UDACITY

< Back to Machine Learning Engineer Nanodegree

Machine Learning Capstone Project

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Just want to say that this is one of the best put together reports I have seen. You really took the time to go above and beyond in all the requirements. Hopefully you have learned a bunch throughout this capstone project(as I can image that you have by reading your report) and you can take some of these techniques further.

If this is your final report, I would like to be the first one to congratulate you on completing this nano-degree! Wish you the best of luck in your future!

Can also check these out

- (http://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.

> "A technology that can accurately predict landmark labels directly from image pixels can broadly benefit applications in various areas such as photo management, maps, aviation, and satellite images."

A high level problem statement is clearly defined. And glad that you mention that this is actually a multi-class classification problem for your image recognition problem.

And very nice job mentioning your machine learning models in the previous section, 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.

> "Although there is imbalance of class, it is not severe enough to cause the model to favor only the class with highest number of classes from training."

This is not really the case here. You have a very unbalanced dataset. Only a few classes over-power the others.

> "Moreover, there is no reason to treat false positives and false negatives differently. Hence use of accuracy, which treats all misclassifications the same, may be justified for the given purpose of this project."

This is good justification for your choice in your accuracy metric. Accuracy is definitely the most 'intuitive' metric to use.

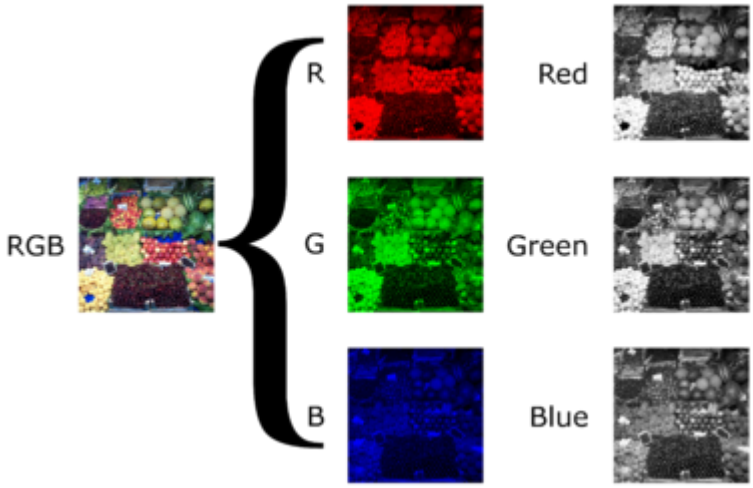
Analysis

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.

Good analysis of your dataset here. Definitely give the reader a solid grasp of what you are working with. Glad that you mention the dataset size (reducing this is a fine idea) and the distribution of the target class. Some analysis of the image size would also be nice.

Another interesting idea to look into would be to display the numerical representation of the images or even break one image into three different color channels and show the difference. Such as



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.

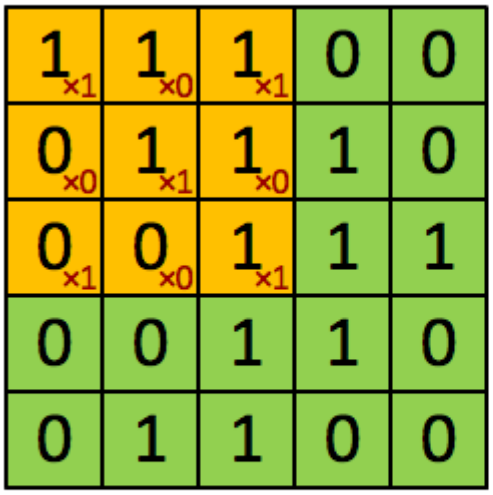
The distribution of target variable is always a good idea.

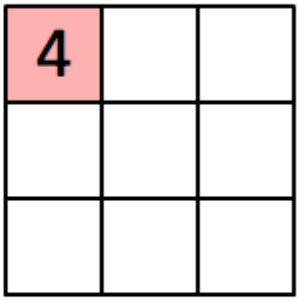
Maybe also plot some Intensity Histograms. In an image processing context, the histogram of an image normally refers to a histogram of the pixel intensity values. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image.

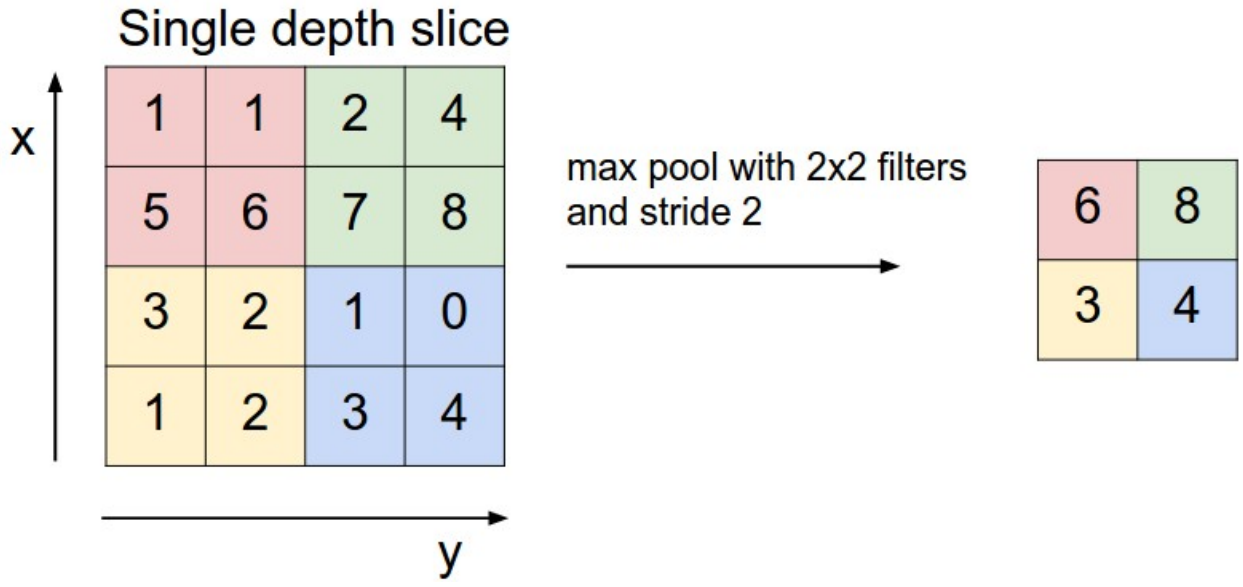
Algorithms and techniques used in the project are thoroughly discussed and properly justified based on the characteristics of the problem.

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. Nice job giving an overview of the entire pipeline.

Maybe also some visuals of each layer







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

Your Monte Carlo simulation is a fun benchmark model idea.

Could also run a simple ML model, such as a logistic regression to get a baseline score for your dataset with a linear model.

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.

All the required steps for building a CNN model.

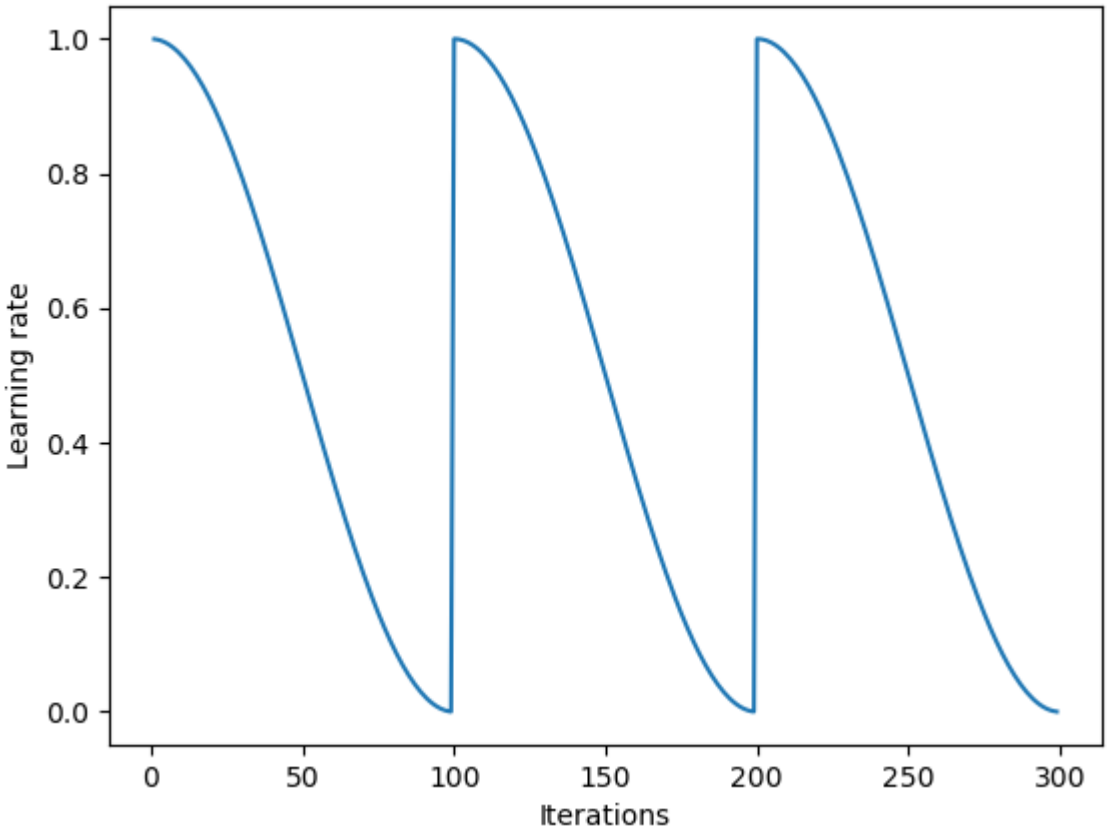
And very nice idea to use data augmentation. As this help the model actually 'learn' features and not just memorize images. The after image could be a nice touch!

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.

Nice work with your hyper-parameter tuning ideas, as this is a great way to improve your model. And you have made it very clear in the parameter you tried and the results with your chart. Very nice!

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 = 're
lu', input_dim = 11))
    classifier.add(Dense(units = 6, kernel_initializer = 'uniform', activation = 're
lu'))
    classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'si
gmoid'))
    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.

Nice discussion of your final model here and the training and validation accuracy / loss plots and overfitting analysis are great to validate the robustness of the model's solution.

Another idea to validate the robustness of the model's solution, would be to try and add some random gaussian noise the images and see how the model performs? Did it actually learn feature or is it just memorizing images?

> "Perhaps, balancing the training dataset using weights may be considered while using the same evaluation metric from the competition."

Typically we just replicate the under-represented class to make the dataset more balanced. Could check out using `class_weight` in keras.

(https://datascience.stackexchange.com/questions/13490/how-to-set-class-weights-for-imbalanced-classes-in-keras)

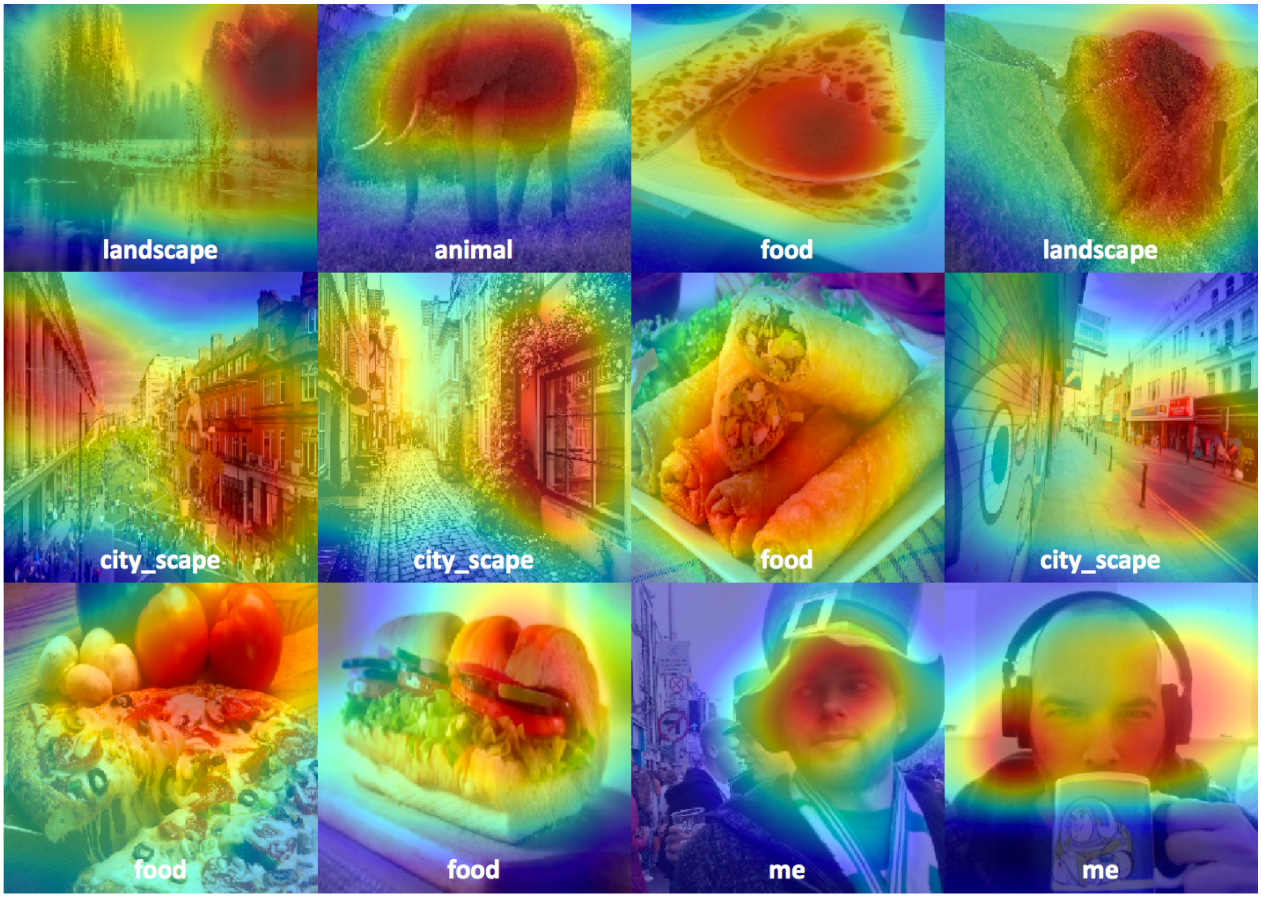
The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.

Conclusion

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

I think your visual could be improved here. Look into showing some image predictions for your model. This is typically the most important part in determining 'where' your model went wrong and how to improve it.

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 bet working with data this big was a good learning experience. 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.

> "One method is to take advantage of transfer learning. Pre- trained models that can identify humans, trees, animals (like pets), and vehicles are available."

For sure! 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. Might want to look into fine-tuning the resnet50 model.

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.

Few more comments would be nice to see. But easy to follow.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

Student FAQ