**УТВЕРЖДАЮ**

Должность

\_\_\_\_\_\_\_\_\_\_\_\_\_ ФИО

“ ” \_\_\_\_\_\_\_\_\_ 2018 г.

**Пояснительная записка 3.**

**Технология обучения нейронной сети на примере двух классов.**

ОКР **«**Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений**»   
(«**SmartGetDistance**»)**

**Н.Новгород**

**2018**

Реферат

Пояснительная записка 27 страниц, 2 источника.

В пояснительной записке в рамках разработки проекта описан процесс обучения нейронной сети NVIDIA DIGITS на примере двух классов

Оглавление

[Термины и определения 4](#_Toc528535838)

[1 Технология обучения нейронной сети на примере двух классов: Cats и Dogs 5](#_Toc528535839)

[1.1 Формирование данных 5](#_Toc528535840)

[1.2 Формирование модели 12](#_Toc528535841)

[2 Тестирование модели 20](#_Toc528535842)

[Список используемых источников 27](#_Toc528535843)

# 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.

**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.

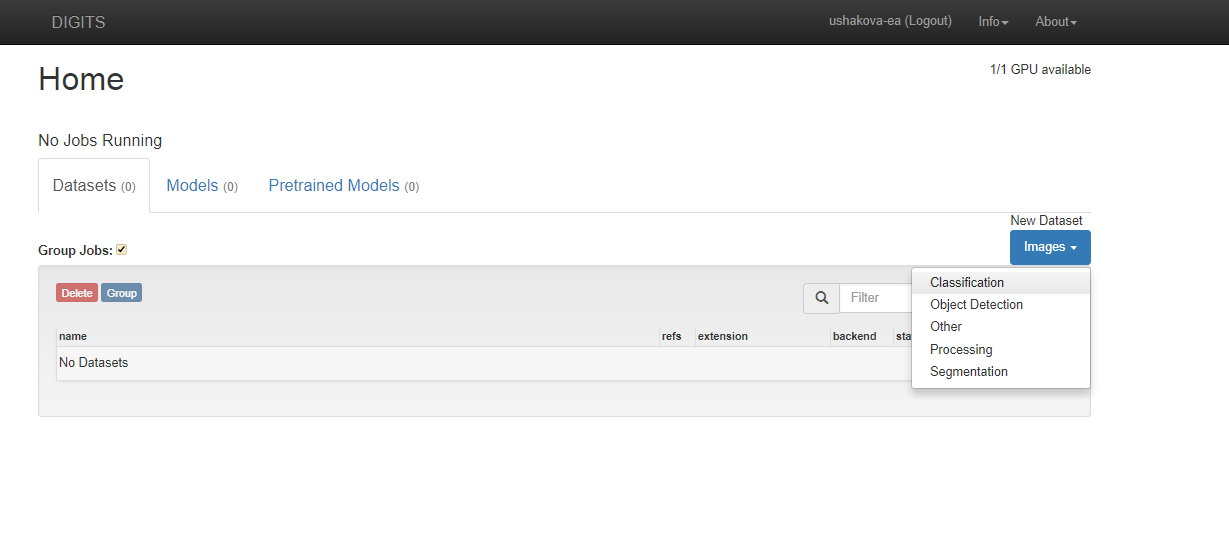
**Эпоха** - одна итерация в процессе обучения, включающая предъявление всех примеров из обучающего множества и, возможно, проверку качества обучения на контрольном множестве.

# Технология обучения нейронной сети на примере двух классов: Cats и Dogs

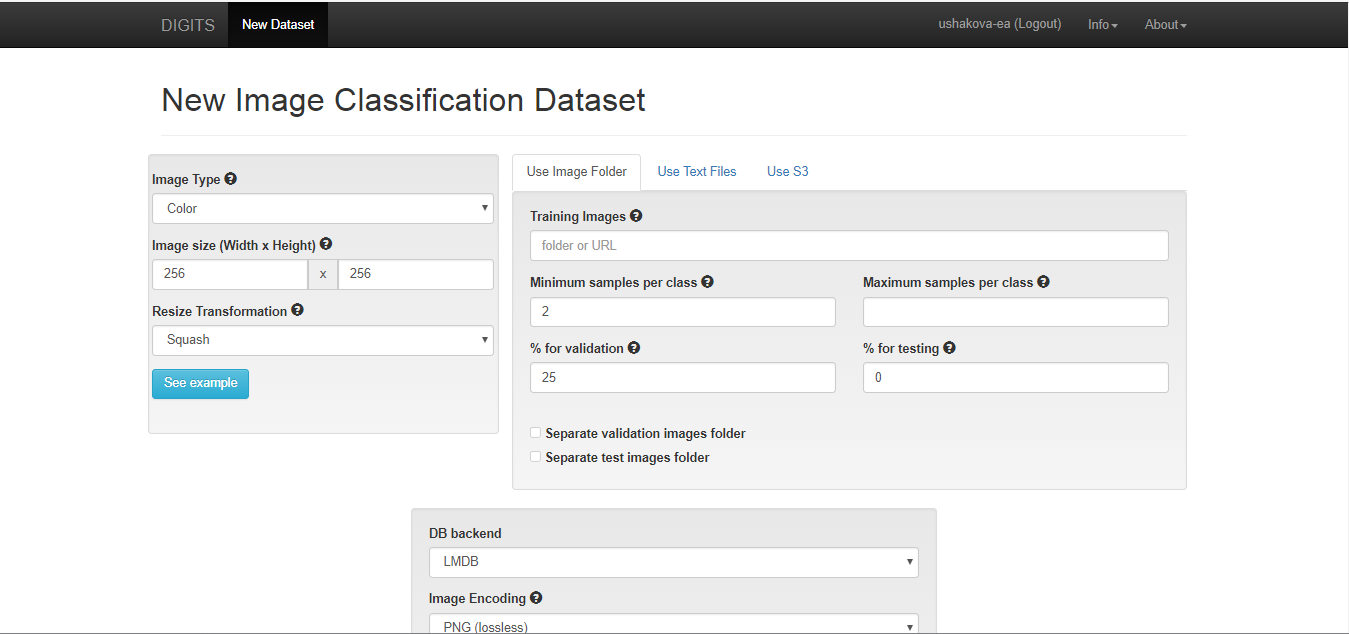
На машине необходимо развернуть нейронную сеть NVIDIA DIGITS (см. Инструкцию по развертыванию нейронной сети NVIDIA DIGITS).

## 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

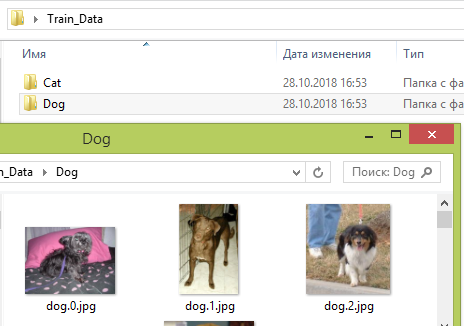


1. В форме присутствуют следующие параметры:

* 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

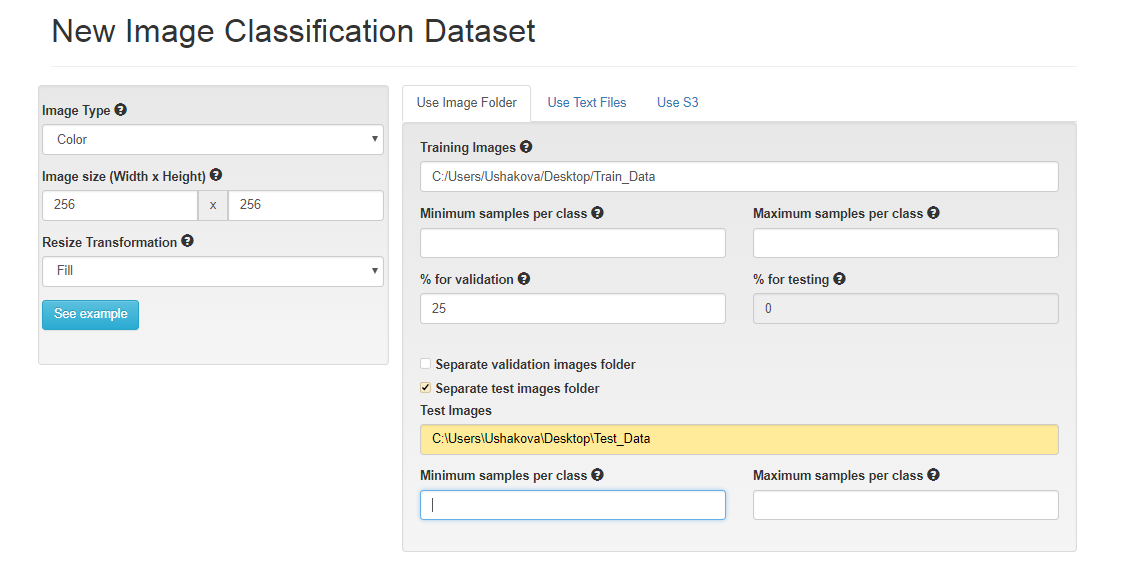
1. Для обучения нейронной сети потребуется DataSet. Для этого необходимо сформировать: трейновую, тестовую и валиационную часть.

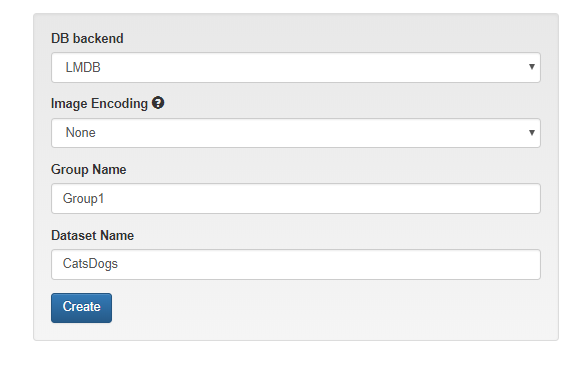
* Изображения для обучения нейронной сети нужно распределить по папкам, название папки – желаемая метка класса (label): Cats и Dogs. В каждой папке находятся изображения членов класса. Полученные папки вложить в одну папку – Train\_Data.



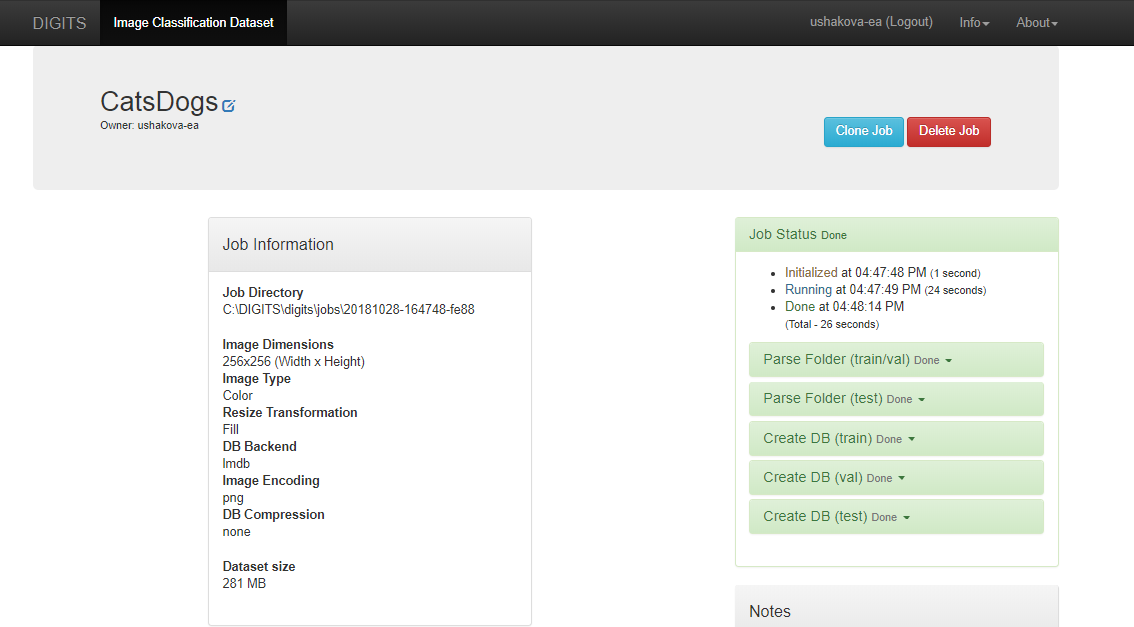
* При необходимости тестовые и валидационные изображения распределяются аналогичным образом в папки – Test\_Data, Val\_Data либо просто указывыется процент от Train\_Data

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



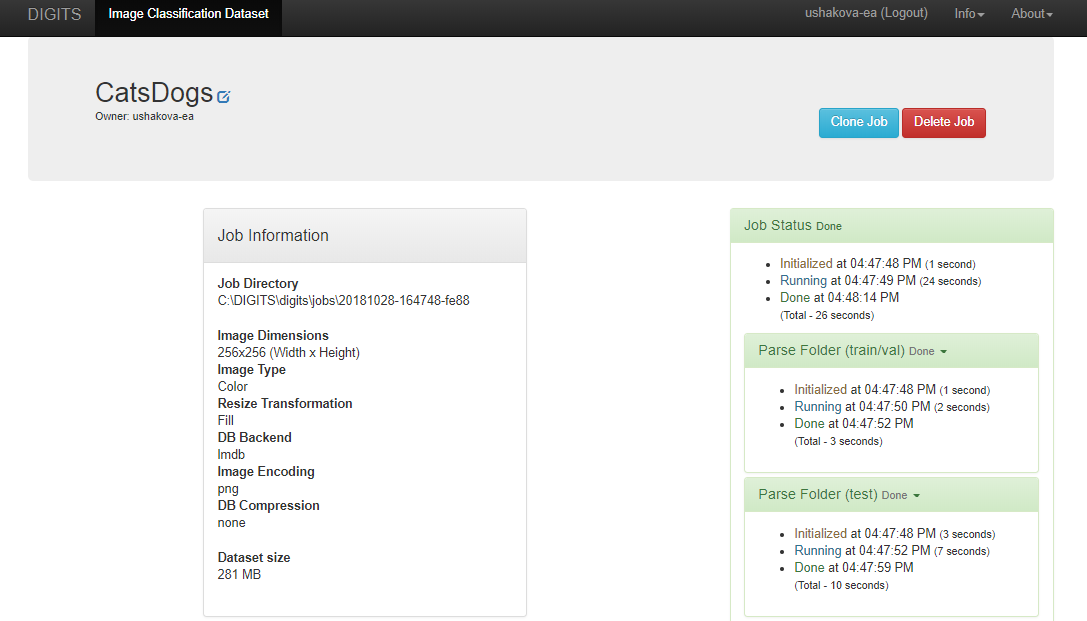


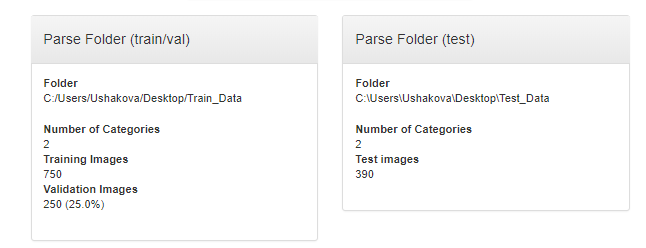
1. Нажать Create → Открывается форма с прогресс барами выполнения каждого из этапов → Об успешном завершении информирует положительный статус.

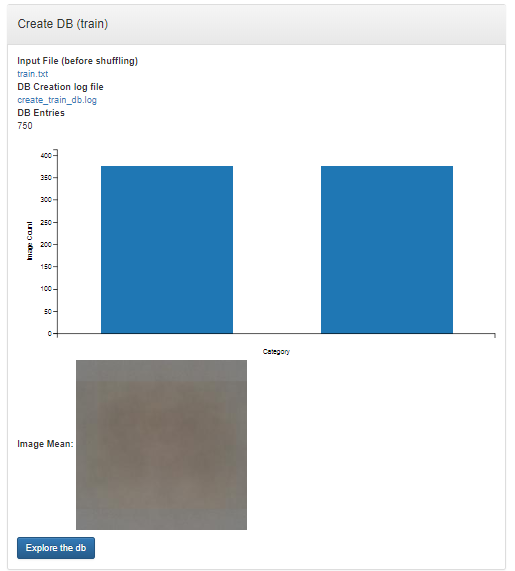


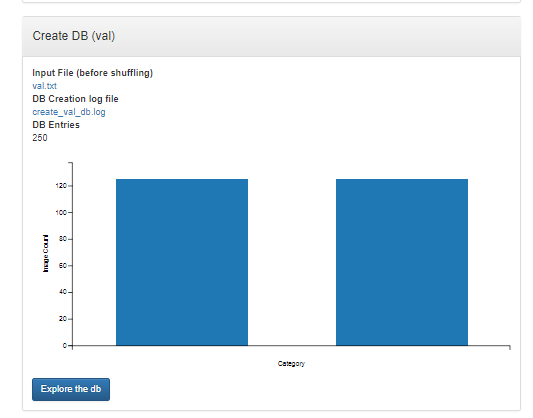
В результате можно увидеть:

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

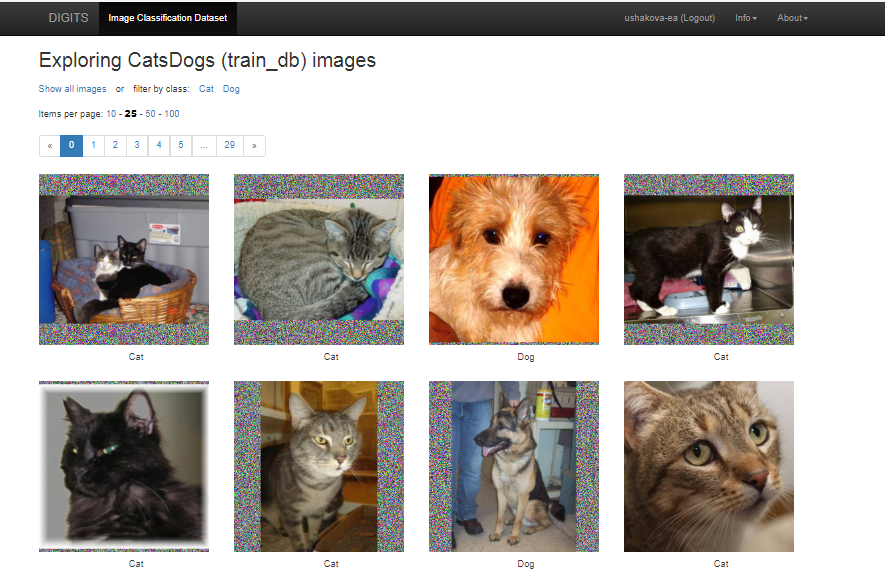




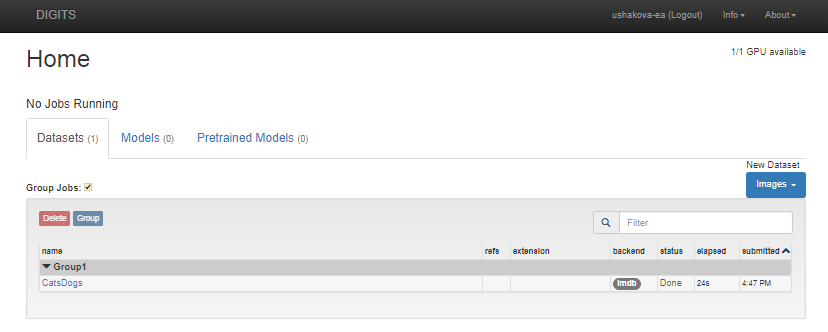






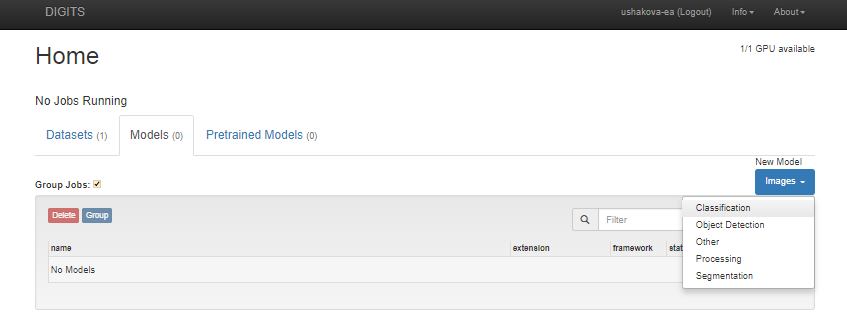


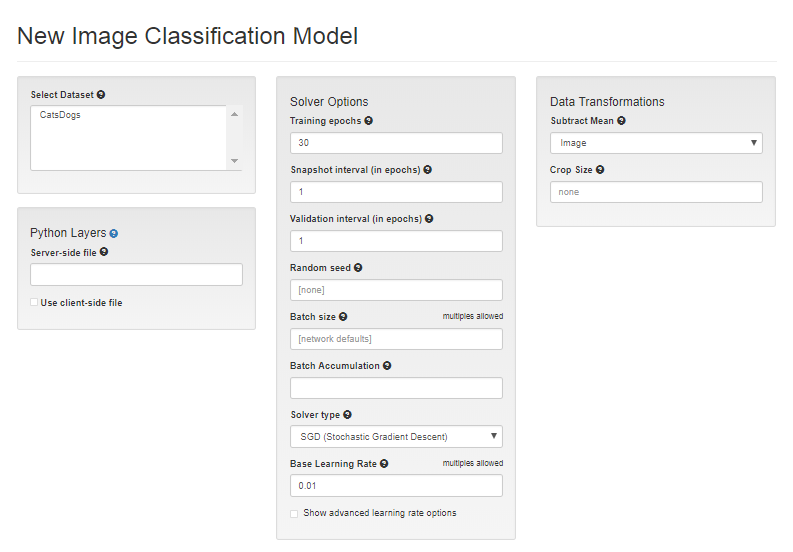
1. Сформированный DataSet будет сохранен во вкладке DataSets

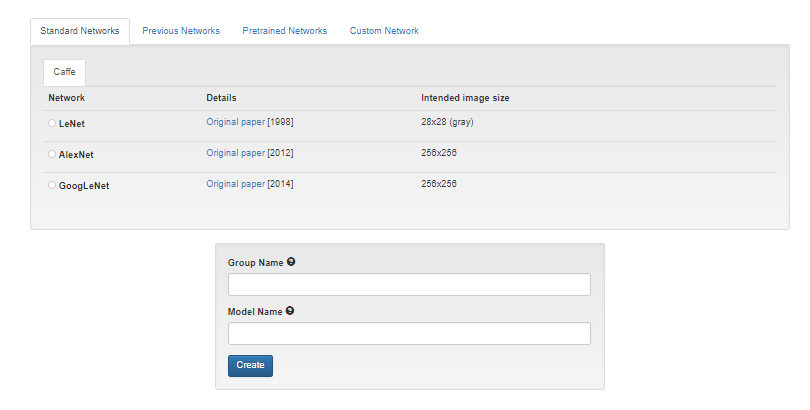


## Формирование модели

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:

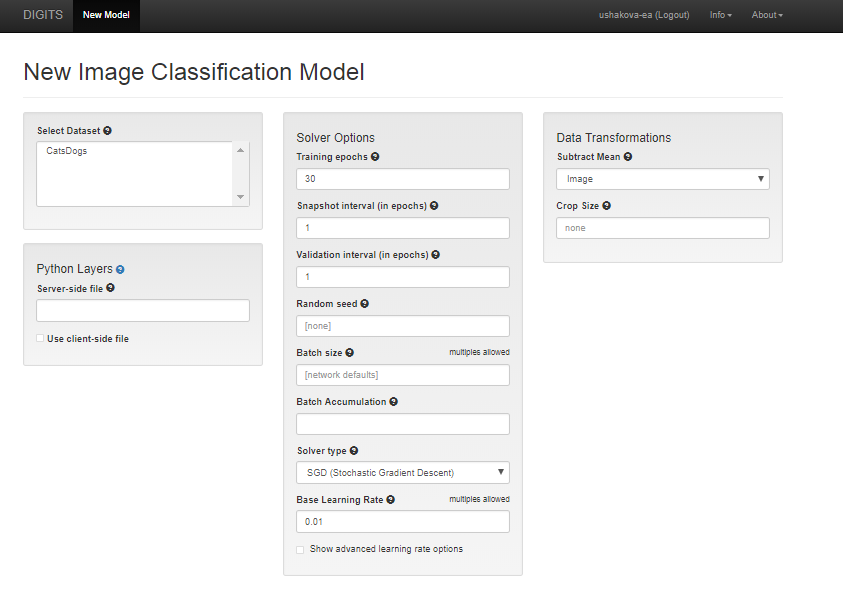
* Training epochs – количество эпох для обучения
* Snapshot interval (in epochs) – количество эпох обучения между съемками
* Validation interval (in epochs) - количество эпох обучения между прохождением через один проход валидации
* Random seed - случайное начальное значение
* Batch size – размер батча (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

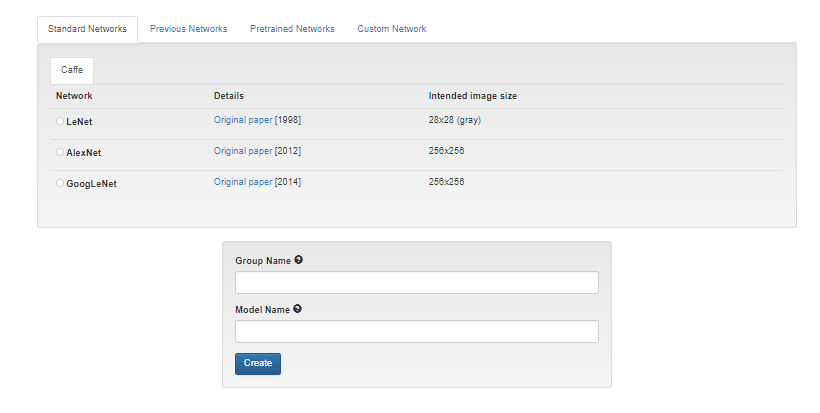
Преобазование данных:

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

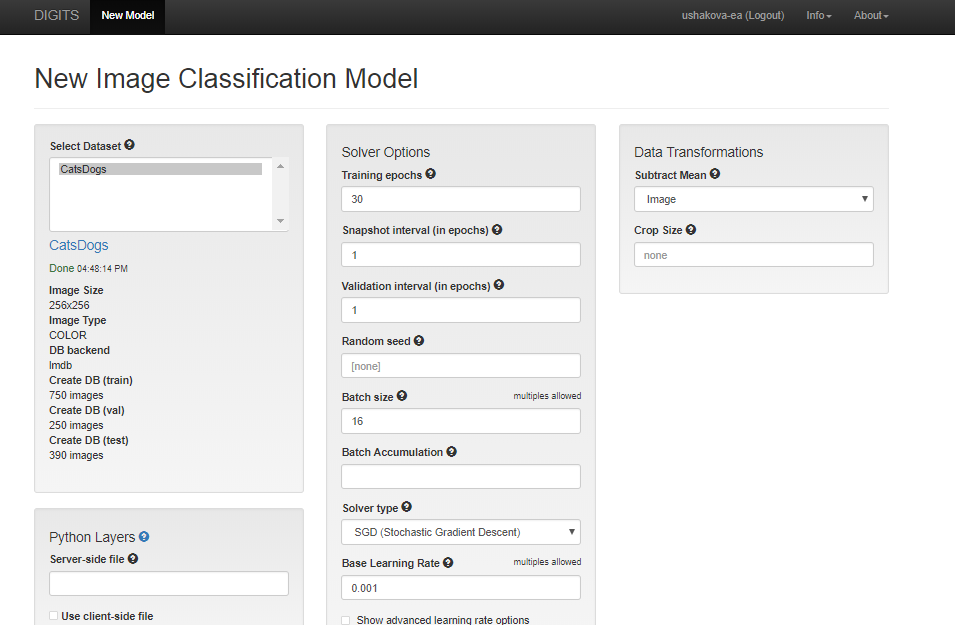


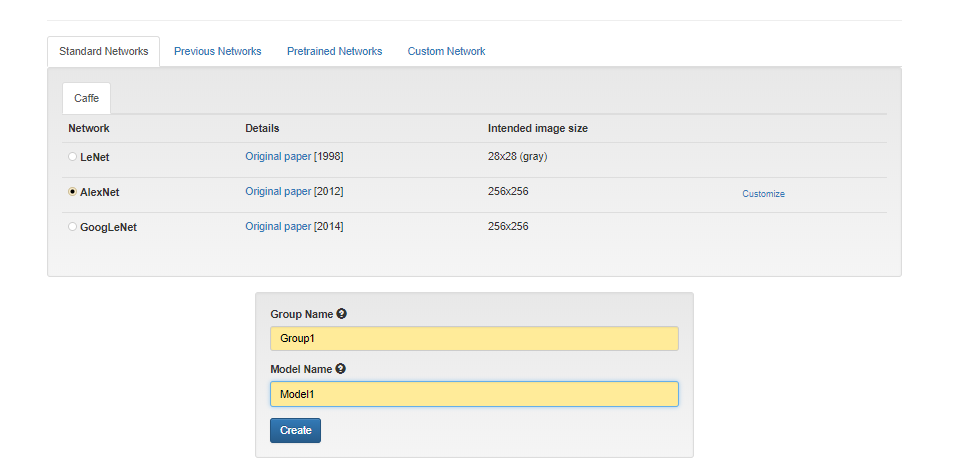
1. Для построения модели можно выбрать:

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

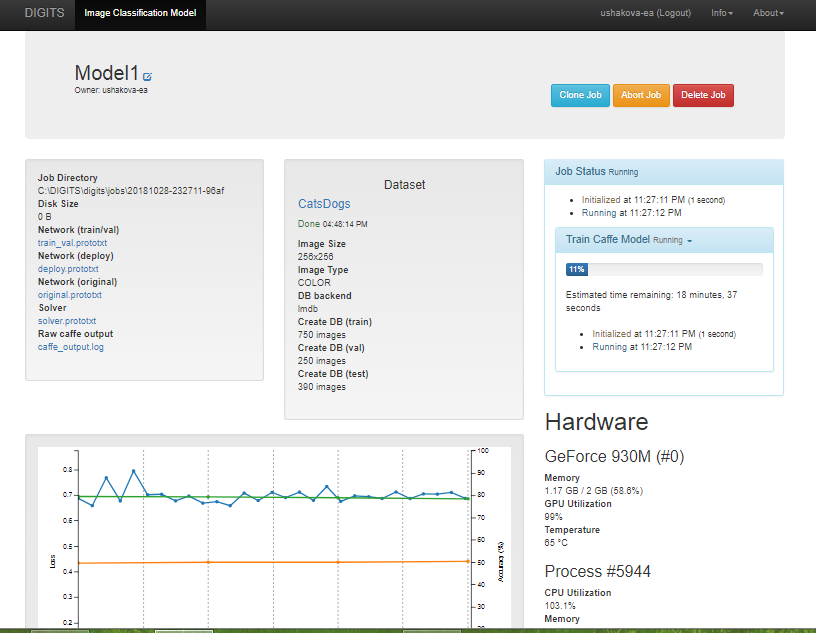


1. Далее необходимо заполнить поля в форме. Пример заполнения:



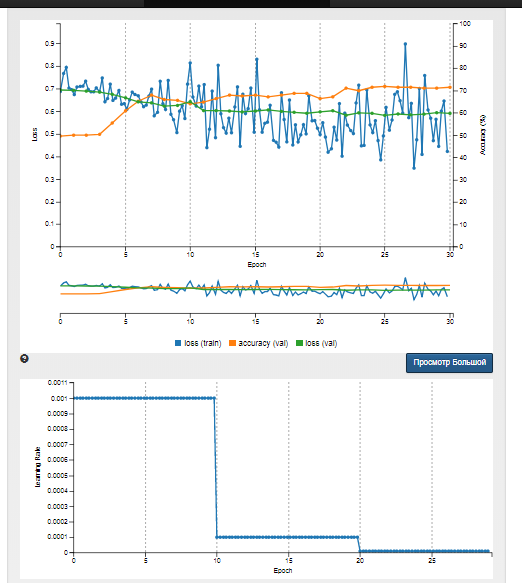


1. Нажать Create → Открывается форма с прогресс баром выполнения → Об успешном завершении информирует положительный статус - Done.

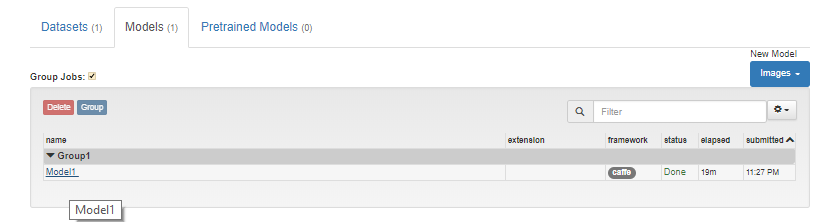


В результате можно увидеть:

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



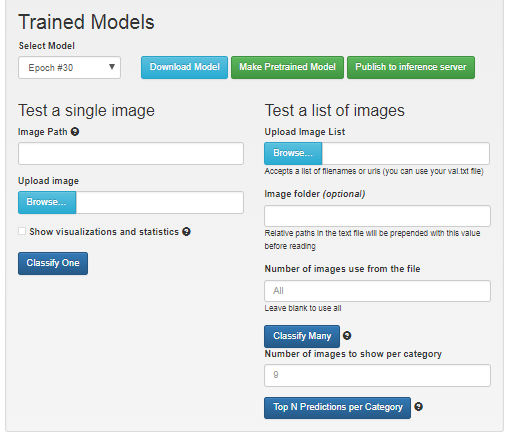
1. Модель будет сохранена во вкладке Models



# Тестирование модели

Существует возможность проверить:

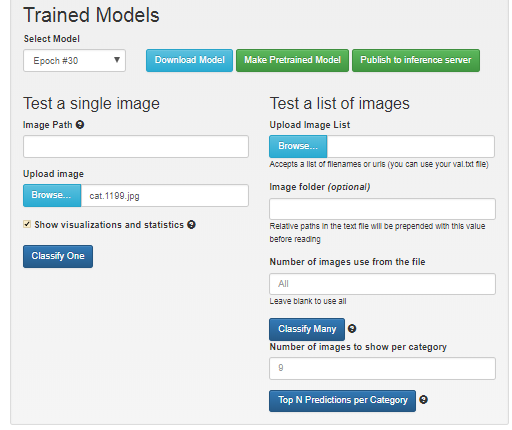
* Список изображений
* Одно изображение



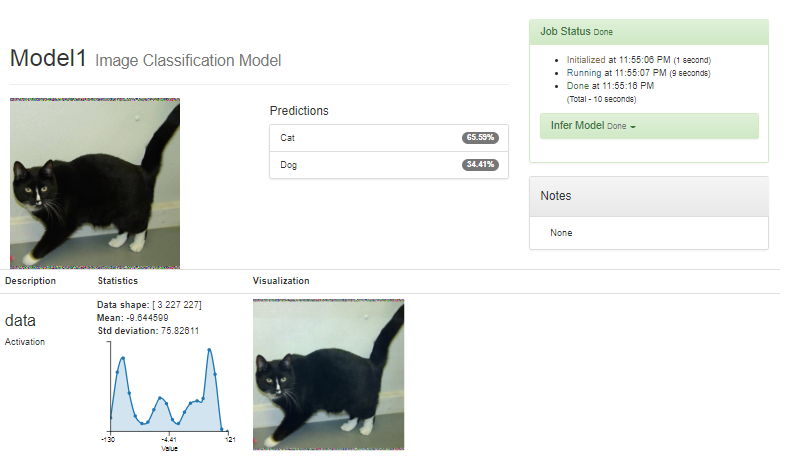
Для тестирования одного изображения

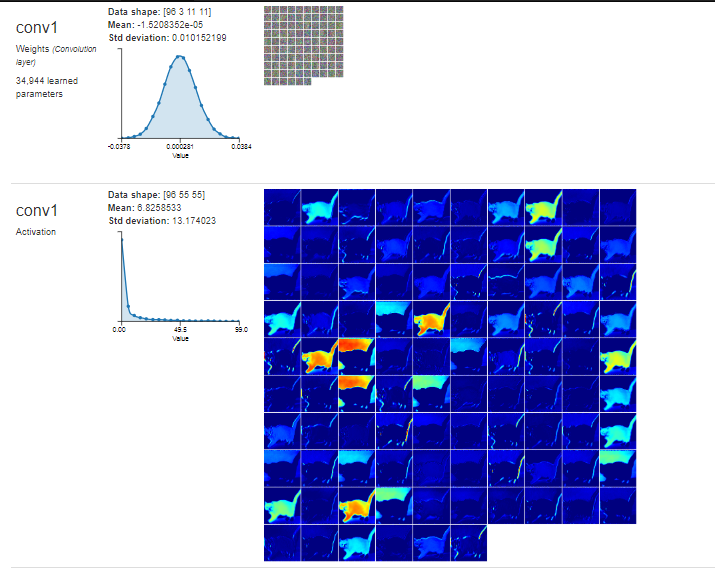
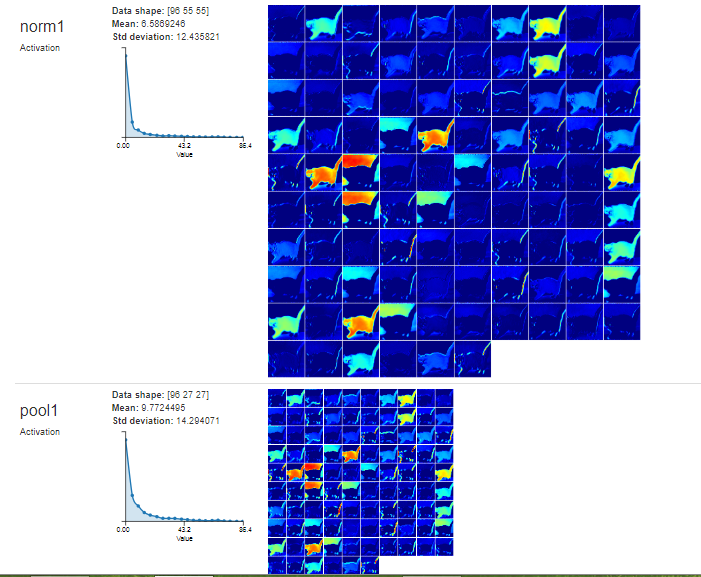
* Загрузить изображение
* Нажать Classify One

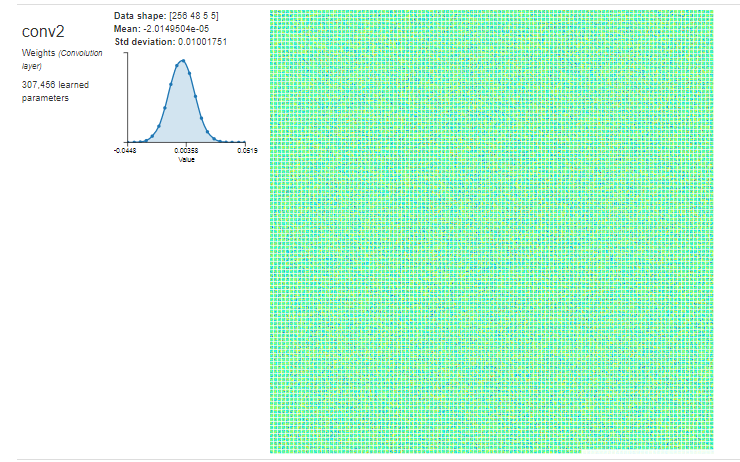
Например:

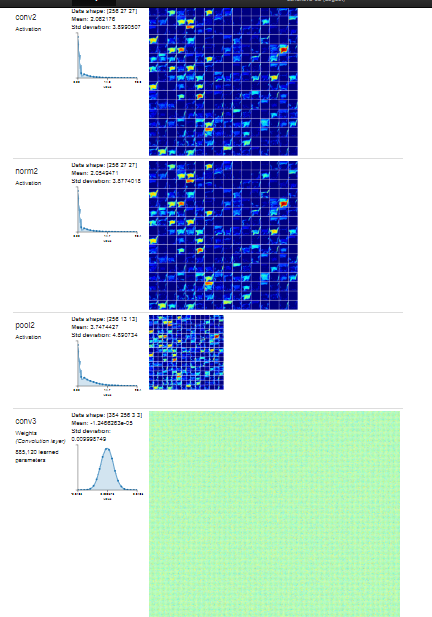


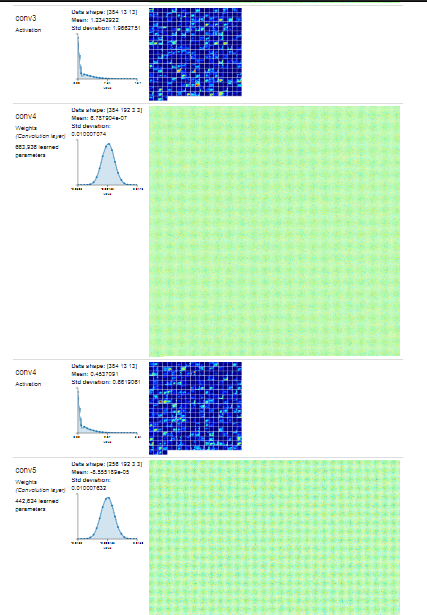
В результате можно увидеть:

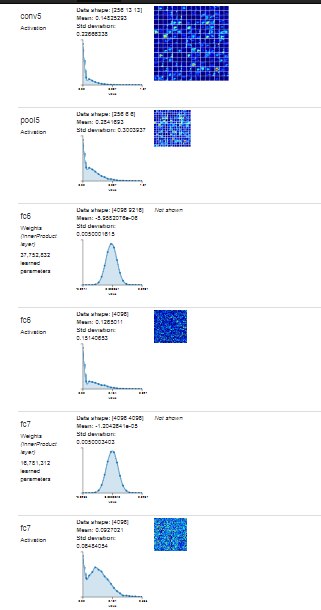
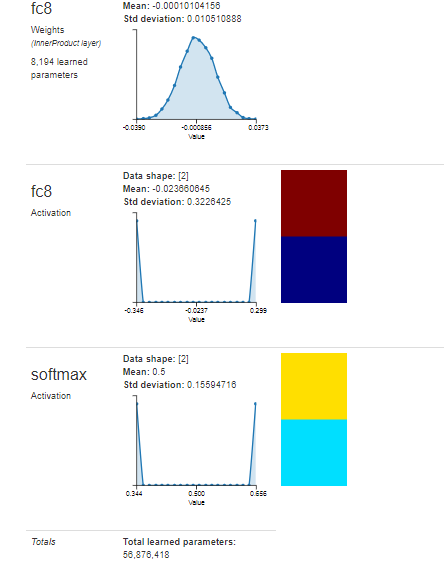
* Процент принадлежности к тому или иному классу
* Статистику и визуализацию по нормализации, пулингу и сверточным слоям
* График функции ошибки
* Количесто изученных параметров







# Список используемых источников

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.