Приложение

к договору №

от\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| СОГЛАСОВАНО | УТВЕРЖДАЮ |
| Сторона ЗАКАЗЧИКА  Старостин Н. В.  «04» декабря 2018 г. | Сторона ИСПОЛНИТЕЛЯ  Губарев С. Ю.  «04» декабря 2018 г. |

**Руководство системного программиста на опытно-конструкторскую разработку**

Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений.

**(Шифр ПО «**SmartGetDistance**»)**

2018

Оглавление

[1. Общие положения 3](#_Toc529372860)

[1.1. Область применения 3](#_Toc529372861)

[1.2. Краткое описание возможностей 3](#_Toc529372862)

[1.3. Назначение и условия применения «SmartGetDistance» 3](#_Toc529372863)

[1.4. Уровень подготовки пользователя 3](#_Toc529372864)

[2. Настройка программы 4](#_Toc529372865)

[2.1. Подготовка исходных данных 4](#_Toc529372866)

[2.2. Развертка и настройка нейронной сети 4](#_Toc529372867)

[Примечание 10](#_Toc529372868)

[2.3. Порядок работы с нейронной сетью 11](#_Toc529372869)

[Формирование модели 18](#_Toc529372870)

[Тестирование модели 25](#_Toc529372871)

[3. Проверка работоспособности программы и дополнительные возможности 27](#_Toc529372872)

## 1. Общие положения

1.1. Область применения

Опытно-конструкторская разработка: «Построение и обучение нейронной сети для идентификации фокусного расстояния по серии изображений. ПО «SmartGetDistance»».

1.2. Краткое описание возможностей

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.

1.3. Назначение и условия применения «SmartGetDistance»

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.

1.4. Уровень подготовки пользователя

Пользователь ПО «Get3DModel» должен иметь опыт работы с ОС MS Пользователь ПО «SmartGetDistance» должен иметь опыт работы с ОС MS Windows (XP/Windows 7/ Windows 8/8.1 /Windows 10), навык работы с командной строкой, а также обладать следующими знаниями:

* знать соответствующую предметную область;
* допустимые параметры входных данных.

1.5. Перечень эксплуатационной документации, с которой необходимо ознакомиться пользователю

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

## 2. Настройка программы

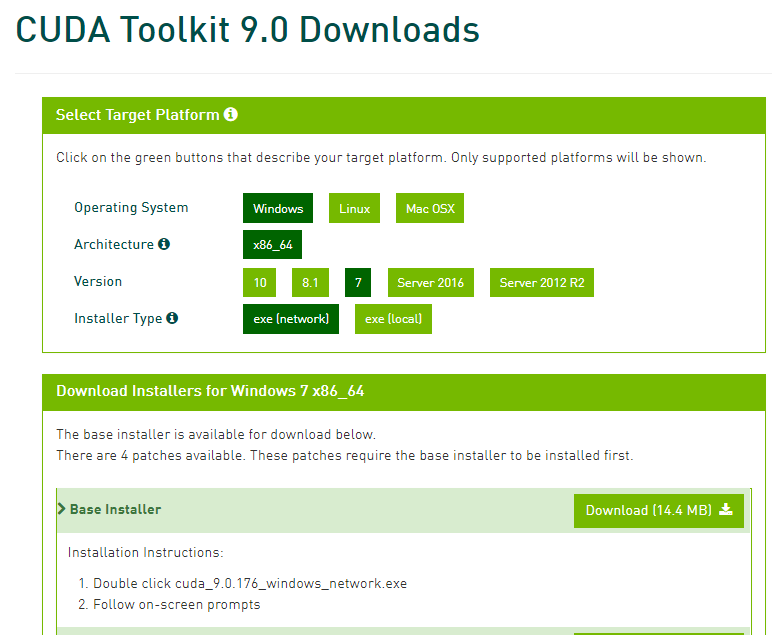
2.1. Подготовка исходных данных

Программка, которая готовит картинки

2.2. Развертка и настройка нейронной сети

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

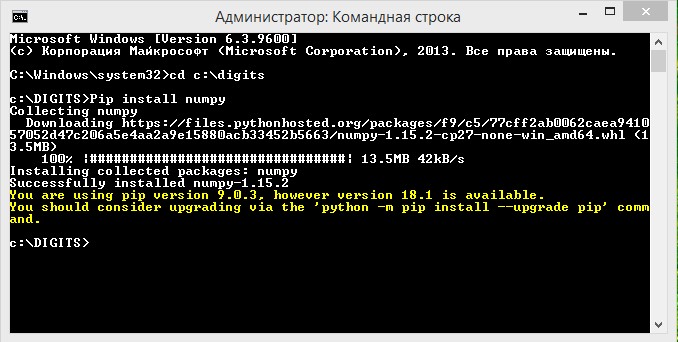
c:\pgramm file(86)\graphviz 2.38\bin

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

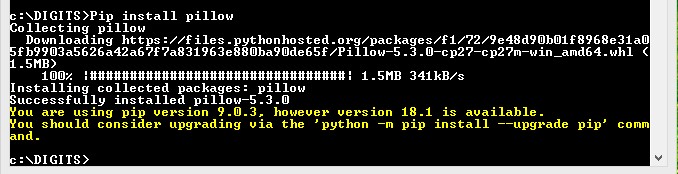
Перейдите в директорию DIGITS: cd c:\digits

Установите следующие библиотеки:

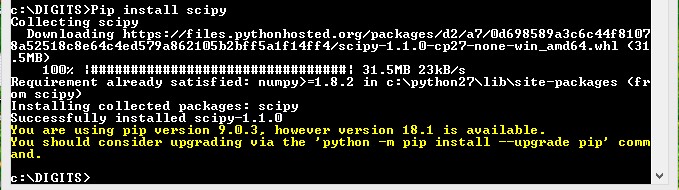
Pip install numpy



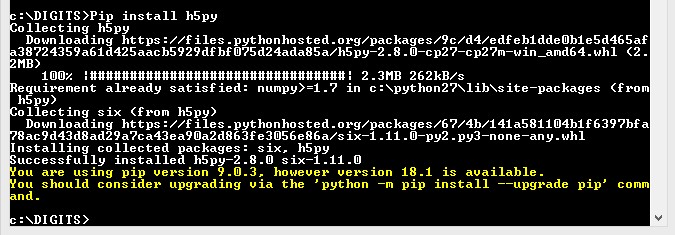
Pip install pillow



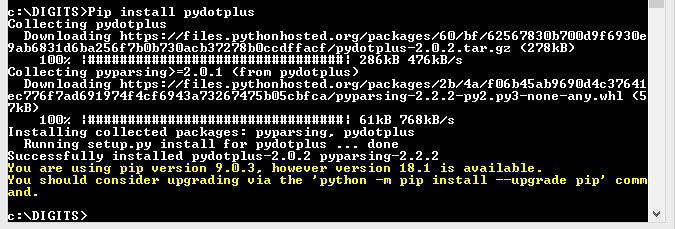
Pip install scipy



Pip install h5py



Pip install pydotplus

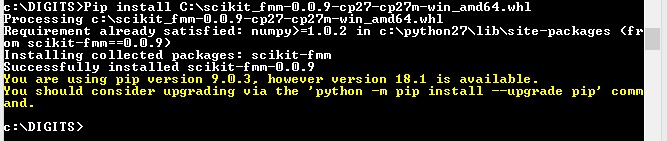


Установите whl пакеты:

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

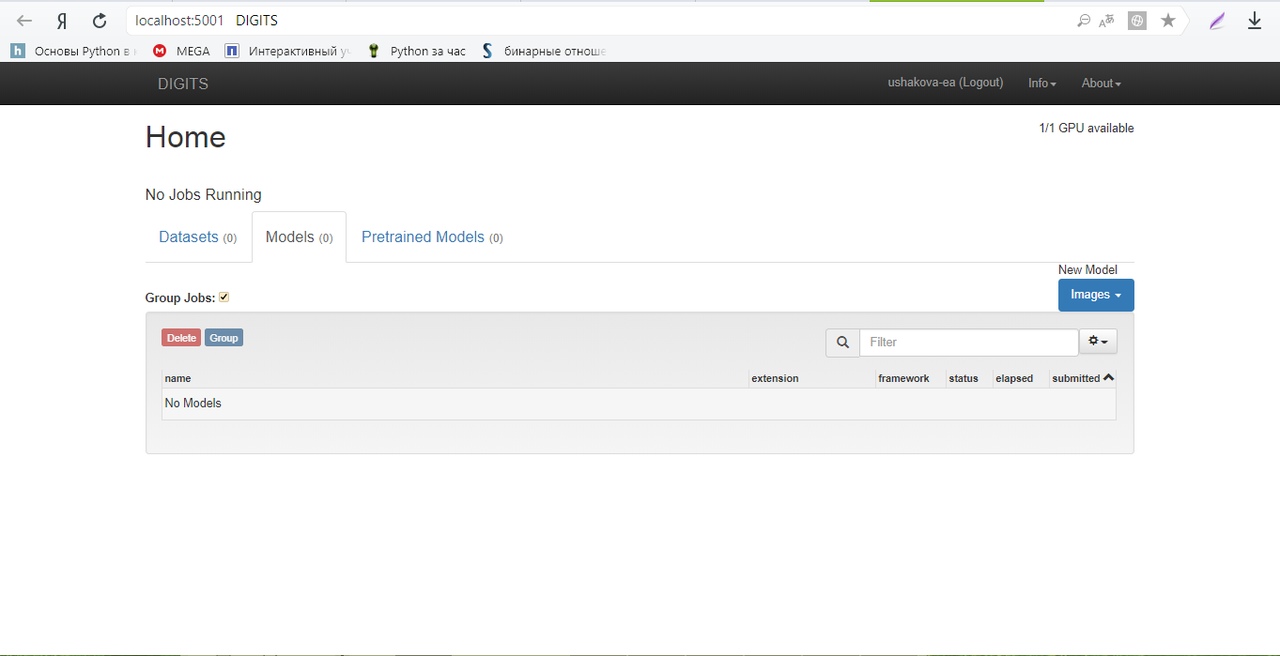


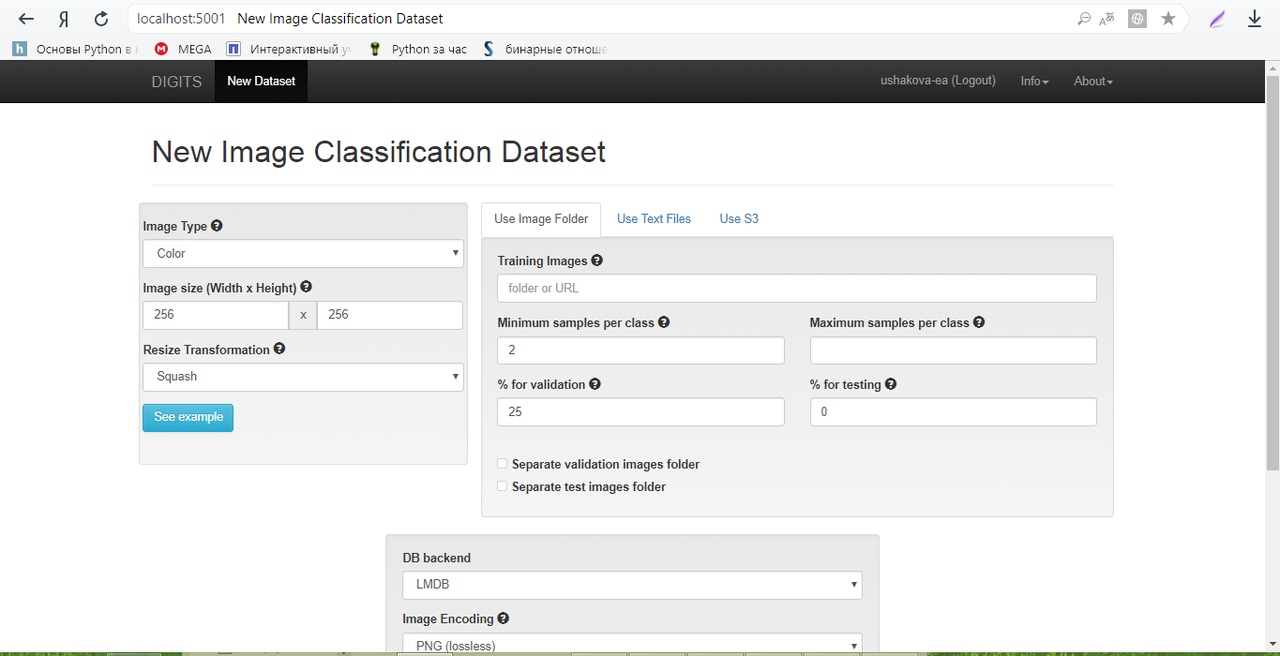
1. Установите оставшиеся Flask библиотеки с помощью команды

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

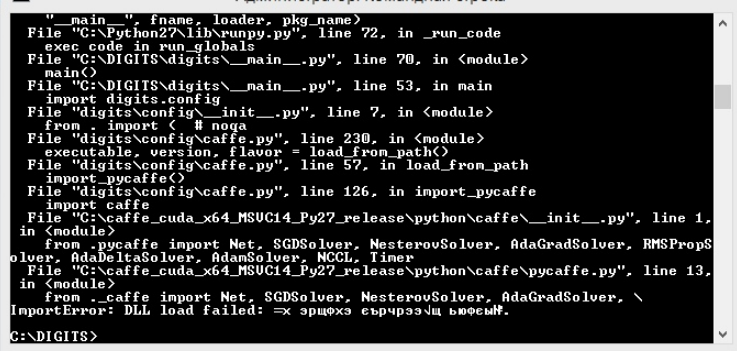
В случае успешной развертки нейронной сети в окне браузера отобразится следующее:

 Здесь можно начинать работу с нейронной сетью.



# Примечание

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

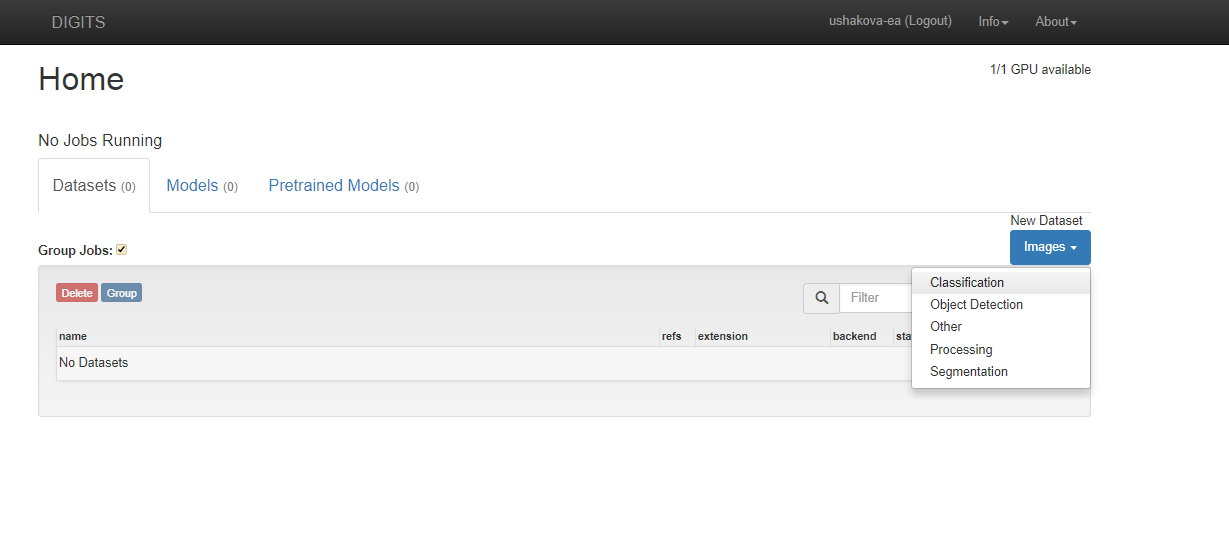


В этом случае скачайте и установите распространяемый пакет пакет Visual C++ для Visual Studio 2015 с ресурса

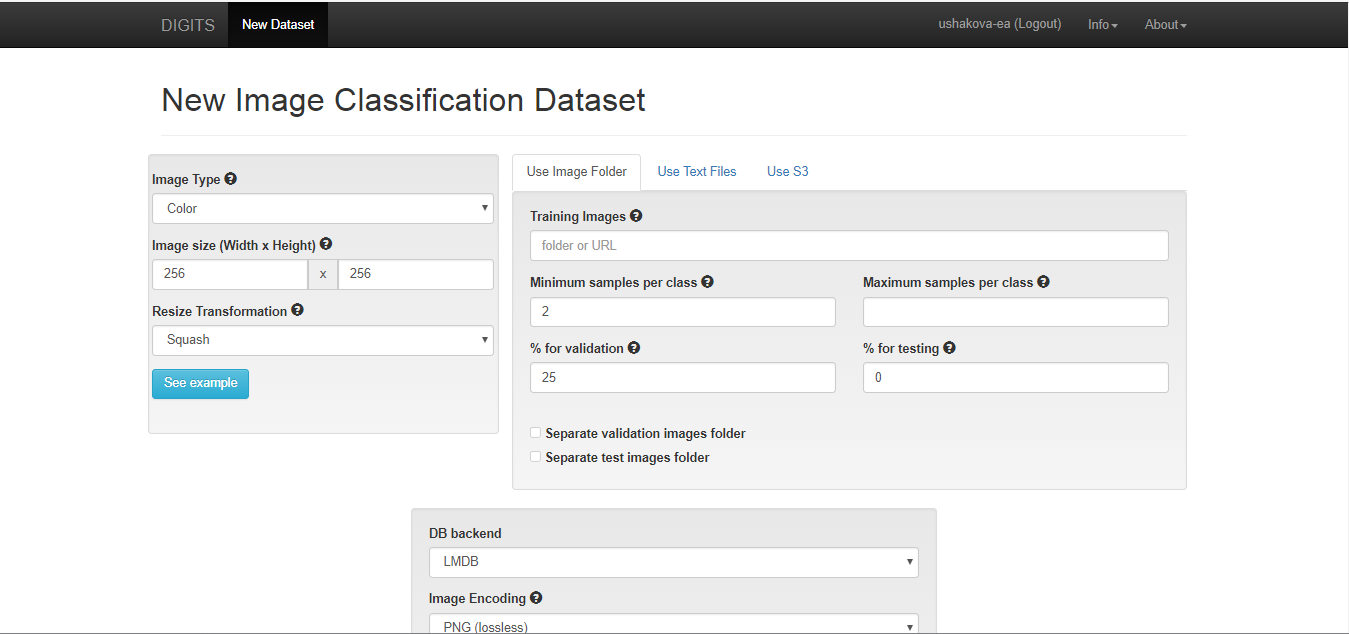
<https://www.microsoft.com/ru-ru/download/details.aspx?id=48145>

## 2.3. Порядок работы с нейронной сетью

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

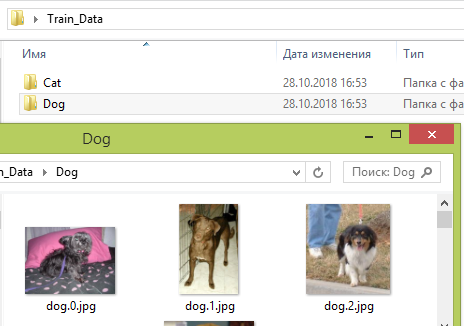


1. В форме присутствуют следующие параметры:

* 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

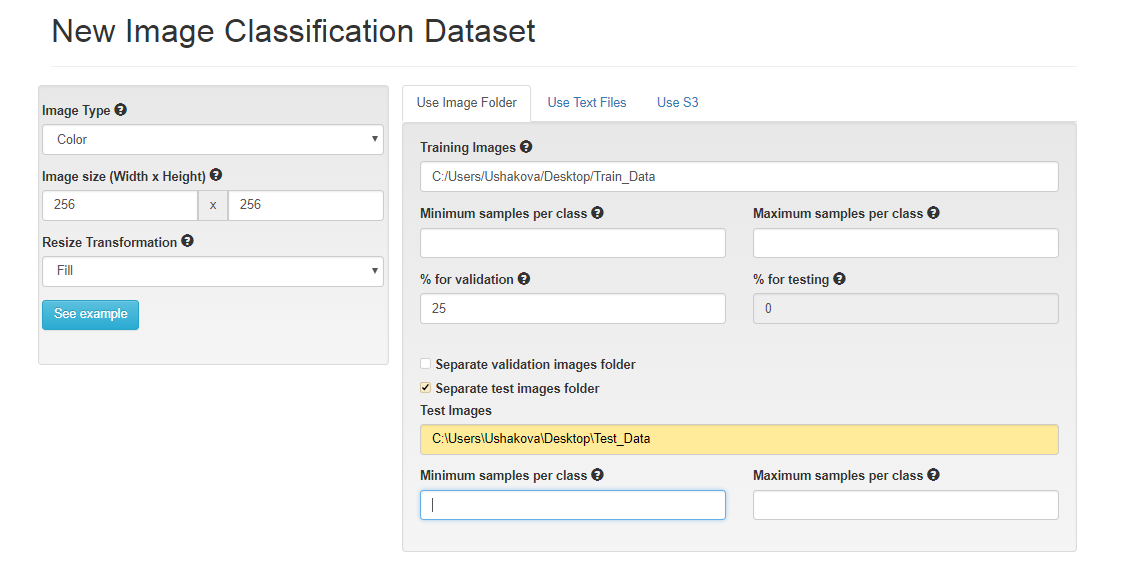
1. Для обучения нейронной сети потребуется DataSet. Для этого необходимо сформировать: трейновую, тестовую и валиационную часть.

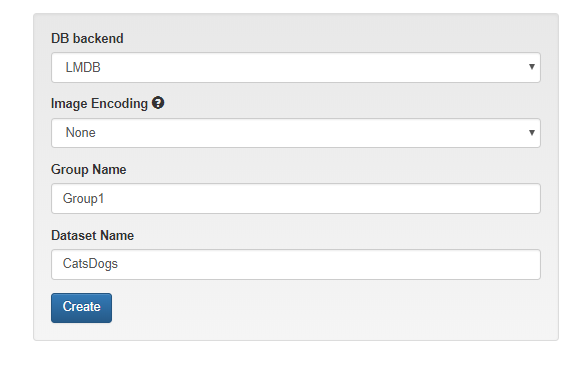
* Изображения для обучения нейронной сети нужно распределить по папкам, название папки – желаемая метка класса (label): Cats и Dogs. В каждой папке находятся изображения членов класса. Полученные папки вложить в одну папку – Train\_Data.



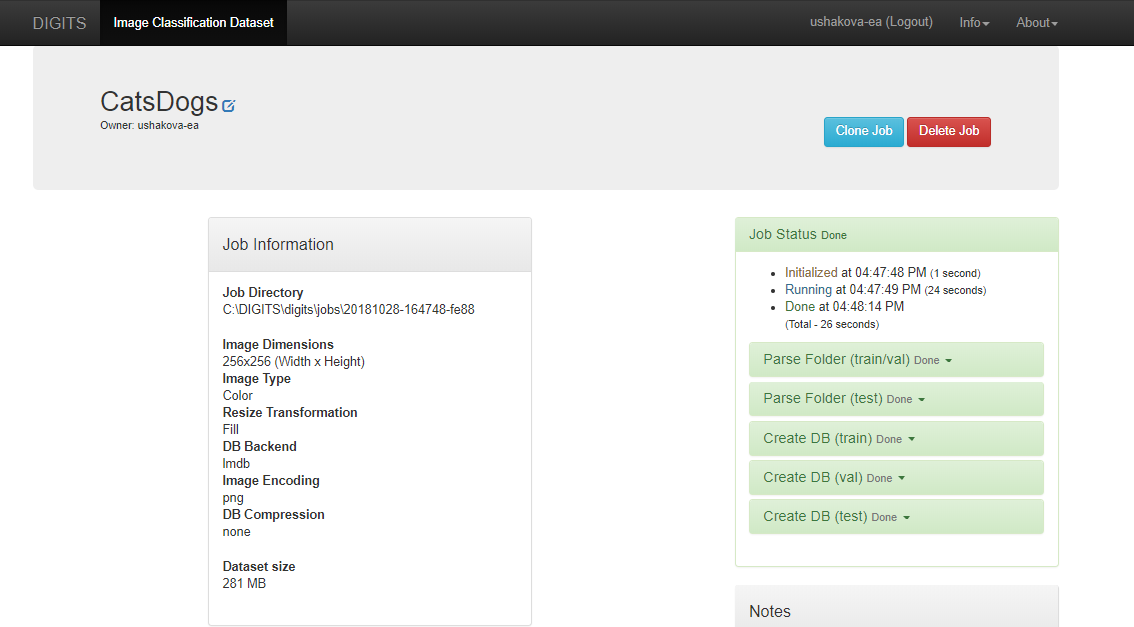
* При необходимости тестовые и валидационные изображения распределяются аналогичным образом в папки – Test\_Data, Val\_Data либо просто указывыется процент от Train\_Data

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



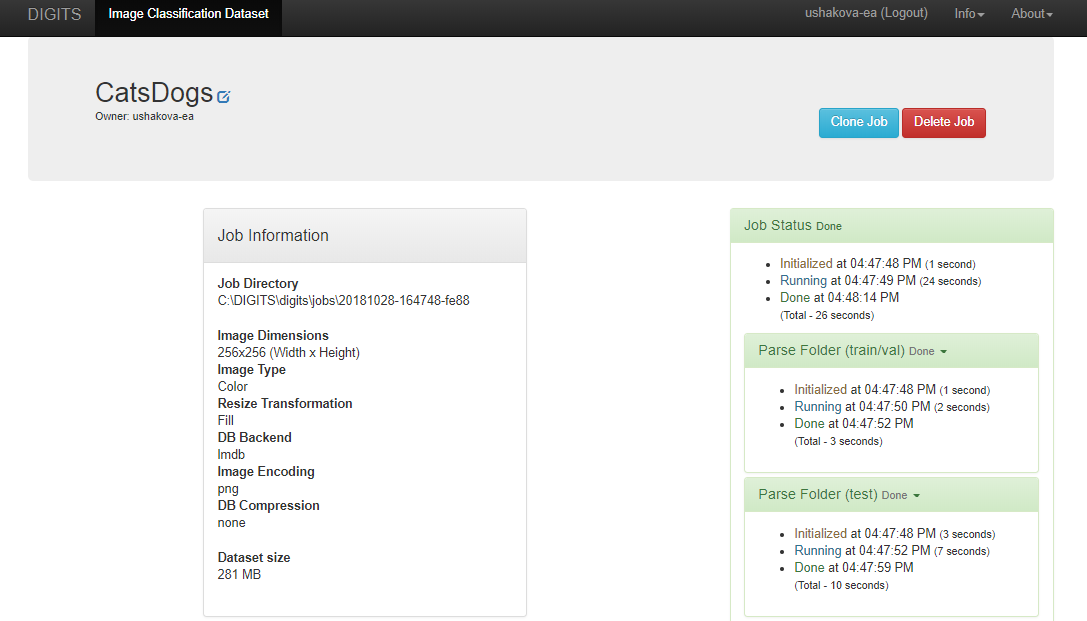


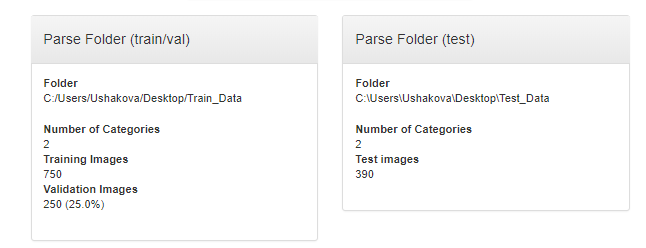
1. Нажать Create → Открывается форма с прогресс барами выполнения каждого из этапов → Об успешном завершении информирует положительный статус.

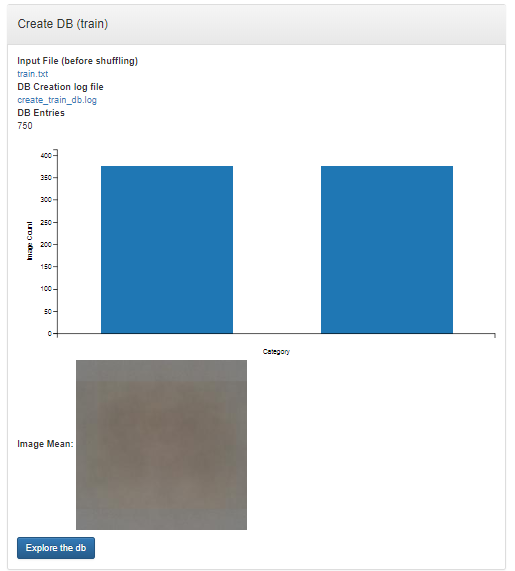


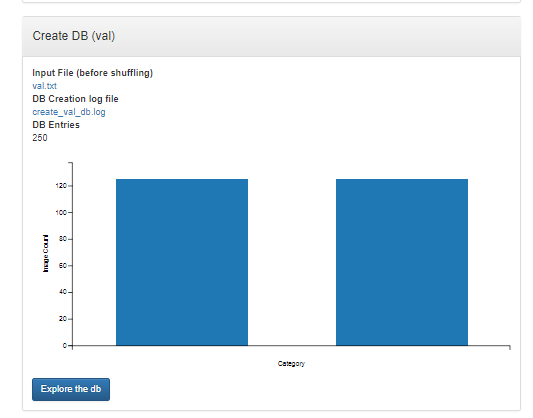
В результате можно увидеть:

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

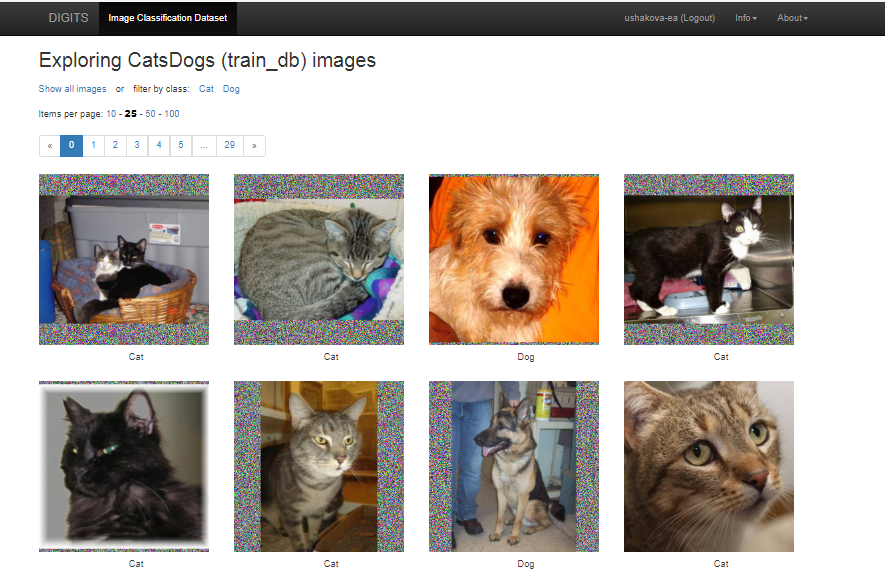




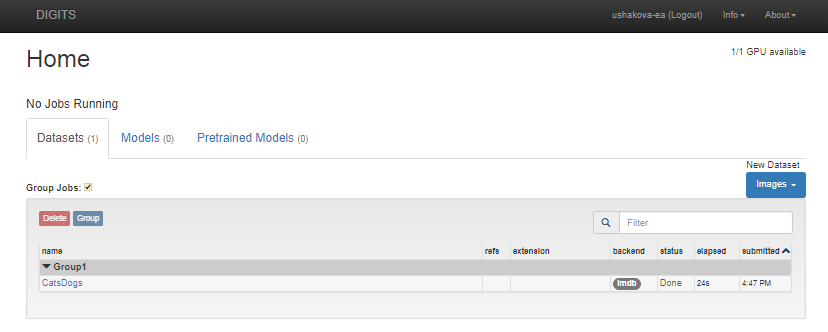






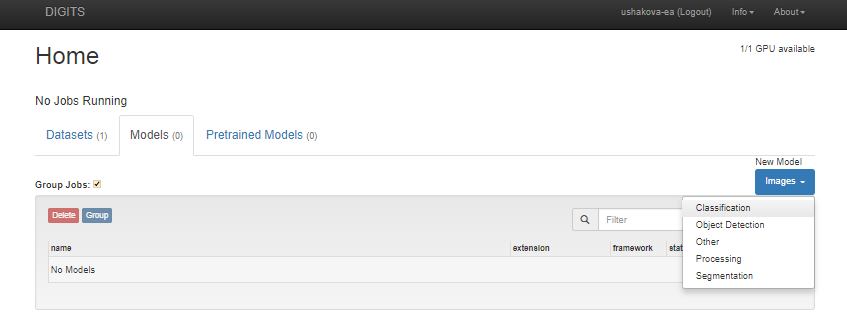


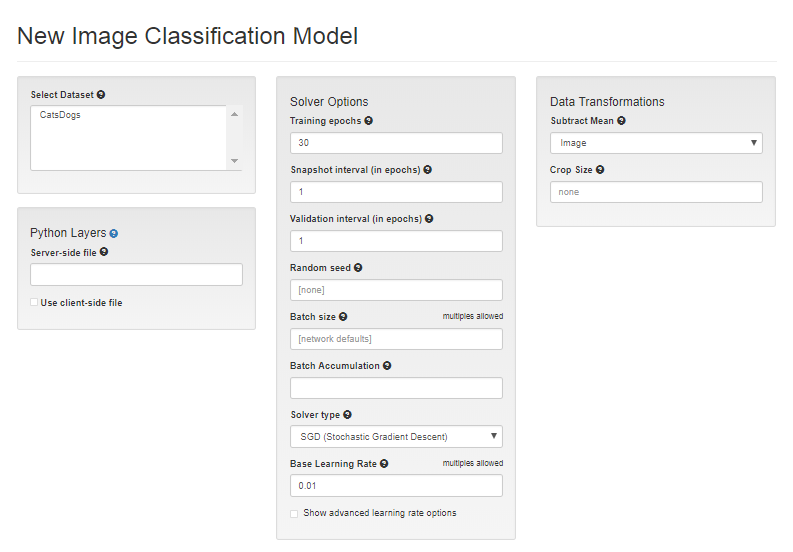
1. Сформированный DataSet будет сохранен во вкладке DataSets

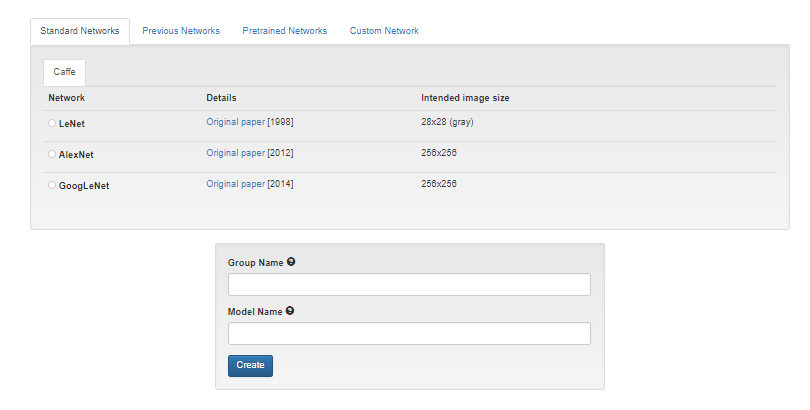


## Формирование модели

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:

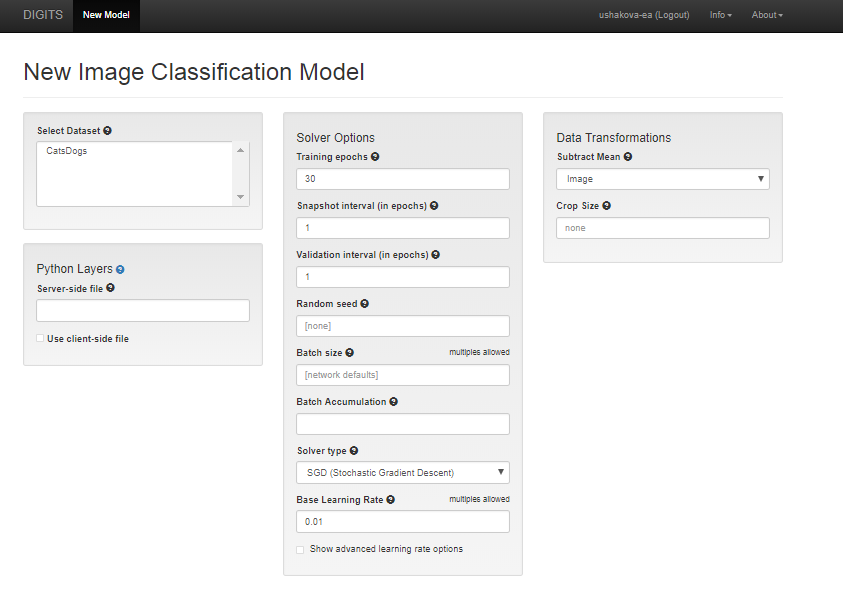
* Training epochs – количество эпох для обучения
* Snapshot interval (in epochs) – количество эпох обучения между съемками
* Validation interval (in epochs) - количество эпох обучения между прохождением через один проход валидации
* Random seed - случайное начальное значение
* Batch size – размер батча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

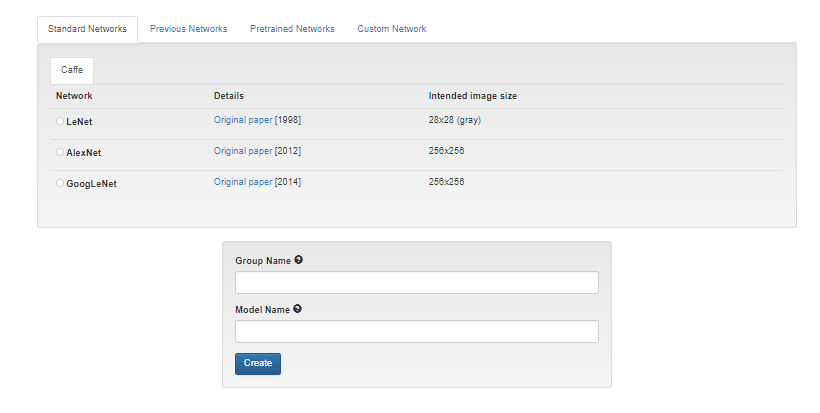
Преобазование данных:

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

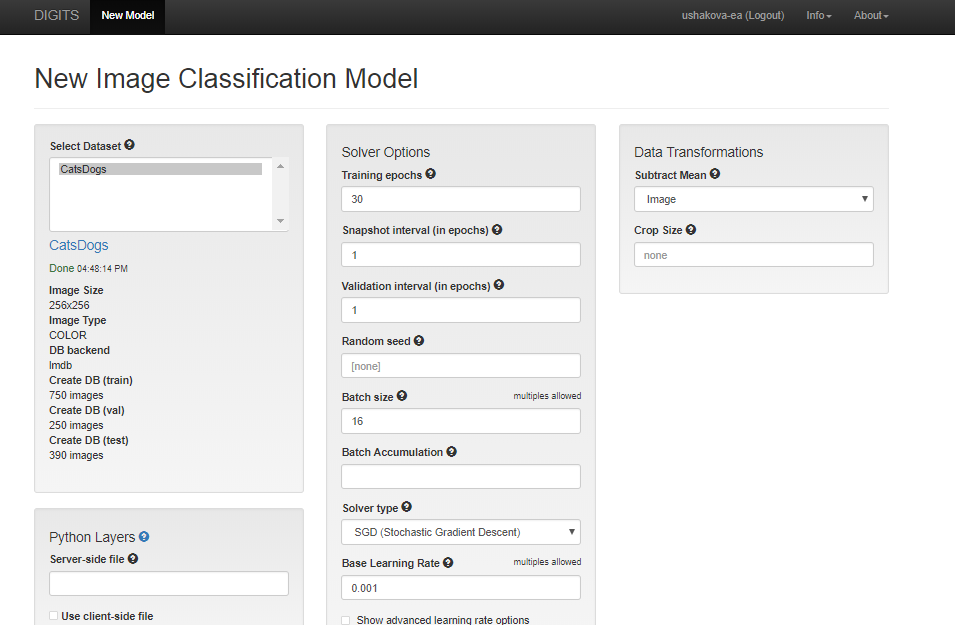


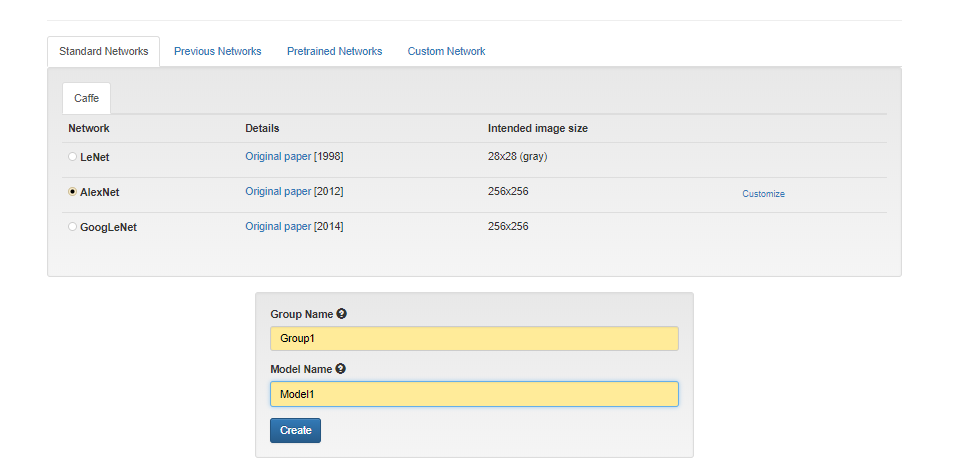
1. Для построения модели можно выбрать:

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

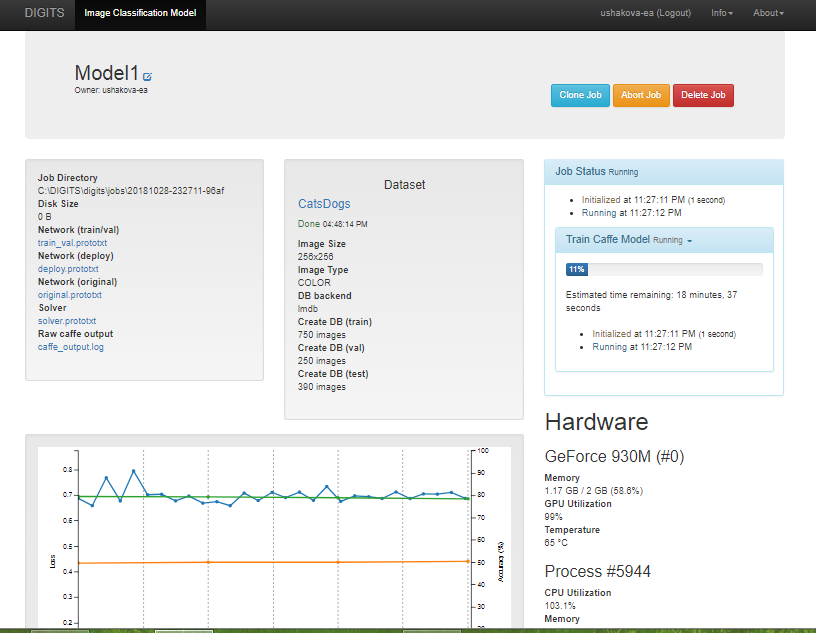


1. Далее необходимо заполнить поля в форме. Пример заполнения:



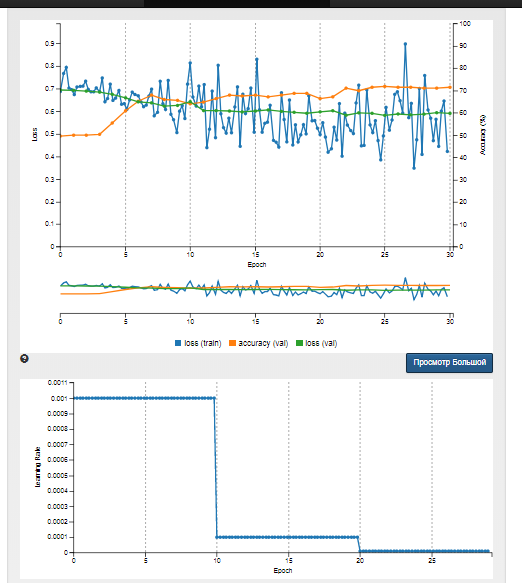


1. Нажать Create → Открывается форма с прогресс баром выполнения → Об успешном завершении информирует положительный статус - Done.

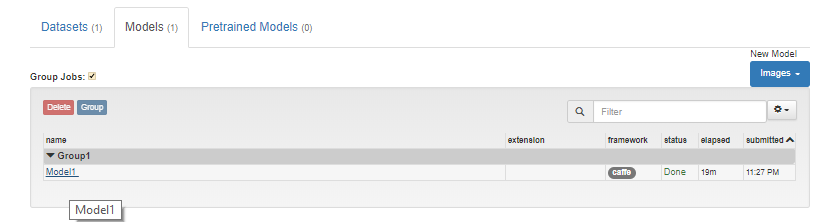


В результате можно увидеть:

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



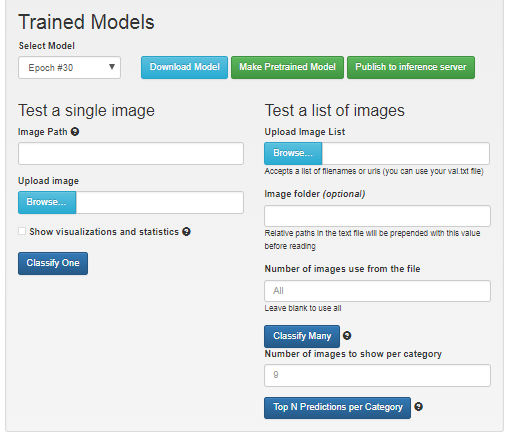
1. Модель будет сохранена во вкладке Models



## Тестирование модели

Существует возможность проверить:

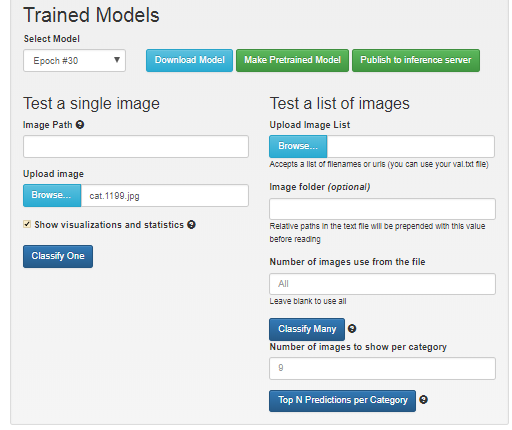
* Список изображений
* Одно изображение



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