An Analysis of "Deep learning Methods for Forecasting COVID-19 Time-Series Data: A Comparative Study"

Gregory Dane Peterson
pet03356@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

Abstract

In a study titled "Deep Learning Methods for Forecasting COVID-19 Time-Series Data: A Comparative Study" five deep learning methods forecast a time-series of COVID-19 cases. The networks include: Recurrent Neural Network (RNN), Long short-term memory (LSTM), Bidirectional LSTM (BiLSTM), Gated recurrent units (GRUs), and Variational AutoEncoder (VAE). The neural networks were trained on daily case numbers spanning from January 22nd, 2020, to June 1st, 2020, while the period from June 1st, 2020, to June 17th, 2020, was reserved for testing the models on unseen data. Data from Italy, France, Spain, China, USA and Australia was used. Performance metrics quantified the accuracy of each forecast and based on the scores the authors concluded that the VAE model performed the best. However, the table of the performance metrics appears to contain errors that makes understanding the authors' claim difficult. The paper also graphs the curves of the forecasts alongside the true data and based on the graphs the VAE does appear to have performed better.

Keywords: neural networks, covid, variational auto encoders, recurrent neural networks

1 Introduction

One of the challenges of the COVID-19 pandemic was making decisions in the present to prepare for an unknown number of COVID-19 cases in the future. A tool that can forecast the future number of cases has the potential to inform decisions in the present. A study conducted by Zeroual et al., titled "Deep Learning Methods for Forecasting COVID-19 Time-Series Data: A Comparative Study" [6] tests five neural networks: Recurrent Neural Network (RNN), Long short-term memory (LSTM), Bidirectional LSTM (BiLSTM), Gated recurrent units (GRUs), and Variational AutoEncoder (VAE) on their ability to forecast COVID-19.

The neural networks were trained on daily COVID-19 cases from six countries: Italy, France, Spain, China, USA and Australia. The training data spans from January 22nd, 2020, to June 1st, 2020, while the period from June 1st, 2020, to June 17th, 2020, was reserved for testing the models on unseen data.

The authors acknowledged that this is a relatively small amount of data to train the neural networks. Forecasting tasks generally have years of diverse and cyclical data to train their neural networks. However, due to the circumstances of the COVID-19 pandemic, the available data was limited, which added to the challenge of making accurate forecasts.

Section 2 will discuss basic mechanics that all neural networks share before discussing two types of neural networks that appeared in the study, RNN and VAE. The other three, LSTM, GRU, and BiLSTM are variants of RNN; for the sake of simplicity and space they will be excluded from discussion. The authors did not discuss the architectures of the neural networks; instead, a brief discussion of the general concepts behind each was given and few parameters. The specifics of the models would've assisted in understanding the study better; in absence of this information, the discussion will be limited to basic mechanics and the underlying concepts of each neural network.

Section 3 will analyze the results of the study. Five performance metrics were used to quantify how good the forecasts were compared to true data, and they will be explained for the sake of understanding results. Unfortunately, the paper's results table appears to contain errors, making analyzing the authors' claims difficult. Discussion of the errors will be part of the analysis.

2 Background

Understanding the basic mechanics of neural networks is a prerequisite for comprehending RNNs and VAEs. An overview of neural networks will be provided as the basis before discussing RNNs and VAEs.

2.1 Neural Networks

A neural network is a series of interconnected functions referred to as nodes. The nodes are organized in layers, where the outputs of nodes in a preceding layer become the inputs of nodes in the following layer. Figure 1 represents a single node within a neural network and will be referenced throughout this section.

In Figure 1 the node has x_n inputs, and each of these is multiplied by a corresponding weight w_n. The weights are negative or positive numbers that act to scale each input's relative contribution to the final output of the node. These

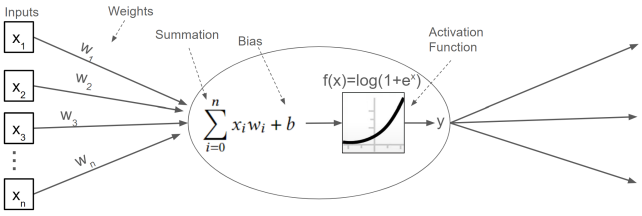

Figure 1. A single node of a neural network.

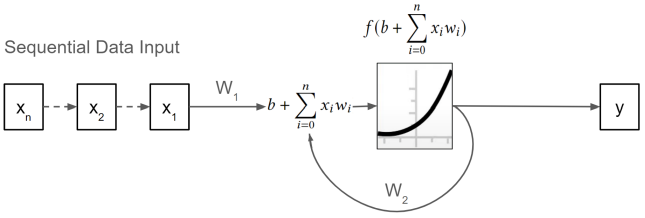

Figure 2. A single recurrent node.

weighted inputs $x_n w_n$ are all added together by the summation into a single number representing all the weighted inputs. A bias term b is added to the summation, acting as a final adjustment before it becomes the input for an an activation function. "Activation" denotes that a node will be more or less activated, produce a bigger or smaller output, depending on the input [3].

An example of an activation function, Softplus was arbitrarily chosen, can be seen in Figure 1 with its corresponding graph. An activation function introduces non-linearity necessary for representing complex patterns and relationships in the data.

In Figure 4, the true data is plotted with the x-axis representing the day, and the y-axis representing the number of cases. The goal of forecasting is to match a curve to the true data as closely as possible. Non-linearity enables the forecast to more closely match the true data. Conceptually, non-linearity is like overlaying an expressive curve that can closely match the true data, while linearity is like overlaying an inexpressive line that poorly matches the true data.

Mechanistically, a neural network is no more complicated than the algebraic operations described above. Each node in a network repeats this process of transforming its inputs into a single number. By changing the weights and biases of a node the output of the activation function can be increased or decreased. The process of calibrating each weight and bias to contribute to a desired final output is referred to as "training" the model.

During the training process, the model's biases and weights are adjusted iteratively using an optimization algorithm. The goal of the optimization is to minimize the difference between the forecasted values and the true values. The difference or "loss" is quantified in a loss function. Training involves feeding the data through the model, computing the loss, and then updating the model's weights and biases to reduce the loss. This iterative optimization allows a neural network to learn and improve its performance.

2.2 Recurrent Neural Network (RNN)

Recurrent neural networks are designed for use with time series or sequential data in which the order of the data matters [4]. An RNN processes inputs sequentially, as illustrated in Figure 2. The dotted arrows indicate each input is being sequentially inputted one at a time: x_1 is inputted, followed by x_2 up to x_n. In the context of forecasting COVID-19, these inputs could represent the daily infected count: day 1 is inputted, followed by day 2 up to day n.

A recurrent node is made up of weights, a summation, a bias, and an activation function, like the node discussed in the prior section about neural networks. The distinguishing characteristic of an RNN is found after the activation function produces an output. In Figure 2, the output is weighted and directed back to the summation as represented by the looping arrow. When the next input is introduced, it is summed with the previous output that represents all the previous inputs in the series. This recurrence is what makes an RNN suited for time-series data, as the node essentially references its own output from the previous time step. This cyclical action can be iterated to influence the current output by a desired number of preceding outputs.

The recurrence plays a crucial role in capturing short-term dependencies in time-series data. Short-term dependencies are patterns that exist in a small sequence of inputs. For example, when forecasting COVID-19 making a prediction about today is dependent on yesterday or the day before. RNNs can capture those short-term patterns between days or maybe weeks.

Long-term dependencies are patterns that span further back in time across a large sequence of data. Traditional RNNs can struggle with learning long-term dependencies. One issue is vanishing gradients, where the influence of early and distant inputs diminish during training [4]. Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) and Bidirectional Long Short-Term Memory (BiLSTM) networks were developed to address this issue by incorporating mechanisms to better control the flow of information as it sequentially receives inputs, enabling early inputs that are important to have an effect on later outputs. This is likely why the authors' included LSTM, BiLSTM, and GRU in the in the study alongside RNN.

2.3 Variational Autoencoder (VAE)

A Variational Autoencoder comprises two neural networks: an encoder and a decoder. The encoder progressively compresses the input in each layer, reducing the dimensionality, or in other words, the number of values make up the encoding. Architecturally this is reflected in a decrease of the

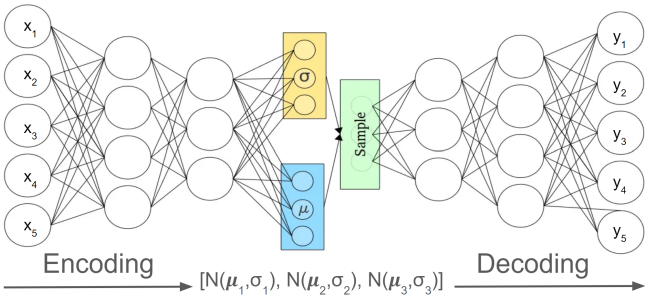

Figure 3. A diagram of a VAE

number of nodes in each following layer of the encoder. Figure 3 depicts a VAE, illustrating the reduction from the first, second, and third layers of the encoder [2].

This compression acts as a mechanism to remove irrelevant and repetitive data. This compressed input captures patterns, relationships, and features that are not apparent in the original input. This compressed representation is referred to as latent because it was hidden or underlying [5]. Once multiple inputs are encoded into their latent form, each dimension of the latent representation can be thought of as a coordinate with each coordinate representing a dimension in latent space. Similar encodings can be described as being closer in this latent space. Ideally, similar input data should have similar encodings in the latent space and dissimilar inputs should have dissimilar encodings in the latent space.

When the desired dimensionality n is reached, one set of n nodes outputs n means based on their connections and the second set n nodes outputs n standard deviations based on their connections. The encoder outputs the parameters of normal distributions, mean and standard deviation, for each dimension of the latent representation.

In a normal distribution, the mean represents the most likely value. The probability of sampling a value decreases the farther away it is from the mean. How drastically the probability decreases is determined by the standard deviation. The normal distributions when sampled give a variation of the encoding; the degree to which the sampled encoding is different from the encoding is dependant on the means and standard deviations.

Conceptually, the normal distributions of an encoding can be imagined as clouds of encodings in a three dimensional space with the mean determining the center of the cloud and standard deviation determining how densely the encodings that can be sampled surround the mean.

Ideally, these clouds will be close enough to overlap while still being distinct; minimizing this distance is an objective of training [1]. Overlap allows interpolation, which means that a meaningful encoding between two distributions of encodings can be found, since gaps don't exist between the distributions in latent space. This allows the creation of new encodings by interpolating in a latent space that smoothly transitions between distributions of encodings [2].

The decoder takes an encoding and decodes it back to a desired higher dimension. In Figure 3 each layer of the encoder progressively decompresses the input, as reflected in the increase of nodes in each following layer of the decoder. The encoder and decoder are usually trained together to balance finding a meaningful latent space that can be decoded and a latent space that allows interpolation [2].

In the paper a short and general description mainly about the probabilistic nature of VAEs was given. Discussion of how the VAE was inputted the time-series and if the architecture was was tailored for temporal data would've been helpful in understanding the study. Because the authors did not discuss those specifics, this section is missing those details.

3 Analysis

The accuracy of the forecasts were evaluated with five performance metrics [6]. Before analyzing the results of the study as well as the authors' claims, the performance metrics will be explained in the following section.

For each formula y_i represents the series of true values and $\hat{y}_i$ represents the series of forecasted values. Each metric computes some measure of how similar y_i and $\hat{y}_i$ are, in other words, how useful $\hat{y}_i$ is as estimates of the true values y_i.

3.1 Performance Metrics

Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) represent the average difference (error) between forecasted values and corresponding true values. A score closer to zero represents a more accurate forecast.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Because RMSE squares the difference, values far from the true data can have a bigger effect on the result. MAE takes the absolute value of the difference and as a result is less sensitive. RMSE is by definition greater than or equal to MAE because it squares the difference.

Mean Absolute Percentage Error (MAPE) represents the average errors as a percentage of the true values, making it useful for determining the relative accuracy of forecasts. A score closer to zero percent represents a more accurate forecast.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%$$

Root Mean Squared Log Error (RMSLE) is an average of errors on a logarithmic scale. RMSLE was likely chosen by the authors for its added benefit of penalizing underestimations

more than overestimations. In the context of using a forecast to inform decisions about COVID-19, an underestimation could lead to being underprepared, while an overestimation and overpreparing are preferred when lives are at stake. A score closer to zero indicates a more accurate forecast.

$$\text{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(y_i+1) - \log(\hat{y}_i+1))^2}$$

Explained Variance (EV) measures how closely the slope of the forecast matches the slope of the true data, ranging from 0 (indicating poor EV) to 1 (representing perfect EV). A good EV score results from consistent differences between actual and forecasted values. As long as the forecast maintains a consistent distance from the true data, even if it's a significant difference, the EV will be closer to 1.

$$\text{EV} = 1 - \frac{\text{Var}(y - \hat{y})}{\text{Var}(y)}$$

3.2 Results

The authors claimed that the VAE gave the most accurate forecast of daily COVID-19 cases.

> It can be easily seen that the VAE model outperformed the other models by providing good forecasting performance with lower RMSE, MAE, MAPE and RMSLE, and EV values closer to 1.

The authors presented graphs of the forecasts and a table of the performance metrics for each neural network in their discussion of results. Figure 4 displays the true curve of each country plotted alongside the forecasts, Table 1 contains the corresponding performance scores.

The scores are unedited from the original paper and as such, the authors use of commas is confusing. Commas often appear where a decimal separator would logically appear. In most English-speaking countries, decimal points are typically used as a decimal separator, but across the world commas are commonly used instead. The authors seem to be from non-English speaking countries, which could be the cause of the confusion. However, for reasons that will be discussed, even if commas are assumed to be the decimal separator, these scores are inconsistent with the graphs.

In Figure 4, the top-left graph of Italy shows an example of forecasts with favorable RMSE and MAE scores. The recurrent models (RNN, LSTM, BiLSTM, and GRU) are all very close to the dotted blue curve, representing the true data. VAE is far off and based off the graph has a RMSE and MAE of around 10,000. Despite this it closely matches the slope of the true curve and it likely has an EV close to 1. The four recurrent models appear to have good RMSE and MAE score of about 2,500, but likely posses poor EV scores since the gap between the forecasts and the true curve is inconsistent.

Table 1. Unedited performance metrics for forecasts of COVID-19 cases using RNN, LSTM, BiLSTM, GRU, and VAE models. RMSLE was excluded for space. Adapted from [6].

Nation	Model	RMSE	MAE	MAPE	EV
Italy	RNN	1,070,474	1,062,061	4,519	0201
	GRU	113,775	1,130,957	4,813	0314
	LSTM	1,054,089	1,046,257	4,452	0267
	BiLSTM	1,041,374	1,033,467	4,398	0269
	VAE	1,386,225	1,385,829	5,901	0951
Spain	RNN	1,683,011	167,719	6,944	0272
	GRU	1,795,678	1,791,683	7,419	0467
	LSTM	1,254,449	1,247,959	5,166	0396
	BiLSTM	1,194,711	1,187,629	4,916	0372
	VAE	5,315,748	5,288,172	2,19	0891
France	RNN	1,287,786	1,279,681	6,827	0224
	GRU	1,204,139	1,196,438	6,383	0311
	LSTM	1,085,008	1,075,795	5,738	0258
	BiLSTM	1,168,893	1,160,923	6,193	0308
	VAE	3,688,083	3,522,353	1,88	0554
China	RNN	1,252,034	1,250,442	1,485	0095
	GRU	1,085,698	1,083,975	1,287	0151
	LSTM	101,482	1,013,002	1,203	0163
	BiLSTM	1,205,955	1,204,413	1,43	0156
	VAE	11,103	107,873	0128	0843
AU	RNN	39,928	397,443	5,47	0279
	GRU	295,978	293,738	4,042	0349
	LSTM	327,123	325,203	4,476	0383
	BiLSTM	335,033	333,098	4,584	0363
	VAE	18,732	17,186	0236	0952
USA	RNN	5,227,287	5,136,497	26,373	0208
	GRU	4,369,108	4,240,145	21,697	0066
	LSTM	1,129,183	1,123,909	58,008	0
	BiLSTM	4,330,228	4,194,141	21,451	0024
	VAE	4,079,244	3,976,682	2,04	0993

For Italy, the graph is inconsistent with Table 1. Instead of being in the thousands or tens of thousands, the RMSE and MAE scores are all, except for GRU, about a million. The scale of these scores is way off based on expectations from the graph. GRU having a much smaller RMSE doesn't make sense since by definition RMSE is greater than or equal to MAE as discussed earlier. An EV score is between 0 and 1, but all the EV scores are missing decimal separators. If a decimal separator was added after the zero for each of these scores the EV scores would make sense and align with the authors' claim of the VAE having EV scores close to 1.

In Figure 4, the-bottom right graph of Australia provides an example of the VAE likely having an EV close to 1. Although the distance between the VAE forecast and the true curve is significant, the VAE forecast appears to have the best EV because it matches the slope of the true curve the best. For these EV scores, if a decimal separator is again added

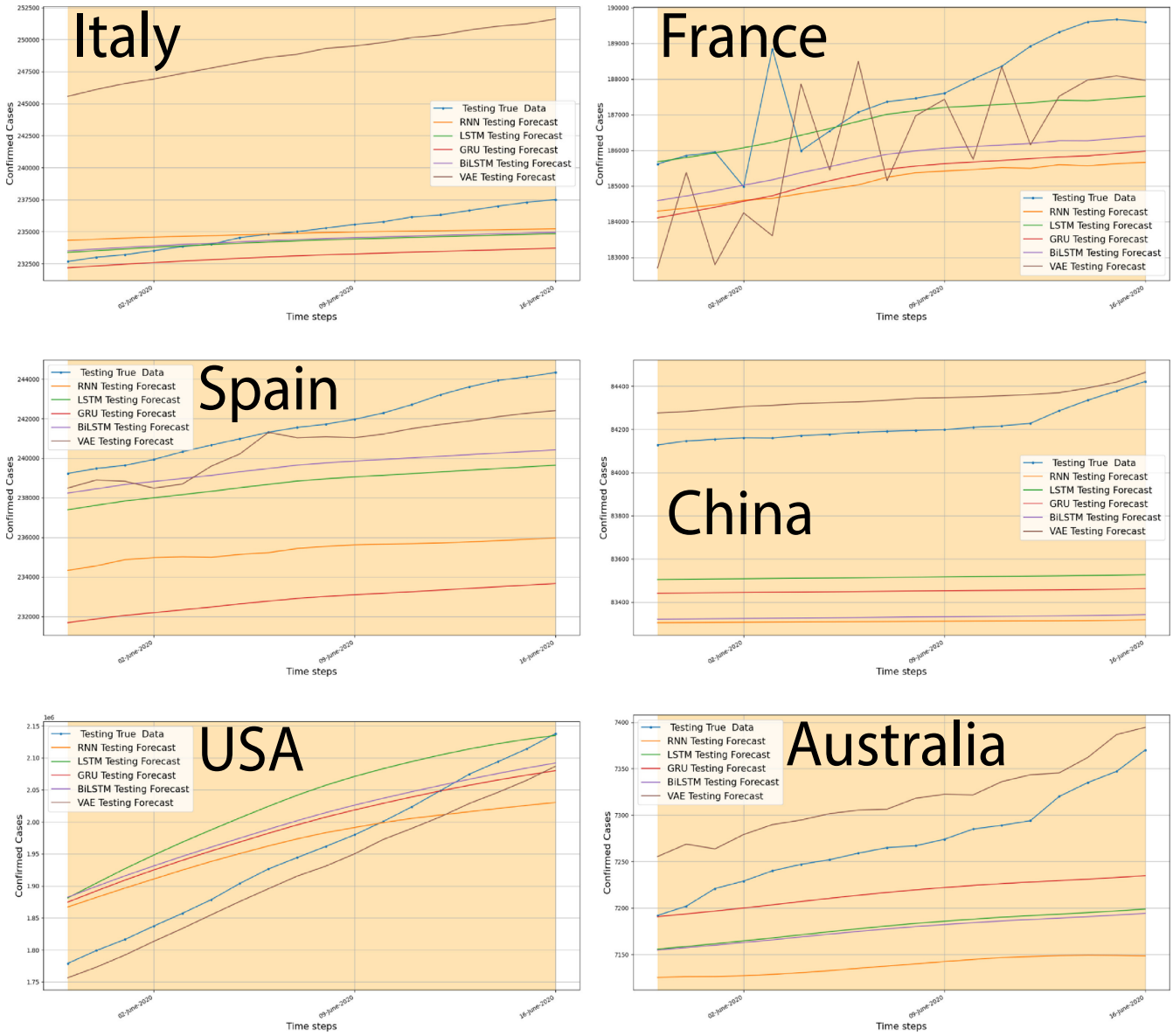

Figure 4. Forecasted cumulative cases from June 1st to June 17th, 2020, using RNN, LSTM, BiLSTM, GRU, and VAE for Italy, France, Spain, China, USA, and Australia. The true data is represented by the dotted blue curve, and the forecast colors are as follows: VAE (brown), BiLSTM (purple), GRU (red), LSTM (green), and RNN (orange). Adapted from [6].

after the zero then this supports the authors' claim that the VAE has EV scores close to 1.

The MAPE scores also appear to be inconsistent with the formulas provided in the paper. MAPE is a percent and should be between 0 and 100, even if it's assumed the commas are decimal separators the scores still appear to be inconsistent with the graphs. For example, based on the top left graph of Figure 4, Italy should have a poor MAPE and the others should have great MAPEs. But if it's Assumed that the comma is a decimal separator, they would all have around the same score which is inconsistent with Figure 4.

For RMSE and VAE scores, commas look like they're following the English norm as delimiters marking the thousands and millions places. This is incompatible with our earlier assumption that commas may act as decimal separators for MAPE scores. But assuming commas are following the English norm, RMSE and MAE scores for the majority of forecasts would suggest that on average the forecasts are off

by millions or hundreds of thousands. But when compared to the graphs the average difference is much smaller.

A pattern to the errors in RMSE, MAE, and MAPE columns are not as obvious as the EV column. The errors may have been caused by a mistake in formatting the table, but any explanation here would be speculation.

4 Conclusion

Zeroual et al. used a 131 day univariate time-series of daily COVID-19 cases to train and test their RNN, GRU, LSTM, BiLSTM, and VAE models. The following 17 days were used to test models in forecasting accuracy. The authors employed RMSE, MAE, MAPE, EV, and RMSLE as performance metrics to quantify the accuracy of the forecasts. According to the study's claims, the VAE was superior based on its performance in the 17 day forecast.

A table containing their performance scores as well as graphs of the forecasts were provided to corroborate their claim. But the performance table appears to contain errors based on the formulas provided and the errors complicate understanding the results of the study making drawing conclusions from the authors' claims difficult. A corrected version of the performance metric table would allow a more accurate assessment of the models' performance.

Despite the discrepancies in the table, inspection of the graphs that plot the forecasts alongside the true data does support the authors' claim that the VAE performed the best in forecasting daily COVID-19 cases.

This is result is interesting, considering the other models benefit from recurrence to capture temporal patterns in the data. The authors acknowledged that the VAE may have outperformed others due to the challenge of forecasting with a relatively small amount of data.

Acknowledgments

I would like to thank Nic McPhee, my senior seminar advisor for his patience and guidance throughout this process. I would also like to thank Peter Dolan for helping me understand the performance metrics. I would like to thank my alumni reviewer for their help. Finally, I would like to thank my friends and family for their continual support.

References

[1] Will Kurt. 2017. Kullback-Leibler Divergence Explained. https://www.countbayesie.com/blog/2017/5/9/kullback-leibler-divergence-explained [Online; accessed 10-October-2023].

[2] Irhum Shafkat. 2018. Intuitively Understanding Variational Autoencoders. https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf [Online; accessed 5-October-2023].

[3] Josh Starmer. 2020. Neural Networks Pt. 1: Inside the Black Box. https://www.youtube.com/watch?v=CqOfi41LfDw&t=847s [Online; accessed 10-September-2023].

[4] Josh Starmer. 2022. Recurrent Neural Networks (RNNs), Clearly Explained!!! https://www.youtube.com/watch?v=AsNTP8Kwu80&t=40s [Online; accessed 11-September-2023].

[5] Ekin Tiu. 2020. Understanding Latent Space in Machine Learning. https://towardsdatascience.com/understanding-latent-space-in-machine-learning-de5a7c687d8d [Online; accessed 20-October-2023].

[6] Abdelhafid Zeroual, Fouzi Harrou, Abdelkader Dairi, and Ying Sun. 2020. Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study. *Chaos, Solitons & Fractals* 140 (2020), 110121. https://doi.org/10.1016/j.chaos.2020.110121