# Data Augmentation

## Introduction

The purpose of this section was to test the impact of data augmentation on the CNN learning speed and eventual accuracy. Due to computational constraints, we only allowed 15 epochs. Note that the neural network used in this section was taken from [here](https://www.kaggle.com/c/cifar-10/discussion/40237). This fits in with the general literature in that most papers are concerned with creating more intuitive and accurate frameworks for the CNN and not necessarily finding the ideal data augmentation to be fed into said networks, which is what we are looking at in this section.

## Details and Methodology

In terms of data augmentation we allowed randomization via the function ImageDataGenerator() in Keras with the following augmentation possibilities:

* Rotation\_angle up to 50 degrees
* Width shift range up to 50% of the width range
* Height shift range up to 50% of the height range
* Horizontal flipping

Specifically this was done by allowing i to go from 0 to 10 and using this code:

datagen = ImageDataGenerator(

featurewise\_center=False,

samplewise\_center=False,

featurewise\_std\_normalization=False,

samplewise\_std\_normalization=False,

zca\_whitening=False,

rotation\_range=50/i if i != 0 else 0,

width\_shift\_range=0.5/i if i != 0 else 0,

height\_shift\_range=0.5/i if i != 0 else 0,

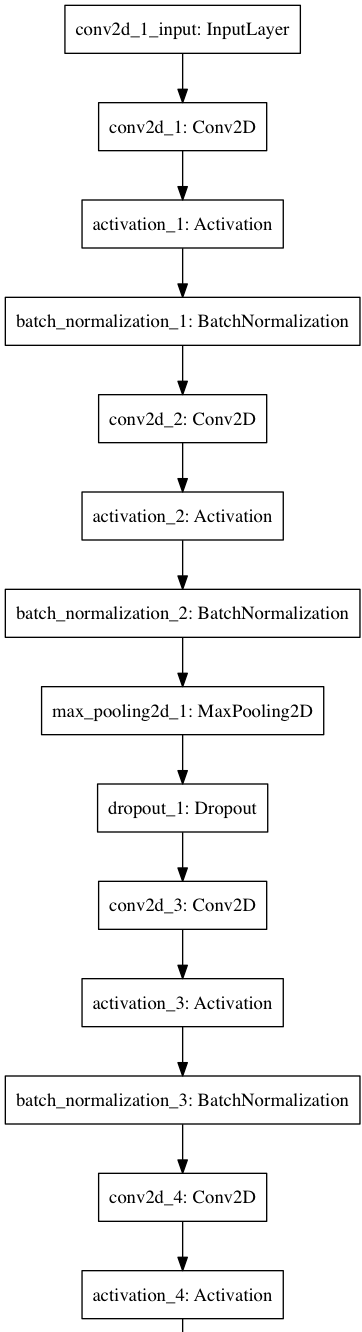
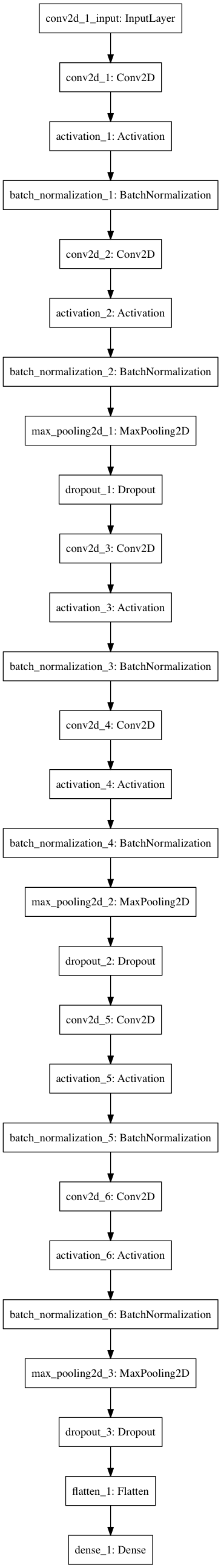
horizontal\_flip=True if i != 0 else False,

vertical\_flip=False

)

datagen.fit(x\_train)

And the model summary is below:

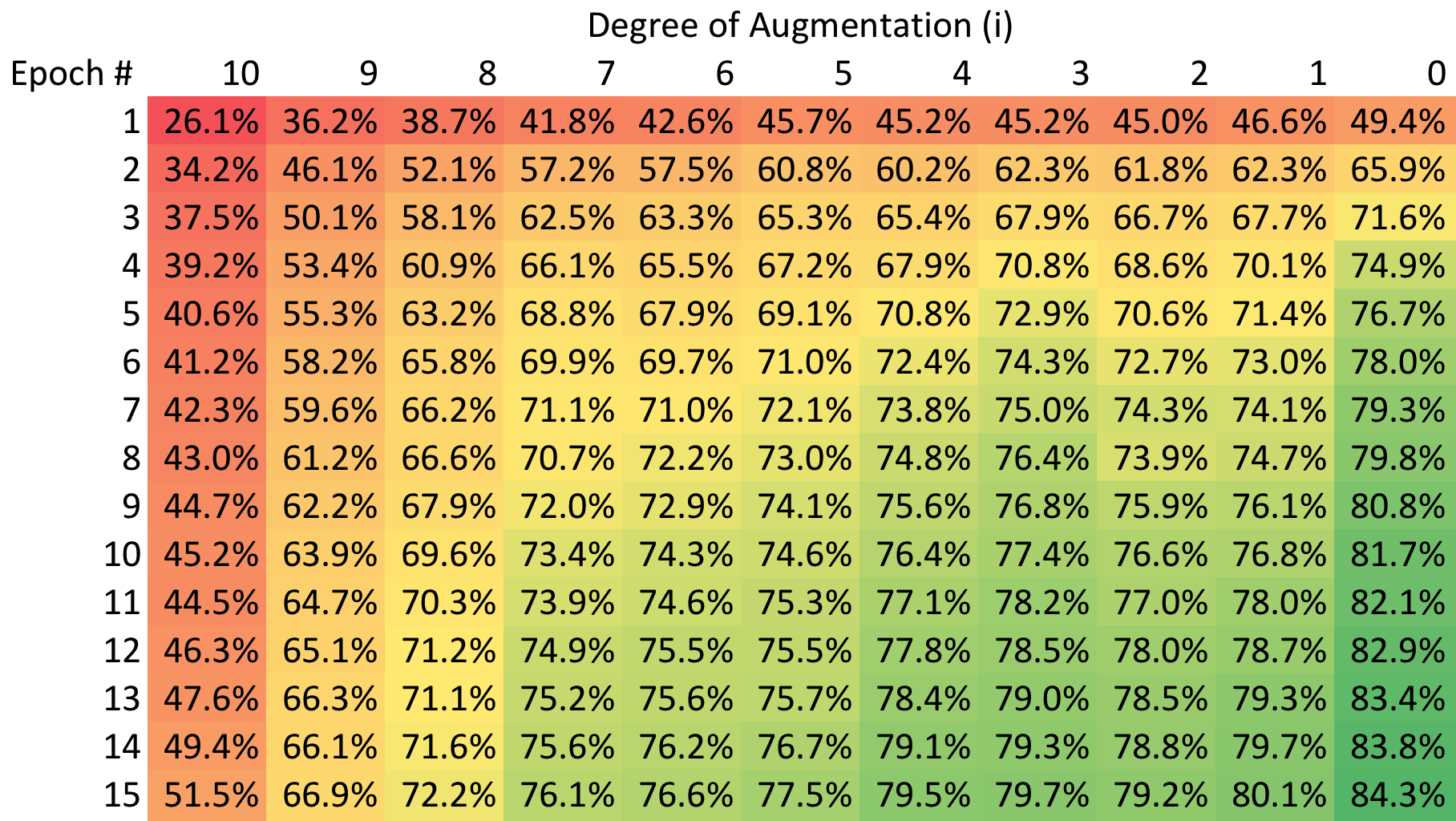
## Constraints

However, due to time constraints, we were unable to isolate the various augmentation parameters (i.e., if we had more computational power we would have run the augmentation parameters in every possible combination such as individually and with all possible combinations of the other augmentation parameters to truly determine what is the ideal augmentation for training). And so, it is possible that some things held constant in all augmentation runs such as horizontal flipping of images is hindering all results. Furthermore, we would’ve liked to run the model for more iterations (i.e. more than 15 epochs), but even 15 epochs took almost 2 hours to run each iteration.

## Results

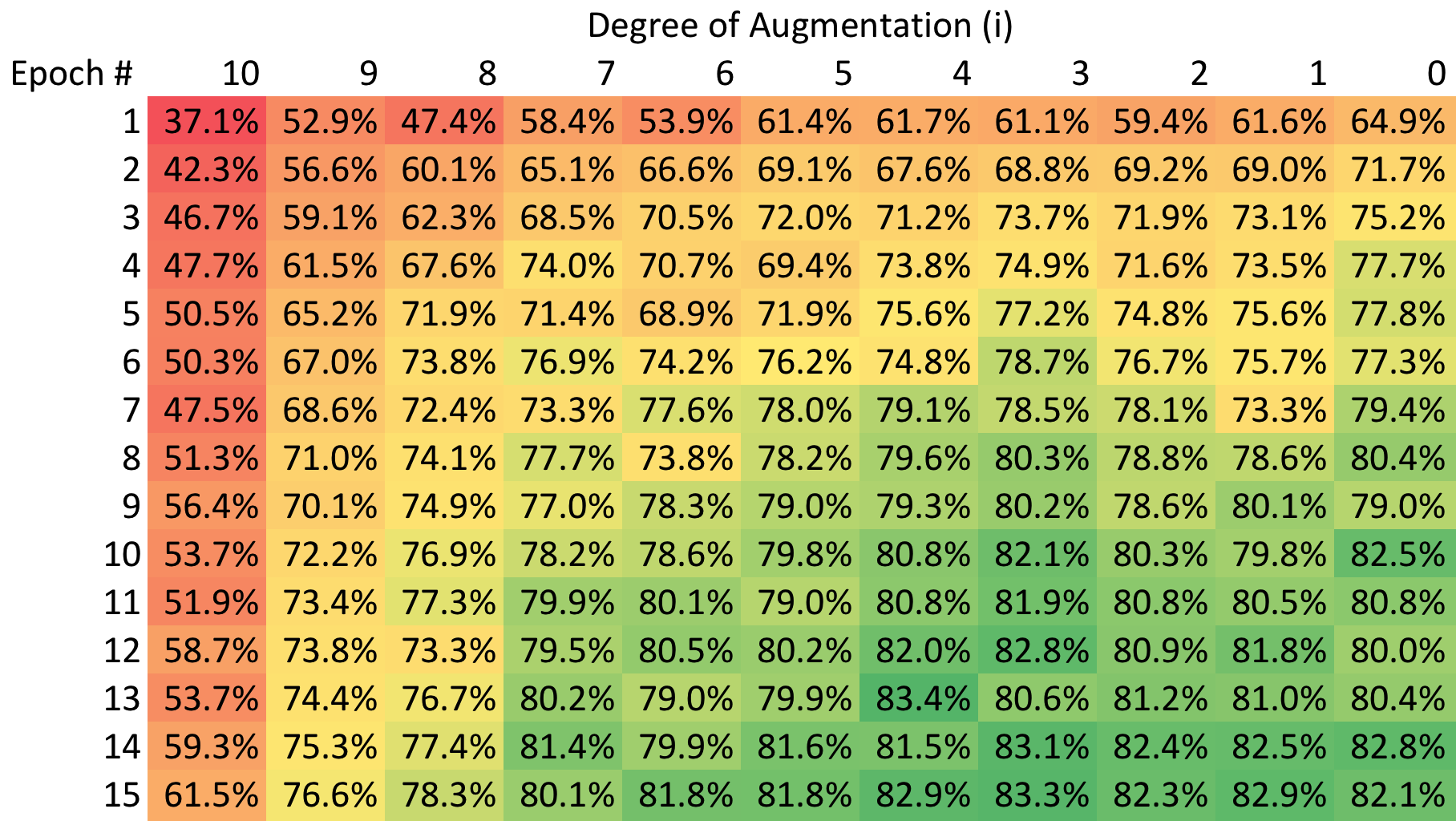
### Training

In terms of results, from the training error (after each epoch) displayed below, we see that data augmentation actually decreases the rate at which the model learns and the final accuracy it reaches.



### Testing

However, the testing error (after each epoch below) seems to suggest that the some degree of data augmentation actually improves final accuracy (i.e. by augmenting the training data, we actually improve generalization of the model, which does make intuitive sense). However, rate of learning for the test error is indeed slower with data augmentation, which does make sense as well since we are adding random noise which the model takes time to adjust to.



## Conclusion

All in all, we conclude in this section that slight data augmentation, while not necessarily good for training accuracy, improves generalization of the CNN to the validation/test set, and thus is ideal. However, the extent to which the data should be augmented and in what ways will need to be further investigated along with allowing more iterations to allow full convergence of the model. Furthermore, these results are for a specific CNN, and other CNN would have to be tested as well to be sure that the results and findings here generalize.