**Machine Learning Convolutional Neural Network For Dog Breed Classification**

**Capstone Project**

Ben Jacobson  
February, 2020

**I. Definition**

**Project Overview**

Classification of objects into categories, groups, or types is essential in technology, business, and biology. It allows us to generalize knowledge for quick understanding or application. As individuals, we are constantly assessing our own environment for safety or danger, or perhaps, clues to success or failure. Though we don’t actively classify our assessments, the concept and its importance remain the same. Classification is essential to nearly all domains of life. It only makes sense we seek technology that aids us to do this.

Thirty years ago, these concepts inspired science fiction. In Terminator 2, Arnold Schwarzenegger had famously quoted “My CPU is a neural net processor… a learning computer.” The real breakthrough in computer vision happened in 2012 and arguably brought forth a wave of confidence with it. The AlexNet neural network outpaced competition by greater than 10% in the ILSVRC (ImageNet Large-Scale Visual Recognition Challenge). The challenge was to accurately classify a set of 1000 classes of images [1,2]. Since this time, neural networks have gained more prominence and acceptance beyond the “black box” it was described as. However, success breeds success, and a slew of deeper and non-sequential neural networks have continued the advancement, led by large multinational companies like Google and Microsoft [3].

Innovative applications have appeared in various domains, such as in healthcare. In 2017, deep convolutional neural networks (CNN) were used for detection of skin cancer with dermatologist-level accuracy [4]. The application wasn’t restricted to CNN application. Neural networks have since been used in healthcare diagnosis and administrative decision making [5].

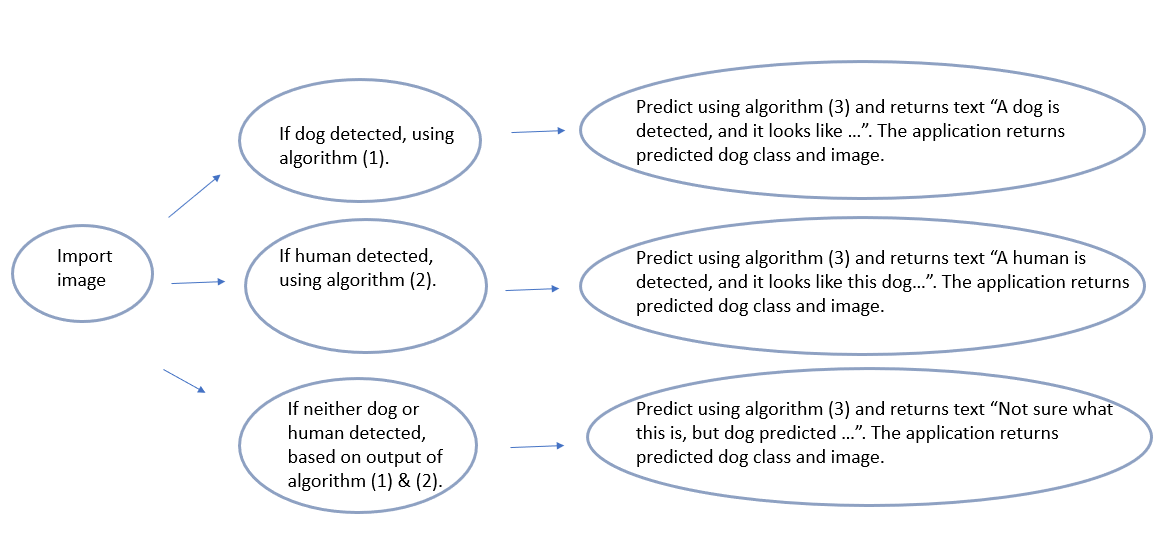
Since that healthcare problem is “solved” barring the obstacles of regulation, we do have one that is quite similar, yet more fun: dog breed classification! We seek to classify pictures of dogs into one of 133 dog breeds in our data. This problem is less abstract than classifying skin lesions, but can have practical application at a veterinary office, provide useful learnings, or a fun application (which is our intent). This paper is not one to be nominated for a Nobel prize, but can be applied to thousands of similar problems in different domains, that perhaps collectively aid mankind. Is this an easy problem? Can you tell whether this is a Komondor or an Old English Sheepdog?



**Problem Statement**

The aim is to accurately detect a dog in an image and predict the breed of that dog, based on 133 classes. This is accomplished in three parts – separate algorithms that determine whether (1) a dog is in an image (2) or a human is in an image and (3) what dog breed is predicted for the image. Our main focus will be on (3).

The final application will be deployed, and use these algorithms in combination, similar to the following flow chart.



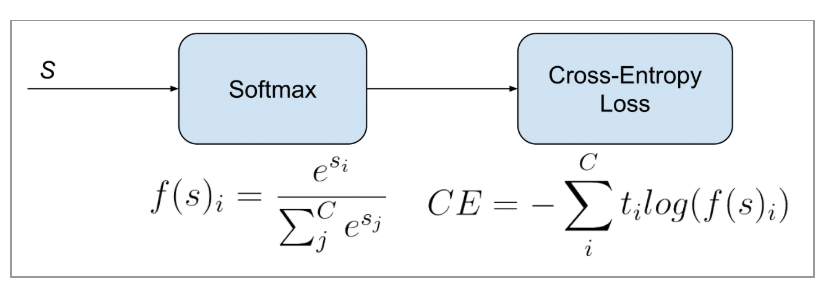
The general framework of the project follows these steps, and iteratively through multiple models as defined in the problem statement. These steps are as follows:

1. Retrieve Input Data
2. Clean and Explore
3. Prepare/Transform
4. Develop/Train Model
5. Validate/Evaluate Model
6. Deploy to Production
7. Monitor and Update Model & Date [7]

**Evaluation Metrics**

Both the benchmark and solution model(s) will be evaluated for accuracy (select correct dog breed class or not) using cross categorical entropy. Compared to MAE and other metrics, cross categorical entropy has been noted to allow better training with more complex datasets, and more generalizable results. [6]

Since this is a multi-label classification problem in which only one predicted class is allowed as an output, so we will use categorical cross-entropy for this type of problem. Categorical cross-entropy loss takes into account the groundtruth and CNN score for each class (133 dogs), with the application of the SoftMax function. The following depicts our loss function applied using pytorch built in functions. [11]



Based on this loss, we calculate our training loss with each iteration, and the running train and validation loss are displayed. The function **test** is defined in the benchmark (scratch) model and applied in the models for that are created in the transfer learning stage.

Add definition of the loss function, loss\_train, and the accuracy

predicting the likelihood of an example belonging to each class

<https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/>

The final calculation for accuracy is fairly simple: if the prediction is correct, then 1, if not 0. This is summed and divided by the total number of predictions, giving accuracy.

**II. Analysis**

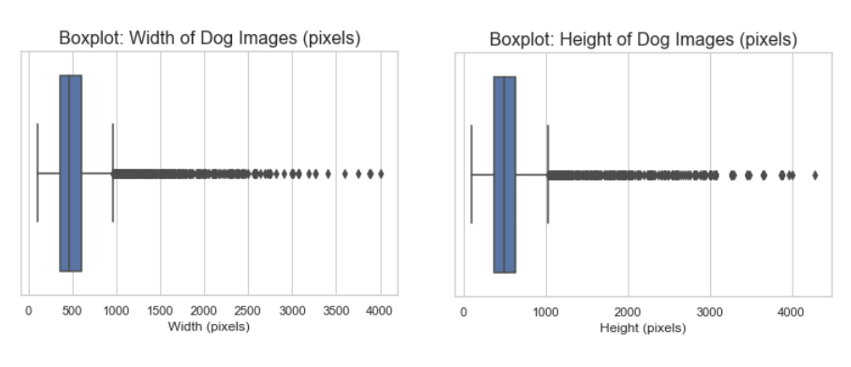
*(approx. 2-4 pages)*

**Data Exploration: Dataset and Inputs**

The following lists the data and inputs for model creation. Both datasets are comprised of images of various size and perspective, but all color (RGB, 3 channel). The datasets are used in the project in order to train, validate, and test predictive models.

* Dog Breed Image Dataset
  + This is a dataset of 133 classes/breeds of dogs, a total of 8,351 images. Each of the 133 dog breeds will be divided into a train, validation and test data set.
* Human Face Dataset
  + This is a dataset of 13233 images of humans.

The following shows the range of input images in our data that we will use to train, validate, and test the model. All images are three channels, which indicates RGB. The "width" dimension has an average of 529 pixels and "height" dimension of 567 pixels. However, statistically speaking, there is a lot of variation in the data, with standard deviation of 333 and 389 pixels, respectively. So, what does this mean to us? Some images have more information than others. Preprocessing steps may affect some images differently as well.



**Exploratory Visualization**

In examining just a few images, there are some important elements to note. From left to right there are many differences, (1) has text in the image, (2) the image is at an angle and just the face, (3) has two dogs, and (4) has plants in the foreground, with many puppies, I think. So, in examining just a few images, we find there is considerable variation in the “framing” of the image and what is in it, not just the dog breed.



In comparison, the human faces dataset is composed of RGB images of the same size 250x250. On initial inspection, they appear a bit cleaner. However, we do see the same rotation, and multiple people in the same image. This is not the focus of the project.

**Algorithms and Techniques**

The anticipated solution will use a machine learning framework based on a convolutional neural network (or multiple) to take an input image, detect if is a dog, and then classify that image based on the available classes. In particular:

* Convolutional neural networks, using transfer learning with well-known architectures, freezing the early layers, and fine tuning the last few layers to classify to dog breeds.
* The recommended solution is intended to be operational, and fast, in order to be used in an application. In particular a MobileNet, VGG, or SqueezeNet model will be trained, tested and selected.

The aim is to measure performance via appropriate metrics, in a replicable and quantifiable manner. As a result, datasets are provided in the GitHub repository, as well as code to replicate the model creation as selected in the final application.

* *Are the algorithms you will use, including any default variables/parameters in the project clearly defined?*
* *Are the techniques to be used thoroughly discussed and justified?*
* *Is it made clear how the input data or datasets will be handled by the algorithms and techniques chosen?*

**Benchmark**

We need a model, preferably parsimonious or historical, to compare the proposed solution with. As requirements for the project, a CNN from scratch is to be created. Random chance indicates that 0.75% of the time the correct class will be selected (based on 133 classes). However, our expectation is to have a benchmark model built and trained from scratch, that achieves greater than 10% accuracy. This will be our benchmark model.

The benchmark model will employ random initialization of weights, approximately five convolutional units, max pooling, batch normalization, dropout, and a final dense layer for classification. Multiple epochs and a smaller batch size will be used. The exact architecture will be built to exceed 10% accuracy based on cross categorical entropy. This is mandated in the project guidelines. (The final model would require greater than 60% accuracy, and be based on transfer learning.)

**III. Methodology**

*(approx. 3-5 pages)*

**Data Preprocessing**

* The dog image set was randomly divided, without replacement, into a train, validation and test set.
* The batch size used is 20 images.
* Each image goes through the following transformations before being batched, and used in the CNN models.
  + The image is resized to 256x256x3
  + The image is then cropped to 224x224x3, in alignment with the transfer learning models used
  + The dataset is augmented by using a Random Rotation of 20 degrees. This is likely important based on the exploratory data analysis, in which most image have the dog at a variety of angles.
  + The dataset is augmented by using a Random Horizontal Flip with a 50% chance of occurring. This allows for mirror images, which makes sense in this context.
  + The image is converted to a tensor.
  + The image is normalized.

**Implementation**

In this section, the process for which metrics, algorithms, and techniques that you implemented for the given data will need to be clearly documented. It should be abundantly clear how the implementation was carried out, and discussion should be made regarding any complications that occurred during this process. Questions to ask yourself when writing this section:

* *Is it made clear how the algorithms and techniques were implemented with the given datasets or input data?*
* *Were there any complications with the original metrics or techniques that required changing prior to acquiring a solution?*
* *Was there any part of the coding process (e.g., writing complicated functions) that should be documented?*

**Refinement**

Refinement was necessary to have better predictive accuracy, but also to deploy the model on Heroku. The initial VGG-26 model was too large to deploy, surpassing the “slug size of 500MB”.

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section:

* *Has an initial solution been found and clearly reported?*
* *Is the process of improvement clearly documented, such as what techniques were used?*
* *Are intermediate and final solutions clearly reported as the process is improved?*

**IV. Results**

*(approx. 2-3 pages)*

**Model Evaluation and Validation**

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model’s solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section:

* *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate?*
* *Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data?*
* *Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results?*
* *Can results found from the model be trusted?*

**Justification**

In this section, your model’s final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section:

* *Are the final results found stronger than the benchmark result reported earlier?*
* *Have you thoroughly analyzed and discussed the final solution?*
* *Is the final solution significant enough to have solved the problem?*

**V. Conclusion**

*(approx. 1-2 pages)*

**Free-Form Visualization**

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section:

* *Have you visualized a relevant or important quality about the problem, dataset, input data, or results?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Reflection**

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section:

* *Have you thoroughly summarized the entire process you used for this project?*
* *Were there any interesting aspects of the project?*
* *Were there any difficult aspects of the project?*
* *Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

**Improvement**

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section:

* *Are there further improvements that could be made on the algorithms or techniques you used in this project?*
* *Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how?*
* *If you used your final solution as the new benchmark, do you think an even better solution exists?*

**Before submitting, ask yourself. . .**

* Does the project report you’ve written follow a well-organized structure similar to that of the project template?
* Is each section (particularly **Analysis** and **Methodology**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
* Would the intended audience of your project be able to understand your analysis, methods, and results?
* Have you properly proof-read your project report to assure there are minimal grammatical and spelling mistakes?
* Are all the resources used for this project correctly cited and referenced?
* Is the code that implements your solution easily readable and properly commented?
* Does the code execute without error and produce results similar to those reported?

Citations

[1] https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neuralnetworks.pdf

[2] http://www.image-net.org/challenges/LSVRC/

[3] https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

[4] https://www.nature.com/articles/nature21056

[5] https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0212356

[6] https://papers.nips.cc/paper/8094-generalized-cross-entropy-loss-for-training-deep-neuralnetworks-with-noisy-labels.pdf

[7] https://Udacity.com

[8] https://cloud.google.com/ml-engine/docs/ml-solutions-overview

[9] https://gombru.github.io/2018/05/23/cross\_entropy\_loss/

[10] <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/CNN20Whitepaper.pdf>

[11] <https://gombru.github.io/2018/05/23/cross_entropy_loss/>

