Le météo: The use of Recurrent Neural Networks to explore climate prediction and uncover the caveats of machine learning

G. Mehan[1]

[1]**UCL Department of Physics and Astronomy,**
Gower Street, Bloomsbury, London WC1E 6BT, United Kingdom

ABSTRACT

Predicting the climate has been a long standing human endeavour. With the introduction of machine learning, this has become somewhat of a different task. Long Short-Term Memory is a popular neural network used to train temporal sequences such as the modelling of the weather. With this, many models were generated and investigations carried out as to the many different ways machine learning can be used in meteorology. Data was supplied by the GCOS Surface Network, and primarily the climate in *Toulouse-Blagnac, France* was considered as the investigative station. Though many of the models ended up predicting a convergent series, a number of possible processes were identified as their causes, and solutions to it made. Specifically relating to the Vanishing Gradient Problem and the 'Dying ReLU'. Thus proving the difficulty in such a task.

Keywords: Recurrent Neural Networks, Long Short-term Memory, Vanishing Gradient Problem, Dying ReLU, Causality

INTRODUCTION

Since the introduction of Machine Learning into our everyday lives, we often under-appreciate and are almost unaware how many processes we come across are based on a Machine Learning Model. Machine Learning is simply the idea of using a computer algorithm to perform a given task without explicit instruction - it learns for itself much like a human baby.

Humans have historically tried to predict the weather and now we have become obsessed with near-perfect predictions so as to not ruin our daily and future plans. What allows us to achieve such high accuracy is indeed the use of Machine Learning in meteorology.

The Global Historical Climatology Network (GHCN) is major database of climate summaries from many stations across the globe, and is often a source to train many climate prediction models. Further, the GCOS Surface Network (GSN) is a subset of these stations that upholds good data taking practice and ensures high-quality of data from a trusted, credible and accountable source. Our goal is indeed to use the GSN to explore the world of climate prediction, and not only to predict the weather, but to discover other ways in which this may help us, such as our quest to

mitigate the effects of climate change.

To highlight explicitly our aims:

- Using data from a given station, we may predict the future weather (as we define as the maximum/minimum temperature and the precipitation) in a number of scenarios.
- In light of climate change, we may unearth the hottest part of the world in the future.
- Explore the many challenges faced in machine learning and such a task as given.

THE DATASET

Within the GSN, we have access to 991 stations across 221 countries. Figure 1 highlights the locations of these stations on a map. For simplicity, I have chosen *Toulous-Blagnac Airport, Toulouse, France*, as our primary station for investigation given its mostly complete dataset, and its location within an urban area of south-west France whose climate is considered 'Cfa: Humid subtropical climates' by the Köppen climate classification (1). This indicates a varied climate where extremes are far and few, and hence more predictable given still that season changes are distinct.

Upon initial inspection, the dataset spans between 01-01-1947 and 13-11-2020. Missing data had been interpolated with the python package *pandas*. Figure 2 contains plots of the datasets for the maximum/minimum temperature and the precipitation.

Having obtained the datasets, we may now proceed to generating the models.

MACHINE LEARNING: THE FINDINGS

Predicting the weather in October and November 2020 using data available up until September 2020

A Recurrent Neural Network (RNN) is one which processes sequential data. Further, a Long Short-Term Memory network (LSTM) is a type of RNN that uses a feedback connection to 'remember' values over a given time series. This is most useful for applications consisting of a temporal sequence. In our case, how the weather changes over time. As such, data must be preprocessed to a time series whose values are shifted within a so-called 'window', which can be likened to how far into the past the network may see into in order to base its prediction off, with a provided offset to label how far into the future we would like to predict.

The way we may approach our available data is to train our network on all the data. This may seem counter intuitive given that traditional machine learning projects are tested on known data.

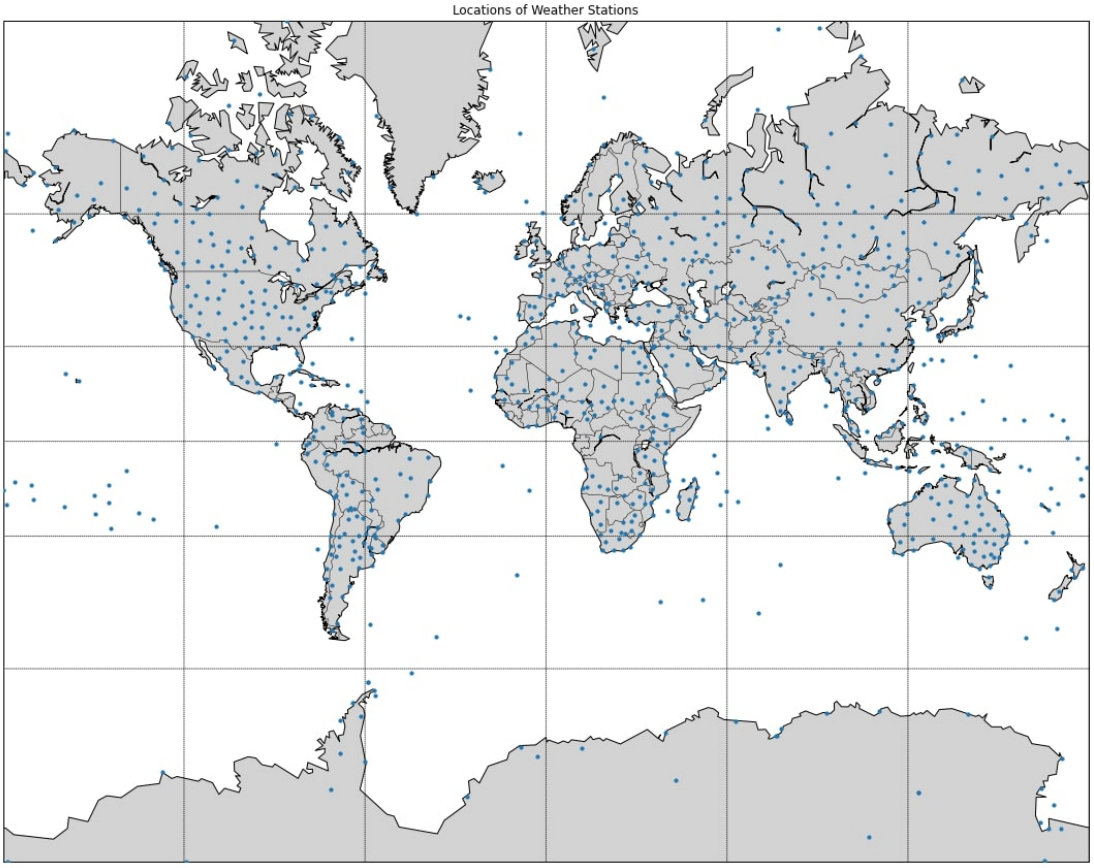

Figure 1. A Basemap Mercator projection of the location of all stations within the GCOS Surface Network.

However, it is my opinion this is not suitable for our purposes given that we do not care for past data, but rather would like to predict future data. To explain further, if we were to push through a test time series dataset into our trained network, then technically, the model will 'see' the correct answers for every value due to the window. So as a result, rather than predict the weather day after day, in this case, it would predict the weather given the true weather recorded. As a result, initial investigations revealed that this produced, unsurprisingly accurate predictions, but of course is highly unrepresentative of the data it would be fed in real-life. In this situation, we may not accurately predict data further than tomorrow given how we chose to test the model each time feeding the correct weather for the past window over an expanded time series. Therefore, we may take a neo approach to testing our model. Having set aside the data we have for October and November 2020, we train our model on all available data with a window size of 31 days, and an offset of 0. In other words, we would like our network to predict the weather for the next day, given the weather for the previous month.

Having trained the model on all the data, aside from October and November 2020 - a separate

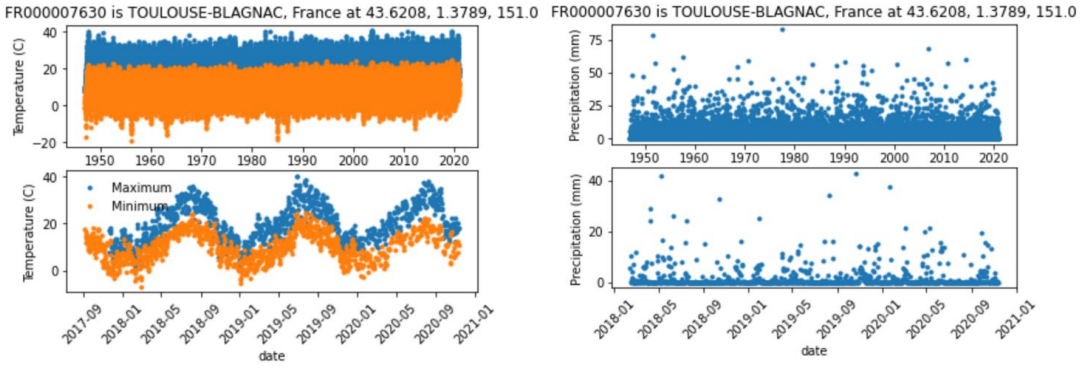

Figure 2. On the left is a plot of the Maximum/Minimum temperature and on the right the precipitation for *Toulouse-Blagnac Airport*, recorded on a mostly daily basis for all the data available from the Global Historical Climatology Network.

model for each the maximum and minimum temperature - the last available item for data training was then fed into the network to produce the prediction for 01-10-2020. This was then fed back into the network, alongside the data for the past 31 days, to produce a prediction for 02-10-2020 and so forth. After 31 iterations of such a feedback, the network was making predictions based off its previous predictions - which is precisely how the weather maybe predicted for the future. Unfortunately, the model generated had very poor accuracy, despite very low loss, given that such a method of prediction often leads to convergence of such a value. In other words, after feeding through a couple of times, the network ended up converging on a given temperature value which it kept reproducing given that the value predicted is just fed through the network again. It is often quite difficult to avoid such convergence in machine learning given its volatility and is quite an often occurrence. Figure 3 documents the predictions for the maximum/minimum Temperature for October and November 2020, alongside the data we had already up until 13-11-2020, using this 'feedback' method described above. Just to prove that it is the feedback why the network performs badly, Figure 4 documents the prediction for the past 100 days of the latest data available, given that it does not predict further than next day, and given the true data for the past 7 days, it performs exceptionally well in relative terms, as expected. However, to emphasise, this is not that useful given our 'thirst' for more extreme weather predictions and this kind of use for our model is quite basic and unfulfilling. In this scenario, however, we may compare the error of such a prediction - predicting the next day given the data for the past 31 days, with the idea that *the weather tomorrow will be exactly the same as the weather today*. Figure 5 documents this error specifically. Unsurprisingly, our model predicts better than the assumed rule. Furthermore, the assuming statement led to more predictions with an error greater than 1 in comparison to

our model, and even a few in the tens, however these have been excluded from Figure 5 for readability due to scaling issues. This shows that such a statement is not really adaptive and is unable to 'predict' outliers in trends due to extreme weather. Our model on the other hand at least tries to accomplish this. However, as mentioned, the true purpose of climate modelling does not just lie in predicting tomorrows weather and with that our assumption may not be used for other predictions, nor can many more assumptions be made for other purpose with regards to climate prediction that are meaningful enough, especially in a world where the climate is forever changing at a rapid rate recently due to climate change and other processes.

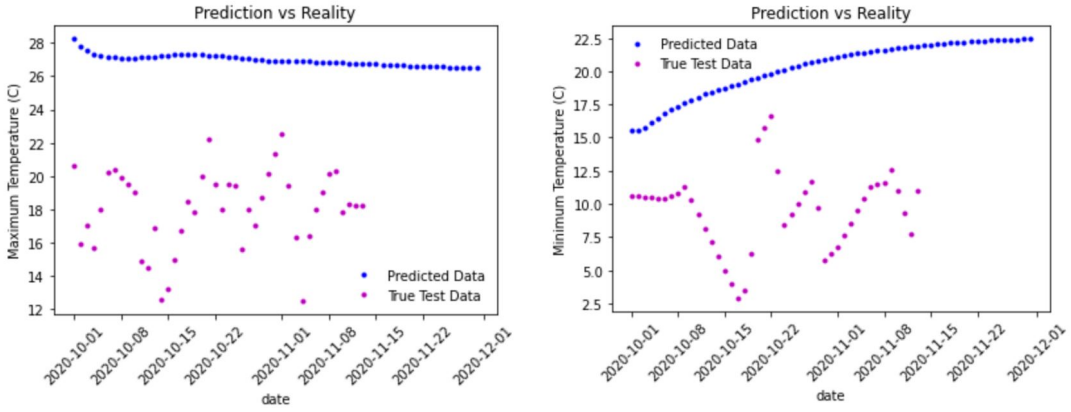

Figure 3. On the left and right are plots of the predictions for the maximum and minimum temperatures respectively for October and November 2020, using a feedback loop to feed our network its own predictions from which it bases it next predictions off.

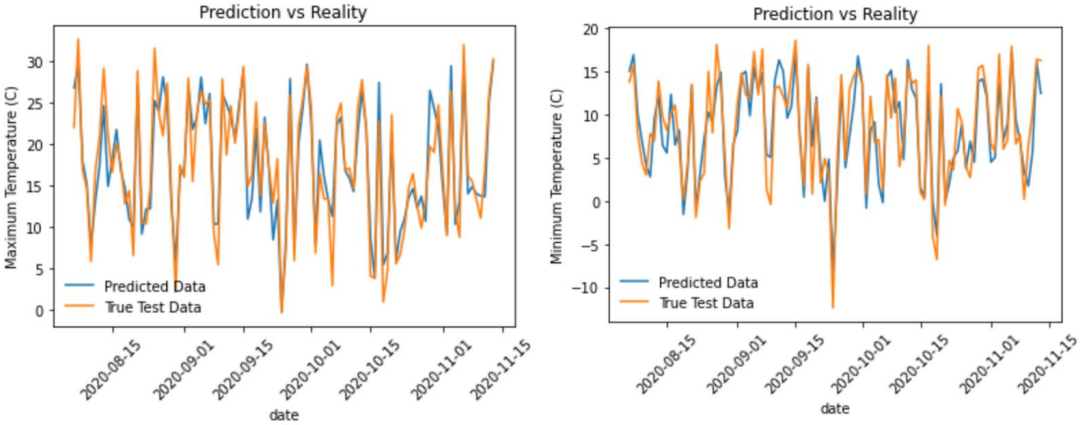

Figure 4. On the left and right are plots of the predictions for the maximum and minimum temperatures respectively for the past 100 days from 30/09/2020 given the true temperatures from the previous 31 days.

Alternatively, another more traditional approach to predicting the maximum and minimum

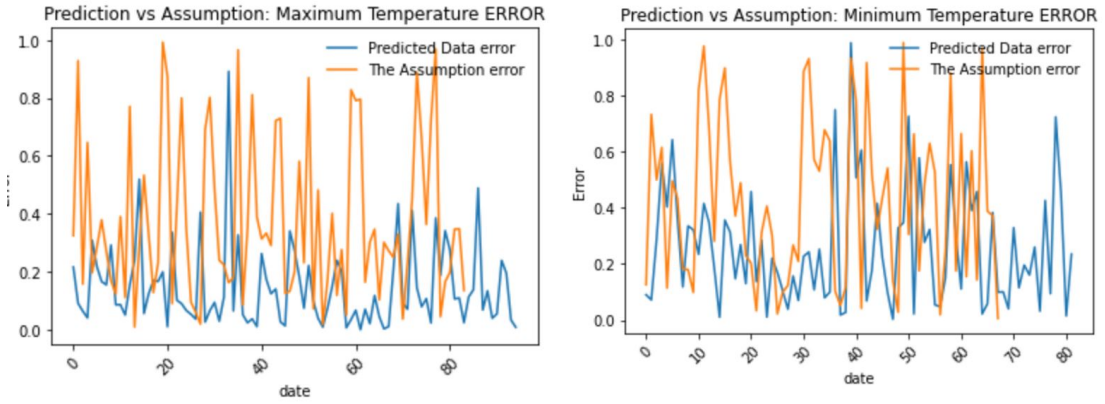

Figure 5. On the left and right are plots of the prediction errors for our model and the statement 'the weather tomorrow will be exactly the same as today'. Any errors above 1 have been excluded given that only a handful of predictions were that awful.

temperature maybe taken to train and test the model that does not involve 'feeding' the network its own predictions. This may come about through training a model that is able to predict two months in advance, in other words, alter to the offset to 61 days. And after decreasing the window size to 7 days, one was able to predict the weather in October and November 2020, with data from August and September 2020, respectively. To comment explicitly, data from August was used to predict October, and likewise data from September to predict November. Figure 6 graphs our model prediction alongside known temperatures, and as expected, the predictions were quite off given the large offset. Nonetheless, the values are reasonable and do not converge as with 'feeding' the network its own predictions.

Figure 6. On the left and right are predictions for the maximum and minimum temperatures respectively for October and November 2020. Though quite reasonable, the predictions fail to pinpoint a greater accuracy normally required to produce weather forecasts.

As far as precipitation goes, it is my belief that it is pointless to predict the amount of rainfall

everyday given the rarity for sizeable rainfall, and how little rain occurs on most days. Having tried to accomplish this, it was found that it was as a result, very difficult to normalize such data given that even if data was normalized to between 0 and 1, the mean was very close to zero, and as a result, only values close to zero were predicted, neglecting the rare events when rain does occur. This completely defeats the purpose of such a model as yes it is accurate in predicting the rain, given that rain does not occur often, but we really need it to tell us when rain DOES occur so we may be better prepared. And so, alternatively, it may be useful instead to normalize the data and treat this as a binary classification problem. We want the probability of rainfall occurring on a given day - which can be seen as much more useful. Having sorted the data to either 0 or 1, with 1 being rain having occurred and 0 as no rain at all, a Binary Cross Entropy loss function was employed. Figure 7 is a histogram plot of our normalized precipitation data. However, much like my previous models, this model decided to converge to around 0.3. Which, once again, defeats the purpose of the model. Figure 8 documents these convergent predictions for rainfall between October and November 2020, alongside the existing data we had access to for these dates.

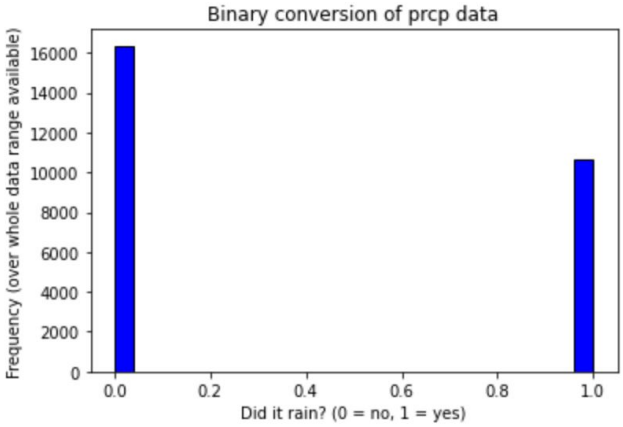

Figure 7. A histogram of the normalized precipitation data given we want to predict the probability of whether or not it will rain on a given day

However, likewise with the temperatures, we may test our model for the past 100 days and each time feeding it the 'correct' data for the past 31 days. Figure 9 documents these predictions for the rainfall as well as the true outcome of the event. To comment specifically, if the prediction probability was greater than 0.5, then this indicates a prediction that it will rain, whereas a prediction less than 0.5 indicates no rain. Our model predictions were scaled in this manner and plotted in Figure 9. To truly compare the accuracy we may plot the accuracy, as well as for comparison once again using our assumption that *the weather tomorrow will be the same as*

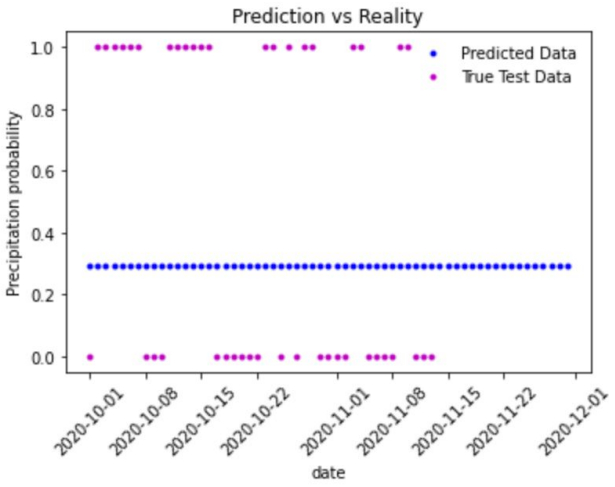

Figure 8. On the left and right are plots of the prediction for the precipitation for October and November 2020 as well as the actual test data.

today, as documented in Figure 10. The errors for our model should mostly lie at zero, however it seems like a 50:50 split indicating at this point at network might as well be as random as a coin toss in deciding whether it rains or not.

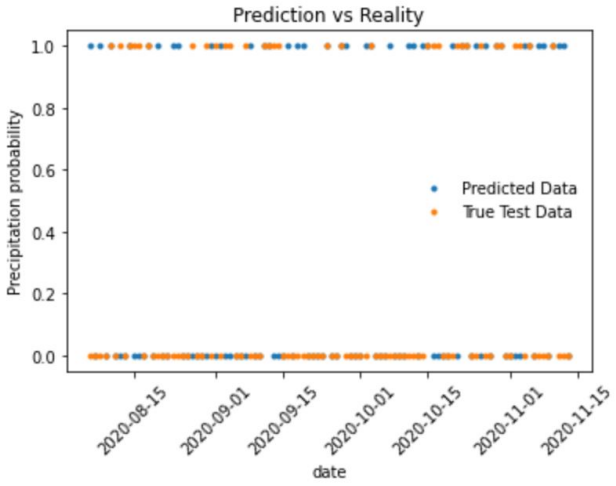

Figure 9. Predictions for the precipitation were made for the past 100 days from 30/09/2020 given the true outcome of the event from the previous 31 days. The probabilities were normalized to predict rainfall or not for a more meaningful comparison.

Predicting the weather a year in advance

Moving on, we may now shift focus into predicting the weather a year in advance. We may firstly approach the problem by calculating the monthly average, then training and testing our model similarly to our 'feedback' convention for predicting the weather in October and November 2020. This time however, we aim to predict the monthly maximum and minimum temperatures for

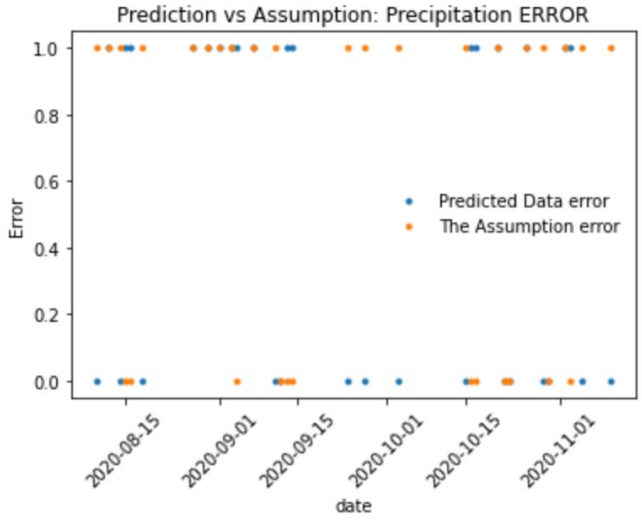

Figure 10. Plots of the prediction errors for our model regarding the probability of precipitation were made, and the statement 'the weather tomorrow will be exactly the same as today'.

October 2020 to September 2021. We have released the precipitation attribute given that now when we take an average, there is not much meaningful data we may extract from the rainfall given our limited resources, time, and undoubtedly lack of expertise. Nonetheless, interestingly enough, our model has implemented what seems to be a sinusoidal model when asked to make predictions for the next year, as shown in Figure 11. For the month of October and November, whose data we have, this seems to be reasonably accurate. And this makes sense that such a parameter would follow a trigonometric function (2).

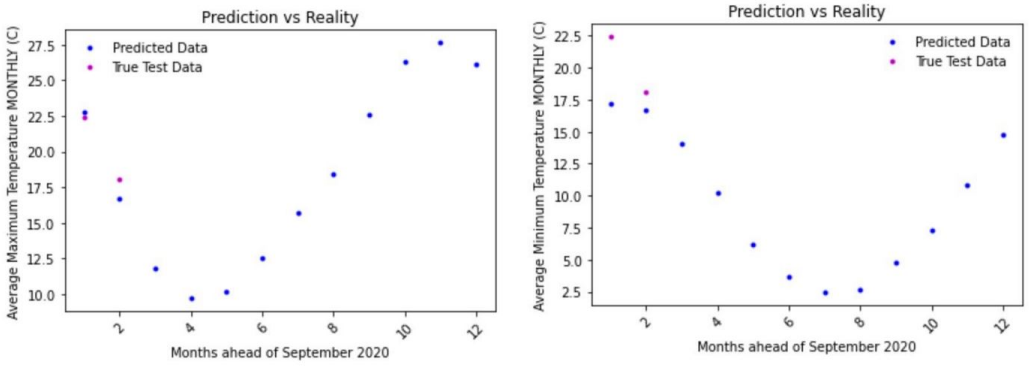

Figure 11. On the left and right are plots of the predictions for the maximum and minimum temperatures respectively for the next year from 30/09/2020 as a monthly average.

Finding the hottest part of the world in 20 years

As a result of climate change, the world has been warming at an increasingly exponential rate. It is now more important than ever to be predict such a phenomenon and do our every best to mitigate against it occurring, otherwise, face devastating consequences.

Nonetheless, one approach could be to train a separate model for each of the 991 GSN stations, and through a feedback loop, predict the maximum yearly temperature year after year. However, training 991 model isn't exactly efficient nor even possible given my current hardware and time available. Therefore, instead, I decided to train one model on data from *Toulouse-Blagnac*, and test on all stations. A window size of 5 (years) was chosen arbitrarily, and to predict the next year requires an offset of zero. Having completed this, I was able to produce a prediction for the maximum yearly temperature in 20 years time for each station, which has been plotted in Figure 12. Quite disappointingly, much like my other 'feedback' models, the model decided to converge and predict the same value for all, but a few stations. This could quite well be due to having trained on only one station, which only has training data in the tens of samples, and from undertraining with the wrong parameters (epochs, batch size,...).

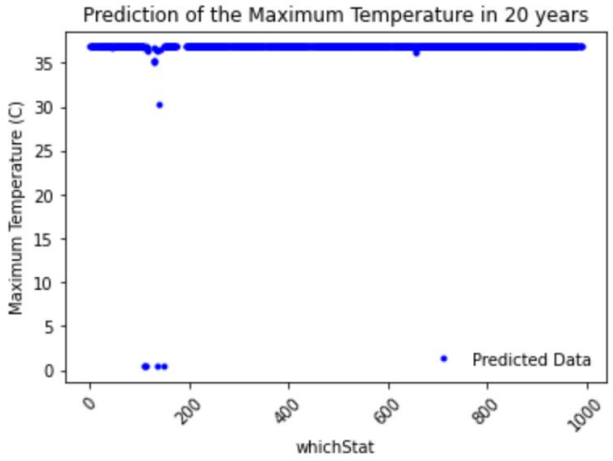

Figure 12. The predictions for the maximum yearly temperature in 20 years time for all 991 GSN stations. whichStat is an arbitrary number unique for each station.

MACHINE LEARNING: A DISCUSSION

The failures of our machine learning models

Disappointingly enough, most of our models had decided to converge on values when asked to predict anything. As suspected, this is due to the implementation of a 'feedback loop' to predict values over a given time series. To reiterate why this was executed, I would like to highlight that

when it comes to most machine learning models, we feed in a test input (which is something that is for definite and known), and expect a prediction of a single or multiple attributes. In our instance, this can not be the case given that firstly, we are implementing a model that predicts for a given time series, and how we decided to train our models (offset, window size,...) meant that if we require a prediction further than offset allows us to give, we must feed the network its own predictions until we reach our desired destination. The test data is for definite for only one instance of the first prediction given that you probably would have data for these past instances that make up the test dataset. That is unless you would like to predict the weather in the past and are satisfied with giving the model the correct test data every time (given that we have this capacity) and of course this produces a very accurate model with not a lot of training, as shown by our tests for the past 100 days. But then again, I personally do not see the point in this. This is because we are trying to predict the future, what worth is it testing the model on basic scenarios that hardly test its limit, nor is representative of how the model will be used in the real world. Of course I could just train a model with an offset equal to how far into the future I would like to predict (thus not requiring an implementation of this feedback loop), as we did show, but this is not feasible nor accurate for predictions in the long-term, we want to be able to use all the data available, especially given the standards of accuracy required by many climate models in this day and age, and a model that is agile in allowing us to predict however far into the future we would like, done through this feedback loop.

The Vanishing Gradient Problem

To refer back to this convergence, I would like to bring 'The Vanishing Gradient Problem' to the table here. This a problem that most machine learning experts are all but familiar with when implementing a Recurrent Neural Network. Primarily though, I would highlight this is a problem encountered during network training, however, with the implementation of a feedback loop as previously discussed, one can liken such a method to the back-propagation of a neural network. And as as result, this problem arises. The problem has been described as arising due to the 'error vanishing as it flows back', (3). In this instance the decaying gradient causes a rough approximation of the true gradient. But I would once again like to highlight, the reference to gradients only applies to training. In this instance our model has already trained successfully. I would like to stress I merely bring up the topic of the Vanishing Gradient Problem to try to explain why when we feed our network its own predictions, do we obtain such a convergence and would like to apologise if this is highly inaccurate and extremely speculative. Despite this, we

can think of the gradients as correlating in some form to the prediction the model makes. During back-propagation of such a model, the algorithm processes these weights down the layer. In our 'instance', we feed the models own predictions through the model but in a forward direction - forward propagation. And as so the gradient would converge, that to on the wrong solution, our model settles on predicting the same value, and is a loop that you cannot escape since this feedback loop relies on good predictions being made earlier on. If this is not the case, then the rest of the predictions will become terrible as well since each successive prediction is dependent on the previous prediction as with our 'window' analogy. It probably is a good idea to leave our explanation here given that such an explanation may prove wrong. Nonetheless, this should serve as some sort of explanation of what I believe is occurring when the model decides to converge on a given value, may that be temperature or precipitation.

Dying ReLU

Another often frustrating problem machine learning experts encounter, and relating to the Vanishing Gradient Problem is the 'Dying ReLU'. This explanation offers the choice of activation functions as a solution. Firstly, I must explain that often, the Vanishing Gradient Problem arises due to the use of activation functions such as sigmoid. The Rectified Linear Unit (ReLU) activation function offers a solution to this most often than not, and as such we have used ReLU for all our activation functions apart from the last layer. However, privy to this, even ReLU suffers from this problem and has been described as 'a kind of vanishing gradient' itself quite surprisingly (4). Essentially, though ReLU is regarded as the gold standard of activation functions for most problems in machine learning, given that it solves many of the problems encountered and mentioned, is quite fragile and if careful attention is not paid to the model, will more often than not have the weights of its neurons converge to zero quite quickly. This can be a result of a large gradient flowing throughout the model. For reference, the ReLU function has been plotted in Figure 13. There are three discussed solutions (4) to this problem of which two are ones that can be implemented easier than the third. One is to 'modify the network architecture', in other words alter parameters such as the number of layers, number of neurons, epoch and batch sizes, and specifically in our case, the offset and window sizes. The second is to introduce 'additional training steps', or employ a better fitted normalization technique, given that as mentioned for precipitation, normalization was quite difficult and for such a variate dataset, is expected to be. Specifically, batch normalization is highly recommended by Lu et al (4) and would involve an extra layer.

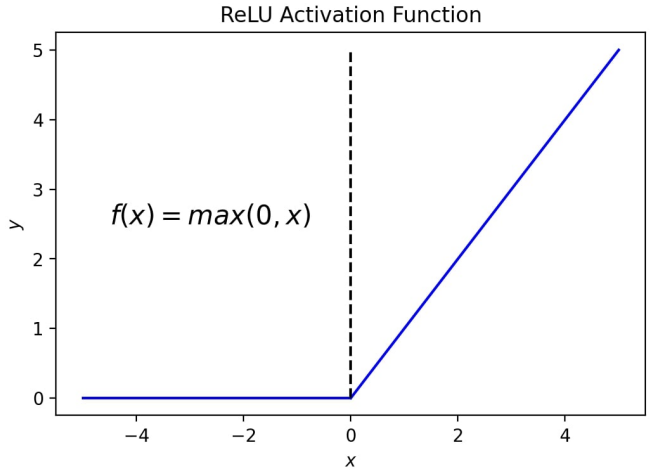

Figure 13. The Rectified Linear Unit (ReLU) Activation Function plotted.

The anomaly

If we go back to Figure 11 when we chose to predict the weather for the next year as a *monthly* average, we avoid such convergence aforementioned when predicting the daily climate for October and November 2020, and yet we are still able to implement this feedback loop with credible predictions. If I may, it is possible to offer an explanation to this, and it may just have to do with our parameters we set for training our model, i.e. the window size, the offset, the epoch and the batch size. To quote the values used, offset: 0, window size: 6 (months), epoch: 100, and batch size: 100. If we also refer back Figure 2, specifically the temperature data available and plotted for the years 2017 to 2020, one can clearly deduce that from one consecutive peak temperature to the next, the trend decreases linearly, and then increases with the gradient at the same order of magnitude. A so called 'V' shaped trend. Reading the time scale, we see that this time period maybe established as a year, or more interestingly enough each section of increase or decrease is roughly 6 months long - the same as our window size. In comparison to our model used to predict the daily climate in October and November 2020, our window size was 31 (days). With the epoch and batch sizes used, it would be very difficult for our model to learn the micro trends that exist on a day-to-day basis, and thus when given 31 days, it would be next to impossible to learn the many trends that may exist, but to us are invisible when we plot the data, especially given how we define the model to learn with the data. Of course, increasing the epochs, altering the batch size to an optimal value, as well as setting the window size to a more suitable value so as to limit these trends together, we may end up with a better performing model, as with our monthly prediction. Given that even we can clearly see the trends in the

monthly data, again referring back to Figure 2, it would be very easy for our model to do so as well, hence explaining why it performs well, even when indeed we still employ a 'feedback loop', and avoiding such convergence of a value. This affirms our convention to liken the failures (and successes) of our models to the 'Dying ReLU' problem and how modifying the network architecture, recommended by Lu et al (4), does really make a difference.

CONCLUSIONS

To write a concluding statement, though most of our models have (mostly) failed to predict the climate, as set out as our primary aim, I have discovered that quite often than not, machine learning will present a number of difficulties. To predict the climate with the use of machine learning, one does not have to be a meteorological expert, nor an expert in many of the other applications of machine learning, such as Natural Language Processing, healthcare, and indeed physics (though of course knowing a little bit helps), but rather needs to have a developed understanding of how machine learning models work, if one is to have a fighting chance of overcoming these difficult challenges. To comment on the meteorological (physics) side to this project, predicting the climate tomorrow is easy, predicting the climate on given day next month is a lot harder. And this is down to fluctuations in meteorological data that at face-value are seen as random. Machine learning has enabled us to uncover (sort of), these trends without theorising the ecological processes involved and allows us to make predictions we would have never been able to. But it is precisely this 'black box' that frightens me, even if we are able to correctly predict the weather using machine learning, we have no way of knowing HOW the model does so, and even though for applicative purposes it is OK, I find it quite ignorant and inhumane to not let curiosity enable you to want to theorise these process we may have not known about for. This project is a perfect example of this, aforementioned, as a physicist, I have limited knowledge as to the meteorological processes involved in climate data, and if I wish to learn about these, machine learning is not the source from which I may (even though these were not my aims).

To comment specifically regarding our aim to predict the hottest part of the world in 20 year time, even though once again we have failed to do so, such a prediction is vitally, if not more important than predicting tomorrows weather and this is because climate change is such a novel, but extremely important concept, and even though we may not know much about compared with classical meteorology, it is precisely this instance where machine learning, in my opinion, shines given that it provides a foundation for us to learn new things at an increasingly accelerating rate than without it. And is especially important for processes that will, and is impacting us right

now.

Considering the Machine Learning side of things, we most definitely have encountered common problems associated with such a task, and though we have failed to mitigate against them, I believe it has enlightened us into this controversial 'black box' innate within machine learning. We have successfully managed to suggest a number of improvements to our models that for our purposes we have been unable to implement due to hardware constraints, time constraints, and indeed lack of expertise in the field of machine learning. Nonetheless, only through implementation and testing may we discover and learn about these challenges. It is therefore my belief that this aspect has been the most successful aim of this project and has indeed shifted my perspective as to the objective of this experiment. Nonetheless, I would still like to emphasises the importance of understanding how specifically, machine learning maybe used in physics and I do not wish to undermine this either.

To summarise my recommendations for the machine learning models, I believe it is important and imperative to play around with the network architecture as often so within machine learning, such a 'game' is considered as 'trial and error'. Explicitly, the window size plays a huge contribution in allowing our models to identify these (micro) trends, as well as how we may 'drill' in these trends through training with increasing the epoch size and appropriately selecting a batch size. Further to this, batch normalization may additionally aid with avoiding convergence of predictions.

Additionally, I would like to highlight additional investigations I was unable to undertake, but would've been desirable to do so. Causality within our ecosystem is something that leaves many dumbfounded. The idea that the climate in one region of the planet may unseemly affect that on the other side of the planet. To comment on a famous example, the wildfires of Australia and the flooding in eastern Africa in 2019, was no coincidence at all. The 'Indian Ocean dipole' (5) is a well documented effect that relates to ocean currents and weather systems far apart from each other to become connected and coupled. One wonders if such a mechanism exists today, then so may be it that many more do too. And machine learning may enable us to uncover these relationships between two given climates that may seem unseemly unrelated. Such an investigation would be carried out by training a machine learning model on one station, and allowing it to predict the weather in another station. And if it proves accurate, then this would incite external investigations into why so to it that a machine learning model is able to do so. Specifically, comparing climates with the same Köppen climate classification (1) but far apart may serve as a starting point.

REFERENCES

[1] Kottek, Markus Grieser, Jürgen Beck, Christoph Rudolf, Bruno Rubel, Franz. (2006). *World Map of the Köppen-Geiger Climate Classification Updated*. Meteorologische Zeitschrift. 15. 259-263. 10.1127/0941-2948/2006/0130.

[2] Per Ottestad. (1986) *Time-series described by trigonometric functions and the possibility of acquiring reliable forecasts for climatic and other biospheric variables*, Journal of Interdisciplinary Cycle Research, 17:1, 29-49, DOI: 10.1080/09291018609359895

[3] Hochreiter, Sepp. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 6. 107-116. 10.1142/S0218488598000094.

[4] Lu, LU. (2020). *Dying ReLU and Initialization: Theory and Numerical Examples*. Communications in Computational Physics. DOI: 10.4208/cicp.oa-2020-0165.

[5] Yan Du, Yuhong Zhang, Lian-Yi Zhang, Tomoki Tozuka, Benjamin Ng, Wenju Cai, (2020). *Thermocline Warming Induced Extreme Indian Ocean Dipole in 2019*, Geophysical Research Letters, 10.1029/2020GL090079, 47, 18,