ОТЧЁТ

по Опытно – Конструкторской Разработке

**«**Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений**»**

**(«**SmartGetDistance**»)**

Нижний Новгород, 2018

Оглавление

[Ведение 3](#_Toc529725560)

[Термины и определения 4](#_Toc529725561)

[Цели, требования и этапы разработки программного изделия 5](#_Toc529725562)

[Основания и назначения для разработки 5](#_Toc529725563)

[Требования к программному изделию и документации и защиты от ИТР 6](#_Toc529725564)

[Стадии и этапы разработки 7](#_Toc529725565)

[Инструкция по развертыванию нейронной сети NVIDIA DIGITS. 8](#_Toc529725566)

[Инструкция по установке на Windows платформу. 8](#_Toc529725567)

[Примечания к установке 14](#_Toc529725568)

[Технология обучения нейронной сети NVIDIA DIGITS 15](#_Toc529725569)

[Формирование данных 15](#_Toc529725570)

[Формирование модели 22](#_Toc529725571)

[Технология принятия входных данных 36](#_Toc529725572)

[Заключение 37](#_Toc529725573)

[Список литературы 38](#_Toc529725574)

[Приложение 39](#_Toc529725575)

# Ведение

Объектом автоматизации является процесс построения и обучения нейронной сети для определения расстояния между фокусом оптической системы и поверхностью рельефа по серии изображений. ПО «SmartGetDistance» должно обеспечивать нахождение данного расстояния.

## Термины и определения

ПО «SmartGetDistance» - разрабатываемое в рамках текущей ОКР ПО предназначенное для построения и обучения нейронной сети для определения расстояния между фокусом оптической системы и поверхностью рельефа по серии изображений.

DIGITS (Deep Learning GPU Training System) - набор программ для создания глубинных нейронных сетей (DNN) в ходе машинного обучения, а также для управления и диагностики данного процесса. DIGITS обладает графическим пользовательским интерфейсом. В состав пакета входит веб-сервер, с помощью которого осуществляется коллективная работа над проектом.

Caffe – программный пакет, предназначенный для работы с графическими ускорителями NVIDIA.

CUDA ( Compute Unified Device Architecture) — программно-аппаратная архитектура [параллельных вычислений](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F), которая позволяет существенно увеличить вычислительную производительность благодаря использованию [графических процессоров](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) фирмы [Nvidia](https://ru.wikipedia.org/wiki/Nvidia" \o "Nvidia).

Python — [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)общего назначения, ориентированный на повышение производительности разработчика и читаемости кода.

Graphviz — комплекс программных элементов для визуализации DOT-графов в автоматическом режиме. Также программа может визуализировать библиотеки, виджеты и TUI-программы.

СПИСОК СОКРАЩЕНИЙ

|  |  |
| --- | --- |
| ОКР | Опытно – конструкторская разработка |
| ПО | Программное обеспечение |
| ПЭВМ | Персональная электронно-вычислительная машина |
| ОС | Операционная система |
| ЕСПД | Единая система программной документации |
| РСП | Руководство системного программиста |
| РО | Руководство оператора |
| ПМИ | программа и методика испытаний |

# Цели, требования и этапы разработки программного изделия

Объектом автоматизации является процесс построения и обучения нейронной сети для определения расстояния между фокусом оптической системы и поверхностью рельефа по серии изображений. ПО «SmartGetDistance» должно обеспечивать нахождение данного расстояния.

## Основания и назначения для разработки

Основание для выполнения ОКР – Решение заказчика

Заказчик: Старостин Николай Владимирович

Исполнитель: гр. 381707м – команда 1, ННГУ им. Лобачевского, гор. Нижний Новгород.

Начало разработки – с момента заключения договора.

Окончание разработки – декабрь 2018

ПО « SmartGetDistance» предназначено для определения расстояния между фокусом оптической системы и поверхностью рельефа по серии изображений, полученного микросъёмкой с малой глубиной резкости.

Результатом решения является ответ нейронной сети по удаленности фокуса от поверхности.

## Требования к программному изделию и документации и защиты от ИТР

Требования к составу и параметрам технических средств

ПО «SmartGetDistance» должно быть разработано с использованием нейронных сетей. ПО «SmartGetDistance» должно быть работоспособно под управлением следующих операционных систем: семейства Windows с 2007г и младше. Машина должна работать на [графических процессорах](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) [NVIDIA](https://ru.wikipedia.org/wiki/NVIDIA) серии GeForce и видеокартах, построенных на официальных спецификациях NVIDIA. На ОС должны быть установлены все библиотеки, необходимые для функционирования ПО «SmartGetDistance».

Требования к входным и выходным данным

ПО «SmartGetDistance» должно загружать необходимые для расчета данные, а именно: 3 изображения формата .png, размером 32х32 px, полученных с разных расстояний одной и той же поверхности. Изображения должны показывать динамику изменения размытости изображений. Структура пакета данных будет уточнена в процессе разработки.

ПО «SmartGetDistance» должно формировать выходные данные, а именно, число, которое показывает расстояние удаленности фокуса от поверхности. Структура выходного файла будет уточнена в процессе разработки.Нейронная сеть предполагает обучение на подготовленных данных, каждый набор включает в себя 3 изображения с эталоном отклика нейронной сети.

Программная документация должна содержать следующие документы:

* руководство системного программиста;
* руководство оператора;
* программа и методика испытаний.

## Стадии и этапы разработки

Этапы и стадии ОКР, их содержание, сроки выполнения, отчетные документы и ответственные за выполнение приведены в таблице 1.

Примечание. Исполнитель в срок до 31.12.2018г бесплатно оказывает сервисные услуги по устранению обнаруженных дефектов.

Приёмочные испытания проводятся комиссией на технических средствах Заказчика на данных от команды №1 и в соответствии с Программой и методикой проведения приёмочных испытаний. Для проведения приемочных испытаний Исполнителем предъявляется следующая документация:

– Техническое задание на ОКР;

– Программа и методика приёмочных испытаний;

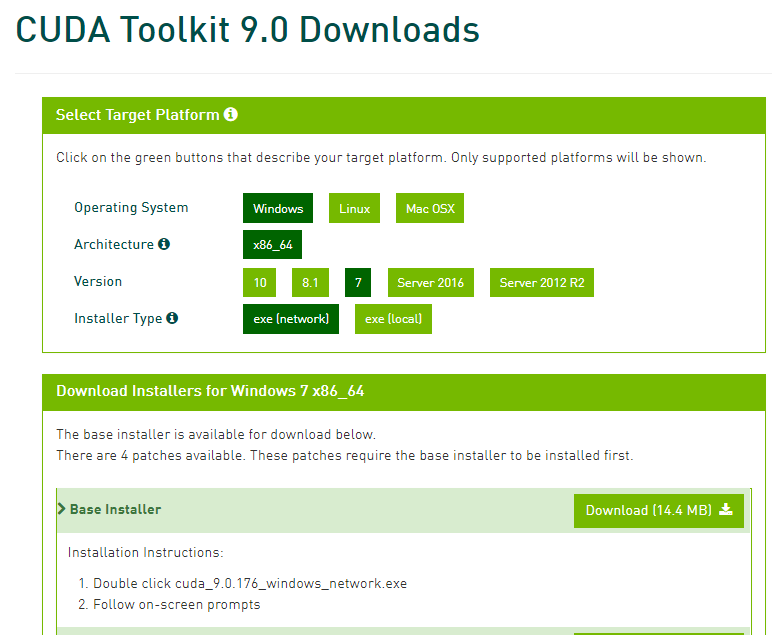
– Программная документация.

# Инструкция по развертыванию нейронной сети NVIDIA DIGITS.

## Инструкция по установке на Windows платформу.

1. Скачайте и установите Python 2.7.14 x64  
   <https://www.python.org/downloads/>
2. Скачайте и установите Сuda 9.0 для своей ОС

<https://developer.nvidia.com/cuda-90-download-archive>



1. Скачайте Visual Studio 2015, CUDA 8.0(GPU), Python 2.7: [Caffe Release](https://github.com/Coderx7/Caffe_1.0_Windows/releases/download/caffe_1.0_windows/caffe_cuda_x64_MSVC14_Py27_release.zip)

с ресурса <https://github.com/Coderx7/Caffe_1.0_Windows> и положите папку в корень локального диска С.

1. Клонируйте репозиторий <https://github.com/NVIDIA/DIGITS> в корень локального диска С
2. Перейдите в директорию с проектом NVIDIA DIGITS. Откройте файл *requirements.txt* и измените содержимое данного файла на следующее:

#Pillow>3.3.1,<=3.3.2

numpy>=1.8.1,<=1.14.0

#scipy>=0.13.3,<=0.17.0

protobuf>=2.5.0,<=3.2.0

six>=1.5.2,<=1.10.0

requests>=2.2.1,<=2.9.1

gevent>=1.0,<=1.1.0

gevent-websocket==0.9.3

Flask==0.10.1

Flask-WTF>=0.11,<=0.12

wtforms>=2.0,<=2.1

Flask-SocketIO==2.6

setuptools>=3.3,<=20.7.0

lmdb==0.87

#h5py>=2.2.1,<=2.6.0

pydot>=1.0.28,<=1.0.29

psutil>=1.2.1,<=3.4.2

matplotlib>=1.3.1,<=1.5.2

#scikit-fmm>=0.0.9

python-magic>=0.2

boto>=2.48.0

1. Откройте:

Панель управления\Система и безопасность\Система -> Дополнительные параметры системы-> Переменные среды

В разделе системные переменные необходимо изменить параметр Рath

В Рath добавляем значения переменной:

С:\Python27

С:\Python27\Scripts

C:\Program Files\NVIDIA Corporation\NVSMI

Далее в переменной среды пользователя создаем переменную PYTHONPATH и задаем значение переменной C:\caffe\_cuda\_x64\_MSVC14\_Py27\_release\python

Добавьте в переменную PATH путь до исполняемых файлов caffe, которые мы скачали (…\caffe\_prebuilt\_binaries\VS\_2015\_CPU\_Python\_2.7\bin) C:\caffe\_cuda\_x64\_MSVC14\_Py27\_release\bin

1. Скачайте и установите программу graphviz2.38

<http://www.graphviz.org/_pages/Download/Download_windows.html>

Пропишите путь в переменную среды Рath путь до graphviz до папки bin

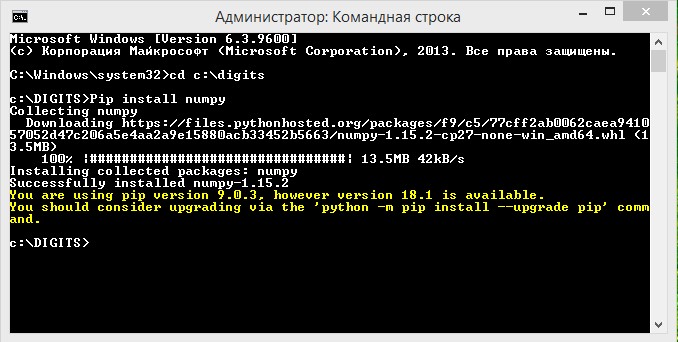
c:\pgramm file(86)\graphviz 2.38\bin

1. Скачайте whl пакеты:  
   scikit\_fmm-0.0.9-cp27-cp27m-win\_amd64.whl  
   scikit\_image-0.14.1-cp27-cp27m-win\_amd64.whl  
     
   отсюда <http://www.lfd.uci.edu/~gohlke/pythonlibs/> и положите файлы в корень локального диска С:\
2. Откройте командную строку cmd.exe от имени администратора

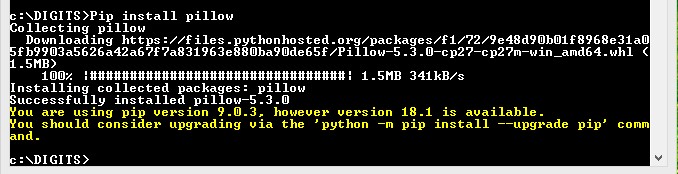
Перейдите в директорию DIGITS: cd c:\digits

Установите следующие библиотеки:

Pip install numpy



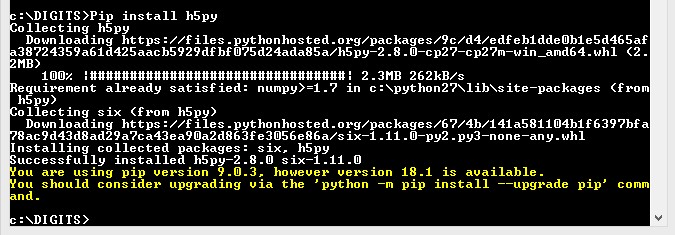
Pip install pillow



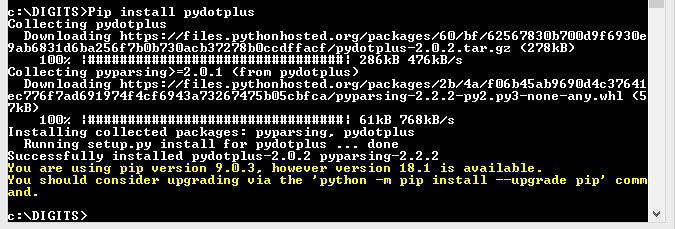
Pip install scipy



Pip install h5py



Pip install pydotplus

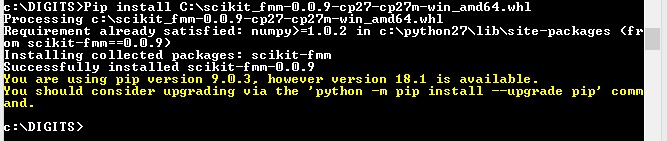


Установите whl пакеты:

Pip install C:\scikit\_image-0.14.1-cp27-cp27m-win\_amd64.whl



Pip install C:\ scikit\_fmm-0.0.9-cp27-cp27m-win\_amd64.whl

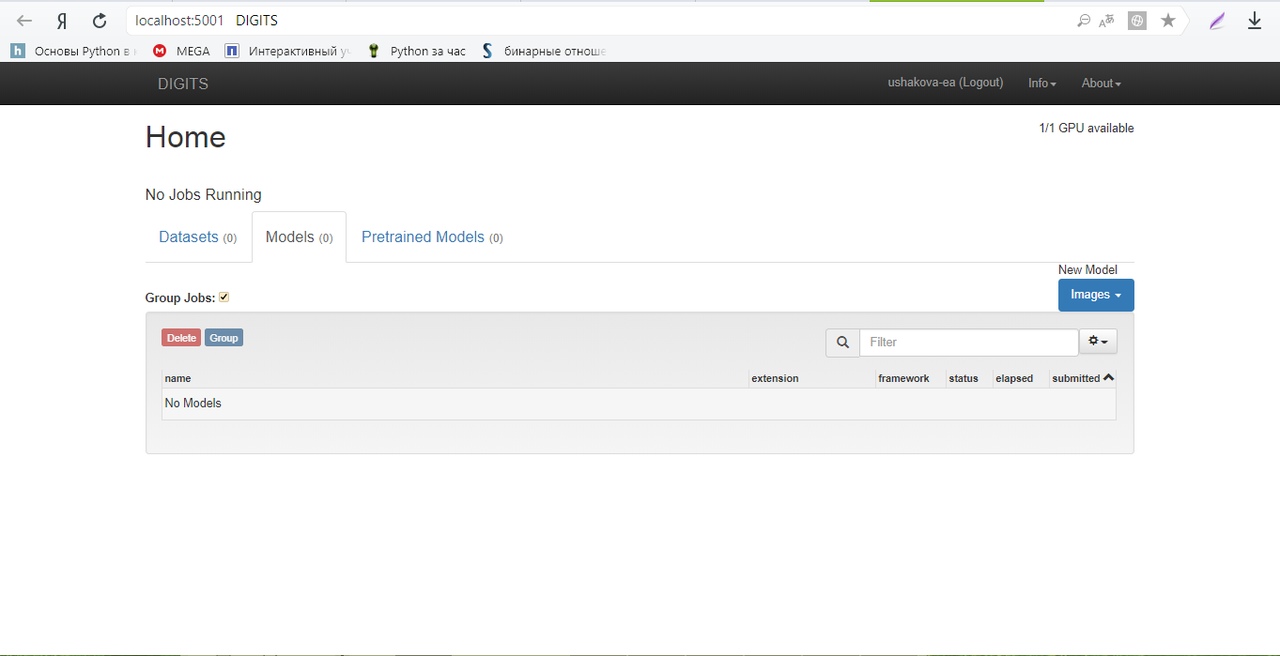


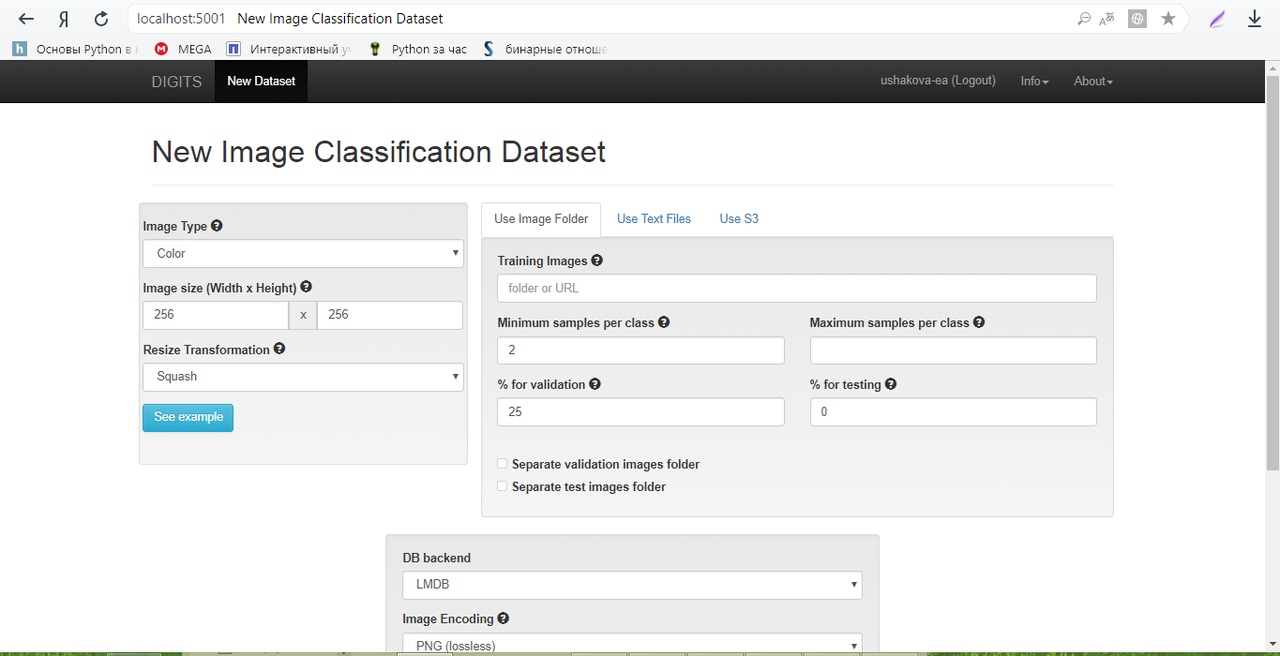
1. Установите оставшиеся Flask библиотеки с помощью команды

python -m pip install -r requirements.txt

1. Запустите NVIDIA DIGITS из директории проекта NVIDIA DIGITS  
   python –m digits –p 5001
2. В браузере откройте станицу по адресу localhost::5001

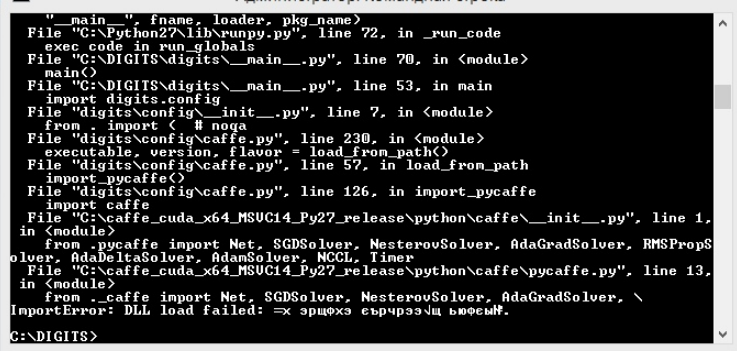
В случае успешной развертки нейронной сети в окне браузера отобразится следующее:

 Здесь можно начинать работу с нейронной сетью.



## Примечания к установке

При установке нейронной сети NVIDIA DIGITS на ОС Windows 8.1 может на шаге 11 возникнуть следующая ошибка:



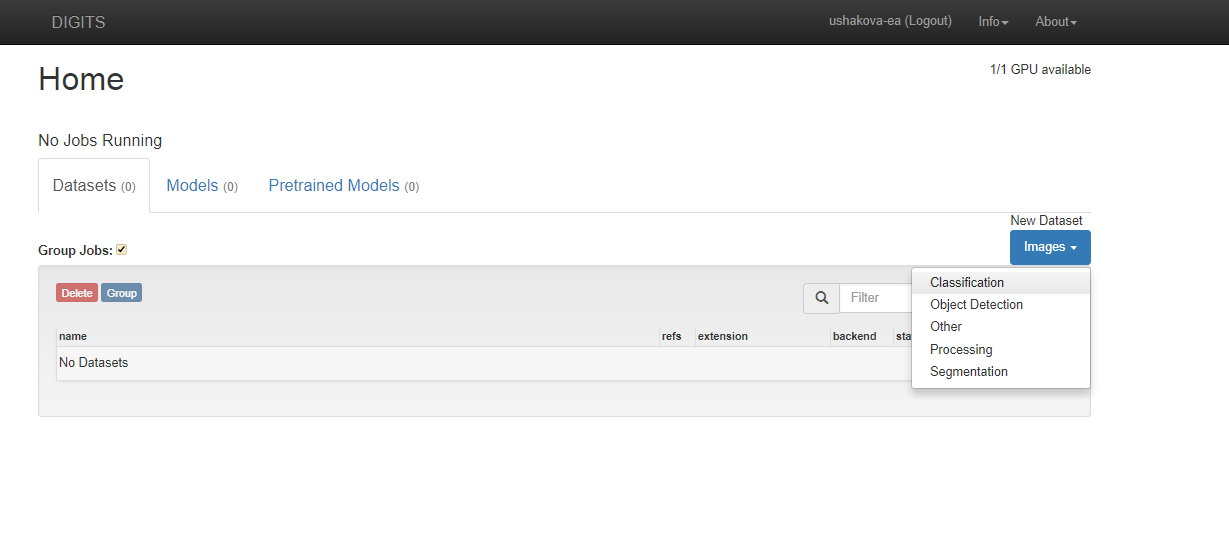
В этом случае скачайте и установите распространяемый пакет пакет Visual C++ для Visual Studio 2015 с ресурса

<https://www.microsoft.com/ru-ru/download/details.aspx?id=48145>

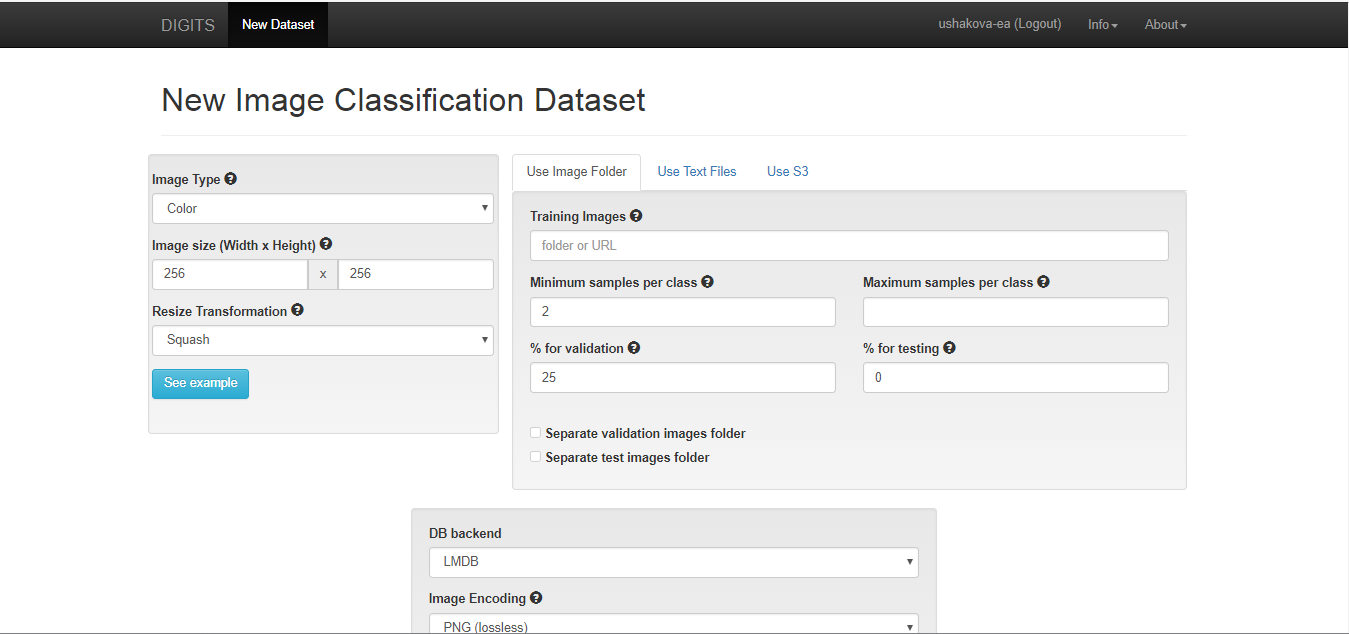
# Технология обучения нейронной сети NVIDIA DIGITS

## Формирование данных

1. Для формирования данных для решения задачи классификации необходимо открыть вкладку Datasets и выбрать Classification.



1. Далее откроется форма, в ней выбрать вкладку Use Image Folder

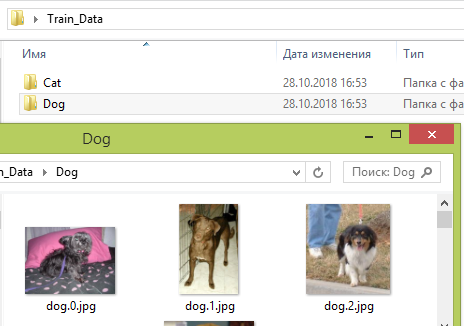


1. В форме присутствуют следующие параметры:

* Image Type – тип изображения (черно-белое или цветное)
* Image size (Width x Height) – Размер изображений (все изображения будут приведены к одномц размеру)
* Resize Transformation – опции для изменения соотношения сторон
* Training Images - папка, содержащая вложенные папки с изображениями.
* Minimum samples per class – минимальное количество изображений для каждого класса (можно оставить пустым – параметр будет проигнорирован)
* Maximum samples per class - максимальное количество изображений для каждого класса (можно оставить пустым – параметр будет проигнорирован)
* % for validation - процент изображений для валидации из обучающего набора
* % for testing – процент тестовых изображений из обучающего набора
* Separate validation images folder – отдельная папка с изображениями для валидации
* Separate test images folder – отдельная папка с тестовыми изображениями
* Image Encoding – тип изображений (все изоражения приводятся к определенному типу)
* Group Name – название группы
* Dataset Name – название набора

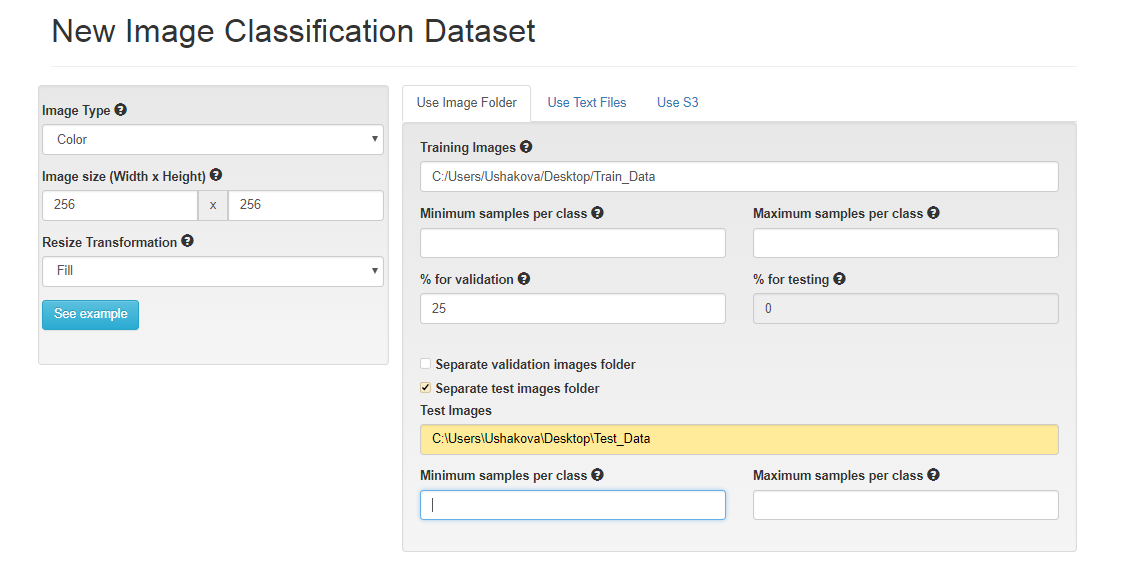
1. Для обучения нейронной сети потребуется DataSet. Для этого необходимо сформировать: трейновую, тестовую и валиационную часть.

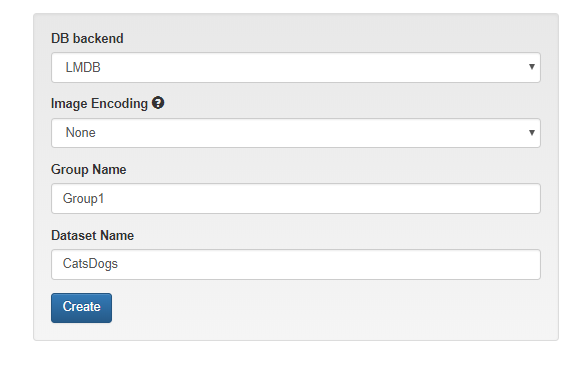
* Изображения для обучения нейронной сети нужно распределить по папкам, название папки – желаемая метка класса (label): Cats и Dogs. В каждой папке находятся изображения членов класса. Полученные папки вложить в одну папку – Train\_Data.



* При необходимости тестовые и валидационные изображения распределяются аналогичным образом в папки – Test\_Data, Val\_Data либо просто указывыется процент от Train\_Data

1. Далее необходимо заполнить поля в форме. Пример заполнения:



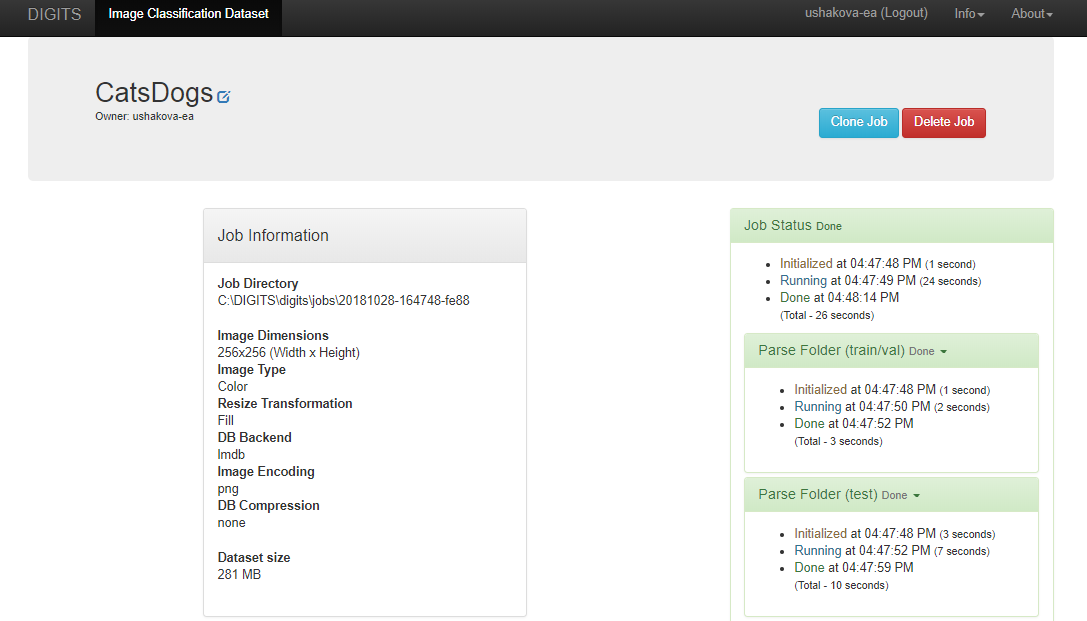


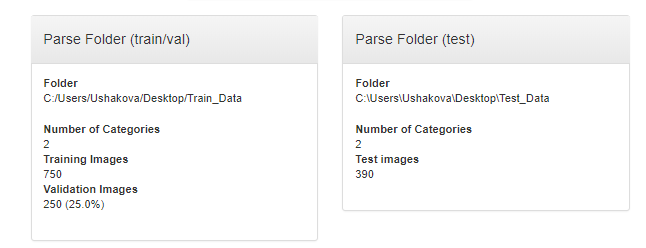
1. Нажать Create → Открывается форма с прогресс барами выполнения каждого из этапов → Об успешном завершении информирует положительный статус.

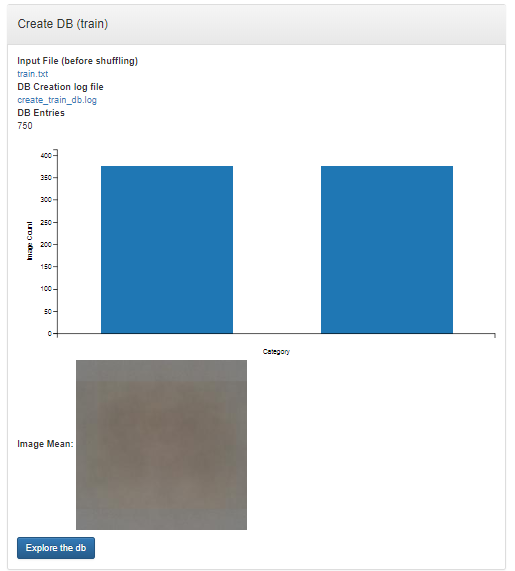


В результате можно увидеть:

* Время, затраченное на парсинг и создание данных.
* Число классов
* Число обучающих, валидационных и тествых изображений
* Среднее изображение классов
* Изображения каждого из наборов

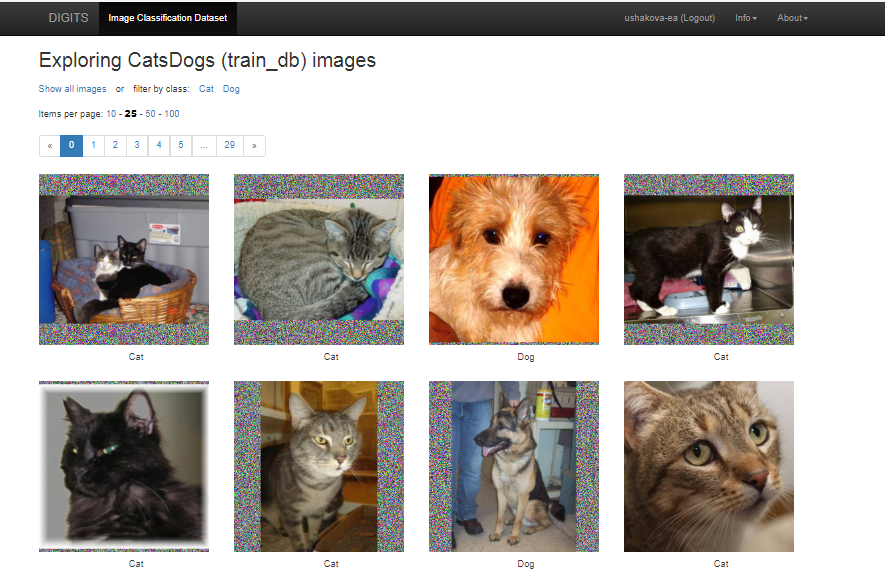




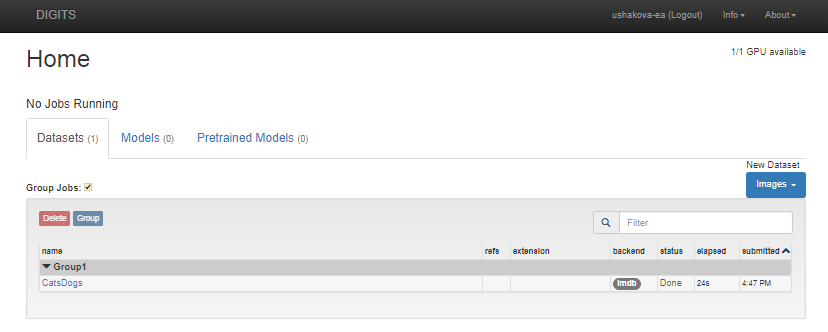






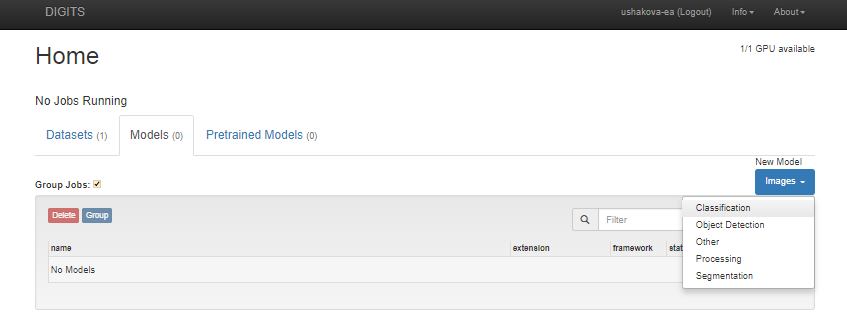


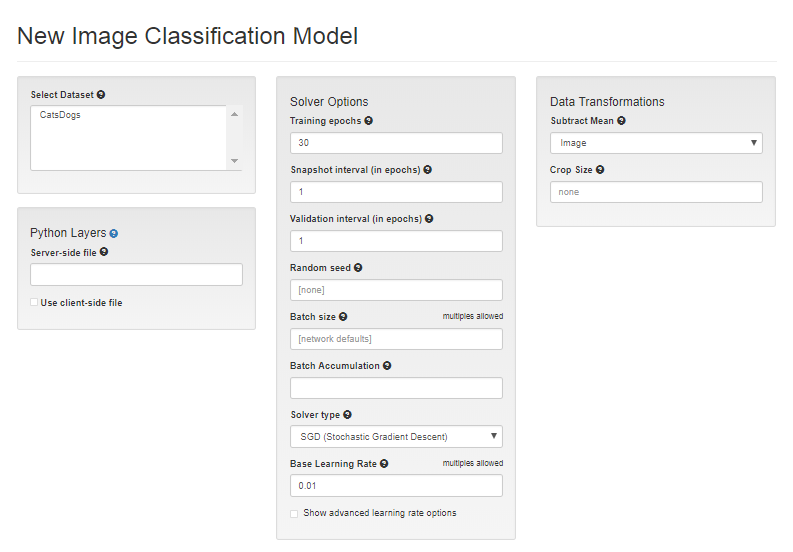
1. Сформированный DataSet будет сохранен во вкладке DataSets

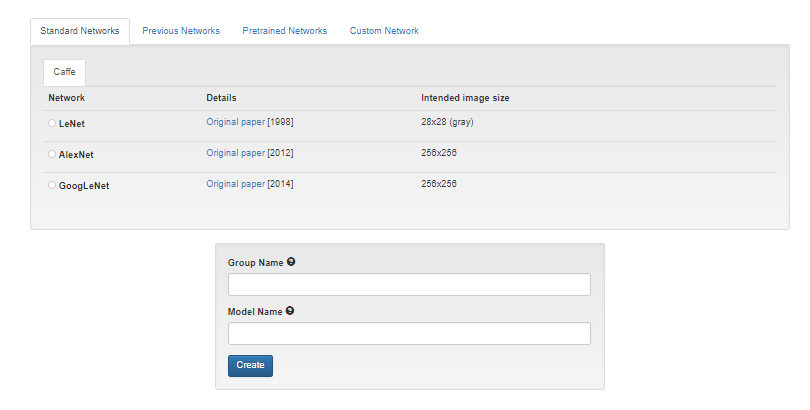


## Формирование модели

1. Для формирования модели для решения задачи классификации необходимо открыть вкладку Models и выбрать Classification.



1. Далее откроется форма



1. В форме присутствуют следующие параметры:

* Select Dataset - набор данных для модели

Слои Python:

* Server-side file – файл на сервере, содержащий определение слоев (можно использовать файл на стороне клиента)

Настройки для решения:

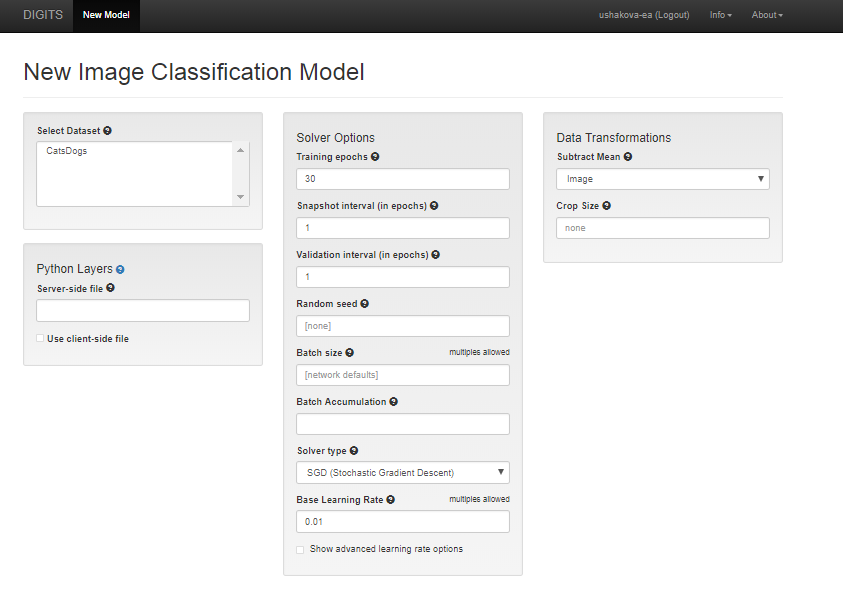
* Training epochs – количество эпох для обучения
* Snapshot interval (in epochs) – количество эпох обучения между съемками
* Validation interval (in epochs) - количество эпох обучения между прохождением через один проход валидации
* Random seed - случайное начальное значение
* Batch size – размер батча (сколько изображений сразу обрабатывается)
* Batch Accumulation – накапление градиентов по нескольким пакетам (полезно, когда вам нужен больший размер пакета для обучения, но он не помещается в память)
* Solver type – стратегия изменения весов нейронной сети
* Base Learning Rate – скорость обучения сети (принимает список через запятую)

Дополнительные параметры скорости обучения:

* Policy – правило
* Step Size - шаг
* Gamma - параметр

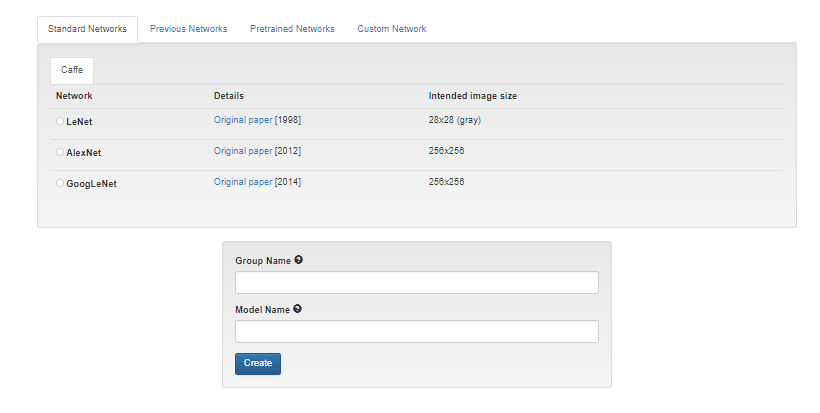
Преобазование данных:

* Subtract Mean – среднее избражение (вычитание среднего изображения из каждого изображения)
* Crop Size – размер кадра

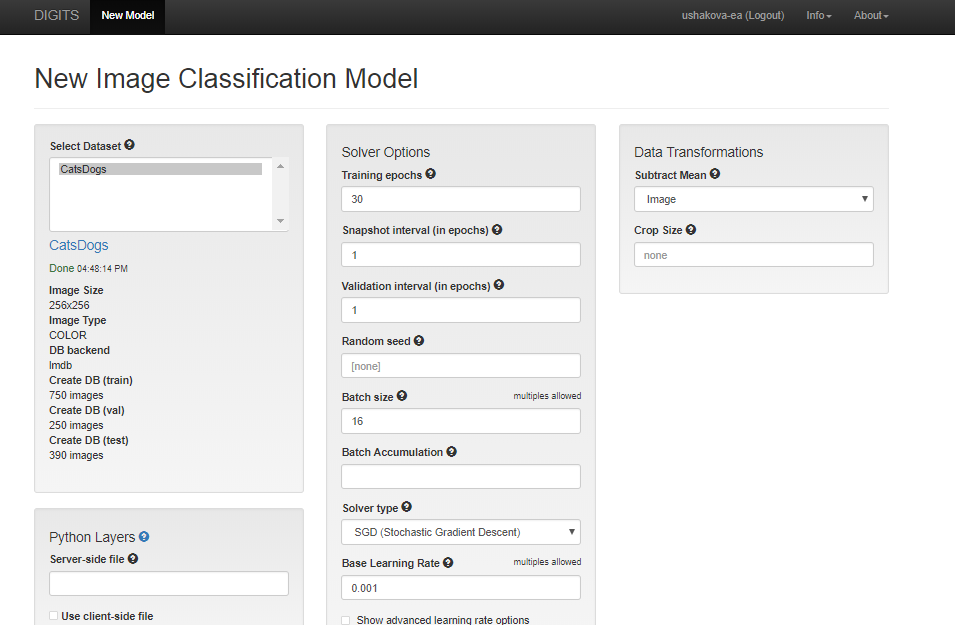


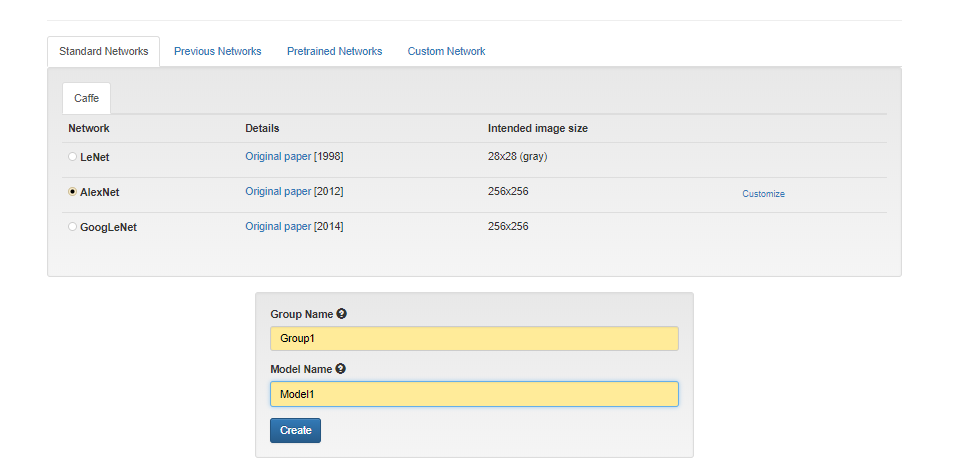
1. Для построения модели можно выбрать:

* одну из трех стандартных сетей **LeNet**, **AlexNet, GoogLeNet**
* ранее сформированную сеть или
* задать свою сеть

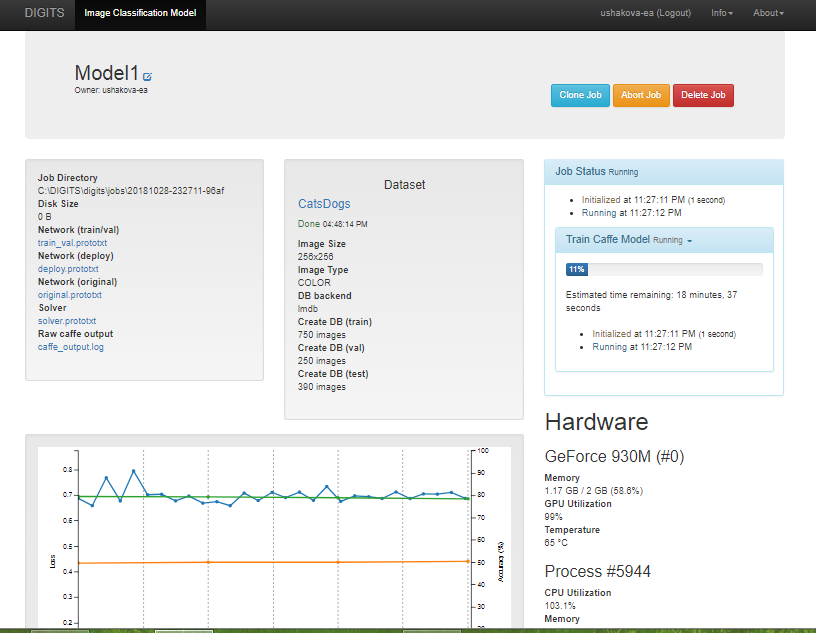


1. Далее необходимо заполнить поля в форме. Пример заполнения:



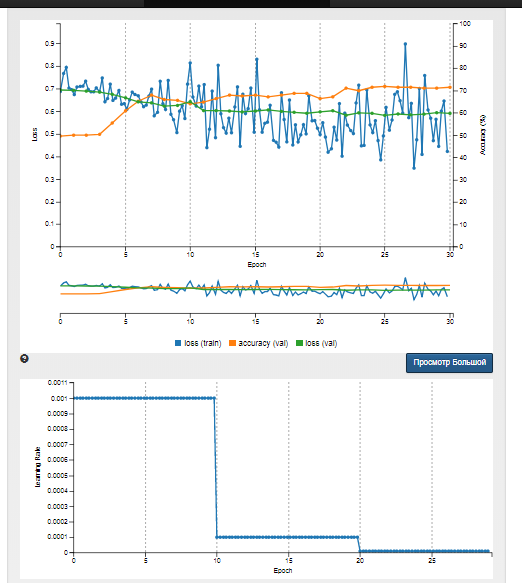


1. Нажать Create → Открывается форма с прогресс баром выполнения → Об успешном завершении информирует положительный статус - Done.

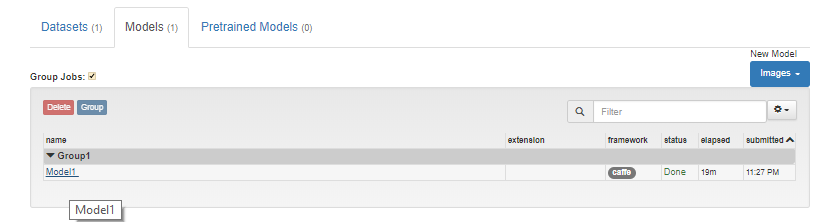


В результате можно увидеть:

* Время, затраченное на формирование модели
* Количество использованной памяти на видеокарте и процессоре
* Ошибку на трейновом наборе
* Ошибку на тестовом наборе
* Точность валидации
* Скорость обучения



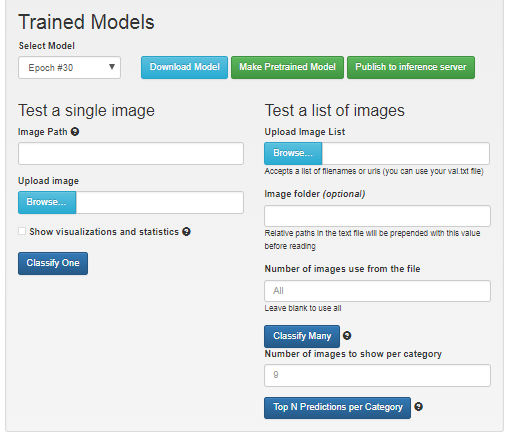
1. Модель будет сохранена во вкладке Models



Тестирование модели

Существует возможность проверить:

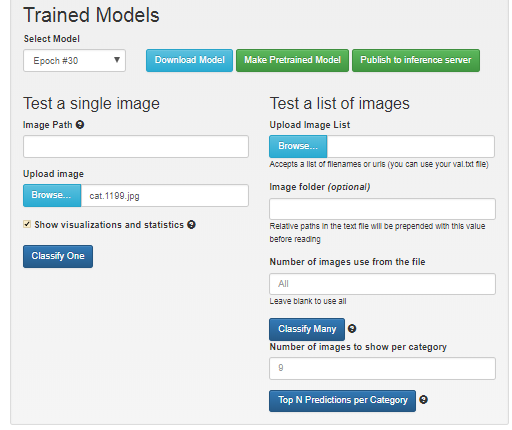
* Список изображений
* Одно изображение



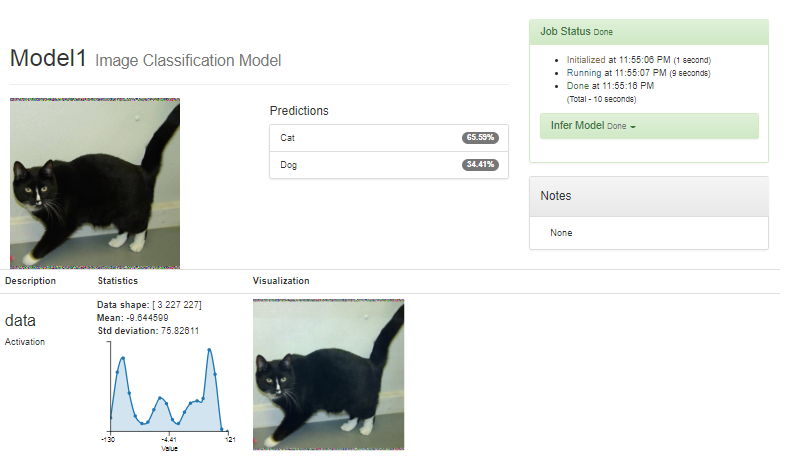
Для тестирования одного изображения

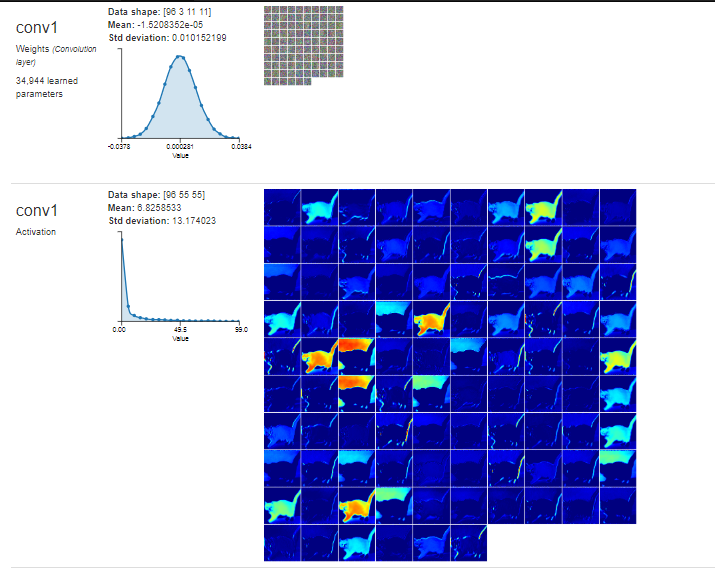
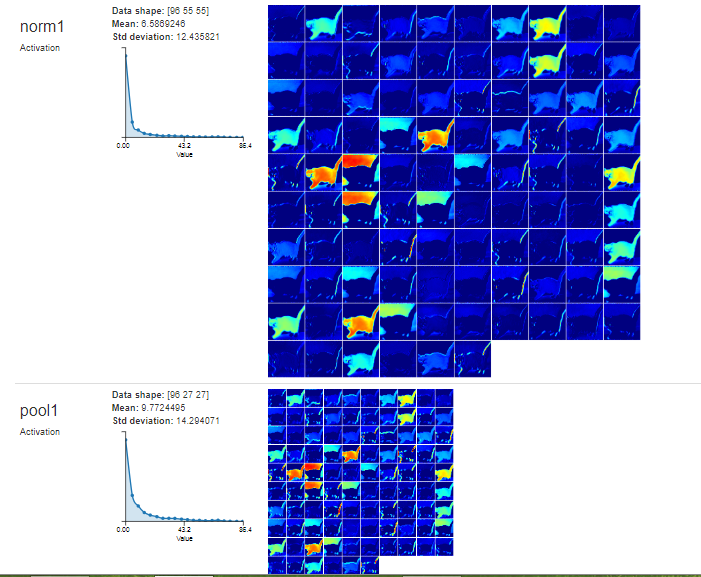
* Загрузить изображение
* Нажать Classify One

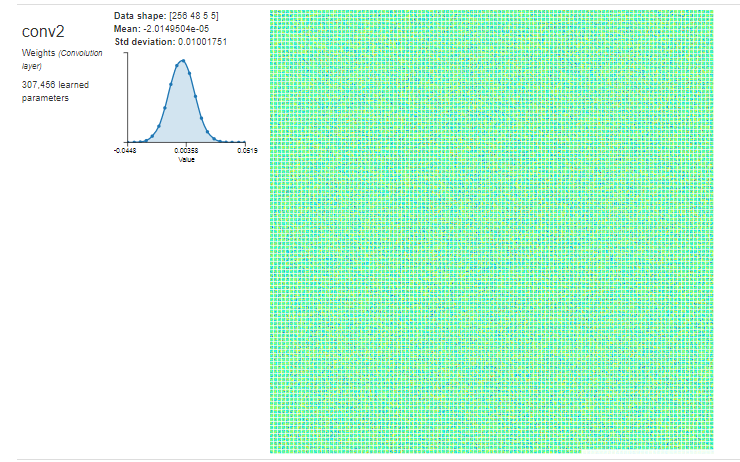
Например:

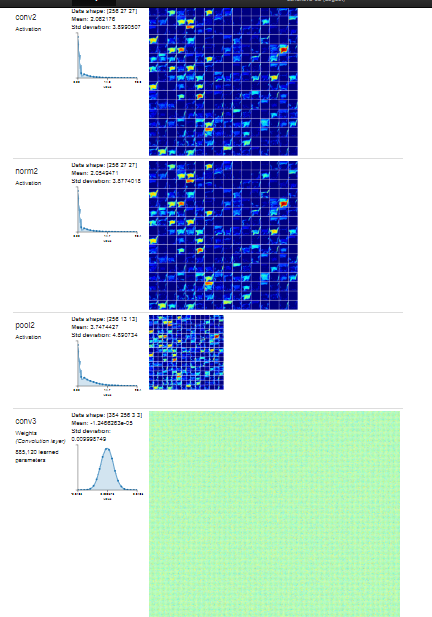


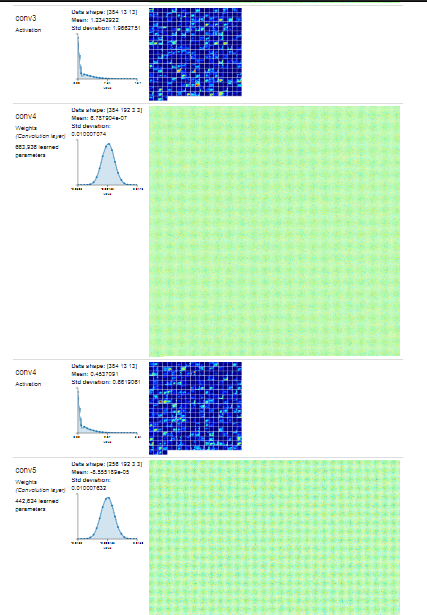
В результате можно увидеть:

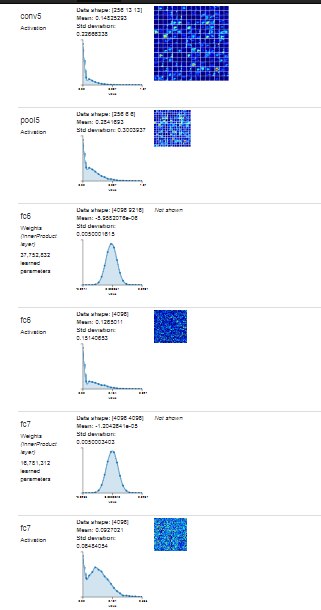
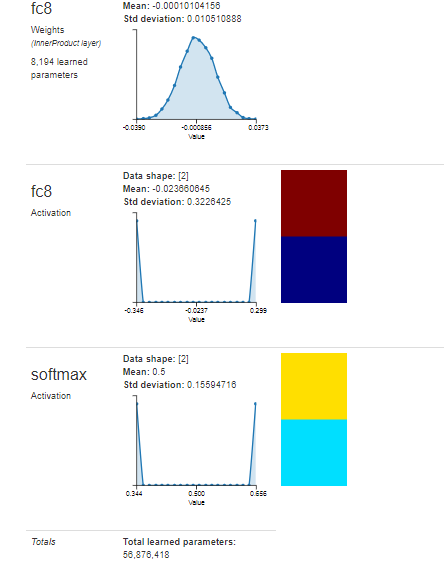
* Процент принадлежности к тому или иному классу
* Статистику и визуализацию по нормализации, пулингу и сверточным слоям
* График функции ошибки
* Количесто изученных параметров







# Технология принятия входных данных

# Заключение

# Список литературы

1. Техническое задание на на опытно-конструкторскую разработку «Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений. (ПО «SmartGetDistance»)». № 1 от 16.10.2018 г. Нижний Новгород, 2018.
2. Инструкция по развертыванию нейронной сети NVIDIA DIGITS **«**Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений**» («**SmartGetDistance**»**), г. Нижний Новгород, 2018.

# Приложение

Таблица 1

| №  модели,  стадии,  этапа | Наименование  модели, стадии, этапа | Сроки выполнения | | Ответственные | Вид  отчетности |
| --- | --- | --- | --- | --- | --- |
| начало | окончание |
| 1 | 2 | 4 | 5 | 6 | 7 |
| 1 | Согласование входных и выходных форматов. | 2.10.2018 | 23.10.2018 | Исполнитель  Заказчик | Пояснительная записка |
| 2 | Подготовка стека технологий (генерация нейроной сети) и тестовых изображений | 2.10.2018 | 23.10.2018 | Исполнитель | Пояснительная записка |
| 3 | 1ая поставка 5тыс. тестовых изображений | 9.10.2018 | 1.11.2018 | Заказчик | Файлы форматов исходных данных, контрольные данные |
| 4 | Обучение нейронной сети, разработка настроечных параметров  1ый релиз на 5тыс. тестовых изображений | 24.10.2018 | 13.11.2018 | Исполнитель | Пояснительная записка |
| 5 | 2ая поставка 100тыс. тестовых изображений | 2.11.2018 | 20.12.2018 | Заказчик | Файлы форматов исходных данных, контрольные данные |
| 6 | Программная реализация ПО. | 16.10.2018 | 30.12.2018 | Исполнитель | ПО на цифровом носителе |
| Разработка программной документации. | РО, РСП, ПМИ |
| Доработка ПО по результатам предварительных испытаний. | ПО на цифровом носителе |
| Проведение приемочных испытаний ПО « SmartGetDistance » | Протокол, Акт испытаний |
| Доклад о результатах ОКР | Отчет по ОКР, презентация |
|  |  |