Quasar Spectra Analysis Using Machine Learning

Jeremy Quijano

June 14, 2019

Abstract

There is a limit to a human's ability to characterize the differences between simulated quasar spectra. We want to see whether we can apply machine learning techniques to improve the accuracy of characterizing different simulated spectra with similar initial parameters. The goal was to accurately differentiate between three simulated quasar spectra with clean and noisy data using machine learning. We developed a CNN model that achieved an accuracy of ∼95% with clean data, and a promising accuracy of ∼74% with $\sigma = 0.15$ noise. These results show that machine learning can be applied in this capacity and may be useful in the future.

1 Introduction

1.1 Quasar Absorption Spectra

The intergalactic medium (IGM) is the structure that makes up the space between galaxies in the universe. The IGM has been studied thoroughly by analyzing the absorption spectra from quasars, specifically the Lyman-alpha (Lyα) forest [1, 2]. The Lyα absorption line is associated with an electron transitioning from its $n = 2$ to $n = 1$ orbital in the Hydrogen atom, corresponding with a wavelength of 1215.67Å [1]. The Lyα forest is a series of absorption lines that are associated with this transition that occurs in the IGM when a photon with the corresponding energy collides with a Hydrogen atom [1, 2].

As stated by Boera et al., the Lyα spectra is influenced by the temperature of the surrounding IGM at the time of absorption, which is known as thermal broadening [3]. Understanding the thermal properties of the IGM at different redshifts may provide better insight to the current conditions. If we could simulate the IGM with different parameters, such as temperature, would it be possible to use machine learning to accurately differentiate between the different spectra?

1.2 Machine Learning

Machine learning is a data analysis tool that is useful when working with large sets. Instead of explicitly providing parameters for the program to run by, we instead let it find patterns in the data provided. The program will then makes decisions on its own based on those patterns. Therefore, machine learning can be thought of as "programming with data" [4]. Instead of laying out specific routines for each step and for each set of data, a machine learning model would allow us to work with different types of data more easily. Therefore, we can have one model that works for many different types of data instead of having one program for each. This streamlines our work flow when analyzing large and different types of data.

1.3 Machine Learning in Astronomy

As stated by Baron, the field of Astronomy has been experiencing a growth in the size and complexity of data in recent years [5]. With the shift towards what is called "big data science," machine learning has been growing in popularity as a tool that researchers are using for various tasks, such as detecting, characterizing, and classifying objects using data gathered by different facilities [5].

As stated before, we ask whether machine learning can be used to differentiate between different simulated spectra. Parks et al. have used machine learning to characterize and identify the existence of Lyα absorption within quasar spectra with an accuracy of ∼90% [6]. This type of application, while different in goals, is similar to what we would like to achieve using machine learning. Their success was motivation for continuing our experiment.

1.4 Machine Learning Applied to Simulated Quasar Spectra

The purpose of this experiment was to test the usefulness of machine learning in analyzing simulated quasar spectra and whether it can be used to research the IGM. The differences between simulated spectra is easy for humans to see when we take extreme examples, such as large differences in temperature. However, it is more difficult if we generate spectra with more similar initial parameters, or introduce noise. The limit for humans to characterize and label these simulations is where we believe machine learning may be useful.

If a machine learning model can accurately characterize and label different simulated spectra, then it would allow us to study the IGM with greater accuracy. We have devised a set of experiments to be done using machine learning and different spectra. We want to test the limits of a machine learning model in this type of application, and evaluate the prospect of using it in the future.

2 Experiment

For this experiment, we used five different simulated quasar spectra and used machine learning to create a model to learn the differences between them. We then gave our model test spectra to see if it accurately labelled them. We trained and tested the model with clean, noisy, and scaled data to see the affect that each had on the accuracy. Finally, we investigated whether there was a skew bias for the middle spectra towards the others. All of these tests were devised to see whether machine learning would be a useful tool in this application for the future.

2.1 Input Data

We started by using two simulated spectra that were extremely different from one another and built a model as a proof of concept. Once we had created a model that returned a reasonable accuracy (>90%), we moved on to using three different sets of simulated data that were more similar to one another. The previous two spectra were not used for the rest of the experiment.

2.1.1 Clean Data

The three different spectra, 0.55HM12_zr9 (0.55), 1.0HM12_zr9 (1.0), and 1.8HM12_zr9 (1.8), were created as 5000×2048 pixel FITS images. We sliced each of the spectra into 5000 one-dimensional images of length 2048 pixels, for a total of 15,000 images. The spectra differ mainly in the temperature of the IGM with a mean density of 5100, 7400, 10900 K for the 0.55, 1.0, and 1.8 spectra respectively. Figure 1 is a sample plot of the simulated spectra.

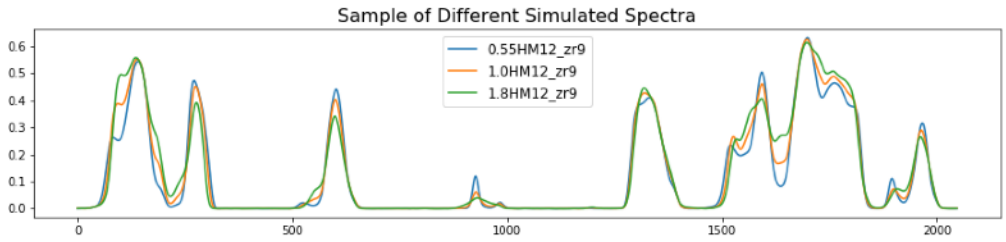

Figure 1: One sample plot from each spectra with difference in temperature.

We can see that as the temperature of the simulated IGM increases, the spectra begins to broaden. There are more distinct features within the 0.55 spectra, while the 1.8 spectra has more gradual peaks. These may be easy to tell apart, however, it is only simulated data without any noise. This is an idealized model because in real observations the data comes in rather noisy. However, there are some questions that can be answered by testing on this type of data, such as: Is it possible to use machine learning with this data? How will a model trained on no noise react to noisy data? Does scaling the data by a multiplicative factor affect the accuracy? Finally, does a model trained on no noise have an inherent skew bias for the middle spectra towards one of the other spectra?

2.1.2 Scaled Data

Another useful test is to try scaling our data. We chose to scale our data by a factor of 0.8, 0.9, 1.1, and 1.2. As you can see in Figure 2 the features of the spectra remain the same and the only difference is the height of the peaks. We want to know how our model deals with these differences, and whether or not it can still accurately label the spectra with scaled data. This would help us in understanding the affect of testing data where the scale is different from the model we chose to train.

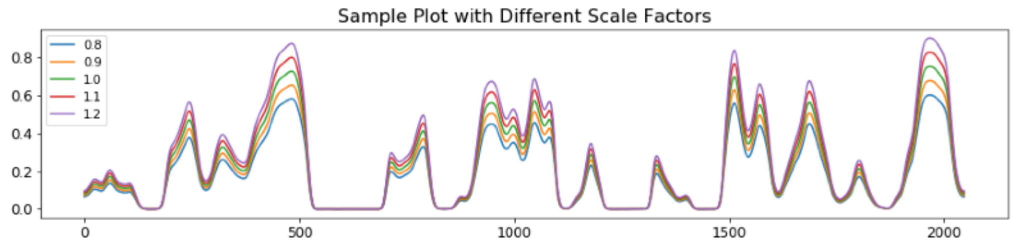

Figure 2: Plot of the multiplicative differences in a simulated spectra.

2.1.3 Noisy Data

We introduced different amounts of noise to simulate the type of data we would receive from a normal observation. We created the noise by adding an array of randomly generated values within a certain standard deviation to each one-dimensional plot. We chose to use sigma values of $\sigma = 0.01$, 0.02, 0.05, 0.10, 0.15, and 0.20. Figure 3 shows how increasing the noise affects the plots visually.

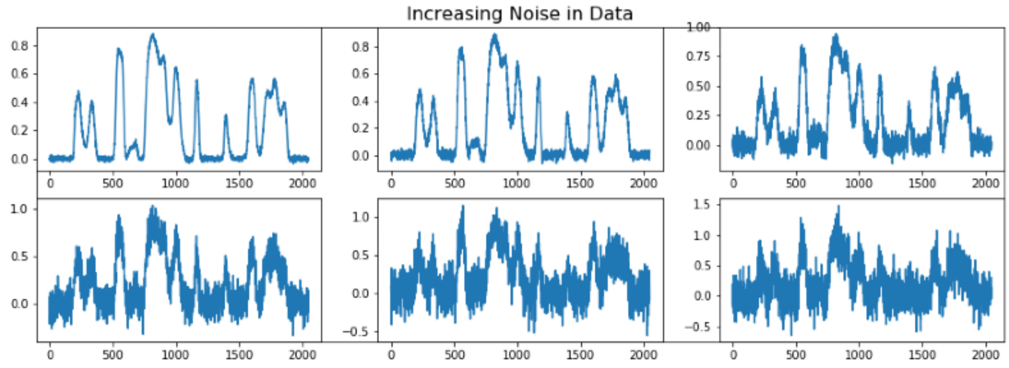

Figure 3: Plot of how the data looks with increasing noise.

As you can see, the unique features of the original spectra begin to drop out as the noise increases. This makes it more difficult for a human to tell the differences between the spectra and where we believe machine learning can be useful. However, there must be a limit to how accurate a model can differentiate between the different spectra, and is the main focus of this experiment. Does noise decrease the accuracy of our model? At what point does our model breakdown? Are models trained on one noise accurate with data of a different noise?

2.2 Convolutional Neural Network Setup

As advised by Dr. Bird, we used a Convolutional Neural Network (CNN) with the TensorFlow package for Python 3 to create our model. This is a common method used for image recognition in machine learning. With help from Dr. Boera, we settled on a structure which used one-dimensional TensorFlow operations for the convolution and pooling steps.

Our neural network consisted of three layers that contained two convolutional operations followed by a max pooling. This structure was proposed by Dr. Boera, as she found that it could increase the accuracy of a model when working with our type of data. After the three layers, we finished with a dense operation to classify the three different spectra. Finally, we used the sparse categorical crossentropy loss function to calculate the error and the Adam optimizer to update the model between each of the 15 epochs. The full parameters of our model are available in Section 6 Table 1.

Following some common machine learning practices, we held 10% of each simulated spectra separate from the training model to test that it was as accurate as it claimed to be. The data was then split into training and validation sets of 80% and 20% respectively. We created different models by training it on data of varying input parameters.

3 Results

The results from this experiment are summarized in sections which correspond to the type of model we used on the test data. We then discuss how the different parameters affected the results, such as scaling it and adding noise. Finally, we discuss the inherent skew bias that our model has on the middle data set, and how it affects this method of analysis.

3.1 Model Trained with No Noise

3.1.1 Results with No Noise Input

Through the testing of our model with no noise, we were able to produce a training accuracy of 96%, validation accuracy of 95%, and a test accuracy of 96%. The accuracy was promising to us because it showed that our model would be useful in this type of application. However, our model was not perfect and has some aspects that need improvement. The model would overfit the data around epoch 13 of 15, but due to the time constraints we were unable to completely avoid overfitting the data. Thus, the training accuracy was usually higher than both the validation and test accuracy's.

Figure 5 is a plot of the accuracy of our models when tested with the same amount of noise, and also shows the same trend of overfitting. Even though our model had issues, we chose to continue with the experiment and propose some optimizational changes for the future. These changes are discussed in Section 4.1.

3.1.2 Results with Scaled Data Input

The next thing we tested was how scaling the data affected the accuracy of our model. For this, we used the model trained with no noise, and tested it on scaled data. We can see how the accuracy is affected by the scale in Figure 4.

An interesting observation is that scaling the data by a factor of 0.8 decreased the accuracy by more than 3%, while 0.9-1.2 affected it by less than 1%. We believe it has to do with the compacting nature of scaling the data. Even though the features are retained when scaled, certain patterns such as height and width of the peaks are changed. These changes to the peaks are what we believe caused the decrease in accuracy for 0.8. However, this does not explain why 0.9-1.2 was still relatively accurate. This may be a unique oddity that occurs with our model, and may be resolved in the future with some improvements.

In spite of the decrease, the model retained an accuracy greater than 92%. This means that we do not necessarily need to give our model the same scaled data for it to accurately characterize and label the spectra. We can be 20% off and still have greater than 90% accuracy when trained on data with no noise. One test that can be done in the future is trying the same process with noisy data.

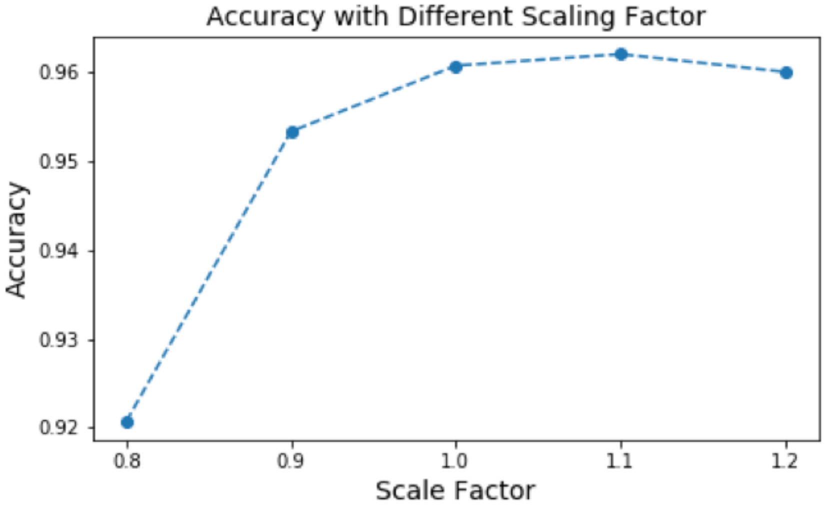

Figure 4: Plot of the accuracy for our model trained on clean data and tested on different scaling factors. Table 2 provides all the values used for the plot.

3.2 Model Trained with Noisy Data

3.2.1 Results with Same Noise Input

By increasing the noise in the data, we expect to see a downward trend in the accuracy of the models. We saw this exact trend during our tests, and Figure 5 shows the training, validation, and testing accuracy of each model when given data with the same amount of noise. As previously stated, we were unable to avoid the overfitting trend. For the noisy models, overfitting occurred around epoch 12 of 15.

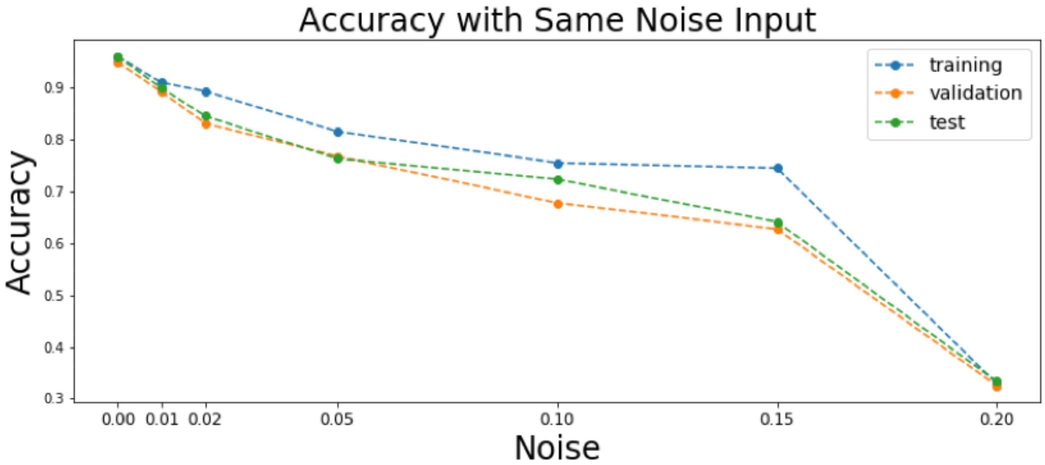

Figure 5: Plot of the accuracy of each model trained on different noise scales, but tested on the same noise. The tabulated data can be viewed in Table 3.

We can see that our model does relatively well in learning the differences between the spectra even with the added noise. A promising result came from our model with $\sigma = 0.15$ that had an accuracy greater than 70%. We believe that with a few adjustments to the model, we could increase the accuracy and have a very useful tool for analyzing simulated spectra. However, the model trained on noise with $\sigma = 0.20$ has an accuracy of 33%. In other words, it failed to train and is where our model broke down.

We believe the decrease in accuracy when adding the noise has to do with the loss of unique features within the data. As seen in Figure 3, the noise masks the overall features to the point where the model trained on $\sigma = 0.20$ was unable to accurately differentiate between the three spectra. When testing that model, it would only label the test spectra as 0.55 and nothing else. This is the one aspect that we could not fix in the allotted time, but the data was still useful to us.

3.2.2 Results with Different Noise Input

For this part, we tested each model with different amounts of noise. The importance of this test was to see whether a model trained on one noise level could be accurate if given data with a different amount. We believed that a model trained on noise should still be accurate with data that had different noise, but we were not sure how accurate the models would be. We have plotted the accuracy of each model when tested on different noise in Figure 6.

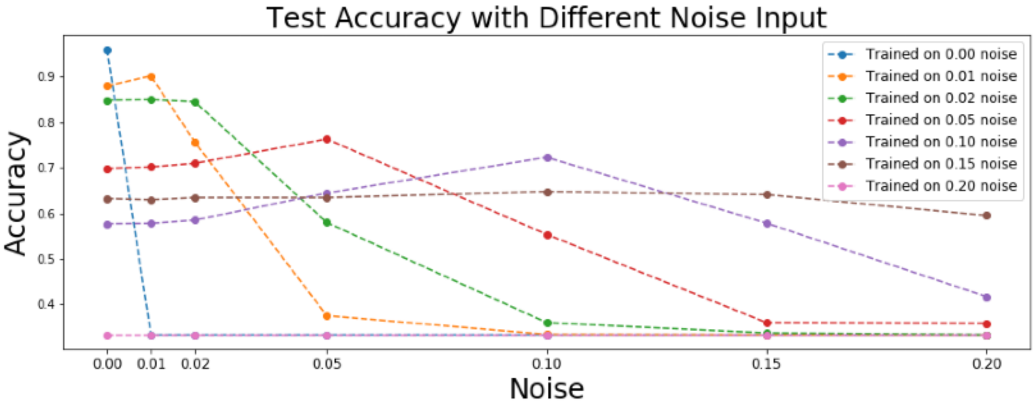

Figure 6: Plot of the accuracy of each model trained on different noise scales, and tested on different noise. The data can also be viewed in Table 4.

We were correct in thinking that models trained on noisy data would still be accurate when tested on a different amount. However, some interesting things to note were the consistency, or lack of consistency, when tested with different levels of noise.

We found that a model trained on no noise fails completely when tested on any amount of noise. Therefore, that model would be useless if we wanted to use simulations to try and fit it to observational data. As stated above, we believe that the breakdown in accuracy is a result of the increased ”features” of the noisy data. The model trained on no noise was not able to characterize the noise and find the unique patterns of the normal simulated data. In other words, the model trained on no noise is only useful in characterizing spectra with no noise.

The more interesting result from our tests was how the consistency of our model increased as we increased the noise of the training data. While the overall accuracy decreased, the model became more consistent when tested with data with different levels of noise. Specifically, the model trained on $\sigma = 0.15$ noise was the most consistent with an average accuracy of $\sim$63% and a high/low percentage change of $\sim$5% across all noise levels. Compared to the others which showed large drop offs, the $\sigma = 0.15$ model showed promising results. This tells us that models trained on noisy data can be used on data with different noise levels, and can therefore be useful with observational data which we do not know the exact amount of noise.

Because this model was so consistent, we believe that if we can increase the accuracy to greater than 90%, then it can be useful for researching the IGM in the future. We could create simulated IGM plots with different parameters and add the necessary noise to those simulations to try and fit the right model to observations. This would make studying the IGM easier, however, we still have some work to do with our model in order for that to happen.

3.2.3 Skew Bias

The final aspect we investigated was how our models dealt with the 1.0 spectra. Because there are two ways for it to skew, how often would it skew and would it favor one direction over the other? Because the 0.55 and 1.0 spectra are closer in temperature than the 1.0 and 1.8 spectra, it would be reasonable to see a skew favored towards 0.55. However, it would also be reasonable to see a skew favored towards the 1.8 spectra because looking at Figure 1, we can see that the 1.0 and 1.8 spectra have more similar features than the 0.55 and 1.0 spectra do.

The first experiment used a model trained on a certain noise, and tested its skew using the 1.0 spectra with the same noise. The results for each model can be seen in Figure 7. The first figure shows the instances where our models inaccurately labelled the 1.0 spectra as 0.55 or 1.8, and the second figure shows the accuracy of the model when it correctly labelled the 1.0 spectra.

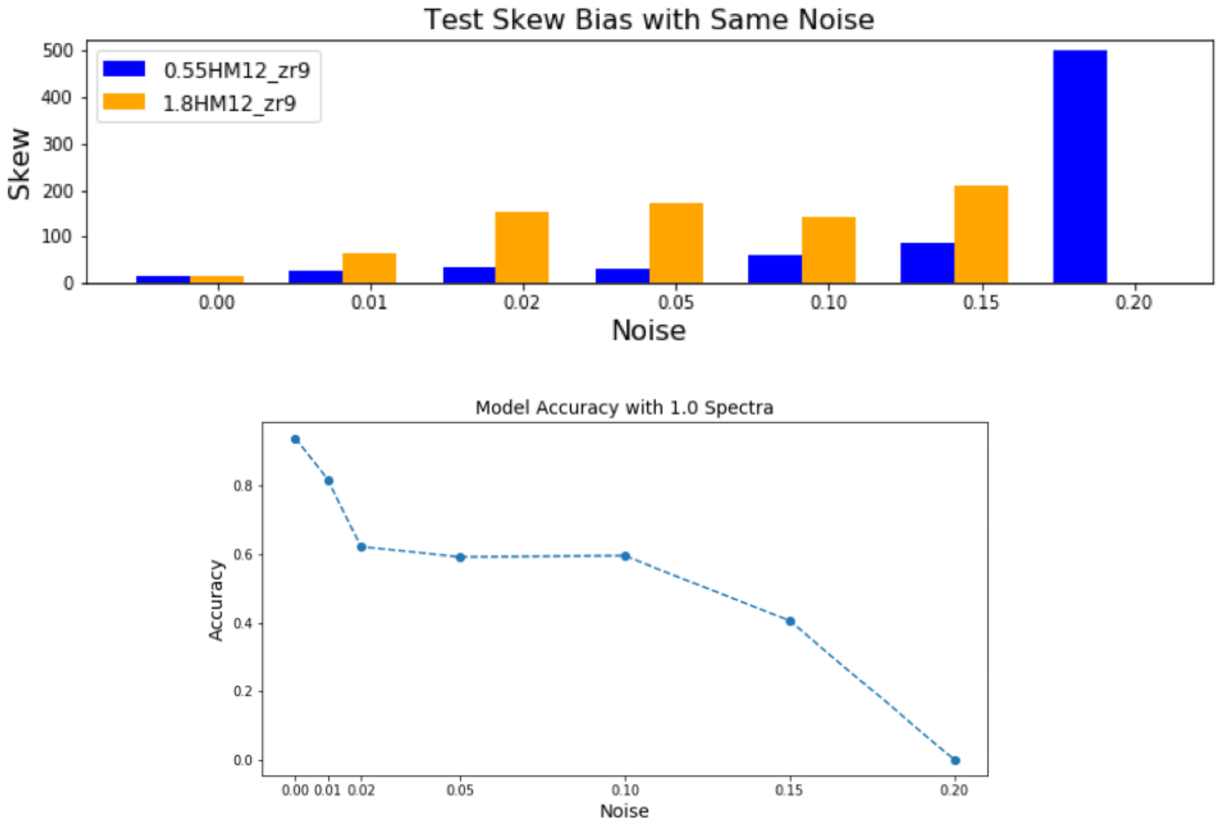

Figure 7: TOP: Plot of the skew that the 1.0 data set had using a model and test data with the same amount of noise. BOTTOM: Accuracy each model when tested on the 1.0 spectra. You can see the data in tabular form in Table 5.

We can see the skew increases as the noise increases, therefore, the accuracy of correctly labelling the 1.0 spectra also decreases. One notable feature is the increasing skew towards the 1.8 spectra as the noise increases. As was previously hypothesized, the similarity between the 1.0 and 1.8 spectra seems to influence the accuracy of our model. Because the spectra were similar to begin with, our model began characterizing them as the same spectra when more noise was added. Another observation is the complete skew for our model trained on $\sigma = 2.0$ noise. The $\sigma = 0.20$ skew is a discrepancy because our results showed that any input would be labeled as 0.55 when tested on the model trained on $\sigma = 0.20$. Therefore, we can still say that it will trend to an increasing skew towards 1.8 as noise increases.

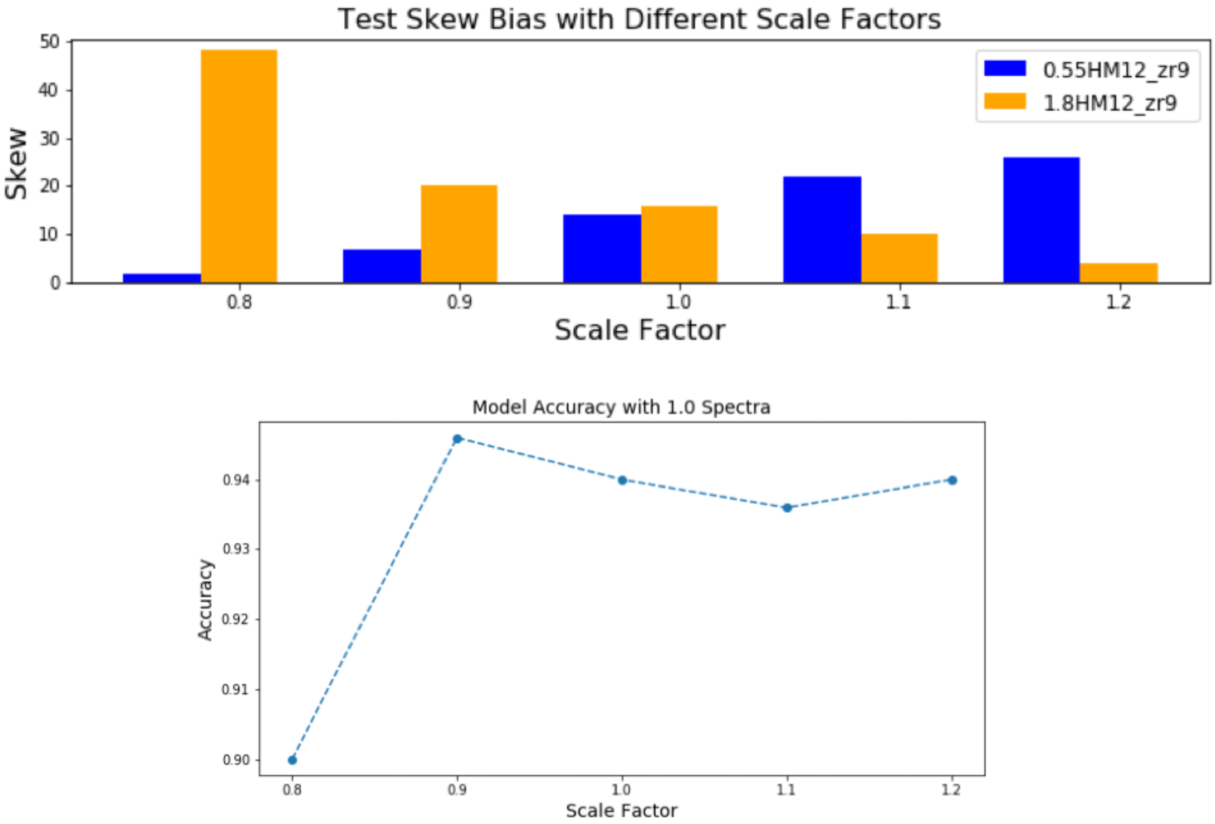

Figure 8: TOP: Plot of the skew that the 1.0 data set. BOTTOM: Accuracy of our model trained on no noise when tested with different scale factors on the 1.0 spectra. Table 6 has the actual results from this test.

The final experiment used our model trained on no noise, and tested it on scaled data. The results in Figure 8 correspond to the same aspects discussed previously for Figure 7.

The results from testing different scale factors was odd. We thought there would be a similar trend as the test before, however, we can see this was not the case. The 1.0 spectra favored 1.8 when scaled by 0.8-1.0, and it favored 0.55 when scaled by 1.1 and 1.2. The bias changed at a scale of 1.0, or the normal scale of the data. We believe the bias switches because there were changes to the features in terms of height and broadness when scaling the 0.55 and 1.8 spectra. Even though the 1.8 spectra has broader features, when scaling it to less than 1.0, it brings the features more in line with the 1.0. The same may also be happening to the 0.55 spectra when scaling it greater than 1.0, and may explain the switch in the bias.

By looking at the results from the 1.0 scale tests, we can see that the bias was less compared to the others. Therefore, if we manage to find data with the same scale factor for our model, then the skew would almost be negligible. This test was still successful, and our model recorded an accuracy greater than 90% for all scale factors, and a greater than 93% accuracy for all but the 0.8 scale. Therefore, when working with data with no noise, our model is fairly accurate across all three spectra, and the skew bias is relatively small compared to the noisy models.

We can see that our model has an inherent skew when working with three data sets. At the moment, it can be noted that if two different spectra have similar features, such as our 1.0 and 1.8 spectra, then it will favor a skew in that direction. Also, there was a larger bias for noisy data than with scaled data. The noisy data skewed 2 to 5 times more than the scaled data would. This is possibly due to the fact that similar data becomes more similar when adding noise. The unique features between them are less distinct, and the model would then characterize them as having the "same" features. We may be able to improve on this issue in the future with some adjustments.

4 Conclusion

We have found that machine learning may be a valuable tool for us to use in the future. We believe that machine learning in this application proved to have marketable qualities, such as accuracy with simulated data, and consistency with noisy data. These are important factors that we wanted to test, and we were happy to see the results given the short time frame.

The accuracy of our initial model gave us the confidence to move forward with this experiment. If we were not able to accurately characterize these spectra with clean data, then there was no hope in using these methods in the future. However, we were able to achieve an accuracy >90% and >70% on clean and noisy data respectively.

Our model showed promising results in terms of accuracy and consistency when working with different types of data. Specifically, the models dealing with noise showed the most promising results for applying this method to observations. The consistency of our model trained on data with noise of $\sigma = 0.15$ shows that there is a possibility to create a better model that can accurately characterize many different types of data with different noise levels in the future. This type of model, if achievable, would be an valuable tool for researchers to use. It would make characterizing the IGM with different parameter easier, and tests involving simulations may be able to fit to observations.

This was only our first step in applying machine learning to the analysis of the spectra from quasars, but we believe that it will prove useful in future experiments. The next steps would be testing with extreme outliers, such as dead or hot pixels and gaps in data. What are the other limits to this model that we may not be able to control? We would like to continue testing the effectiveness of machine learning in this application in the future, however, we will need to make a few improvements before we can continue.

4.1 Proposed Improvements

Due to the time constraints of this experiment, we were not able to fully optimize our model. Therefore, we have come up with a few methods that may help increase the accuracy and can be implemented later to see how they work.

To decrease the overfitting trend, the easiest method would be implementing a dropout or penalty to the model. The dropout avoids overfitting by randomly dropping out some of the connected neurons between layers. This decreases the chance of overfitting by ”forgetting” certain factors between layers or epochs. We were not able to implement this in any meaningful manner, but is a possible solution to our issue. Overfitting could also be a side effect of the amount of convolutional layers we used. By decreasing the layers and increasing the number of epochs, it could affect how quickly the model characterizes and fits the data.

In order to increase the accuracy of our model, especially when using noisy data, we believe that increasing the window for the convolutional layers would allow more unique features to be characterized at once. Looking back at Figure 3, we can see that even with increased noise the large peaks are still noticeable to some extent. By increasing the window to view more of these features, we believe our model would more accurately characterize and label these noisy data sets. This may have some effect on the skew bias, but it is just a hypothetical side-effect from the increased window size. This could also affect the overall accuracy of our model, and will need to be tested to find the best method of implementation.

5 Acknowledgements

This project was proposed and headed by Dr. George Becker, Assistant Professor at University of California, Riverside (UCR). We would like to thank Dr. Simeon Bird, Assistant Professor at UCR, and Dr. Elisa Boera, Postdoctoral Scholar at UCR, for their knowledge and guidance in machine learning, and for providing the necessary references and tools for this project. We developed our model through the use of the TensorFlow package for Python 3: https://www.tensorflow.org/install, and the full code for this project is available on github: https://github.com/JeremyQuijano/Quasar-Spectra-Analysis-Using-Machine-Learning.

References

[1] Michael Rauch. The lyman alpha forest in the spectra of qsos. 1998. arXiv:astro-ph/9806286. doi:10.1146/annurev.astro.36.1.267.

[2] Jill Bechtold. Quasar absorption lines, 2001. arXiv:astro-ph/0112521.

[3] Elisa Boera, George D. Becker, James S. Bolton, and Fahad Nasir. Revealing reionization with the thermal history of the intergalactic medium: new constraints from the lyman-α flux power spectrum. 2018. arXiv:1809.06980. doi:10.3847/1538-4357/aafee4.

[4] TensorFlow. Why tensorflow. https://www.tensorflow.org/about/.

[5] Dalya Baron. Machine learning in astronomy: a practical overview, 2019. arXiv:1904.07248.

[6] David Parks, J. Xavier Prochaska, Shawfeng Dong, and Zheng Cai. Deep learning of quasar spectra to discover and characterize damped lya systems. 2017. arXiv:1709.04962. doi:10.1093/mnras/sty196.

6 Tables

Parameters	Value	Description
Convolutional Kernel	16	Size for each convolutional instance.
Convolutional Filters	32	Number of filters for the first convolutional instance per layer.
Convolutional Filters	16	Number of filters for the second convolutional instance per layer.
Pooling Kernel	4	Size of the pool with softmax activation.
Dense	3	Connected layer used for classifying the three spectra.

Table 1: List of the full parameters used for our CNN model. Corresponds with Section 2.2.

Scale Factor	Accuracy
0.8	.9207
0.9	.9533
1.0	.9607
1.1	.9620
1.2	.9600

Table 2: Accuracy for our clean data model when tested with scaled data. Corresponds with Figure 4.

Noise	0.00	0.01	0.02	0.05	0.10	0.15	0.20
Training	.9606	.9099	.8931	.8156	.7550	.7450	.3291
Validation	.9496	.8919	.8311	.7678	.6778	.6274	.3252
Testing	.9607	.8993	.8453	.7633	.7233	.6413	.3333

Table 3: Accuracy for each model created with different noise. Corresponds with Figure 5.

Noise	0.00	0.01	0.02	0.05	0.10	0.15	0.20
0.00	.9607	.3333	.3333	.3333	.3333	.3333	.3333
0.01	.8800	.9020	.7567	.3753	.3340	.3333	.3333
0.02	.8487	.8500	.8453	.5800	.3600	.3373	.3333
0.05	.6987	.7020	.7100	.7633	.5540	.3600	.3587
0.10	.5773	.5780	.5853	.6447	.7233	.5787	.4173
0.15	.6327	.6300	.6347	.6347	.6480	.6413	.5953
0.20	.3333	.3333	.3333	.3333	.3333	.3333	.3333

Table 4: Accuracy for each model created when tested with different noise. Corresponds with Figure 6.

Noise	0.00	0.01	0.02	0.05	0.10	0.15	0.20
Spectra 0.55	14	28	35	30	60	85	500
Spectra 1.8	16	63	154	174	142	212	0
Accuracy	.9400	.8180	.6220	.5920	.5960	.4060	0

Table 5: The amount of 1.0 skew towards the 0.55 and 1.8 spectra when the clean model was tested with clean data. Corresponds with Figure 7.

Scale Factor	0.8	0.9	1.0	1.1	1.2
Spectra 0.55	2	7	14	22	26
Spectra 1.8	48	20	16	10	4
Accuracy	.9000	.9460	.9400	.9360	.9400

Table 6: The amount of 1.0 skew towards the 0.55 and 1.8 spectra when the clean model was tested with scaled data. Corresponds with Figure 8.