# Machine Learning Capstone Project

REVIEW | HISTORY

## Requires Changes

4 SPECIFICATIONS REQUIRE CHANGES

This is a very cool analysis and a great read to a very real world and practical problem. You have demonstrated a full understanding of the entire machine learning pipeline and your report definitely gets the readers attention with the results you have achieved. You just have to expand in some of these sections, but will greatly improve your report. Please check out some of these other ideas presented here and we look forward in seeing your next submission!!

Can also check these out

- (https://course.fast.ai/)
- (https://www.youtube.com/playlist?list=PLkt2uSq6rBVctENoVBg1TpCC7OQi31AlC)
- (https://www.youtube.com/playlist?list=PL9Hr9sNUjfsmEu1ZniY0XpHSzl5uihcXZ)

### Definition

**Student provides a high-level overview of the project in layman's terms. Background information such as the problem domain, the project origin, and related data sets or input data is given.**

Very solid opening section here, as you have done a great job describing the problem and that machine learning can solve this. Image classification is definitely growing in terms to different industries.

And you have provided good research to back your claims. It is always important to provide similiar research on such a topic.

**The problem which needs to be solved is clearly defined. A strategy for solving the problem, including discussion of the expected solution, has been made.**

> "This paper outlines a convolutional neural network which uses patient cell image data to classify whether a cell is infected with malaria or not. It outlines one customised architecture and three architectures which use pre-trained networks including ResNet-50, NasNetMobile and InceptionV3. The results show that the customised architecture obtains significantly better results than those obtained by the pre-trained networks. Compared to previous literature, it obtains an improved sensitivity score."

Although, not in a corresponding **Problem Statement** section, A high level problem statement is clearly defined. I would recommend mentioning that this is actually a binary classification problem as well.

And very nice job mentioning your machine learning pipeline and models used, as this gives the reader some ideas in what is to come in your report and how you plan on solving this important task.

**Metrics used to measure performance of a model or result are clearly defined. Metrics are justified based on the characteristics of the problem.**

Good analysis and justification for your metrics. Sensitivity and Specificity are typically used in the medial field.

Tying your metric choice into your particular problem and problem domain is actually the single most important thing to do in any machine learning project. If you optimize a model based on the incorrect metric, your model might not be suitable for the business goals.

## Analysis

**If a dataset is present, features and calculated statistics relevant to the problem have been reported and discussed, along with a sampling of the data. In lieu of a dataset, a thorough description of the input space or input data has been made. Abnormalities or characteristics about the data or input that need to be addressed have been identified.**

Glad that you discuss the distribution of the target class and total number of images. Therefore lastly for this section, make sure you also show

- some sample images from your dataset
- document the image sizes. All the same size? Varying sizes? etc... Some descriptive stats would also be nice to see here.

As this allows the reader to get an understanding of the structure of the data you are working with.

**A visualization has been provided that summarizes or extracts a relevant characteristic or feature about the dataset or input data with thorough discussion. Visual cues are clearly defined.**

Make sure you include an **Exploratory Visualization** section. In this section, you will need to provide some form of visualization that summarizes or extracts a relevant characteristic or feature about the data. The visualization should adequately support the data being used. Discuss why this visualization was chosen and how it is relevant.

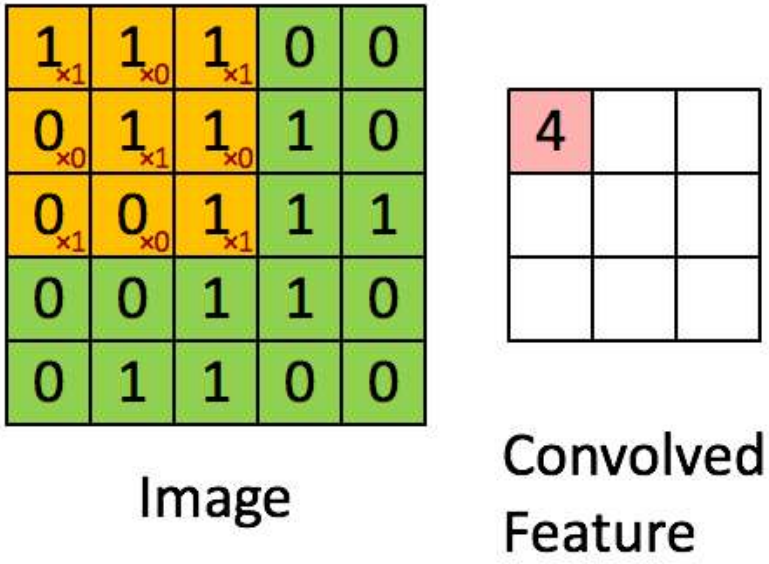
Idea

- Maybe plot some Intensity Histograms. In an image processing context, the histogram of an image normally refers to a histogram of the pixel intensity values. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image.

**Algorithms and techniques used in the project are thoroughly discussed and properly justified based on the characteristics of the problem.**

You have given very solid ideas in how each subset of your CNN architecture will work, as it is clear that you have a good understanding of all these steps.
(http://cs231n.github.io/convolutional-networks/)

Maybe even some visuals could help explain each step as well.




Student clearly defines a benchmark result or threshold for comparing performances of solutions obtained.

> "Rajaraman et al (2018) utilised a customised CNN with three convolutional layers and two fully connected layers. They also carried out transfer learning using pre-trained models such as AlexNet, VGG-16, ResNet-50, Xception, DenseNet-121. In this paper, I utilise their results as a benchmark for the performance of the CNN."

External benchmarks are great for such a problem. Glad that you mention their model architecture.

Benchmarking is the process of comparing your result to existing method or running a very simple machine learning model, just to confirm that your problem is actually 'solvable'.

## Methodology

All preprocessing steps have been clearly documented. Abnormalities or characteristics about the data or input that needed to be addressed have been corrected. If no data preprocessing is necessary, it has been clearly justified.

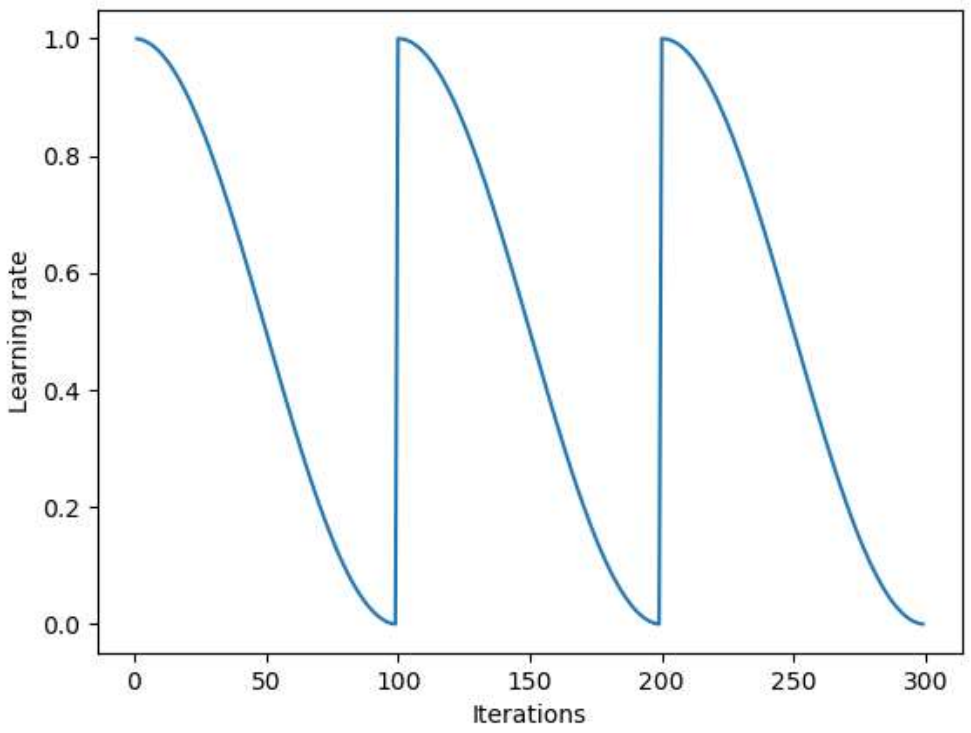
Could also use some data augmentation. As this help the model actually 'learn' features and not just memorize images. And also helps generate more training data.

If you would like to learn how to do this is straight tensorflow, check out this notebook (https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/06_CIFAR-10.ipynb)

**The process for which metrics, algorithms, and techniques were implemented with the given datasets or input data has been thoroughly documented. Complications that occurred during the coding process are discussed.**

Very solid step by step process here, as it is quite clear in how you approached this problem. Your results would definitely be replicable.

Another idea would be to check out using Cyclical Learning Rates for Training Neural Networks. This is where we simply keep increasing the learning rate from a very small value, until the loss stops decreasing and then bump it up once more. We can plot the learning rate across batches to see what this looks like.



**The process of improving upon the algorithms and techniques used is clearly documented. Both the initial and final solutions are reported, along with intermediate solutions, if necessary.**

I will take your transfer learning approach as model refinement here. Transfer learning is a good idea. Most of the time you won't want to train a whole convolutional network yourself. Modern ConvNets training on huge datasets like ImageNet take weeks on multiple GPUs. Instead, most people use a pretrained network either as a fixed feature extractor, or as an initial network to fine tune.

Could also look into something like gridSearch to tune hyper-parameters for your model, such as the learning rate, optimizers, batch_size, etc... As we can actually use Sklearn with Keras! Check out this example

```
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV
from keras.models import Sequential
from keras.layers import Dense

def build_classifier(optimizer):
    classifier = Sequential()
    classifier.add(Dense(units = 6, kernel_initializer = 'uniform', activation = 'relu', input_dim = 11))
    classifier.add(Dense(units = 6, kernel_initializer = 'uniform', activation = 'relu'))
    classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'sigmoid'))
    classifier.compile(optimizer = optimizer, loss = 'binary_crossentropy', metrics = ['accuracy'])
    return classifier
classifier = KerasClassifier(build_fn = build_classifier)
parameters = {'batch_size': [25, 32],
              'epochs': [100, 500],
              'optimizer': ['adam', 'rmsprop']}
grid_search = GridSearchCV(estimator = classifier,
                           param_grid = parameters,
                           scoring = 'accuracy',
                           cv = 10)
grid_search = grid_search.fit(X_train, y_train)
best_parameters = grid_search.best_params_
best_accuracy = grid_search.best_score_
```

## Results

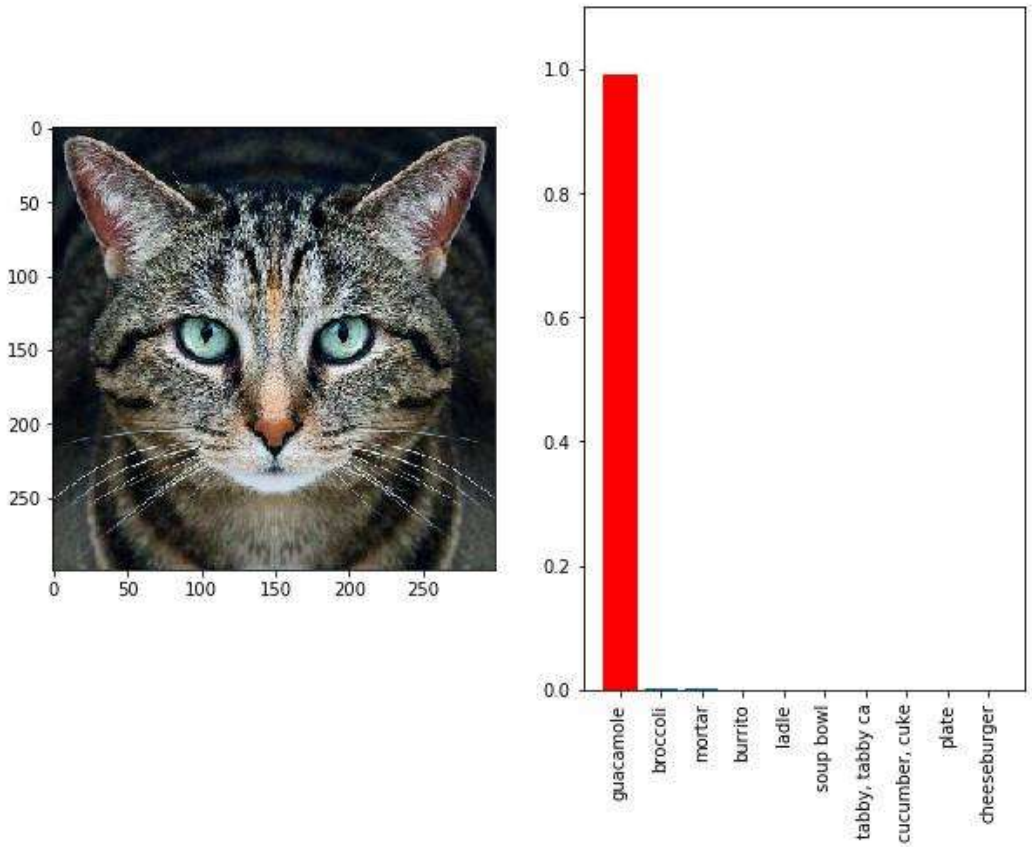
**The final model's qualities — such as parameters — are evaluated in detail. Some type of analysis is used to validate the robustness of the model's solution.**

You have done a good job describing your final models, however also for this section, you need to also provide some analysis to validate the robustness of the model's solution.

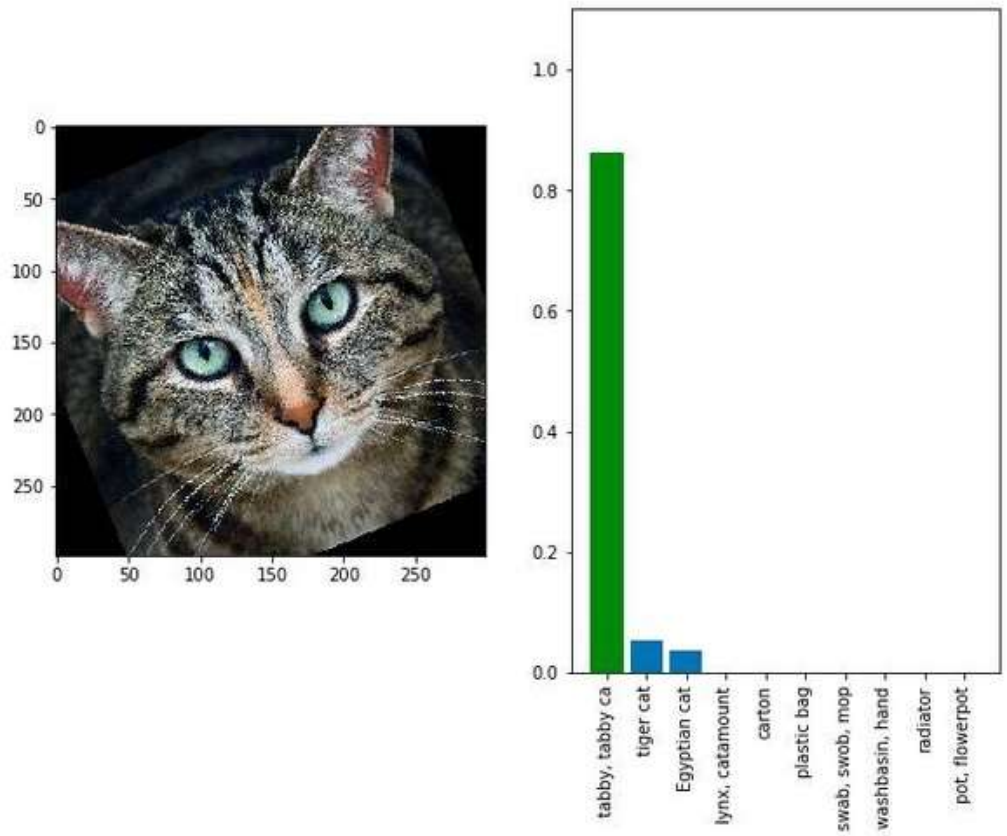
Some ideas to discuss / do to validate your model

- Could look into using KFold CV and show the different folds scores
- Maybe describe the training / validation / testing results. Any overfitting happening here? Do we need even more regularization? etc....
- add some random gaussian noise the images and see how the model performs
- run some images through your model and analyze the predictions

**Why we need to validate our models?**




Here's a picture of a cat right? Google's Inception model thinks it's a guacamole. As much as the image looks like a cat, the image is digitally altered which confused the model.




Slightly rotating the image led the model to correctly classify the image as a cat (and as an animal)

The above image is what's called as an adversarial image, trying to fool your model into thinking the image is something you want it to be instead of what the image actually is.

The real danger is in the application, especially in healthcare and defense. For example, how would you convince that your model for predicting cancer actually works? How do you know your model is not susceptible to noise? How do you know that your model has actually learnt what it is supposed to be learning? This is why it is always important to validate the robustness of the model's solution.

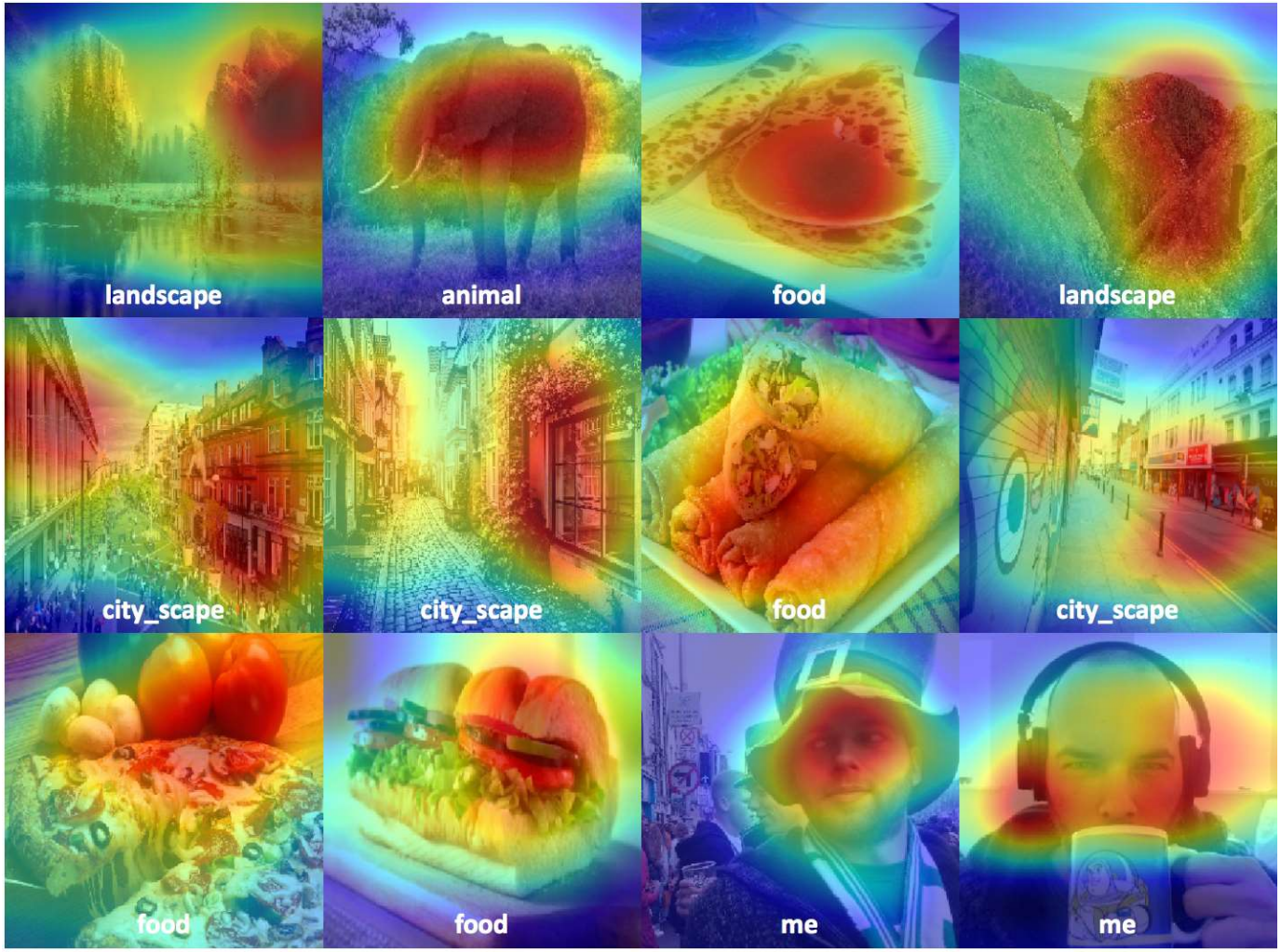
**The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.**

## Conclusion

**A visualization has been provided that emphasizes an important quality about the project with thorough discussion. Visual cues are clearly defined.**

Make sure you include a **Free-Form Visualization** section. 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.

This could be a confusion matrix or one other idea would be to visual the convolution layers and maybe see where the 'activations' are. For example



(https://hackernoon.com/visualizing-parts-of-convolutional-neural-networks-using-keras-and-cats-5cc01b214e59)
(http://cs231n.github.io/understanding-cnn/)

**Student adequately summarizes the end-to-end problem solution and discusses one or two particular aspects of the project they found interesting or difficult.**

Nice work discussing your final end-to-end problem solution as this reads quite well. I can definitely tell that you have spent a long time on this project as it really shows.

**Discussion is made as to how one aspect of the implementation could be improved. Potential solutions resulting from these improvements are considered and compared/contrasted to the current solution.**

> "Training this on a CPU limits the number of epochs it could train on due to the longer amount of time required for training. In general, I faced hardware issues with this projects even on pre-processing tasks. I attempted to use three different laptops and Google Colab."

Could also look into using floydhub or paperspace or vectordash(which have 1080Ti) to train your models in the cloud.

## Quality

**Project report follows a well-organized structure and would be readily understood by its intended audience. Each section is written in a clear, concise and specific manner. Few grammatical and spelling mistakes are present. All resources used to complete the project are cited and referenced.**

Your writing is very clean and it is very easy to understand what you are saying. I personally thank you as this report is very easy to read :)

**Code is formatted neatly with comments that effectively explain complex implementations. Output produces similar results and solutions as to those discussed in the project.**

Code is formatted neatly. You might also check out these links for some best practices when commenting your code

- this post regarding Docstrings vs Comments.

- Google Style Python Docstrings
- This Best of the Best Practices" (BOBP) guide to developing in Python

RESUBMIT PROJECT

DOWNLOAD PROJECT


## Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)