#_Toc11867426)

[**ЗАКЛЮЧЕНИЕ** 57](#_Toc11867427)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 58](#_Toc11867428)

[**ПРИЛОЖЕНИЕ A** 61](#_Toc11867429)

# 

# **ВВЕДЕНИЕ**

**Цель работы:** создание метода повышения производительности в задаче обучения нейронных сетей с использованием вычислительного кластера.

**Задачи:**

* Проанализировать методы повышения производительности в решении задачи обучения нейронных сетей. Проанализировать методы взаимодействия с вычислительным кластером по внешнему программному интерфейсу;
* Создать метод взаимодействия с кластером в рамках задачи распределенного обучения нейронных сетей с использованием среды контроля заданий;
* Протестировать разработанную систему в решении практических задач обучения нейронных сетей;

**Объект** – вычислительные кластеры и методы повышения производительности при их использовании.

**Проблема** – ресурсо-затратность и время-затратность в задаче обучения нейронных сетей без использования кластера или с использованием кластера без использования системы обработки заданий.

В настоящее время особую популярность в решении самых разных задач получили нейронные сети. С их помощью можно и определять наличие объектов на изображениях и выявлять паттерны и закономерности во временных рядах и обрабатывать и распознавать любой звук. Но перед использованием нейронных сетей их необходимо обучить на, зачастую, огромных массивах данных. Это могут быть тысячи изображений или миллионы строк данных. А для обучения нейронных сетей на них может потребоваться несколько десятков (а иногда даже тысячи) эпох (одна эпоха – полное прохождение всего набора данных в прямом и обратном направлении).

Основной проблемой является тот факт, что суммарное время обучения нейронной сети может достигать месяцев даже с использованием GPU-ускорителей, что в ряде задач может превышать время, по которому наступает «concept drifting» и не является приемлемым.

По этой причине целесообразно проводить вычисления не на машине, на которой уже развернута нейронная сеть и происходит обработка в реальном времени, а отправлять вычислительное задание на высокопроизводительный кластер. Также для контроля статуса выполнения заданий, устранения дублей заданий, подбора вычислительных ресурсов кластера, которые можно использовать для решения задания и соответствующей корректировки задания-скрипта обучения нейронной сети (который также необходимо реализовать и запускать соответствующим образом для использования всех доступных заданию мощностей и использования распределенных параллельных вычислений) целесообразно использовать систему планирования и управления заданиями, отправляемыми на кластер.

В работе проиллюстрировано взаимодействие приложения на рабочей машине пользователя, удостоверяющее его личность по клавиатурному почерку посредством отправки запросов на сервер, на котором развернута нейронная сеть. Сервер с нейронной сетью должен инициировать отправку задачи дообучения нейронной сети в систему управления заданиями на кластере. Из системы управления заданиями вычислительная задача отправляется на кластер для ускорения вычислений. Затем обученная нейронная сеть заменяет активную (по какому-либо методу разворачивания) на сервере.

# 

# **1 АНАЛИТИЧЕСКИЙ РАЗДЕЛ**

## **1.1 Кластерные вычисления для ускорения вычислений. Виды кластерных систем.**

Существует множество вычислительно сложных задач. К ним можно отнести и компьютерное моделирование, в ходе которого решается ряд научно-технических или инженерных задач. Также к ним относятся и обучение нейронных сетей и классических алгоритмов машинного обучения. В ходе этих процессов производится огромное количество математических операций, для вычисления которых нужно большое количество ресурсов.

Обычные серверные или рабочие станции далеко не всегда обладают объемом вычислительных ресурсов, которые позволили бы быстро и эффективно решить какую-либо вычислительно сложную задачу.

Именно для решения таких задач конструируются вычислительные кластеры, состоящие из большого количества узлов (базовых инженерных объектов), связанных между собой на базе шинной архитектуры или коммутатора. С помощью использования кластера возможно производить вычисления не только параллельно, но и распределенно  
[ГОСТ Р ИСО/МЭК 10746-3-2001].

Также кластерные системы позволяют обеспечить повышенную надежность в решении ряда задач (например, производя вычисления одновременно на двух и более узлах, чтобы, в случае отказа одного из них, вычисления не были прерваны).

Кластерные системы обладают также большим потенциалом к масштабированию, которое реализуется путем увеличения количества вычислительных узлов.

Разделяют два вида кластеров:

* Гомогенные кластерные системы – системы, состоящие из одинаковых узлов;
* Гетерогенные кластерные системы – системы, состоящие из разнотипных узлов;

Отличительной особенностью кластерных вычислительных систем является высокий уровень доступности. Так как каждый отдельный вычислительный узел функционирует самостоятельно, на основе своей операционной системы, как правило, Unix-подобной (отсутствует единая операционная система), а также отсутствует совместно используемая память, это приводит к отсутствию проблемы когерентности кэшей. У каждого узла вычислительного кластера своя память. А результат вычислений на узле по некоторому интерфейсу отправляется на управляющую машину кластера, которая, как правило, агрегирует результаты, полученные на других узлах.

Для обеспечения высокого уровня надежности, доступности и бесперебойности на кластерных системах необходимо наличие специализированного программного обеспечения, которое организовывало бы бесперебойную работу при отказе одного или более узлов. Такое программное обеспечение осуществляет механизм передачи сообщений поверх стандартных сетевых протоколов.

При обмене информацией между узлами используются, как правило, два подхода:

* Передача сообщений (как правило, с помощью MPI – Message Passing Interface – то есть программный интерфейс для обмена сообщениями между процессами, выполняющими одну задачу);
* Совместно используемая память;

Неотъемлемая часть кластерных систем – программное обеспечение, организующее бесперебойную работу в случае отказа одного или более узлов. Такое программное обеспечение должно быть предустановлено на каждом узле вычислительного кластера. Это программное обеспечение способно перераспределить нагрузку в случае аварийной ситуации, а также восстановить вычисления в случае сбоя.

Также в кластерных системах необходимо программное обеспечение, которое могло бы контролировать работоспособность всех узлов. Контроль этот основан на периодической рассылке каждым узлом сигнала “keepalive” или “heartbeat”. В случае отсутствия сигнала в течение некотрого времени – узел считается неактивным и вышедшим из строя.

### [Источник: <http://www.nsc.ru/win/elbib/data/show_page.dhtml?77+858> Кластерные вычислительные системы]

### [Цилькер Б.Я. Организация ЭВМ и систем : Учебник для вузов / Б.Я. Цилькер, С.А. Орлов. - 2-е изд. - СПб.: Питер, 2011. - 688 с. - ISBN 978-5-49807-862-5.]

### Brewer E. Clustering: Multiply and Conquer // Data Communications, July, 1997.

## **1.2 Обучение нейронной сети как пример вычислительно сложной задачи, эффективно решаемую на кластере.**

Как уже было сказано ранее, одной из вычислительно сложных задач в настоящее время является обучение нейронных сетей.

Существует большое число задач, которые могли бы решаться при помощи нейронных сетей. Также в большом количестве задач требуется поддержание нейронных сетей в актуальном состоянии, вследствие “concept drifting”.

“Concept drift” – явление в статистической аналитике и машинном (а также глубоком) обучении, при котором статистические свойства целевой переменной, которые модель (к примеру, нейронная сеть или другая математическая модель) пытается предсказать – изменяются со временем некоторым непредвиденным образом. Это влечет понижение качества работы модели. Ярким примером является модель машинного обучения, распознающая голос конкретного человека. С годами голос человека склонен изменяться, что приведет к тому, что модель перестанет корректно распознавать этот голос. Другим примером может являться модель, распознающая человеческое лицо. Оно также склонно к изменениям, что также повлечет ухудшение распознавательной способности модели применительно к человеку, которого необходимо было распознавать. К другим примерам можно отнести и появление новых игроков на рынке ценных бумаг. Некоторая математическая модель, прогнозирующая курс котировок, станет менее точной вследствие изменения ситуации на рынке и появления других, неучтенных ранее факторов.

### “Concept drifting” может появляться во многих задачах, которые решаются с помощью машинного обучения и нейронных сетей. Поэтому следует учитывать его возможное появление в ряде задач, а в случае его появления – либо дообучать модель, либо обучать модель заново. [The problem of concept drift: definitions and related work , Alexey Tsymbal, 2004, <https://www.researchgate.net/publication/228723141_The_Problem_of_Concept_Drift_Definitions_and_Related_Work>]

### <https://towardsdatascience.com/concept-drift-and-model-decay-in-machine-learning-a98a809ea8d4>

Само по себе обучение представляет из себя большое число математических операций и преобразований, в ходе которых, в случае нейронных сетей, определяются веса между нейронами, содержащими некоторую функцию активации. Обучение проходит на обучающей выборке, качество обучения и способности решать поставленную задачу проверяется на тестовой выборке. Затем обученная нейронная сеть получает способность эффективно решать задачи на подобных объектах в реальном времени.

Для обучения нейронных сетей требуется большое количество информации на вход. Это могут быть сотни изображений, аудиофрагментов или миллионы строк табличных данных.

Обучение нейронных разделяют на следующие категории:

* Обучение с учителем;
* Обучение без учителя;
* Обучение с подкреплением;

Обучение с учителем предполагает наличие размеченных данных, содержащих как данные, которые должны идти на вход, так и данные, которые необходимо получить (например, набор изображений и текстовый файл со списком названий изображений и информацией, есть ли на этих изображениях некоторый объект для задачи распознавания объектов на изображениях). Такое обучение применяется в задачах классификации (к какому классу принадлежит объект, в том числе, к классу изображений с наличием чего-либо) и регрессии (задача выявления зависимости некоторой случайной величины от ряда других величин при помощи аппроксимации функции).

Обучение без учителя применяется в ряде задач, где либо нет размеченных данных, либо их необходимо, некоторым образом, разметить в ходе самой задачи. Примером таких задач может являться кластеризация. В некотором пространстве имеется ряд точек, которые необходимо объединить в кластеры из точек. Также может применяться для обнаружения аномалий на объектах, построения ассоциаций (например, для рекомендательных систем) и автокодировщиков (которые применяются в дальнейшем, например, для машинного перевода).

Обучение с подкреплением применяется для задач, в которых есть некоторая среда с конечным числом возможных состояний. Оно применяется, например, для обучения роботов совершать некоторые действия в ходе большого числа итераций.

Рассмотрим задачу обучения с учителем. Каждый объект при обучении сначала проходит через нейронную сеть, получает некоторый ответ, а затем происходит, так называемое, обратное распространение ошибки, в ходе которого веса нейронной сети корректируются в соответствии с градиентом ошибки.

Часто обучение происходит в течении большого числа эпох (одна эпоха - полное прохождение всей обучающей выборки через нейронную сеть – в ходе прямого и обратного распространения). Так как даже одно вычисление прямого и обратного прохода одного объекта может иметь большое число математических вычислений – то все обучение, содержащее большое число эпох, может происходить в течение продолжительного времени. На мощнейших CPU (central processing unit) это может занимать недели и даже месяцы. С использованием GPU время обучения может значительно сокращаться благодаря большему количеству блоков арифметико-логических устройств, которые эффективно и быстро производят вычисления матричных операций. Однако даже с использованием нескольких GPU ускорителей на одной вычислительной машине сложно достичь хорошей скорости вычислений.

В настоящее время эта проблема решается специализированными и промышленными GPU-ускорителями, а также с помощью кластерных вычислений. [Аль-хулайди Абдулмаджид Ахмед Галеб (2011) – «Распределенные вычисления (кластерные вычисления) с использованием пакета параллельного программирования»;]

Многие современные фреймворки для машинного и глубокого обучения также имеют встроенную поддержку распределенных кластерных вычислений (pytorch.distributed) или же существуют программные библиотеки, используя которые возможно обучать нейронные сети и классические алгоритмы машинного обучения (такие как градиентный бустинг) распределенно (horovod, dask).

## **1.3 Фреймворки для глубокого обучения, архитектуры нейронных сетей и способы организаций распределенных вычислений.**

ТУТ НАДО НАПИСАТЬ

## **1.4 Выводы**

ТУТ НАДО НАПИСАТЬ

# **2 КОНСТРУКТОРСКИЙ РАЗДЕЛ**

## **2.1 Архитектура системы для ускоренного вычисления задач при помощи кластера.**

Рассмотрим в следующую ситуацию: на некотором сервере развернута нейронная сеть. Эта нейронная сеть подвержена “concept drift” и её необходимо обучать заново с некоторой периодичностью. Так как ресурсы сервера загружены вычислениями результата прогноза для объектов, подающихся на вход посредством API сервера в нейронную сеть, то обучение нейронной сети следует провести на удаленной вычислительной машине. По причине того, что сама операция полного обучения нейронной сети достаточно ресурсозатратная, необходимо использовать высокопроизводительные вычислительные машины. Примером таких вычислительных машин может являться кластер. Обучение будет производиться на вычислительном кластере с использованием распределенных и параллельных вычислений для повышения скорости решения задачи обучения.

Для эффективности использования вычислительного кластера целесообразно использовать систему контроля заданий. В обязанности такой системы должно входить следующее:

* Контроль дублей задач (удаление лишних);
* Контроль, логирование и учёт списка задач;
* Управление запуском задач и запуск с параметрами, соответствующим текущему статусу кластера;
* Отслеживание загрузки/выгрузки файлов в хранилища и запуск событий по окончанию;
* Перезапуск задач в случае необходимости;
* Мониторинг статусов и смены статусов выполняемых задач;
* Управление статусами задач при необходимости;
* Контроль обновления и замены обученной нейронной сети на сервере;
* ???

Все задания с сервера, которые необходимо будет обработать на кластере, сначала посылаются на эту систему контроля заданий.

Рассмотрим детально все обязанности этой системы.

**Контроль дублей задач** — эта обязанность вызвана необходимостью контролировать ситуацию, при которой в ходе какой-либо ошибки (прикладного ПО сервера, сетевой и т.д.) поступает более одного одинакового задания с необходимостью вычисления на кластере. Если задания абсолютно идентичны – необходимо обрабатывать на кластере лишь одно из них.

**Контроль и логирование списка задач** — данная обязанность является следствием необходимости учета всех задач за некоторый промежуток времени для дальнейшего возможного анализа, изучения истории и непосредственно учёта задач, отправляемых на вычисление на кластер.

**Управление запуском задач и запуск с параметрами, соответствующим текущему статусу кластера** — одна из ключевых функциональностей системы управления заданиями – это корректировка отправляемых заданий в соответствии с текущей нагрузкой и свободными ресурсами кластера. Данный функционал должен обеспечивать отправку задания на кластер, принимая в учет другие задания в списке, в очереди, свободные ресурсы кластера, оценочное время выполнения.

**Отслеживание загрузки/выгрузки файлов в хранилища и запуск событий по окончанию** — эта функция системы управления заданиями должна контролировать старт, окончание отправки файлов с кластера на объектное хранилище, с объектного хранилища на кластер, с объектного хранилища на сервер или с сервера на объектное хранилище и вызывать необходимые процедуры по окончанию загрузки.

**Перезапуск задач в случае необходимости** — данный функционал должен обеспечивать гарантированную выполнимость нужного задания вне зависимости от каких-либо факторов (в частности в случае аварии на некоторых кластерных узлах, компонент должен отследить это событие и запустить задание на активных узлах повторно, возможно, внеся корректировки в параметры задания.

**Мониторинг статусов и смены статусов выполняемых задач** — наличие данного функционала обусловлено тем, что далеко не все кластерные вычислительные системы обладают функционалом обратного вызова по наступлению каких-либо событий, поэтому отслеживать задания или процессы необходимо самостоятельно.

**Управление статусами задач при необходимости** — эта функция при появлении более актуального задания останавливать старое (при соблюдении некоторых условий, например, если задание отрабатывало менее некоторого временного периода) и заменять на более актуальное. Затем необходимо изменять условие, блокирующее запуск нового более актуального задания (например, уменьшать время, по истечению которого, задание считается незаменяемым), с целью предотвращения бесконечного чередования актуальных заданий без фактического окончания какого-либо из них.

**Контроль обновления и замены обученной нейронной сети на сервере** — данная функциональность не является обязательной, но в рамках этой работы было целесообразно реализовать контроль обновления нейронной сети как на самом сервере, так и на системе контроля заданий (чтобы внести соответствующую информацию в поле в списке заданий в базе данных).

Диаграмма потоков данных, отражающая данный подход и метод, а также саму реализацию, между компонентами системы отслеживания заданий, внешнего хранилища и вычислительного кластера изображена ниже:

# 

Рисунок X. Диаграмма потоков данных

Архитектура взаимодействия кластера и системы отслеживания заданий приведена ниже:

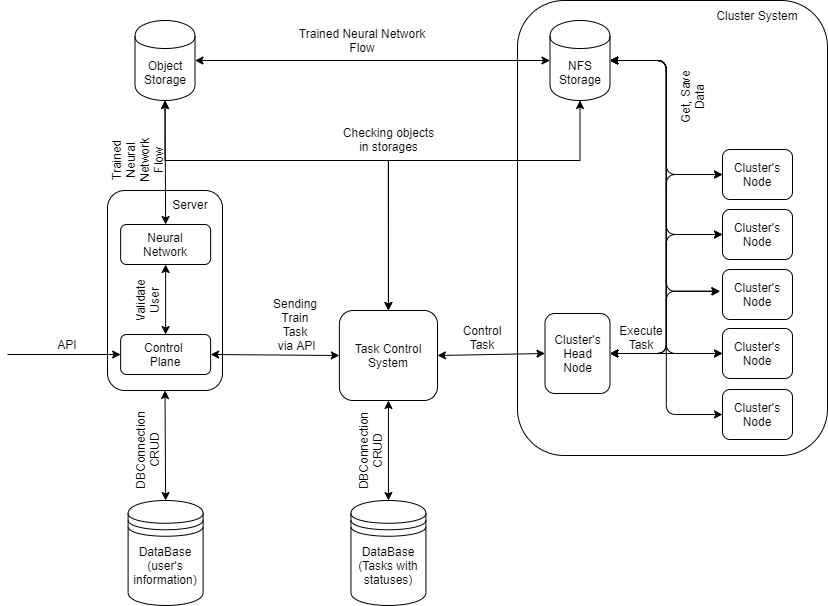


Рисунок X. Архитектура взаимодействия кластера и системы отслеживания заданий

На ней отображены все необходимые базы данных и все хранилища, которые взаимодействуют с теми или иными компонентами. Также на ней отображено ключевое взаимодействие компонентов сервера с развернутой нейронной сетью и вычислительного кластера с компонентом Task Control System.

Архитектура всей системы, включая компоненты на клиентской части (а именно на пользовательском десктопе) отображены ниже. На ней также отображено взаимодействие любого открытого на пользовательском компьютере приложения и приложения, собирающего метрики клавиатурного почерка и отправляющего на вход нейронной сети на сервер для проведения фоновой аутентификации по самому клавиатурному почерку.

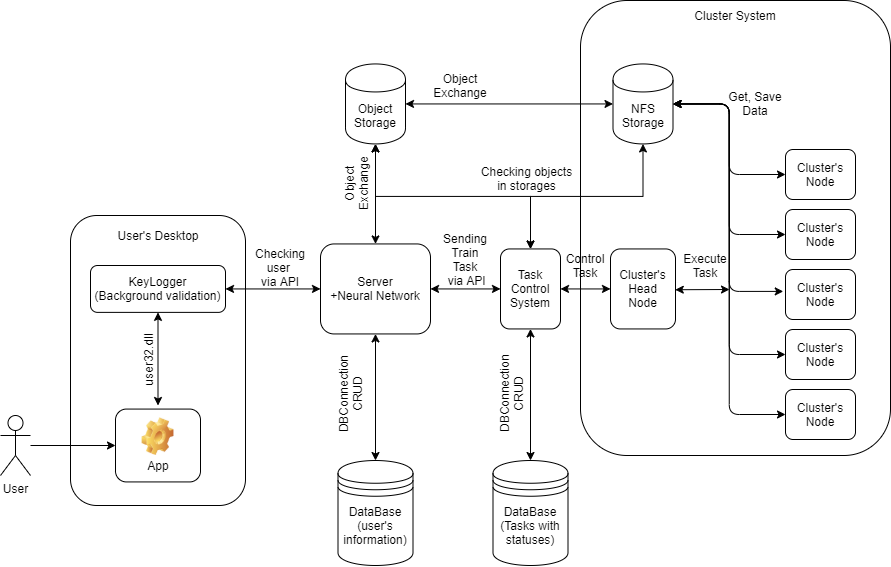


Рисунок X. Архитектура всей системы, представленной в работе

## **2.2 Пользовательское приложение для сбора метрик и взаимодействие его с системой контроля заданий.**

ВОТ ТУТ ПОНАПИСАТЬ ДА

## **2.3 Архитектура нейронной сети и методы распределенных вычислений на кластере, применительно к ней.**

ВОТ ТУТ ПОНАПИСАТЬ ДА

# **3 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ**

## **3.1 Детали подготовки данных для обучения и тестирования системы**

Какой там язык, какая база данных, какой брокер сообщений (редис давай да), какой фреймворк для нейросетки и детали реализации всего и вся.

## **3.4 Выводы**

# 

# **ЗАКЛЮЧЕНИЕ**

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Князев А. В. Распознавание слитного рукописного текста / Вестник МЭИ, 2013. С. 15.
2. Попова В. В. Распознавание рукописного текста на китайском языке на основе ключей при помощи нейронных сетей // Информационные технологии XXI века, 2013. С. 47
3. РАСПОЗНАВАНИЕ РУКОПИСНЫХ СИМВОЛОВ С ПРИМЕНЕНИЕМ НЕЙРОСЕТЕВОЙ ТЕХНОЛОГИИ, Андреев Всеволод Владимирович, Журавлев Михаил Сергеевич, // Прикладная Информатика № 2, с. 115-117
4. Исрафилов Х. С. Применение нейронных сетей в распознавании рукописного текста // Молодой ученый. - 2016. - №29. - С. 24-27 [Электронный ресурс]. – Ресурс доступа: <https://moluch.ru/archive/133/37372/> (Дата обращения: 2018-11-15)
5. Кулакова О.А., Воронова Л.И. // Московский Технический Университет Связи и Информатики. - 2017. С. 10-11.
6. Comp. nanotechnol., 2016, выпуск 3, страницы 224–241. Защита информации. Биометрические технологии идентификации в системах контроля и управления доступом. С 224-241.
7. Алгоритмы оффлайн-распознавания рукописных цифр. Козлов Владимир Дмитриевич. Выпускная квалификационная работа. Научный руководитель к. ф.–м. н, доцент С. И. Гуров. Московский государственный университет имени М. В. Ломоносова. С. 73.
8. Маркова С.В., Жигалов К.Ю. Применение нейронной сети для создания системы распознавания изображений // Фундаментальные исследования. – 2017. – № 8-1. С 60-64.
9. Geoffrey E. Hinton. Dynamic Routing Between Capsules // Google Brain, 2017. С 15-18.
10. 36 материалов о нейросетях: книги, статьи и последние исследования [Электронный ресурс]. – Ресурс доступа: [https://habrahabr.ru/company/neurodatalab/blog/336218](https://habrahabr.ru/company/neurodatalab/blog/336218/) (дата обращения: 11.02.2019)
11. Применение нейросетей в распознавании изображений [Электронный ресурс]. – Ресурс доступа: https://geektimes.ru/post/74326 (дата обращения: 19.01.2019)
12. Метрики качества [Электронный ресурс]. – Ресурс доступа: <http://bazhenov.me/blog/2012/07/21/classification-performance-evaluation.html>(дата обращения: 21.01.2019)
13. Калькулятор рецептивных полей [Электронный ресурс]. – Ресурс доступа: https://fomoro.com/projects/project/receptive-field-calculator (дата обращения: 11.12.2018)
14. Pooling // объединение слоев [Электронный ресурс]. – Ресурс доступа: http://datareview.info/article/eto-nuzhno-znat-klyuchevyie-rekomendatsii-po-glubokomu-obucheniyu-chast-2 (дата обращения: 21.01.2019)
15. Набор данных для обучения и тестирования [Электронный ресурс]. – Ресурс доступа: http://www.iapr-tc11.org/mediawiki/index.php/ICDAR\_2011\_Signature\_Verification\_Competition\_(SigComp2011) (дата обращения: 15.02.2019)
16. Сверточная нейронная сеть на PyTorch[Электронный ресурс]. – Ресурс доступа: [https://neurohive.io/ru/tutorial/cnn-na-pytorch](https://neurohive.io/ru/tutorial/cnn-na-pytorch/) (дата обращения: 10.03.2019)
17. Как работает FaceID в iPhone X: алгоритм на языке Python[Электронный ресурс]. – Ресурс доступа: <http://ai-news.ru/2018/03/kak_rabotaet_faceid_v_iphone_x_algoritm_na_yazyke_python.html> (дата обращения: 14.04.2019)
18. Face ID Security Guide. – Ресурсдоступа: <https://www.apple.com/business/site/docs/FaceID_Security_Guide.pdf>
19. Зарплаты ИТ-специалистов на середину 2018 года [Электронный ресурс]. – Ресурс доступа: <https://vc.ru/flood/43849-zarplaty-it-specialistov-na-seredinu-2018-goda> (дата обращения: 01.05.2019)
20. EnterpriseAndroidvs IOS [Электронный ресурс]. – Ресурс доступа: <https://medium.com/predict/enterprise-android-vs-ios-which-is-more-secure-51a95cac53c2> (дата обращения: 11.03.2019)
21. Сверточная нейронная сеть, часть 1: структура, топология, функции активации и обучающее множество [Электронный ресурс]. – Ресурс доступа: <https://habr.com/ru/post/348000/> (дата обращения: 23.02.2019)
22. Градиентый бустинг [Электронный ресурс]. – Ресурс доступа: <https://neurohive.io/ru/osnovy-data-science/gradientyj-busting/> (дата обращения: 26.04.2019)
23. Разновидности нейронных сетей [Электронный ресурс]. – Ресурс доступа: <https://tproger.ru/translations/neural-network-zoo-1/> (дата обращения: 05.05.2019)
24. Открытый курс машинного обучения. Тема 5. Композиции: бэггинг, случайный лес [Электронный ресурс]. – Ресурс доступа: <https://habr.com/ru/company/ods/blog/324402/#1-begging> (дата обращения: 03.05.2019)
25. Логистическая функция ошибки [Электронный ресурс] <https://dyakonov.org/2018/03/12/логистическая-функция-ошибки/> (дата обращения: 03.05.2019);
26. Майков К.А. Козлова Ю.А. Метод идентификации пользователя по рукописному кодовому слову // Новые информационные технологии в автоматизированных системах, М.- 2019, с.19-21;

# **ПРИЛОЖЕНИЕ A**