COS 711 Assignment 3: Predicting Air Quality with Deep Learning Models

Stefanie Y. Wannenburg
15013198
24 June 2020
Dept. of Computer Science, University of Pretoria
github link:https://github.com/StefanieWannenburg/COS711Ass3.git

***Abstract*—In this paper three different deep neural network architectures were developed to predict the air quality for the next day for five different locations in Uganda. The goal is to take informed precautionary actions to protect the health of the communities. The models were trained on a combination of hourly weather recordings and air quality contributing features from each location. Two convolutional neural networks were developed by viewing the problem domain as regression and classification, respectively. Following this, a long-short term memory model was developed as a regression problem. Data preparation, hyperparameters and model parameters choices were based on the best found research available. It was found that a regression technique is more suitable to solve this type of problem, and techniques to mitigate imbalanced continuous data proved critical for a well performing network. From the three models the long-short term memory model performed the best. However, it was concluded that none of the three models are adequate for practical use, and further development should be done on the critical areas identified.**

***Index Terms*—Convolutional Neural Network, Long Short-Term Memory Network, Regression Problem, Multi-Class Classification Problem, Multi-Variate Time-Series Data**

I. Introduction

This report documents the use of two different deep neural network (NN) architectures to predict air quality over different locations in Uganda. Both a convolutional neural network (CNN) and a long short-term memory network (LSTM) model were built for this problem. The input data is formatted as a multi-variate time-series. This document outlines all the steps, including data preparation, hyperparameters choices and a comparative results discussion.

II. Background

Due to the expense of reference grade monitors, there has been a lack and need for air quality data in sub-Saharan Africa. Access to this data is essential to understand which air quality improvement actions to take to protect the health of the communities. AirQo, a research initiative of the College of Computing and Information Science in Uganda, has installed low-cost sensors across Uganda [1].

'AirQo Ugandan Air Quality Forecast Challenge' was an open-online challenge led by the University of Birmingham, UK [1]. The goal of the challenge was to gain insight into the relationship between historical data, generated by five of the sensors, and air quality, so that health improvement actions can be taken. To achieve this a generalized NN that predicts the air quality of the next day, much like how weather models are used, was built. The data consists of 5-day hourly weather readings and air quality contributing features from each sensor's location. The data can be found on the challenge page [1]. Air quality targets are indicated by PM2.5 (particulate matter smaller than 2.5 micrometers in diameter) and given 24 hours after the last weather reading. Classes of hazardous levels of PM2.5 can be seen in Table I.

TABLE I
Classes of hazardous levels of PM2.5

Health concern	$\mathbf{PM_{2.5}}$ $(\mu \mathbf{gm^{-3}})$	**Precautions**
Good	0 -12	None
Moderate	13-35	Unusually sensitive people should consider reducing prolonged or heavy exertion
Unhealthy for sensitive groups	36-55	Sensitive groups should reduce prolonged or heavy exertion
Unhealthy	56-150	Everyone should reduce prolonged or heavy exertion, take more breaks during outdoor activities
Very unhealthy	151-250	Everyone should avoid prolonged or heavy exertion, move activities indoor or reschedule
Hazardous	250 +	Everyone should avoid all physical activities outdoors

These classes will be used as a guideline to inform Ugandan citizens of the hazardous level expected for the next day and enforce precautionary health actions. This problem can be viewed as either a regression or a classification problem, with either a continuous air quality value prediction or a hazardous level prediction of independent classes, being the difference.

The addition of more than one hidden layer in NNs allows a NN to learn more complex hierarchical representations of data. This capability has allowed the application of deep NNs to become very popular. CNN models where specifically designed to handle image data by detecting spacial dependence in pixelated inputs.

Multi-variate temporal data can be used as input in the same sense as images. Each observation consists of many variables that share the same time-steps and this can be treated like a one-dimensional image that a CNN can read and filter into the most important elements. Deep CNNs use these filters to find the hierarchical relationships and make accurate predictions [2].

Recurrent NN like LSTMs has been successfully used in complex language processing problems, due to it's ability to handle order between observations when mapping from inputs to outputs. Unlike CNNs, LSTMs remembers useful context from the input data, and can dynamically change that context as needed [2].

Both of these models have been developed for the 'AirQo Ugandan Air Quality Forecast Challenge' as described above. A comparison of both the regression and classification problem domain was done with a CNN model and a regression problem was applied to the LSTM model.

III. Experimental Set-Up

A. Data Preparation

Analysis and pre-processing were performed on the data sets. Data transformation steps are outlined in the following subsections.

a) Data Inspection: There are two sets of input information. The first is a set of 5 consecutive days of hourly weather recordings. Table II explains the different weather variables given.

TABLE II
Weather Variables

Variables	**Description**
Temp	Mean temperature recorded at the site over the hour (^oC)
Precip	Total rainfall in mm recorded at the site over the hour (mm)
Re humidity	Average relative humidity over the hour (%)
Wind dir	Mean direction of the wind over the hour (^{o})
Wind spd	Mean wind speed at the site over the hour (m/s)
Atmos press	Mean atmospheric pressure (atm)

Each weather series is associated with an air quality target value and a unique sensor in Uganda. The target value represents the PM2.5 air quality prediction 24 hours after the last weather recording. Secondly, a list of contributing features about the location of each sensor, as seen in Table III, was also given.

NNs are designed on mathematical principles, requiring numerical inputs. The given input data has one categorical feature, namely *'Location'*. Integer encoding was used to convert the string set $\{A,B,C,D \text{ and } E\}$ to the corresponding range $\{1,2,3,4,5\}$. Although there are many different conversions found in research such as one-hot encoding, this simple transformation was deemed sufficient since the locations are independent and few.

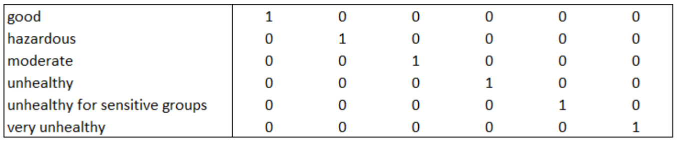
TABLE III
Location Features

Feature	**Description**
Location	One of either A,B,C, D or E referring to the different locations selected
Loc altitude	The height above sea level (m)
Km2	The area of the parish in which the device is located (m^2)
Aspect	The direction of the slope on which the device is located faces
Dist motorway	The distance of the device from the nearest motorway (m)
Dist primary	The distance of the device from the nearest primary road (m)
Dist secondary	The distance of the device from the nearest secondary road (m)
Dist tertiary	The distance of the device from the nearest tertiary road (m)
Dist unclassified	The distance of the device from the nearest unclassified road (m)
Dist residential	The distance of the device from the nearest residential road (m)
Popn	The population of the parish in which the device is located
Hh	The number of households in the parish in which the device is located
Hh cook charcoal	Number of households in the parish in which the device is located which cook using charcoal
Hh cook firewood	Number of households in the parish in which the device is located which cook using firewood
Hh burn waste	Number of households in the parish in which the device is located which dispose of solid household waste by burning

The weather series has also been given with a number of missing values. Generally, weather variables within the same time frame tend to be alike. The hourly data was transformed to averaged daily data and NaN values were forward filled by the first available recording of the specific weather variable. Observations with initiating NaN values were back filled.

The distance features $\{$*'Dist primary', ...*$\}$ within the locational dataset contains empty cells that actually represent values greater than 5000m. These empty cells where thus filled by the value of 5000 as a cap for those inputs. *'Dist motorway'* was completely empty, thus this feature was removed, since it would add no value.

An additional set of multiple-class labels where created for the classification problem by transforming the target values into the hazardous levels seen in Table I. This set was fitted using a LabelEncoder into a format required by the NN. The LabelEncoder essentially quantifies the levels of different labels. Figure 1 illustrates this transformation.



good	1	0	0	0	0	0
hazardous	0	1	0	0	0	0
moderate	0	0	1	0	0	0
unhealthy	0	0	0	1	0	0
unhealthy for sensitive groups	0	0	0	0	1	0
very unhealthy	0	0	0	0	0	1

Fig. 1. LabelEncoder transformation

b) Data Sampling: The weather series dataset was merged with the locational features. To obtain an amalgamated three-dimensional input, the locational features were duplicated over time.

The amalgamated data set was randomly separated, based on typical values seen in literature. The chosen criteria is listed in Table IV. The training set is the sample of data used to train the model. The testing set is used to test the generalization, under-fitting and over-fitting of the model, while tuning the model hyperparameters. Visual inspection was done to ensure that the training and testing sets distributions are good representatives of the entire dataset.

TABLE IV
DATA SET SEPARATION

Data set	**Separation criteria**	**Input dimensions**
Training set	80% of the dataset	(12431,5,21)
Test set	20% of the dataset	(3108,5,21)

[a]Dimensions (patterns,time-steps,features).

The 21 features as seen in the dimensions are {*'temp', 'precip', 'rel humidity', 'wind dir', 'wind spd', 'atmos press', 'loc altitude', 'km2', 'aspect', 'dist trunk', 'dist primary', 'dist secondary', 'dist tertiary', 'dist unclassified', 'dist residential', 'popn', 'hh', 'hh cook charcoal', 'hh cook firewood', 'hh burn waste', 'location'*} and the 5 time-steps represents 5 days.

c) Imbalanced Data: The imbalanced distribution of the training set target values was plotted as continuous values and as classes, seen in Fig 2. Learning a supervised NN on imbalanced target values, has the consequence of false metric results. The majority ranges or classes dominates the learning process and the NN disregards the minority ranges or classes. In Fig 2 it can be seen that the *'very unhealthy'* or *'hazardous'* air quality levels as well as the good levels fall into the minority. The importance of accurately predicting the extreme unhealthy cases is essential to protect the health of the community and for the goal of this model.

Luckily, imbalanced classes is a common issue in machine learning and the Synthetic Minority Over-sampling Technique (SMOTE) is popularly used to overcome such an issue. SMOTE basically, under-samples majority classes and over-samples minority classes by creating synthetic minority class examples [3]. For imbalanced continuous target values however, only a few solutions exist. Among them is a pre-processing method called SMOGN which combines two over-sampling procedures and an under-sampling strategy [4]. To over-sample, interpolation with SMOTER is done when the seed example and k-nearest neighbour is close enough. Otherwise the introduction of Gaussian Noise is used. Under-sampling is done randomly. SMOTE was applied for the classification task, and SMOGN with a relevance threshold of 0.7, to the regression tasks. These transformations were only applied to the training set so that the testing set stays untouched for evaluation purposes. Figure 3 depicts the transformed continuous target values and classes. The training set dimensionality changed to (19325,5,21) and (24939,5,21) for the different regression and classification problem domain respectively.

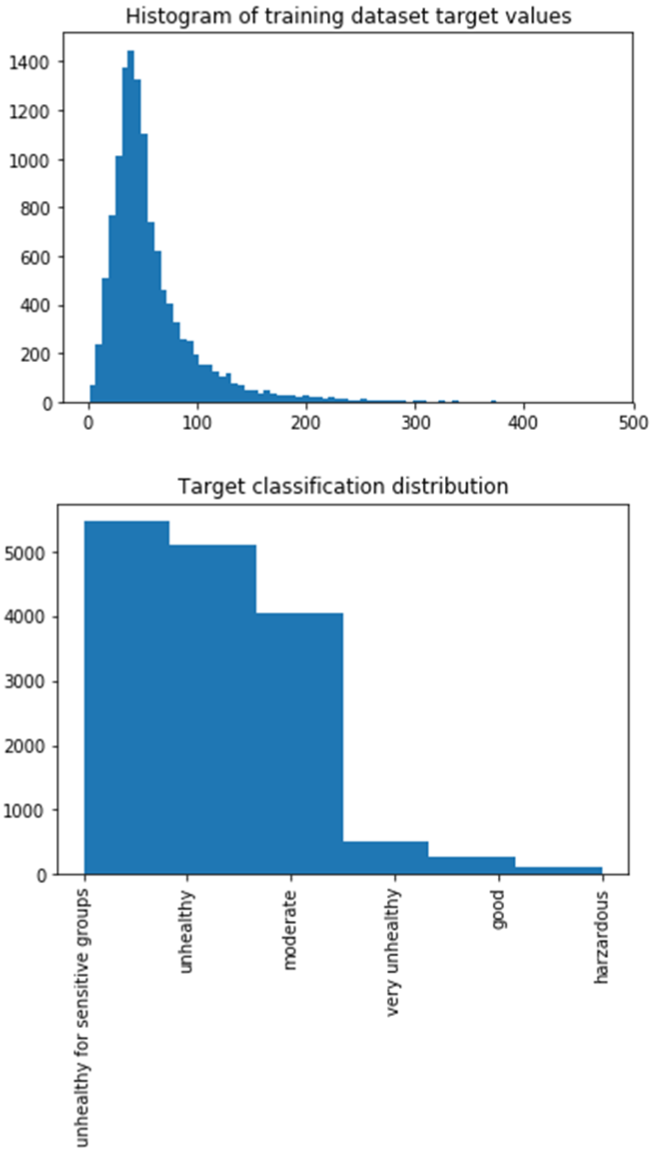


Fig. 2. Target value distributions

The lower air quality values that represented the majority have been under-sampled, while the additional synthetic examples have been generated for the higher values. From the application of SMOGN the extreme cases, values between (300-400) have also increased, but not by as many.

d) Data Scaling: Each set of input features and the continuous target values are in different ranges of mainly random distributions. This increases the difficulty of the problem, since unscaled attributes can result in a slow and unstable learning of the model [5]. The dataset was scaled using a MinMax scaling technique, where feature values are essentially normalized into a range between {0-1}. The training set parameters were stored and used to scale both the training and testing set. This was done to avoid data leakage during model evaluation.

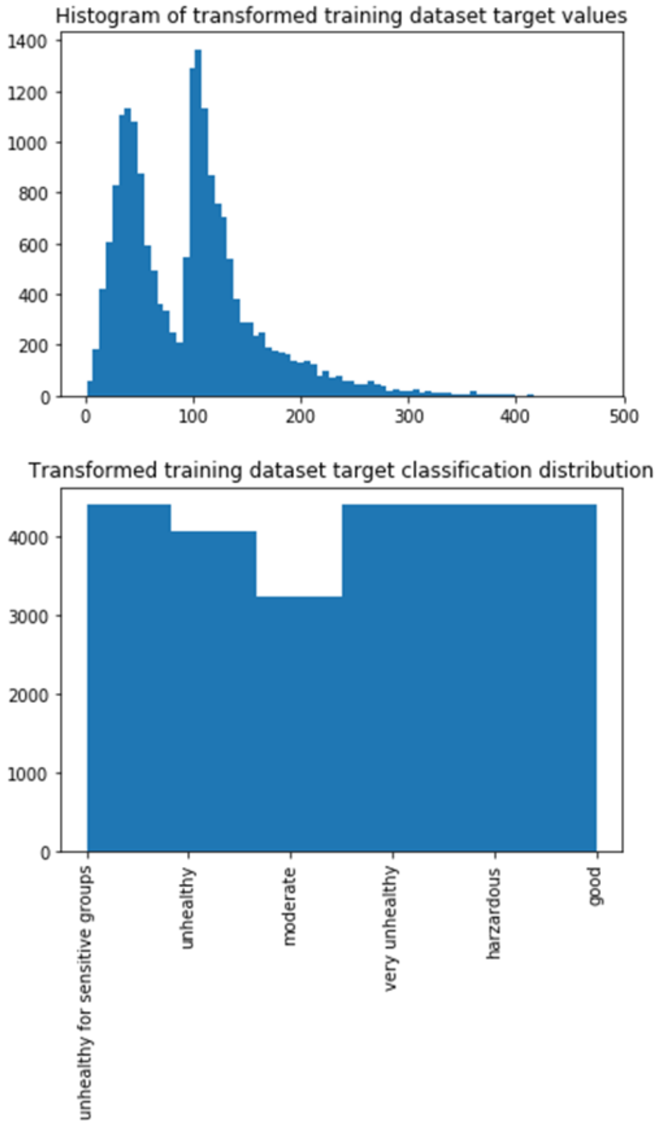


Fig. 3. Transformed target value distributions

B. Neural Network Parameters

The following sections outlines and justifies the architecture structure, and list of model parameters and hyperparameters chosen for both the regression and classification CNNs and the LSTM model.

a) Dimensionality Of Neural Network: Both CNNs structures for the regression and classification tasks can be viewed in Table V. For visualization purposes Fig. 4 illustrates the structures as well. The first convolutional layer has 20 filters and the second convolutional layer has 10 filters, both have a kernel size of (2, 21). The convolutional output is a "filtered" univariate time-series. Max-pooling is then applied to each filtered time-series and used as input to a fully-connected dense layer. Dropout regularisation is applied to the 20 input nodes of the dense layer and in between the convolutional layers. Dropout probabilistically removes inputs in an attempt to make the network more robust to the inputs [5]. The regression problem CNN output layer consists of one neuron for the continuous value output, while the classification problem CNN output layer consist of the 6 nodes for the hazardous classes. The total trainable parameters for the two CNN models are 1511 and 1616, respectively. Having less trainable parameters then patterns/observations in your training set is a good check to avoid over-fitting your model. The choice of two convolutional layers was to allow the NN to find the hierarchical representation, while the number of filters and nodes chosen are guestimates.

TABLE V
CONVOLUTIONAL NEURAL NETWORKS STRUCTURE SUMMARY

Layers	**Shape (Regression)**	**Shape (Classification)**
Input layer	(None, 5, 21)	(None, 5, 21)
Convolutional layer 1	(None, 4, 20)	(None, 4, 20)
Dropout (50%)		
Convolutional layer 2	(None, 3, 10)	(None, 3, 10)
Max-pooling layer 1	(None, 1, 10)	(None, 1, 10)
Flattening layer 1	(None, 10)	(None, 10)
Dropout (50%)		
Dense layer 1	(None, 20)	(None, 20)
Output layer	(None, 1)	(None, 6)

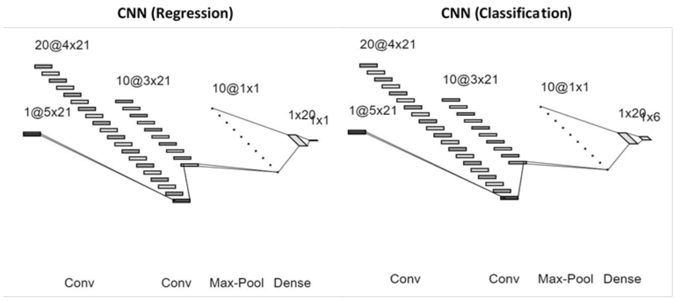


Fig. 4. CNN Visual Structures

The LSTM model structure summary can be seen in Table VI and the visual representation in Fig. 5. This model consists of two hidden layers with 20 and 10 LSTM nodes, respectively. Regularised dropout has also been applied to the last hidden layer. The output layer consists of one perceptron which allows for a continuous value output. The total number of trainable parameters are 4611.

b) Activation Functions: The activation function employed on the hidden layer feature maps and neurons is called the Rectifier Linear function (ReLu). Various articles suggests that this activation function performs the best in deep NN models, mainly because of it's linear behaviour [4]. As seen in (1) the function is linear for positive weighted sum input values (*net*) and *0* for negative values and values equal to *0*.

$$f(net) = \begin{cases} net & \text{if } net > 0 \\ 0 & \text{if } net <= 0 \end{cases} \tag{1}$$

TABLE VI
LONG-SHORT TERM MEMORY MODEL (REGRESSION) STRUCTURE SUMMARY

Layers	Shape
Input layer	(None, 5, 21)
LSTM layer 1	(None, 5, 20)
Dropout (50%)	
LSTM layer 2	(None, 10)
Output layer	(None, 1)

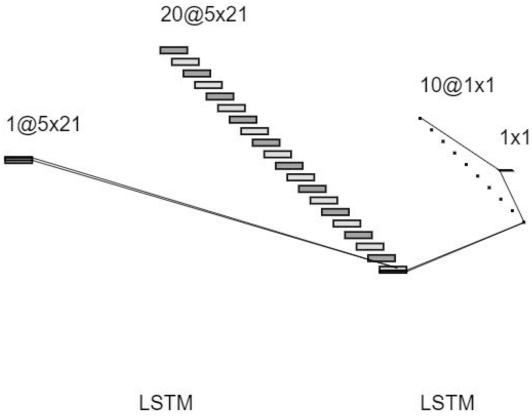


Fig. 5. LSTM Visual Structure

H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian found that other initialization techniques were not appropriate for the ReLu activation functions, they suggested a technique now commonly referred to as ”He initialization” (He normal) [6]. The initial weights are randomly sampled from a standard normal distribution and multiplied by $\frac{\sqrt{2}}{\sqrt{fanin}}$, where *fanin* is the number of incoming connections for the given unit. The bias weights are initialized to zero.

A linear function was employed on the output layer of the regression NNs, since it is only one node outputting a set of granular continuous values between $\{0,...,475.82\}$. A softmax function was employed on the output layer of the classification NN. A softmax function turns the vector numbers into probabilities that sum to one, to ensure that the maximum probability connected to a class is predicted.

c) Metrics And Loss Function: The mean square error (MSE) function is the most widely used in gradient descent optimization for regression tasks. Equation (2) shows how the loss is calculated, where P_T represents the number of patterns in the training set and J the number of output units. The root of the mean square error (RMSE) was used to assess the performance of the regression NN.

$$MSE = \frac{\sum_{p=1}^{P_T} \sum_{j=1}^{J} (t_{jp} - y_{jp})^2}{P_T J} \qquad (2)$$

Cross-entropy is the default loss function used for multi-class classification problems [5]. Cross-entropy calculate the average error or surprisingness between the target and predicted probability distributions for all the classes in the output. Equation (3) shows how the loss is calculated, where P_T represents the number of patterns in the training set and J the number of output units. Although classification accuracy is usually used as the performance metric for classification problems, this is not deemed appropriate for imbalanced classification problems. Since imbalanced sets are dominated by a majority class, the classification accuracy will only be a representation of the majority class accuracy. Precision is an evaluation metric that quantifies the number of correct predictions per class and takes the average of the classes [8]. Even though attempts were made in the pre-processing phase to balance out the classes, the use of precision as the evaluation metric was still deemed necessary.

$$E_T = -\frac{\sum_{p=1}^{P_T} \sum_{j=1}^{J} t_{jp} ln y_{jp} + (1 - t_{jp}) ln(1 - y_{jp})}{P_T J} \qquad (3)$$

d) Learning Approach: The mini-batch learning approach was used since this is most frequently used in deep learning NNs and is a combination of both the stochastic and batch training approaches. The average gradient over a subset of patterns of batch size *100* is calculated. The weights are updated after each mini-batch for *100* epochs. An epoch represents one cycle through the training set. The epoch and batch size were chosen through trail-and-error.
This approach is less computationally heavy than the batch training approach, where weights are updated after each epoch. It is also more reliable than the stochastic approach, where updates happen after each pattern, which is bound to cause fluctuation.

A successful optimization algorithm called Adam was used. This method uses gradient descent backpropagation, but computes separate adaptive learning rates for each parameter from estimates of first and second moments of the gradients. A learning rate (η) is defined as a hyperparameter that controls how much the model with each weight should update [5]. Adam uses a initial default learning rate of 0.01.

The LSTM model updates weights by using backpropagation through time (BPTT). Essentially, BPTT computes a running average of all past gradients. The LSTM unrolls the recurrent relations and the Adam algorithm is applied in the same way [7].

IV. RESEARCH RESULTS

The NNs were trained with the parameters as discussed in the previous sections. The github link to all the models created on python, can be found in the title section of this paper. Ten simulations were run for each model to assimilate the general stochastic nature of NNs. The results for each of the models will be discussed in the following sub-sections.

A. *Convolutional Neural Network (Regression)*

Figure 6 and 7 depicts a single simulation of the MSE and RMSE over the number of epochs. From these graphs it can be seen that the model converges and displays a good generalization. Generalization can be measured as the difference between training and validation loss. This essentially means that other input data of the same nature and distribution, that goes through the same data preparation steps, should perform the same. The mean and standard deviation of the evaluated RMSE simulations where rescaled by the MinMax inverse transformation function, to present a realistic idea of how the model is performing. The re-scaled error is 59.5147 +/- 1.8478. Being out with roughly 60PM2.5 can be interpreted as follows. Referring again at the hazardous levels provided in Table VII, each level has a different range. This range can be viewed as a grace span with which the prediction is allowed to be off. The grace span for the healthier levels is lower than that of the more hazardous levels. This means the model predicts very poorly for the healthier levels, but for unhealthy levels the model can still be informative, even though it might be 60PM2.5 off. For the goal of this model the importance sway on the higher PM2.5 predictions. It can thus be argued that this model performs reasonably well.

TABLE VII
CLASSES OF HAZARDOUS LEVELS OF PM2.5

Health concern	$\mathbf{PM_{2.5}}$ $(\mu\mathbf{gm^{-3}})$	**Difference**
Good	0 -12	12
Moderate	13-35	22
Unhealthy for sensitive groups	36-55	19
Unhealthy	56-150	94
Very unhealthy	151-250	99
Hazardous	250 +	Infinite

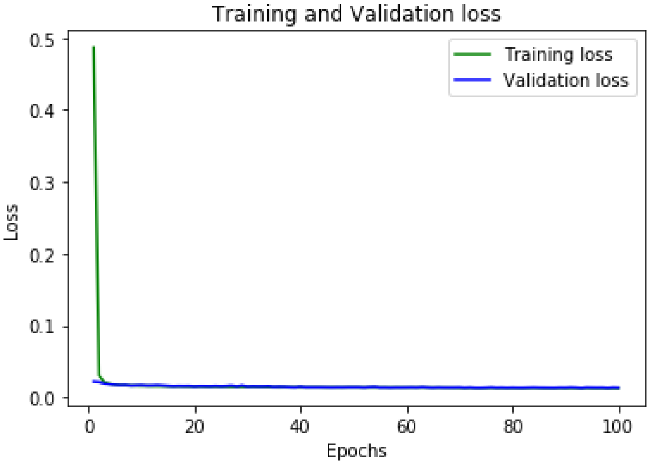


Fig. 6. Mean square error over number of epochs

However, further investigation was done to see if the model accurately predicts the extreme values higher than 150PM2.5. Figure 8 displays the comparison between target values higher than 150PM2.5 and the predicted values. It is clear that even though SMOGN was used to mitigate the imbalanced input sets, the model still did not learn the minority data points and represents false negatives. For the goal of the model, which is to predict the air quality so that precautionary actions can be taken for the community's health, the CNN regression model is inadequate.

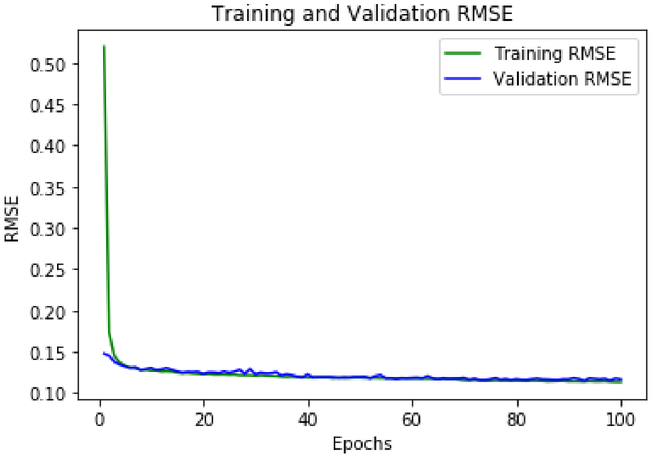


Fig. 7. Root mean square error over number of epochs

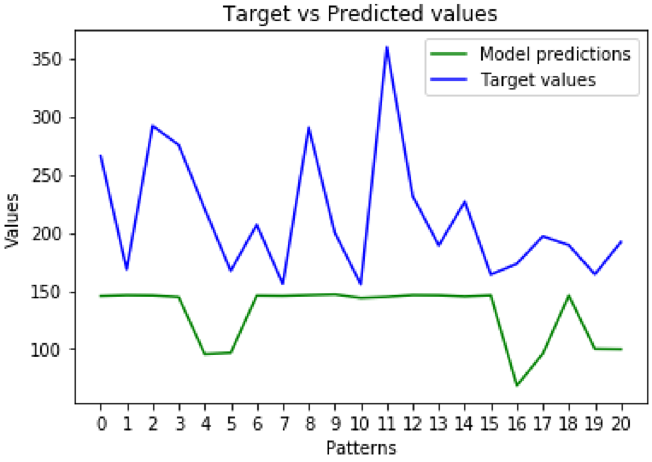


Fig. 8. Predictions for high air quality values

B. *Convolutional Neural Network (Classification)*

Figure 9 depicts a single simulation of the cross-entropy error over the number of epochs. From this graph it can be seen that the model converges with a generalization error of 0.1, which is not too large. Figure 10 gives an example of how the model predicts. From this figure it can be seen that the model does in fact predict minority classes, which is an indication that the use of SMOTE to mitigate the class imbalance was effective. However, the mean precision results for the simulation is 21.757% +/- 5.904%. The goal would be to get the precision as close to 100% as possible. The hypothesis for why the classification CNN performs poorly, could be because the classifier views the classes as independent, like one would predict an image as either a cat or a dog. It can be informative for the NN to learn how far exactly from the target value it predicted, in order for it to improve it's predictions. Thus, viewing this as an regression problem is expected to give better results.

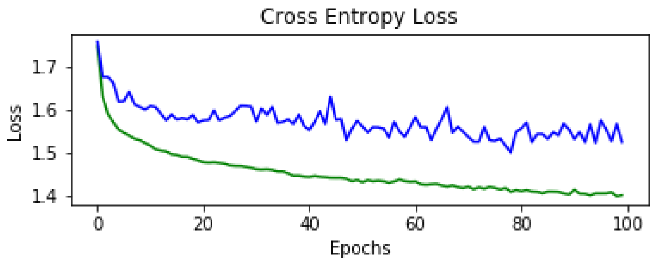


Fig. 9. Cross-entropy error over number of epochs

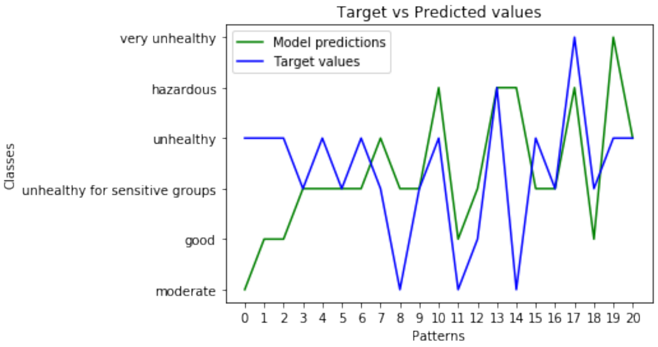


Fig. 10. Target values vs predicted values

C. Long-Short Term Memory Model (Regression)

Figure 11 and 12 depicts a single simulation of the MSE and RMSE over the number of epochs. From these graphs it can be seen that the model converges with a generalization error of 0.011 which is relatively small. The rescaled RMSE results for the simulations is 56.9128 +/- 2.1459. The same issue from predicting the minority data point can be seen in Fig. 13. The same argument would apply for this model as for the CNN (regression) model, since the same imbalanced dataset is used.

D. Comparative Study

For the purpose of having a common metric with which to compare the three models, precision evaluation was used on all three. Table VIII holds these comparison. Although the LSTM compares the best out of the three, this model is inadequate for the goal. This is due to the fact that the results depict a false positive. In general none of the three different models performed well.

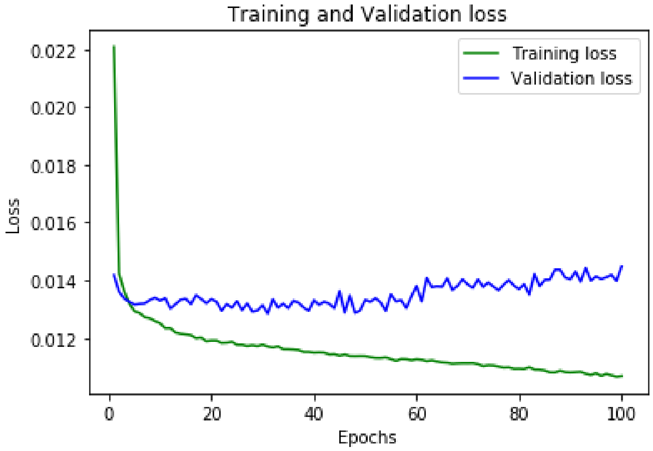


Fig. 11. Mean square error over number of epochs

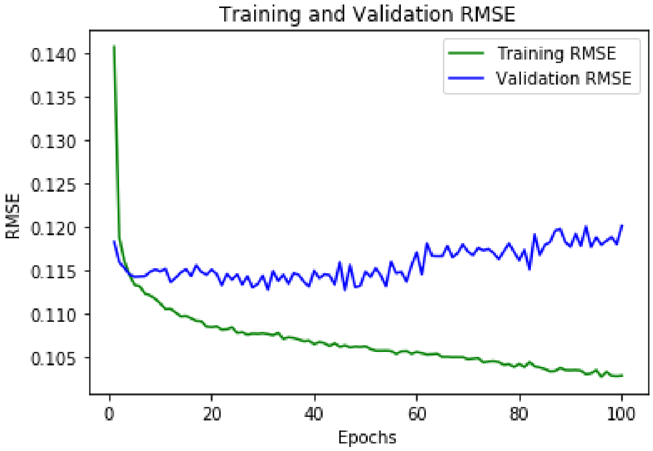


Fig. 12. Root mean square error over number of epochs

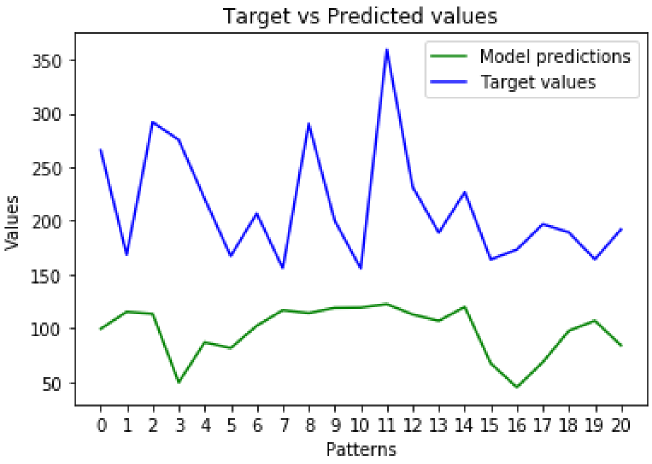


Fig. 13. Predictions for high air quality values

TABLE VIII
PRECISION RESULTS

Model	**Precision**
CNN (regression)	33.880%
CNN (classification)	21.757%
LSTM	43.275 %

V. CONCLUSION

From the results discussion it was realised that building a NN that predicts air quality in levels is limiting. A classifier NN views the target classes as independent. This means that the model does not take into consideration that an incorrect prediction of a *'very unhealthy'* level instead of a *'hazardous'* level is closer to correct that an incorrect prediction of *'good'*. This limitation is proven by the poor performance score of the CNN classifier model.

For regression tasks, there is a lack of research done for problems with imbalanced continues target values [4]. Where the most relevant cases are the most rare, such as in this report, two critical issues where identified. Firstly, the observed performance is degraded since the error of the minority cases is overpowered by the majority cases. Secondly, the NN does not have enough of the minority cases to learn from and thus disregards the most relevant data. A recently proposed technique called SMOGN was used to mitigate the imbalance, however this technique proved to be ineffective. This is thus identified as a critical area of further research to improve the performance of these models.

With the regression tasks the LSTM performs better, with a precision of 43.275%. This could be because recurrent neural networks are designed for cases with temporal data, such as this, and the memory function is more effective than the filters from the CNN.

Although these models are not adequate to be used, it cannot be concluded that there exists no relationship between the data collected and the air quality values. Rather the insight gained from this report can be leveraged for further work, such as improving the data imbalance and tuning more parameters. After the improvements have been made however it would be expected that the LSTM model will perform the best.

The 'AirQo Ugandan Air Quality Forecast Challenge' keeps a leadership board of model performances [1]. The best performing score last stood at an error of 31.9702. Comparing it to the LSTM error of 56.9128 +/- 2.1459 indicates that significantly better performing models for this problem can be achieved.

REFERENCES

[1] Zindi, "AirQo Ugandan air quality forecast challenge,", June 2020. https://zindi.africa/competitions/airqo-ugandan-air-quality-forecast-challenge

[2] J Brownlee, "Deep learning for time series forecasting: Predict the future woth MPLs, CNNs and LSTMs in Python,", Augustus 2018.

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique,", June 2002.

[4] P. Branco, L. Torgo, R. P. Ribeiro , "SMOGN: A pre-processing approach for imbalanced regression,", 2017.

[5] J. Brownlee, "Better deep learning: Train faster, reduce overfitting and make better predictions,", December 2018.

[6] H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet Classification," February 2015.

[7] J. Brownlee, "Long short-term memory networks with python,", July 2017.

[8] J. Brownlee, "Imbalanced classification with python: Choose better metrics, balance skewed classes, and apply cost-sensitive learning,", January 2020.