NEURAL SUPREMACY, BUT SCARCE AND UNEVEN: MODELLING AND FORECASTING WORLDWIDE REMITTANCE PRICES

A COMPARATIVE STUDY BETWEEN STATISTICAL AND NEURAL NETWORKS MODELS FOR REMITTANCE PRICE FORECASTING

CHUNXI ZHAO

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
AT THE SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
OF TILBURG UNIVERSITY

STUDENT NUMBER

938301

COMMITTEE

Dr. Giovanni Cassani
Rotney

LOCATION

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science &
Artificial Intelligence
Tilburg, The Netherlands

DATE

May 20th, 2024

WORD COUNT

8750

ACKNOWLEDGMENTS

I would like to thank the support of the World Bank's Remittance Prices Worldwide dataset team, especially Mr. Karpinski, for their proposal of the idea of researching remittance, provision of the dataset and communication of the social significance of the project. I am especially grateful to Dr. Cassani, who offered strict but invaluable guidance, detailed instructions, and prompt responses. His dedication to his academic responsibilities, despite the demands of new fatherhood, has been commendable. The support from these individuals and organizations was crucial to the completion of this research.

NEURAL SUPREMACY, BUT SCARCE AND UNEVEN: MODELLING AND FORECASTING WORLDWIDE REMITTANCE PRICES

A COMPARATIVE STUDY BETWEEN STATISTICAL AND NEURAL NETWORKS MODELS FOR REMITTANCE PRICE FORECASTING

CHUNXI ZHAO

Abstract

This study investigates the effectiveness of various forecasting models for remittance prices using the World Bank's Remittance Prices Worldwide database. We compare conventional ARIMA models and neural networks (LSTM, CNN, and CNN-LSTM) against simple baselines. The findings reveal that while LSTM neural networks marginally outperform conventional ARIMA and naive models, the improvement is less substantial than anticipated. Notably, the Naive Last Value model performs surprisingly well, challenging assumptions about the superiority of complex models in this domain. Significant disparities in forecasting error exist across different country groups, with higher errors for non-G8 G20 countries and sub-Saharan Africa. The analysis of model robustness under data corruption scenarios demonstrates LSTM's superior resilience compared to CNN, albeit inconclusively. This study contributes to the ongoing discussions and studies about the comparative performance, disparate impact and robustness of models in financial forecasting and highlights that the study's results are not extraordinary but remain socially significant and scientifically encouraging for remittance service researchers, users and regulators alike.

1 DATA SOURCE, ETHICS, CODE, AND TECHNOLOGY STATEMENT

The dataset analysed in the thesis has been obtained from the World Bank's Remittance Prices Worldwide database through an online request and the World Bank owns the dataset. The dataset was available for download after the request was approved and a private email link was sent. The data is publicly accessible, and its use is subject to the terms and

conditions outlined by the World Bank, which include proper attribution and disclaimers regarding endorsement and warranties. The project thus does not involve data collected from human participants or animals.

All figures in the thesis are produced by the author, and there are pictures from external sources. All such pictures are within the public domain and the use abides by the license under which the pictures were distributed.

The code used to generate the results is available through the provided Github page here: https://yaqovz.github.io/personal_website/. The thesis referenced code snippets from *Darts* and *StatsForecast* libraries' tutorials, with the assistance of generative models, ChatGPT and Claude AI, for debugging and corrections. The adapted code fragments are indicated in the notebook through the use of the libraries. All libraries and frameworks used, along with their version numbers, are to be seen in the thesis.

In terms of writing, the thesis used the Oxford American Writers' Thesaurus and Google for paraphrasing and accurate expression. As for grammatical and syntactical guidance as well as spelling checks, Grammarly, Claude AI and ChatGPT were employed. The thesis was formatted using Overleaf, with the guidance of ChatGPT for LaTeX coding. No other reference management software was used other than the LaTeX template provided by the university.

2 INTRODUCTION

The age of migration is arriving. At the beginning of the decade, 281 million people live in countries where they are not born, comprising 3.5 per cent of the world population (United Nations, 2023). Amid the hectic political debates about the cultural dimensions of immigration, remittances — the transfer of money by migrant workers to their home nations — have been overlooked. Remittance volume in 2023 recorded $669 billion dollars, exceeding foreign direct investment and development aid for lower- and middle-income countries as a source of financing (KNOMAD, 2023). However, the excessive costs of remittance services, averaging 6.18% and over double the UN Development Goal for 2030, hinder global development and poverty reduction.

In time-series forecasting, the dataset presents a valuable opportunity to test various models on brand-new real-world data. The state-of-the-art of the area is generally accepted as N-Beats, a deep learning model based on multiple layer perceptrons (Oreshkin et al., 2020); but the current development in time-series forecasting has issues with performance and generalisability in real-world datasets existing out there.

Thus, for data scientists, this thesis contributes to the current debate on the performances of a wide range of time-series forecasting models. Given the conflicting evidence and lack of consensus in the literature, the author hopes to add empirical evidence on model performance facing various assessments of comparison, generalisability and robustness. For future research, it aims to highlight pitfalls and inspire methodological improvements for future studies of remittance prices' variance and evolution.

Societally, this thesis strives to change the remittance industry for the better through data-grounded policy-making. The remittance industry has been long censured for opaque pricing at high costs of migrant workers. Insights into future pricing dynamics enable authorities to monitor remittance costs proactively and negotiate more effectively with service providers, ensuring affordable channels. Within a remittance ecosystem where governments have a stronger leverage, policy-making can create an environment where remittances flow affordably and reliably, contributing to financial inclusion and human development of the developing world. Moreover, evaluating forecasting models paves the way for robust early-warning systems. When price fluctuations are anticipated, timely preparations can be made in light of the potential disruptions in state finance and family budgets. Thus, this research could provide foresight benefiting all stakeholders - migrant workers, regulatory bodies, service providers, and recipient communities.

However, this thesis also acknowledges the potential disparate impacts. Low-income countries might also lack the necessary technological infrastructure and data processing capabilities to fully leverage the findings, thereby exacerbating existing inequalities. Conscious of the potential disparate impacts, the thesis also incorporates the perspective of inequality into its research questions. This ensures that data-driven policy-making advances inclusively and equitably, benefiting all communities involved.

2.1 *Research Strategy and Questions*

To investigate the performance of different time-series forecasting models on the multiple time-series forecasting task regarding remittance costs, this thesis asks the following questions:

> *To what extent can one forecast remittance prices using the World Bank's Remittance Prices Worldwide database?*

To better understand the forecastability of remittance prices, the thesis assesses the performance, generalizability, and robustness of various models through the following questions:

RQ1 *How well do conventional ARIMA and neural networks forecast the following quarter's remittances given all the preceding data?*

RQ2 *How do the best baseline model and the best-trained model compare in terms of performance across different types of countries?*

RQ3 *To what extent does the corruption of recent input data affect the forecasting performance of neural network models for remittance prices, and how does this impact vary with the length of the corrupted period?*

The thesis begins with RQ1 comparing models with different assumptions to identify promising approaches for forecasting remittances in an unfamiliar dataset. RQ2 then examines the disparate social impact through their performance across geographically and economically different country groups, crucial for detecting potential biases that may exacerbate inequality between countries. In this way, RQ2 provides multifaceted insights into the forecastability of remittance prices with attention to its heterogeneous social implications.

RQ3 evaluates forecasting models' robustness against compromised how data integrity impacts remittance price forecasting, crucial in real-world scenarios. Under scarce data regimes, maintaining accuracy despite compromised data is essential, particularly for remittance data which war and diseases can interrupt or distort in its collection and accurate predictions could impact financial planning for vulnerable populations. Answering RQ3 will clarify the extent to which remittance prices can be forecast under varying data quality conditions more realistically, identifying current limitations and guiding improvements upon model robustness.

2.2 *Findings*

- Among the trained models, LSTM performed best, matching and surpassing the best baseline, showing the potential of modelling and forecasting remittance prices. But the baseline using the last value only performed surprisingly well, achieving the lowest in two of the three metrics.
- ARIMA showed mediocre performance, underperforming the best baseline and the best neural network model by approximately 7% to 10% in terms of error.
- Significant disparities in forecasting performance exist across different geographic and economic groupings, and error is particularly high for certain less developed countries.

- Data shuffling experiments revealed the importance of the last value. LSTM showed the relative robustness of LSTM to data distortions compared to CNN, although both LSTM outperformance's generalisability and robustness await further research for more conclusive answers.

3 RELATED WORK

This section first discusses current approaches to remittance price analysis and highlights the importance of time-series forecasting. Next, it explains the significance of the thesis's model comparison in the ongoing debate. Finally, it emphasizes the importance of assessing the optimal amount of information for neural models.

3.1 *Literature on Remittance Prices*

As important as remittances are in international economics, remittances remain an understudied topic in related disciplines. The majority of research employs cross-sectional approaches by pooling the data points together. The pioneering study (Beck & Peria, 2009) involved cross-sectional regression of socioeconomic factors in 2009 on remittance costs. It suggests that corridors, where more banks run remittance services and a larger immigrant population, tend to have lower costs for sending money transfers back home, a conclusion echoed by Bersch et al. (2021). Further, Beck et al. (2022) added that urbanisation, the Internet and robust financial development all make international transfers cheaper, using regression analysis involving a cross-sectional sample of 2018 and a pooled panel from 2011 to 2020.

As Beck and Peria (2009) suggested that panel variation be exploited, the thesis contrasts with existing panel regression approaches and tries to estimate the next quarter's value using the inherent dynamics of time-series. Existing studies on remittance pricing have relied on cross-sectional or pooled data, treating remittance costs as static or without patterns of evolution over time.

Additionally, remittance costs vary significantly with the location and economic status of the sending or receiving countries (Beck et al., 2022; KNOMAD, 2023), with impoverished African nations bearing the highest costs of remittances. Consequently, temporal dynamics may also be related to such variables and impact the models' generalizability. The thesis, therefore, examines the forecasting performance of the models more granularly to account for these factors.

3.2 *Advances and Controversy in Financial Time-series Forecasting*

Time-series forecasting began with autoregressive integrated moving average models - ARIMA (Box & Jenkins, 1976) and now see a boom in deep neural networks. The former handles data series individually, while the latter ones - being global - identify hidden patterns across all series and deliver outstanding performances (Semenoglou et al., 2021).

The Fourth Makridakis Competition's (M4) results - as the established yardstick in time-series forecasting - prove the excellence of neural networks (R. Hyndman, 2020; Makridakis, Spiliotis, & Assimakopoulos, 2018a). The state-of-the-art remains N-Beats (Oreshkin et al., 2020) which achieved 11.135 in symmetrical Mean Average Percentage Error (sMAPE), a 21.6% decrease in this error metric compared to the Naive benchmark. On the same M4 dataset, the performance is followed by that of the competition year's winner, a hybrid model, ES-RNN, which Smyl (2020) proposed and achieved 11.720 in sMAPE.

Neural networks appear superior when dealing with financial data, too. (Siami-Namini & Namin, 2018) showed a simple Long Short-Term Memory (LSTM) model with 4 neurons already surpasses ARIMA by 80% in Root Mean Sqaure Error (RMSE). Similarly, Latif et al. (2023) demonstrated that an RNN-LSTM model improved the prediction to 0.27% from 1.79% in Mean Average Percentage Error compared to ARIMA.

Despite the growing evidence in favour of neural networks, their superiority remains contested. Green and Armstrong (2015) and Makridakis, Assimakopoulos, and Spiliotis (2018) noted that complex methods are often favored in journals over simpler models, even when the former were not adequately benchmarked or replicated. A comprehensive comparison (Makridakis, Spiliotis, & Assimakopoulos, 2018b) showed ARIMA outperformed top neural network models by 12% in symmetrical Mean Absolute Percentage Error (sMAPE) on a large competition dataset, while most basic neural networks underperformed naive baselines. Notably, in the M4 competition, half of the neural network models failed to surpass naive baselines (Makridakis, Spiliotis, & Assimakopoulos, 2018a), which Makridakis et al. (2020) attributed to the large number of parameters and difficulty with cross-validation if the series are short.

Makridakis et al. (2023) recently revised his opinion, arguing that the latest deep learning models can achieve the best performance on the same dataset. However, the practical superiority of neural networks remains controversial, given the significantly longer training time demanded by the latest deep models.

Building on the literature above, this thesis compares statistical models against simple neural networks, excluding the demanding latest models. It

differs from competition datasets which may suffer from the lack of real-life data representativeness (Spiliotis et al., 2020), over-research and the overfitting of other models, hence the inflated performances (R. Hyndman, 2020). Thus, with the remittance dataset, this thesis aims to provide meaningful evidence to the debate on the practical utility of advanced models.

In addition to the LSTM, this thesis implements a simple CNN as a pattern-based alternative to recurrent models for time-series forecasting. Semenoglou et al. (2021) provide theoretical backing for the efficacy of CNNs in this domain, arguing that neural networks can benefit from cross-learning globally latent patterns, a strength of architectures like CNNs. By comparing models from diverse architectural families, the thesis aims to gain diverse perspectives and a more well-rounded understanding of the complexities inherent in the remittance prices dataset.

3.3 *Data Informativeness in Forecasting*

The debate surrounding neural network performance in finance is intertwined with economic theories, particularly the Efficient Market Hypothesis (EMH). EMH posits that prices immediately incorporate all relevant information, questioning the utility of historical data in forecasting (Fama, 1965). Supporting this, Kraehenbuehl and Osterrieder (2022) found no improvement in Bitcoin price forecast accuracy when neural networks incorporated additional stock data, contradicting studies advocating for LSTM models (Latif et al., 2023). This aligns with EMH, as it suggests long-term information—a key feature of LSTM models—may be redundant.

However, other studies emphasize that optimal input length varies, and excessive information can hinder performance. Azlan et al. (2019) demonstrated that a 100-day window outperformed shorter periods in predicting S&P 500 trends by approximately 20% in accuracy, while longer windows misled models. Extending similar trials across multiple models, Shi et al. (2022) found that in air quality forecasting, a 3-day input window was optimal for 3-hour predictions, generally outperforming both shorter and longer inputs.

These conflicting findings raise important questions about how important the later and earlier data are respectively and how robust the model is in case later, more informative data are unavailable. However, the typical approach of using large variations in input range is not feasible for our remittance data due to 1) its scarcity and brevity, and 2) the tendency of simple LSTM models to "replicate" the last input given adverse data situations (Kosma et al., 2022). To address these constraints while still examining the impact of historical data length on forecasting performance,

this thesis proposes an alternative approach using data shuffling, and comparing LSTM and CNN models illustrates how very distinct types of model architectures - one sequential and one spatial - respond differently to information corruption.

4 METHODOLOGY

This section introduces the models used in this thesis for forecasting remittance prices. Starting with the baseline, it then describes the principles behind autoregressive integrated moving average models (ARIMA) and explains the architecture of the neural network models employed in the thesis.

4.1 *Baseline Models*

Following R. Hyndman and Athanasopoulos (2018), the thesis applies three baselines: Mean, Naive Method, and Drift. They are simple but effective methods in financial time-series forecasting and especially the naive method has been used as a benchmark in established forecasting competitions (Makridakis & Hibon, 2000).

Suppose one predicts t_n. The mean method means taking the average of all the values preceding $(t_1, \ldots, t_{n-1})$ to be predicted as the prediction. The naive method is to take the value immediately preceding as the forecast, which is to set the forecast of t_n as t_{n-1}. The drift method is equivalent to drawing a straight line between t_1 and t_{n-1} and extrapolating it to t_n as its forecast value.

4.2 *Statistical Model: ARIMA*

The ARIMA model, chosen for its parsimony and strong forecasting performance, is fitted to each time-series and compared against the three baselines. Introduced by Box and Jenkins (1970), ARIMA forecasts future values using three components: autoregression (AR), integration (I), and moving average (MA) - the specific orders of these components are denoted by the parameters p, d, and q, respectively.

The autoregressive part (AR) utilizes past values of the time-series to forecast the next value, while the moving average (MA) part captures the dependencies between the current value and past errors. Given AR and MA only capture linearity, capturing non-linear trends requires the integration component (I), which ensures that non-stationary time-series are made stationary by differencing. For a detailed explanation, see Appendix C.

ARIMA is selected as a meaningful benchmark against neural network models, particularly because it is akin to the linear form of the simplest neural network model, the Multiple Layer Perceptron (MLP) (Makridakis, Spiliotis, & Assimakopoulos, 2018b). While ARIMA captures linear relationships, neural networks are theoretically capable of modeling non-linear interactions. Thus, comparing ARIMA to neural networks assesses whether adding non-linear complexity brings additional benefits.

4.3 *Neural Networks*

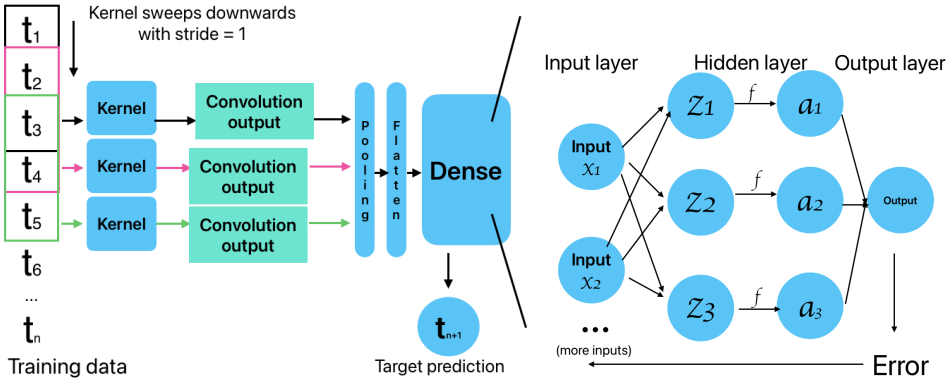
The models compared to the conventional ARIMA model are neural network models, Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM). Both models have their own characteristics, with LSTM better at capturing long-term dependencies and CNN better at learning local patterns. This section introduces the mechanism of both models for 1-D data.

4.3.1 *CNN and Dense Layers*

Multilayer perceptron (MLP) (Werbos, 1970), from which the Dense layer in the model is derived, is a fully-connected network where every neuron in the hidden layer receives information from all input values (Figure 1b). It works through the following two steps:

FORWARD PROPAGATION: The input values are multiplied by a weight matrix and added to a bias vector. A non-linear activation function, such as ReLU, is then applied to the result, producing the output.

BACKPROPAGATION: The algorithm adjusts the weights and biases by calculating the error gradients with respect to these parameters, aiming to minimize the error. This process iteratively updates the parameters to improve the model's performance on the training data.



(a) CNN Layer (b) Dense Layer (MLP)

In a CNN architecture, introduced by Fukushima (1980), the convolutional layer precedes the Dense layer to extract local patterns. As shown in Figure 1a, a one-dimensional convolutional kernel sweeps across the input sequence, performing dot multiplication at each stride with the covered area, capturing local features. This process, known as "local connectivity," links each convolutional node to a small subset of the sequence. Thus, neurons in the same convolutional layer apply the same kernel across the sequence, detecting patterns and forming a feature map. The feature map then undergoes pooling (optional) and flattening for feature and dimension reduction. The transformed convolutional outputs are fed into the dense layer for further processing and backpropagation. This design allows CNNs to extract low-level patterns using convolutions and combine learned features in the Dense layer for predictions.

CNN models are computationally efficient due to local connectivity and shared weights. Each layer's nodes are only locally connected to the input, and only one weight matrix is used per layer for the entire input, significantly reducing computation—beneficial for a student thesis project. Additionally, CNNs are suitable for financial data, as market prices are seldom impacted by events from the distant past, making long-term dependencies less useful (Cao & Wang, 2019).

4.3.2 *LSTM*

An LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) designed to understand long-term dependencies in sequential data. It is a special kind of Recurrent Neural Network, designed to solve the traditional RNN's problems with vanishing or exploding gradients in long sequences (Gers et al., 2000; Hochreiter & Schmidhuber, 1997). Unlike

traditional RNNs, LSTMs have a more complex cell structure that helps to determine which information to forget or to pass on.

The LSTM cell, comprising a cell state and three gates (Figure 2), processes past information through these gates at each time step. The Forget Gate determines how much of the previous cell state to retain, the Input Gate updates the cell state with new information, and the Output Gate filters the cell state to produce the current hidden state. For a more technical explanation, see Appendix C.

The three gates allow for selective retainment or forgetting over a long sequence without gradient problems, capable of capturing both short-term and long-term dependencies if need be. It is particularly beneficial if long-term seasonality or trends are as important as short-term patterns to be encoded, and it does prove highly competent through its wide application in different domains of time-series forecasting and advanced forecasting models (Hewamalage et al., 2021).

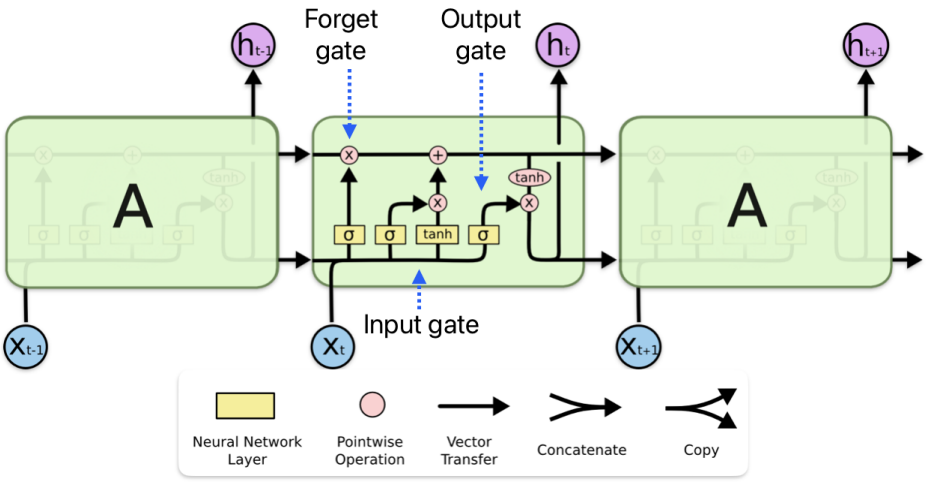


Figure 2: LSTM structure, image from Olah, 2015.

4.3.3 *CNN-LSTM*

The one-dimensional feature map that a 1-D CNN kernel outputs is simultaneously a sequence that has the pattern of the information condensed, and it is suitable to be handled by LSTM for learning of sequential patterns. The combined CNN-LSTM model leverages the strengths of CNN and LSTM to capture both local patterns and long-term dependencies simultaneously. This hybrid architecture is particularly useful for time-series forecasting tasks, where the data may exhibit intricate patterns at multiple scales.

5 DATASET PROCESSING

This section describes preprocessing. It begins with the script libraries, and then proceeds to the description and preprocessing choices of the research. For a high-level flowchart showing the steps, see Figure 3.

5.1 *Script information*

This project is done in Python. For library versions, see Appendix A. *SciPy* (Virtanen et al., 2020) generated Figure 12. *StatsForecast* (Garza et al., 2022) created statistical plots in Section 6.2, and *darts* (Herzen et al., 2022) selected the best ARIMA model and calculated all metrics. *scikit-learn* (Buitinck et al., 2013) handled data scaling. *keras* (Chollet et al., 2015) built neural network models, optimized with *hyperopt* (Bergstra et al., 2013). *deep-significance* (Ulmer et al., 2022) performed significance tests between neural network models. For reproducibility, the script sets the random seed to 88 wherever possible.

5.2 *Dataset Description*

The dataset used in this thesis is the World Bank's Remittance Prices Worldwide (RPW) dataset (The World Bank, 2024). The RPW dataset provides data on the costs of sending and receiving remittances across international borders from 48 sending countries to 105 receiving countries, covering 367 corridors. Data collection started biannually in 2011 and has been updated quarterly since 2013, with the latest data available up to Q3 2023.

The raw RPW dataset is split into two panel data sheets: one up to Q1 2016 and the other from Q2 2016 onwards. Each row shows the remittance costs (as a percentage of the amount sent) between two countries for a specific quarter. The dataset includes variables such as the region of the sending and receiving countries and their G20/G8 membership status. For a complete list of variables, see Table 3 in Appendix A. The remittance costs are recorded in two versions: "cc1 total cost%" (based on sending $200) and "cc2 total cost%" (based on sending $500), as sometimes there is a limit on the amount sent.

5.3 *Dataset Preparation*

The dataset preparation involved merging, variable selection, pivoting, and cleaning missing values. The final dataset was produced by merging

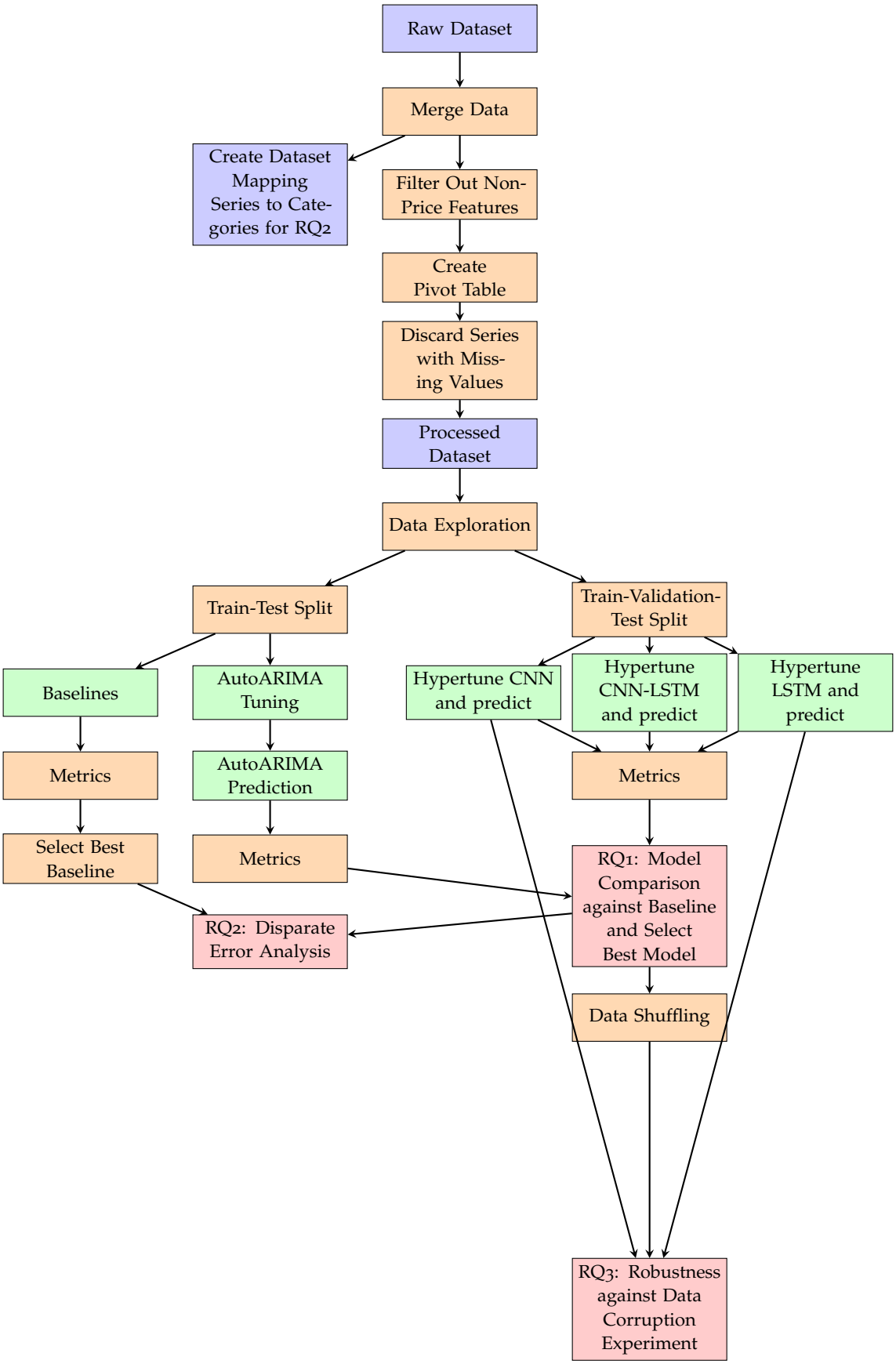


Figure 3: Flowchart illustrating the experimental setup of this thesis project. Blue blocks indicate the creation of a dataset; red blocks indicate experiments answering research questions; green blocks indicate model building.

two sheets into a single DataFrame through vertical concatenation and discarding non-shared columns. After analyzing missing values, 'cc2' columns were excluded due to significantly more missing values, and repetitive columns like 'source_code' and 'destination_code' were removed.

For analyzing total cost variations over time, a pivot table was created with 'timestamp' as the index, 'corridor' (remittance channel between two countries) and 'firm' as columns, and the mean of 'cc1 total cost %' as values. The initial four biannual timestamps were dropped for quarterly data consistency. Table 4 in Appendix A shows an example of the transformed data for different companies' transfer costs between South Africa and Zimbabwe. Another table is created mapping each series to the static categories, such as region. It serves for disparate error analysis in RQ2, grouping the error metrics per category.

Among all series, 774 firm-corridor time-series were complete (see Figure 11 in Appendix B for the missing data distribution). Many incomplete series had interruptions due to missing values, and the thesis decided to only select time-series with complete data across the entire period, since missing data imputation is out of the thesis's scope. Moreover, most missing data resulted from discontinued data recording at different time points due to changing economic and political significance. Focusing on coherent series ensures the coverage of corridors the World Bank deems most important, an approach the data collecting team approved after discussion.

5.4 *Dataset Visualisation*

Figure 4 shows the first 5 time-series from the Algeria-Egypt corridor, illustrating the typical variation and potential for outliers in the selected set of series without missing data. The series generally fluctuate between 2% and 6%, with occasional spikes like Al Fardan Exchange's price reaching 16% in 2022, indicating prominent variations likely due to external events.

Overall, the dataset comprises 774 time-series exhibiting diverse statistical properties, as visualized in Figure 12 in Appendix B. The average series value is around 5%, with a right-skewed tail indicating some higher-priced corridors. Most series demonstrate moderate variation, with the coefficient of variation peaking around 0.2. However, a fair number of corridors see unstable variation and skewness beyond ±1, pointing to the presence of outliers within each series. Indeed, 231 out of 774 series contain at least one outlier compared to the general trend.

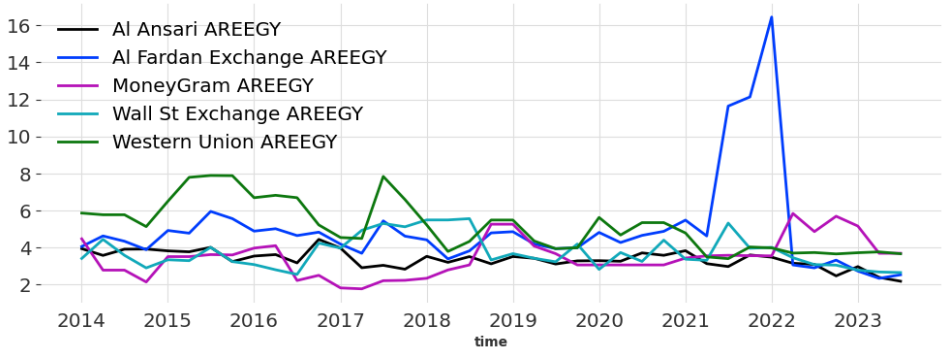


Figure 4: Plot showing the first 5 time-series

6 EXPERIMENTAL SET-UP

This section delves into the practical and describes the experimentation set-up and process step by step: setting up ARIMA and various neural network models and their train-test split approaches respectively for model comparison in RQ1. It then explains the metrics and significance checks used before detailing how disparate error analysis (RQ2) and robustness tests (RQ3) are conducted.

6.1 *Overview of Train-Test Split*

The goal is to predict the last three values (t_{37}, t_{38}, t_{39}) in a time-series of 39 points. The data split for ARIMA and neural networks (CNN, LSTM) differs but follows a unified principle for fair comparison. Both models train and/or validate up to t_{36}, using the best parameters to predict t_{37}, t_{38}, and t_{39}. Each model uses a single trained instance for predictions, given the actual values immediately preceding without updating parameters. This ensures both models have equal access to information, maintaining a controlled experimental setup. Figure 5 provides a high-level illustration of the difference with descriptions following below.

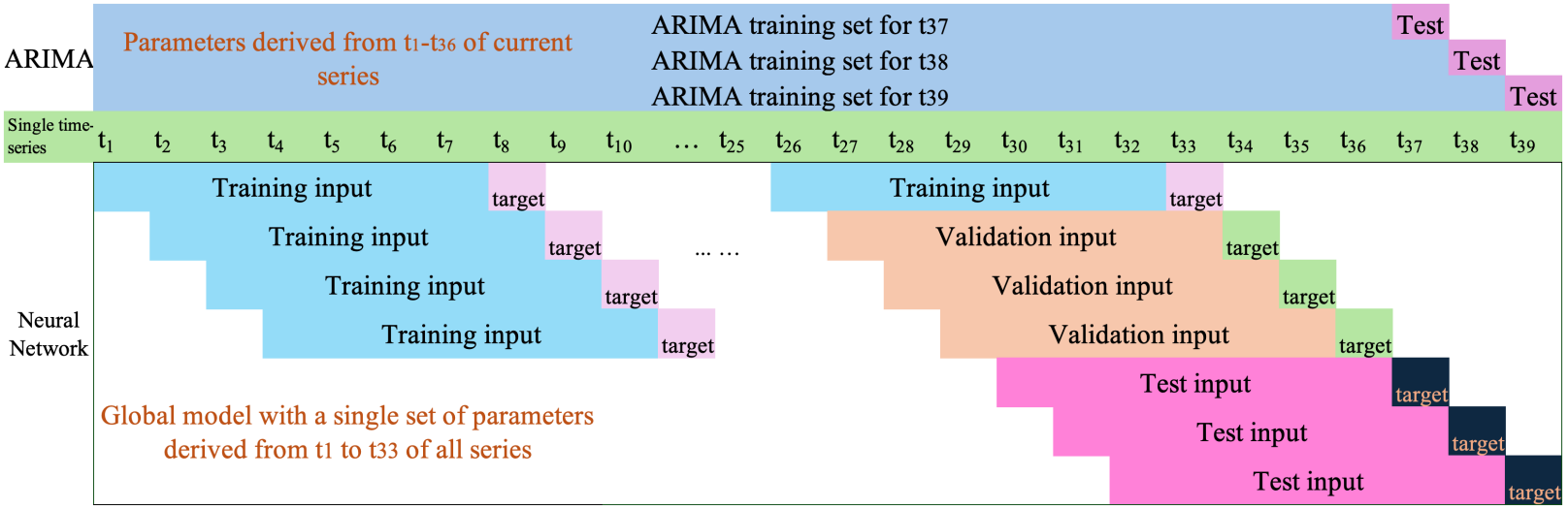


Figure 5: Train-Validation-Test Split

6.2 *ARIMA Set-up*

The key to model a series using ARIMA is to determine its three parameters, p, d, and q. Traditionally, ARIMA models require meticulous manual tuning. First, stationarity testing is conducted using the Augmented Dickey-Fuller (ADF) test. For non-stationary data, differencing is applied until stationarity is achieved. The PACF and ACF plots then guide the selection of p and q parameters (See Appendix C for details). Yet, tuning these parameters can be arduous due to multiple significant lags, making model testing across large datasets time-consuming.

Instead of manual labour, the thesis utilizes *StatsForecast*'s *AutoARIMA* procedure. This procedure automates the traditional tuning of the potential combinations of parameters and the model with the lowest Akaike Information Criterion (AIC) using cross-validation. The steps are as follows:

- For each individual time-series, ARIMA parameters are determined with *AutoARIMA* using data up to t_{36}, not subject to change during forecast.
- To predict t_{37}, the model is given time steps t_1 to t_{36}.
- For predictions t_{38} and t_{39}, the same model is provided with all time steps up to t_{37} and t_{38} respectively.

This iterative retraining approach is adopted to produce the next-quarter forecast without cascading errors from previous predictions. Unlike neural networks, ARIMA models do not require a separate validation set. They are trained on the entire available data up to the prediction point and make forecasts iteratively, utilizing the previously predicted values as inputs during training.

6.3 *Neural Network Set-up*

In this thesis, three neural network architectures were employed: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and a hybrid CNN-LSTM model. *hyperopt* tunes each model for 100 times in a Bayesian manner for the parameters minimizing validation loss, within a user-defined search space. The search space of the tuned hyperparameters and the best values are listed in Appendix A.

All models use Huber loss, preferred over mean squared error (MSE) for its higher robustness to outliers present in the data (Jadon et al., 2022). Huber loss applies MSE to normal values and switches to a linear loss function for large outliers, reducing the outliers' impact on the loss and improving model robustness. Empirical experiments using Huber also

proved a small improvement upon the metrics compred to MSE. Mean absolute percentage error (MAPE) is not selected because it penalizes positive errors less - not suitable for this dataset where most values are low positives and outliers are high positives.

Each model trains for a maximum of 30 epochs, with early stopping (patience set to 5) to prevent overfitting due to the short input size. Training halts if validation loss does not decrease for 5 consecutive epochs.

6.3.1 *Data Transformation*

The thesis makes no *a priori* assumption about the best data transformation method, and it prioritizes model performance over the indiscriminate implementation of standard practices. The following three data transformation methods are used before the data are fed into the model:

1. Raw data
2. Logarithmic transformation
3. Min-max scaling to range [0, 1] after logarithmic transformation

The latter two methods follow the literature comparing statistical and neural network models (Makridakis, Spiliotis, & Assimakopoulos, 2018b; Nesreen K. Ahmed & El-Shishiny, 2010). If log transformation is employed, the whole dataset is transformed. In the case of min-max scaling, the thesis fits *scikit-learn*'s *MinMaxScaler* on t_1 to t_{36} (the training set's range) and transforms the train, validation and test sets separately with the same scaler - to scale series by series and prevent data leakage.

However, empirical evaluation showed that these transformations did not improve model performance compared to the original, untransformed data. The lack of improvement from data transformations was confirmed as statistically significant by the deep-significance library, which compared 20 result sets after each transformation.

Outliers were also retained, as they represent genuine external shocks that a forecasting model should be able to handle and do not indicate entry errors. In fact, external shocks like COVID and warfare and policy changes easily impact remittance costs. The outliers therefore contain information of the data-producing process and should be taken into account (R. Hyndman & Athanasopoulos, 2018).

Therefore, the thesis proceeded with the untransformed data with potential outliers, and all reported results are based on this approach.

6.3.2 *Train-Test Split for neural networks*

In contrast to ARIMA, the train-test split for neural networks uses a sliding window approach. A chronological train-validation-test split assesses the

models' ability to generalize to future data points, avoiding data leakage that could occur with conventional cross-validation if later data points were used for training and earlier data points for testing. For each series:

- Input is a window of k consecutive time steps (e.g. $t_1 \dots t_k$, $t_2 \dots t_{k+1}$), with the target being the following value (t_{k+1}, t_{k+2}, respectively).
- To avoid leakage, training inputs/targets are derived from t_1 to t_{33}.
- t_{34}, t_{35}, t_{36} are validation targets with preceding data of length k as input.
- Testing is on predicting t_{37}, t_{38}, t_{39} with preceding data of length k as input.

Thus, out of each series, six arrays are created: three arrays of sliding windows for training/validation/testing, respectively and another three arrays of their corresponding targets. The neural network model then at once trains on all the series' training window-target pairs, validates on the validation counterparts and forecasts given the test sliding windows. The forecast results and the test targets are used to calculate metrics. This approach trains one model on all series segments ("global training"), enabling cross-learning from unrelated series.

6.3.3 *CNN model*

The CNN architecture adopted is stacked as follows: an initial Conv1D layer with a specified number of neurons, n, followed by another Conv1D layer with half the number of neurons and the same kernel size, a Flatten layer, and one or two Dense layers with neurons equal to the second Conv1D layer, concluding with an output layer. All layers, except the output layer, use 'relu' as activation function. This design, inspired by Fungtammasan and Koprinska (2023), omits pooling layers to avoid information loss given the short input window. Omitting pooling prevents length reduction and information loss in the already short input window for time-series analysis (Liu et al., 2020).

The hyperparameter ranges and best values of all three neural networks are shown in Table 6, 7, 8 in Appendix A.

6.3.4 *LSTM model*

The LSTM model consists of one, two or three LSTM layers, followed by a dense output layer. The number of LSTM layers is a tunable hyperparameter. When multiple LSTM layers are used, each subsequent layer has half the number of units as the previous layer. The LSTM layers, except

for the last one, are set to return the full sequence to preserve temporal information for subsequent layers, concluding with a Dense layer. The search space of possible input window lengths, optimiser, and its learning rate is kept the same as for CNN.

6.3.5 *CNN-LSTM model*

The CNN-LSTM model comprises a CNN layer followed by one or two stacked LSTM layers and a dense output layer. The number of LSTM layers is a tunable hyperparameter. When multiple LSTM layers are used, each subsequent layer has half the number of filters as the previous layer, except the last layer, which returns the full sequence to preserve temporal information for the dense layer. The input window length, optimizer, learning rate, and kernel size are hyperparameters tuned within the same search space as the CNN model.

6.4 *Metrics*

The chosen metrics for this thesis are symmetric Mean Absolute Error (MAE), Mean Absolute Percentage Error (sMAPE), and Mean Absolute Scaled Error (MASE), derived from the *darts* library (Herzen et al., 2022). MAE is selected for its interpretability: it uses the same units as the original data, thus easy to understand the forecast deviation. sMAPE is also used and calculated as a percentage as below:

$$sMAPE = 200 \cdot \frac{1}{T} \sum_{t=1}^{T} \frac{|y_t - \hat{y}_t|}{|y_t| + |\hat{y}_t|}$$

where y_t represents the true series and $\hat{y}_t$ the predicted values.

MASE compares the forecast error to the error obtained by a naive model on the in-sample data and is particularly useful for evaluating if the model outperforms a simple one. It is computed as:

$$MASE = \frac{MAE(y_{t_p+1:t_p+T}, \hat{y}_{t_p+1:t_p+T})}{E_m}$$

with E_m being:

$$E_m = MAE(y_{m:t_p}, y_{0:t_p-m})$$

Here, $y_{t_p+1:t_p+T}$ is the true test series, $\hat{y}_{t_p+1:t_p+T}$ the predicted test series, t_p the prediction time (before the first forecast), and m the length of a season (defaulting to 1 if unspecified).

The chosen metrics — sMAPE and MASE — are selected for their scale independence, unbiased nature, and ability to account for data complexity.

Both metrics are suitable for cross-series comparison, particularly important for remittance prices ranging from below 5 to above 30. sMAPE corrects the bias inherent in MAPE, providing a fair scale-independent percentage-based evaluation without favoring low forecasts (R. J. Hyndman & Koehler, 2006). MASE, in contrast, offers insight into a model's performance relative to a naive benchmark, considering the dataset complexity's impact on the degree of inflation of other metrics.

Diverging from common practices, RMSE and R square are excluded due to their volatility and unreliability. RMSE is sensitive to outliers and gives evaluations inconsistent with other established metrics, while R square can misleadingly suggest high explanatory power even with random variables and thus fail to illustrate the model's real competency (Armstrong, 2001).

6.5 *Significance Check*

The median of the metrics are reported to facilitate comparison between traditional and neural network models. Neural networks are non-linear and heavily reliant on numerous stochastic parameters, causing metrics from individual trials to fluctuate due to randomness. To mitigate this, all reported neural network metrics use the median of 20 runs with the same hyperparameters.

Whenever statistical significance is noted for comparisons between neural networks, *deep-significance* is employed, accounting for their deep learning models' mathematical particularities. Given their stochastic nature, relying on additional summary statistics cannot yield definitive conclusions (Dror et al., 2019), hence the use of *deep-significance* to ensure claimed superiority follows rigorous analysis tailored to neural networks.

6.6 *RQ2: Error Analysis by Class*

Building upon the model comparison results, the research proceeds to address RQ2 and RQ3. The best-performing neural network model and the top baseline model are selected based on established performance metrics. For each model, Mean Absolute Error (MAE) is chosen to illustrate the uneven social impact due to the straightforwardness of the metric in real-life interpretations. To assess potential disparate impacts, the errors are aggregated based on two key factors: the receiving countries' geographical regions and the sending countries' economic status (specifically, G8/G20 membership). This grouping strategy is designed to reveal any systematic differences in forecasting accuracy across different global contexts. Other

potential categorizations were not chosen due to extreme class imbalances and negligible numbers in most classes in the dataset.

6.7 *RQ3: Data Disruption Design*

RQ3's experimental design employs data shuffling to investigate the diverging impact of data integrity on CNN's and LSTM's forecasting performance (See Figure 15 in Appendix B for example). For an input window of length k, the following series are generated:

- **Original**: The original ordered sequence from t_n to t_{n+k}.
- **All Shuffled but Most Recent**: A shuffled version of Series 1, with t_{n+k} maintained in its original position.
- **Recent Shuffled, Old Kept**: A group of progressively shuffled series, each disturbing an additional 10% of data points preceding the target.

In this experiment, k is set to be 20, as all three models optimised the input window length to be 20 time stamps. This methodology serves two primary objectives:

1. To assess whether neural networks only replicate recent values, as suggested by Kosma et al. (2022). If it is true, shuffling previous data points in the input window should make no difference. This is evaluated by comparing model performance on the first two series.
2. To "break" the inherent meaningful patterns lying closer to the target, and reveal how far back in time the models effectively utilize information. Gradually introducing disorder into the recent period mitigates the insufficiency of the first experiment that the most recent value is factually very informative in the dataset.

By varying the extent of data corruption, this approach aims to provide a nuanced analysis of how effectively models utilize historical information and their robustness against disruption.

7 RESULTS

This section reports and analyses the forecasting performances of the chosen models on the remittance price dataset from the World Bank. It then examines neural network performance under different data transformations and tests model validity with shuffled input windows.

7.1 *RQ1: Model Comparison*

	Model	MAE	MASE	sMAPE (%)
Baseline	Naive Last Value	**1.14**	**1.11**	*14.56*
	Mean Model	2.26	2.58	29.43
	Naive Drift	1.18	1.16	15.41
Statistical	ARIMA	1.26	1.19	16.04
Neural Network	Optimized CNN	1.15	1.14	14.79
	Optimized LSTM	**1.14**	1.13	**14.48**
	Optimized CNN-LSTM	*1.15*	*1.13*	*14.56*

Table 1: Performance Metrics of Different Models

Table 1 compares the performance of baseline, statistical, and neural network models. The best value is in bold, with the second italised. The Naive Last Value model surprisingly outperforms all others, achieving the lowest MAE at 1.14, MASE at 1.11, and the second lowest sMAPE at 14.56. In contrast, the Mean Model underperforms drastically, with error metrics approximately double those of the best baseline across MAE, MASE, and sMAPE.

As for the representative of statistical models, ARIMA demonstrates mediocre performance, significantly better than the Mean Model but moderately worse than the remaining models and best baseline. Compared to the best baseline, ARIMA roughly underperforms by 7% in MAE and 10% in MASE and sMAPE.

Inspecting the ARIMA parameter distribution reveals that nearly a fifth of the series follow an ARIMA(0,1,0) pattern, where the current value depends solely on the last observed value plus a random component. This indicates *AutoArima* found the most recent value highly informative across the dataset.

Moving on to the neural network models, Table 1 reveals that all neural network models considerably surpass the traditional ARIMA and approach or match the best baseline more closely than ARIMA. Among them, the Optimized LSTM performs on par with Naive Last Value, achieving an sMAPE of 14.48% and MAE of 1.14, though a slightly higher MASE of 1.13 compared to 1.11 for the baseline. As for CNN, it fails to surpass either of the architectures with Recurrent Layers and seems counterproductive in general, even if minimally. The reason is that the hybrid model underperformed the individual LSTM model in terms of MAE and sMAPE with statistical significance. Statistical significance tests show none of the

performance metrics of CNN is statistically significantly better than the other two neural network models. LSTM are statistically significantly better than CNN-LSTM in terms of MAE and sMAPE, but not in terms of MASE despite the same level of performance.

One may notice LSTM's MASE of 1.13 is slightly higher than the Naive Last Value's 1.11, despite outperforming all baselines on MAE and sMAPE. This discrepancy arises because MASE assesses performance relative to the in-sample naive forecast error. When the naive model performs exceptionally well, as with datasets showing minimal fluctuations, its error is low, reducing the MASE denominator and inflating the metric. Accounting for this bias, the LSTM likely surpasses the baseline overall, highlighting its strength in capturing long-range dependencies in financial forecasting.

Lastly, all three model architectures — LSTM, CNN, and the hybrid CNN-LSTM — intriguingly converged on a 20 time-step input window during hyperparameter tuning, indicating that the past 5 years of data provide the richest information without excessive distortion. On top of this, CNN selected a kernel size of 3 quarters, meaning the model will analyze data points in groups of three-quarters to extract features.

7.2 *RQ2: Disparate Error Analysis*

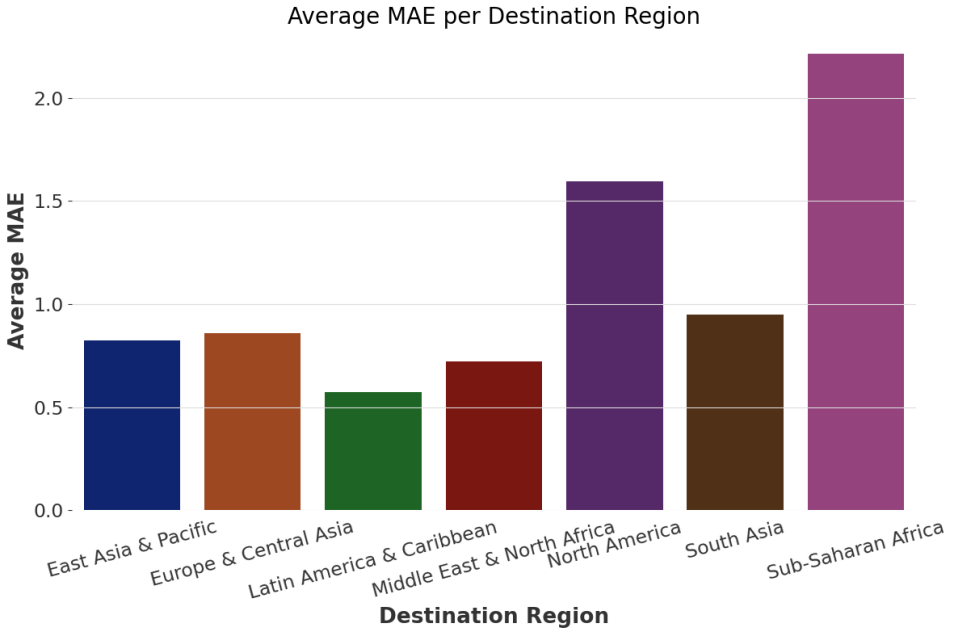


Figure 6: Uneven MAE in LSTM's forecasting across the region of remittance destination region

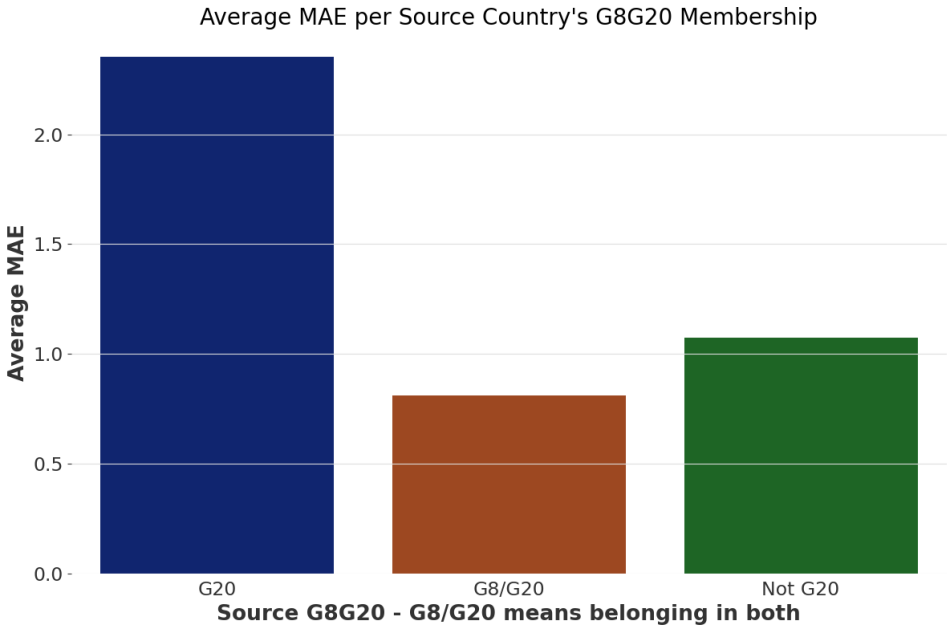


Figure 7: Uneven MAE in LSTM's forecasting per remittance source's economic status

Figures 6 and 7 reveal concerning disparities in the model's remittance cost forecasting performance across different geographic and economic groupings. For most country subgroups, the LSTM model performs decently with an error of around or less than 1% of the total transfer amount. But notably, the model exhibits substantially higher average Mean Absolute Errors (MAEs) when predicting remittance outflows from non-G8 G20 countries (MAE = 2.35) and inflows to sub-Saharan Africa (MAE = 2.21). This means if a worker plans to send 1000 USD, the forecast could deviate from the actual cost by more than 20 USD.

As the baseline on par with LSTM, Naive Last Value shows overall similarly disparate error distribution. Figure 13 and 14 in Appendix B show that the Naive Last Value baseline exhibits a similarly disparate error distribution, with MAEs not diverging from LSTM's by more than 0.1 for most groups, except North America as a destination region. Interestingly, for the two categories where LSTM performed worst - sub-Saharan Africa as destination and non-G8 G20 as source - it outperformed the best baseline by over 0.075 and 0.045 in MAE, respectively.

7.3 *RQ3: Robustness against Data Disruption Analysis*

Table 2 and Figure 8, 9 and 10 illustrate CNN's and LSTM's responses to various data disturbance scenarios, with models re-trained using identi-

Model	Shuffling	MAE	MASE	sMAPE%
CNN	without shuffling	**1.15**	**1.14**	**14.79**
	shuffling all but the last value	1.22	1.28	15.81
LSTM	without shuffling	**1.14**	**1.13**	**14.48**
	shuffling all but the last value	1.14	1.13	14.60

Table 2: Testing the significance of the Last Value

cal optimal hyperparameters. Table 2 reveals the last value's significant contribution to prediction accuracy. Overall, while correct sequencing yielded optimal results for both models, shuffling the initial 19 time stamps (approximately 5 years' data) did not greatly deteriorate performance, with metrics held better than ARIMA. Interestingly, CNN saw a larger deterioration in sMAPE, by 6.9% (from 15.81 to 14.48) compared to what the original series yielded. Meanwhile, LSTM's metrics almost did not worsen.

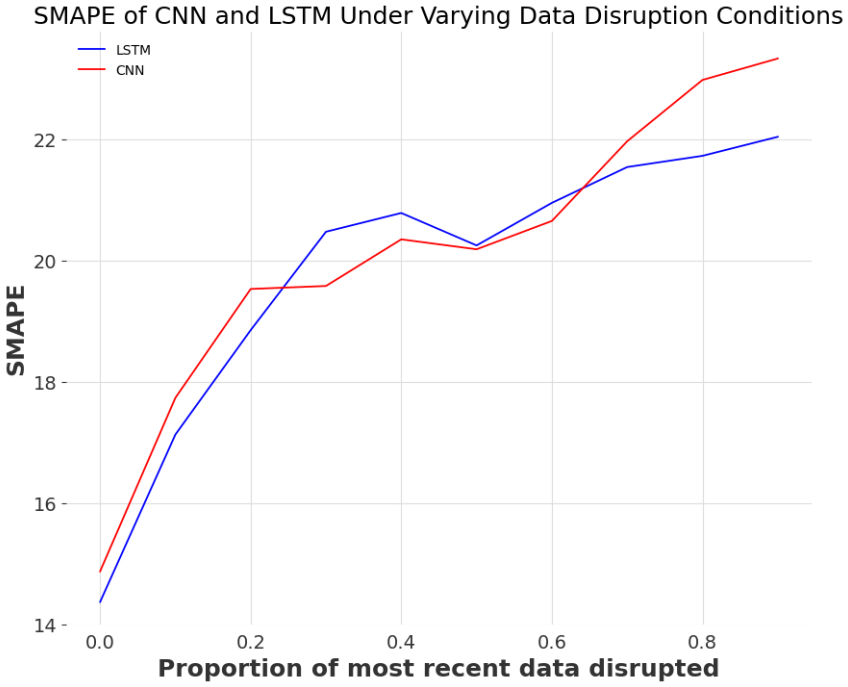


Figure 8: sMAPE of CNN and LSTM under different data disruption regimes

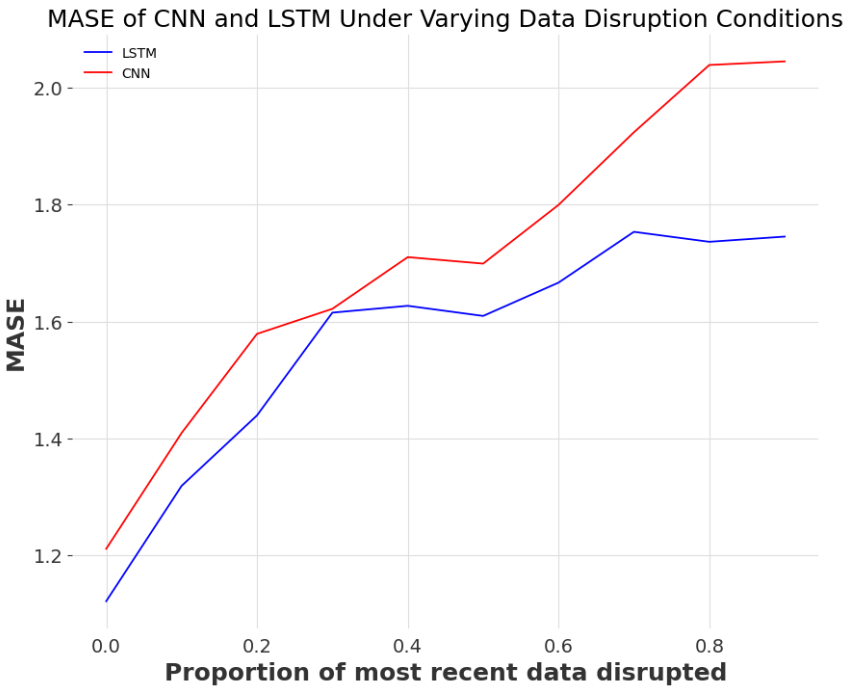


Figure 9: MASE of CNN and LSTM under different data disruption regimes

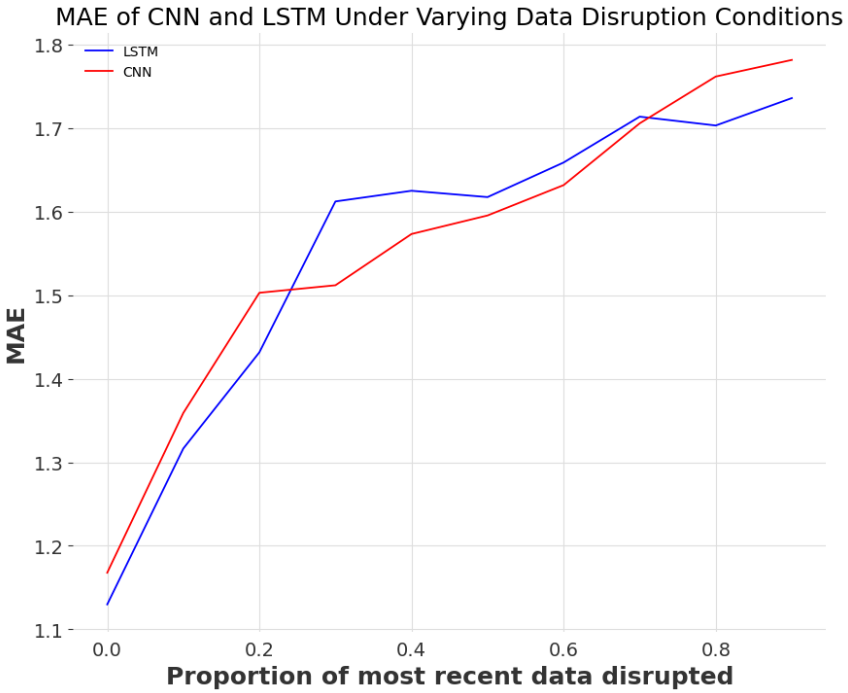


Figure 10: MAE of CNN and LSTM under different data disruption regimes

Both models exhibit an upward trend in error as data disruption increases, indicating declining forecast accuracy. For sMAPE (Figure 8), CNN and LSTM show similar trajectories of performance deterioration. LSTM maintains slightly better performance overall, except in the 30% to 40% disruption range where its performance temporarily drops below CNN before recovering. Aligning closely with sMAPE, MAE trends (Figure 10) show comparable degradation patterns for both models. MASE results (Figure 9) reveal a more pronounced difference between the models. LSTM consistently outperforms CNN across all disruption levels, demonstrating greater robustness as data disruption increases. CNN's MASE rises more steeply, particularly beyond 60% disruption.

The performance deterioration rates of the two models also show marked differences across the metrics. In the case of MASE and MAE, the patterns diverge more distinctly: LSTM's performance drops sharply during the initial 0-30% disruption but stabilizes as disruption increases, indicating its resilience as long as earlier data are intact. Crucially, beyond 40% disruption, CNN's performance declines at a faster rate than LSTM while LSTM maintains relatively stable performance. This divergence is especially prominent in the MASE metric (Figure 9), where LSTM's error starts to plateau around 40% disruption, while CNN's error surges.

8 DISCUSSION

This section provides a discussion of our results in Section 7 regarding the three research questions. For each research question, a concise answer is followed by an in-depth discussion of the contribution to the literature, their scientific limitation or social significance, and how they suggest directions for future research.

8.1 *RQ1: Model Comparison*

- **RQ1**: How well do conventional ARIMA and neural networks forecast the following quarter's remittances given all the preceding data?
- **Answer**: Overall, judged from the forecasting metrics, LSTM seems capable of learning patterns in remittance price data, but the marginal superiority questions the generalizability. In contrast, ARIMA considerably underperforms the best naive baseline.

The results provide new evidence for the contested superiority of neural networks in real-life datasets of remittance prices. LSTM shows significant improvement over ARIMA approaches, which in turn underperformed

the baseline considerably, contrasting with Makridakis, Spiliotis, and Assimakopoulos (2018b)'s findings. Simultaneously, the Naive Last Value model surprisingly outperforms more complex models. This outcome likely stems from the inherent stability of the remittance price dataset, as evidenced by ARIMA's preference to use the last value only.

By comparing LSTM, CNN, and CNN-LSTM, this study extends the work of Siami-Namini and Namin (2018) and Latif et al. (2023) to remittance price forecasting. LSTM's supremacy suggests pure temporal modeling may be more effective than pattern recognition for this data, and that added complexity can be counterproductive, echoing Green and Armstrong (2015)'s finding. Both the baseline's good performance and CNN-LSTM's underperformance contribute to the ongoing debate on model complexity, challenging the trend towards complex models noted by Makridakis, Assimakopoulos, and Spiliotis (2018) and Green and Armstrong (2015), while aligning with R. Hyndman and Athanasopoulos (2018)'s and Fama (1965)'s observations on the efficacy of recent data in economic forecasting.

However, LSTM's marginal gains may not be definitive or generalizable. The improvement is far less than reported in Latif et al. (2023) and Siami-Namini and Namin (2018), and was obtained without conventional data scaling methods. These minor gains may not generalize beyond the specific validation and testing periods used. Regrettably, no consensus exists on the optimal model selection in time-series forecasting, and high generalisability is inherently difficult given external shocks and possible changes in the remittance market structure (Makridakis et al., 2020).

Additionally, the model complexity leads to high computational demand and training time cancelling out such marginal superiority of advanced models (Makridakis et al., 2023). Future research should thus focus on both improved preprocessing and model selection methods for more generalizable performance and cost-efficient architecture for practicality. The latter is vital for countries with higher remittance dependency which usually command fewer resources.

8.2 *RQ2: Disparate Error Analysis*

- **RQ2**: How do the best baseline model and the best-trained model compare in terms of performance across different types of countries?
- **Answer**: Notably higher errors were observed for remittance outflows from non-G8 G20 countries and inflows to sub-Saharan Africa, but LSTM outperformed Naive Last Value slightly in these two groups.

It highlights the risk of focusing on overall metrics at the expense of the social implications of these disparities. Non-G8 G20 nations in-

clude major remittance sources such as oil-rich Gulf states and emerging economies, while sub-Saharan Africa comprises numerous heavily remittance-dependent countries. Thus, the models' limitations in accurately predicting these critical financial flows may disproportionately harm the vulnerable populations.

On the other hand, LSTM encouragingly performed slightly better than the baseline in these two categories. It indicates the utility of complex models for targeted inequality-reducing forecasting. Since global LSTM already shows potential, there is a need for economists to investigate the pattern of co-variation between remittance price series and to develop targeted approaches considering regional and economic factors. This geographical and economic pattern extends the argument of Beck and Peria (2009) and Beck et al. (2022) that remittance price variation correlates with socio-economic variables, suggesting that temporal dynamics may exhibit patterns clustered by development level or geographical proximity, shared among socio-economically similar countries and corridors. As Semenoglou et al. (2021) suggested, cross-learning facilitates pattern extraction and modeling, so could future research explore training specialized models for each category to improve accuracy.

8.3 *RQ3: Robustness against Data Disruption Analysis*

- **RQ3**: To what extent does the corruption of recent input data affect the forecasting performance of neural network models for remittance prices, and how does this impact vary with the length of the corrupted period?
- **Answer**: The most recent input value proves crucial, with model performance remaining stable when it is unaffected. However, when earlier data is progressively corrupted, a more complex picture emerges: LSTM demonstrates greater resilience, particularly under severe disruption, while CNN's performance degrades more uniformly. Such complexity only allows tentative interpretation.

Section 7.3 shows that both CNN and LSTM perform well when the most recent data point remains in place. This finding aligns with the strong performance of the Naive Last Value model and the financial theory (Fama, 1965) discussed earlier, reinforcing the idea that recent data carries significant predictive power in remittance price forecasting. Table 2 further reveals interesting differences between CNN and LSTM. While CNN's performance dropped significantly, disruption of more historical patterns seems not to affect LSTM at all. Thus, it seems to agree with Kosma et al.

(2022) that LSTM indeed is reliant on the last input value facing noisy and scarce data settings, more so than CNN.

Figures 8, 9, and 10 paint a more complex picture. Both LSTM and CNN deteriorated non-linearly. LSTM's superior robustness at higher disruption levels suggests it leverages longer-term dependencies in the earlier data left undisturbed. This ability aligns with LSTM's theoretical strengths in processing sequential data. Conversely, CNN's more consistent degradation indicates its reliance on local patterns, which are progressively disrupted as data corruption increases. The results potentially challenge the conclusion that the model **solely** learned the last value. The observed non-linear deterioration patterns and the differences between LSTM and CNN indicate that both models leveraged information from multiple time steps in complex and different ways.

A key limitation of this study is the uncertain real-world applicability of our simulated data disruption. Notably, LSTM's unexpected outperformance of CNN under severe disruption challenges our intuitions about model behavior with corrupted data. This could reflect LSTM's capacity to capture global statistical properties or be an artifact of the disruption method, where models might treat shuffled data as genuine and attempt to model random noise. This highlights a potential disconnect between the experimental design and real-world data disruptions.

Future research should therefore focus on developing models resilient to real-world data issues. This could involve data collectors and economists identifying authentic disruption scenarios, missing values imputation in different manners, and the exploration of static or textual variables (e.g. policy news) or covariates available to enhance performance and robustness alike. Such efforts would bridge the gap between theoretical modelling and practical applicability in real-world price forecasting.

9 BROADER SOCIAL IMPLICATIONS

The superiority of LSTM over traditional models, albeit modest, offers encouragement for future researchers and could still benefit millions of migrant workers, as even small improvements in forecasting accuracy could translate to substantial savings given the large volume of global remittances. However, the strong performance of the Naive Last Value model means that, in the short run, it remains challenging to develop models that excellently foresee short-term remittance market dynamics. This implication limits the leverage of international agencies and governments in negotiations with remittance service providers. While this market stability may reassure remittance senders and governments, it also suggests resistance to rapid positive changes. Consequently, remittance

users and observers might need to consider non-financial indicators for potential market shifts. Overall, these results emphasize the importance of balancing advanced and simple forecasting techniques in developing strategies to make remittance services more affordable and accessible. The principles of accuracy, equity and practicality should simultaneously guide future model development of remittance prices, ensuring such forecasting benefits the vulnerable most and contributes to financial inclusion and economic development in remittance-dependent communities and regions worldwide.

10 CONCLUSION

This thesis presents the first comprehensive attempt to forecast remittance prices using time-series analysis, providing a valuable baseline for future research. Its findings demonstrate that simple LSTM can forecast the next quarter's remittance prices with marginally better accuracy than using the last known value. The study contributes additional evidence to the debate in financial forecasting, highlighting the superiority of LSTM over other neural network architectures, especially ARIMA, and re-emphasizing the usefulness of the Last Value. The observed disparities in forecasting accuracy across different country groups underscore the need for tailored approaches in remittance price prediction. Furthermore, the investigation into model robustness against data corruption reveals LSTM's greater resilience, but the practical model robustness requires future research into more realistic solutions involving optimal imputation and external data. Overall, the results provide insights for policymakers in managing and predicting remittance costs and are encouraging for future model development, while the remaining questions over generalizability and robustness await deeper investigation.

REFERENCES

Armstrong, J. (2001). Evaluating forecasting methods. https://doi.org/10.1007/978-0-306-47630-3_20

Azlan, A., Yusof, Y., & Mohsin, M. F. M. (2019). Determining the impact of window length on time series forecasting using deep learning. *9*(44), 260–267. https://doi.org/10.19101/IJACR.PID77

Beck, T., Janfils, M., & Kpodar, K. R. (2022). *What explains remittance fees? panel evidence* (IMF Working Papers No. 2022/063). International Monetary Fund.

Beck, T., & Peria, M. S. M. (2009). *What explains the cost of remittances? an examination across 119 country corridors* (Policy Research Working Pa-

per No. 5072). The World Bank. https://documents.worldbank.org/en/publication/documents-reports/documentdetail/678731468176731953/what-explains-the-cost-of-remittances-an-examination-across-119-country-corridors

Bergstra, J., Yamins, D., & Cox, D. (2013, 17–19 Jun). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th international conference on machine learning* (pp. 115–123, Vol. 28). PMLR. https://proceedings.mlr.press/v28/bergstra13.html

Bersch, J., Clevy, J. F., Muhammad, N., Perez Ruiz, E., & Yakhshilikov, Y. (2021). *Fintech potential for remittance transfers: A central america perspective* (IMF Working Papers No. 2021/175). International Monetary Fund. https://www.imf.org/en/Publications/WP/Issues/2021/07/16/Fintech-Potential-for-Remittance-Transfers-A-Central-America-Perspective-50260

Box, G. E. P., & Jenkins, G. M. (1976). *Time series analysis: Forecasting and control* (Revised). Holden-Day.

Box, G. E., & Jenkins, G. M. (1970). *Time series analysis: Forecasting and control*. Holden-Day.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 108–122.

Cao, J., & Wang, J. (2019). Stock price forecasting model based on modified convolution neural network and financial time series analysis. *International Journal of Communication Systems*, *32*(12), e3987. https://doi.org/https://doi.org/10.1002/dac.3987

Chollet, F., et al. (2015). Keras [Version 2.4.0].

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, *74*(366a), 427–431.

Dror, R., Shlomov, S., & Reichart, R. (2019, July). Deep dominance - how to properly compare deep neural models. In A. Korhonen, D. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 2773–2785). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1266

Fama, E. F. (1965). Random walks in stock market prices. *Financial Analysts Journal*, *21*(5), 55–59. Retrieved May 24, 2024, from http://www.jstor.org/stable/4469865

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*, 193–202. https://api.semanticscholar.org/CorpusID:206775608

Fungtammasan, G., & Koprinska, I. (2023). Convolutional and lstm neural networks for solar power forecasting. *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–7. https://doi.org/10.1109/IJCNN54540.2023.10191813

Garza, F., Canseco, M. M., Challú, C., & Olivares, K. G. (2022). StatsForecast: Lightning fast forecasting with statistical and econometric models. https://github.com/Nixtla/statsforecast

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural computation*, *12*(10), 2451–2471. https://doi.org/10.1162/089976600300015015

Green, K. C., & Armstrong, J. S. (2015). Simple versus complex forecasting: The evidence [Special Issue on Simple Versus Complex Forecasting]. *Journal of Business Research*, *68*(8), 1678–1685. https://doi.org/https://doi.org/10.1016/j.jbusres.2015.03.026

Herzen, J., LÃ¤ssig, F., Piazzetta, S. G., Neuer, T., Tafti, L., Raille, G., Pottelbergh, T. V., Pasieka, M., Skrodzki, A., Huguenin, N., Dumonal, M., KoÅ›cisz, J., Bader, D., Gusset, F., Benheddi, M., Williamson, C., Kosinski, M., Petrik, M., & Grosch, G. (2022). Darts: User-friendly modern machine learning for time series. *Journal of Machine Learning Research*, *23*(124), 1–6. http://jmlr.org/papers/v23/21-1177.html

Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, *37*(1), 388–427. https://doi.org/https://doi.org/10.1016/j.ijforecast.2020.06.008

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hyndman, R. (2020). A brief history of forecasting competitions [M4 Competition]. *International Journal of Forecasting*, *36*(1), 7–14. https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.03.015

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/https://doi.org/10.1016/j.ijforecast.2006.03.001

Hyndman, R., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice* (2nd). OTexts.

Jadon, A., Patil, A., & Jadon, S. (2022). A comprehensive survey of regression based loss functions for time series forecasting.

KNOMAD. (2023, June). *Migration and development brief 38* (tech. rep.) (Accessed: 2023-09-20).

Kosma, C., Nikolentzos, G., Xu, N., & Vazirgiannis, M. (2022). Time series forecasting models copy the past: How to mitigate.

Kraehenbuehl, M., & Osterrieder, J. (2022). The efficient market hypothesis for bitcoin in the context of neural networks.

Latif, N., Selvam, J. D., Kapse, M., Sharma, V., & Mahajan, V. (2023). Comparative performance of lstm and arima for the short-term prediction of bitcoin prices. *The Australasian Accounting Business and Finance Journal*, *17*(1), 256–276.

Liu, S., Ji, H., & Wang, M. C. (2020). Nonpooling convolutional neural network forecasting for seasonal time series with trends. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(8), 2879–2888. https://doi.org/10.1109/TNNLS.2019.2934110

Makridakis, S., Assimakopoulos, V., & Spiliotis, E. (2018). Objectivity, reproducibility and replicability in forecasting research. *International Journal of Forecasting*, *34*(4), 835–838. https://doi.org/10.1016/j.ijforecast.2018.05.001

Makridakis, S., & Hibon, M. (2000). The m3-competition: Results, conclusions and implications [The M3- Competition]. *International Journal of Forecasting*, *16*(4), 451–476. https://doi.org/https://doi.org/10.1016/S0169-2070(00)00057-1

Makridakis, S., Hyndman, R. J., & Petropoulos, F. (2020). Forecasting in social settings: The state of the art [M4 Competition]. *International Journal of Forecasting*, *36*(1), 15–28. https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.05.011

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018a). The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802–808. https://doi.org/https://doi.org/10.1016/j.ijforecast.2018.06.001

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018b). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLOS ONE*, *13*(3), 1–26. https://doi.org/10.1371/journal.pone.0194889

Makridakis, S., Spiliotis, E., Assimakopoulos, V., Semenoglou, A.-A., Mulder, G., & Nikolopoulos, K. (2023). Statistical, machine learning and deep learning forecasting methods: Comparisons and ways forward. *Journal of the Operational Research Society*, *74*(3), 840–859. https://doi.org/10.1080/01605682.2022.2118629

Nesreen K. Ahmed, N. E. G., Amir F. Atiya, & El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, *29*(5-6), 594–621. https://doi.org/10.1080/07474938.2010.481556

Olah, C. (2015). *Understanding lstm networks* [Image from the blog]. Retrieved May 19, 2024, from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-beats: Neural basis expansion analysis for interpretable time series forecasting.

Semenoglou, A.-A., Spiliotis, E., Makridakis, S., & Assimakopoulos, V. (2021). Investigating the accuracy of cross-learning time series forecasting methods. *International Journal of Forecasting*, *37*(3), 1072–1084. https://doi.org/https://doi.org/10.1016/j.ijforecast.2020.11.009

Shi, J., Jain, M., & Narasimhan, G. (2022). Time series forecasting (tsf) using various deep learning models.

Siami-Namini, S., & Namin, A. S. (2018). Forecasting economics and financial time series: ARIMA vs. LSTM. *CoRR*, *abs/1803.06386*. http://arxiv.org/abs/1803.06386

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting [M4 Competition]. *International Journal of Forecasting*, *36*(1), 75–85. https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.03.017

Spiliotis, E., Kouloumos, A., Assimakopoulos, V., & Makridakis, S. (2020). Are forecasting competitions data representative of the reality? *International Journal of Forecasting*, *36*(1), 37–53.

The World Bank. (2024). *Remittance prices worldwide* [Available at: http://remittanceprices.worldbank.org]. http://remittanceprices.worldbank.org

Ulmer, D., Hardmeier, C., & Frellsen, J. (2022). Deep-significance-easy and meaningful statistical significance testing in the age of neural networks. *arXiv preprint arXiv:2204.06815*.

United Nations. (2023). Global issues: Migration [Accessed: 2023-05-10]. https://www.un.org/en/global-issues/migration

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2

Werbos, P. (1970, January). Applications of advances in nonlinear sensitivity analysis. https://doi.org/10.1007/BFb0006203

APPENDIX A

Field	Description
id	id number applied to each service
period	data collection period
source	country where money is sent from
destination	country where money is sent to
_code	ISO 3166-1 alpha-3 country code
_name	country name
_region	country's World Bank region
_income	country's income group
_lending	country's lending category
_G8G20	country's membership of G8 and/or G20 group
firm	remittance service provider offering the service
firm type	type of remittance service provider offering the service
product	type(s) of product offered
speed actual	time it takes for the money to be available for the receiver
sending location	type of location where the service is available
cc(X) denomination amount	surveyed amount sent in USD
cc(X) lcu amount	surveyed amount sent in local currency of the sending country
cc(X) lcu code	local currency of the sending country
cc(X) lcu fee	fee in local currency
cc(X) lcu fx rate	foreign currency exchange rate applied to the transaction by the remittance service provider
cc(X) fx margin	percentage difference between the foreign currency exchange rate applied to the transaction by the remittance service provider and the interbank exchange rate
cc(X) total cost %	total cost of the transaction in percentage
inter lcu bank fx	interbank exchange rate
transparent	if yes, indicates that the RSP provided the researcher with the exchange rate applied to the transaction; if no, indicates that this information was not provided
standard note	provides additional information on the service
standard note 2	provides additional information on the service
coverage	ranks the extensiveness of the network in the receiving country
pickup method	indicates how the money can be picked up in the receiving country
date	day when the information was collected
NOTE: as of Q2 2016, categories changed as follows	
product	dropped
sending location	renamed "access point"
coverage	renamed "receiving network coverage"
payment instrument	NEW: instrument used by the sender to pay for the transaction
sending network coverage	NEW: ranks the extensiveness of the network of the firm in the sending country

Table 3: Description of the Remittance Prices Worldwide dataset fields

corridor	ZAFZWE			
firm	**MoneyGram**	**Mukuru**	**Nedbank**	**SFX**
timestamp				
2011-01-01	11.31	NaN	17.80	NaN
2011-07-01	13.15	19.86	17.24	NaN
2012-01-01	10.57	10.00	18.26	NaN
2012-07-01	11.37	19.63	17.73	NaN
2013-01-01	12.64	14.82	18.23	NaN
2013-04-01	9.64	12.955	17.37	NaN
2013-07-01	13.37	14.62	19.05	NaN
2013-10-01	11.54	12.20	18.14	NaN

Table 4: Remittance Prices for ZAFZWE corridor by different firms over time

Software Versions	
Python Version	3.10.13 [Clang 14.0.6]
NumPy Version	1.26.4
Pandas Version	2.2.1
SciPy Version	1.13.0
Scikit-learn	1.2.2
Statsmodels Version	0.14.2
Keras Version	2.11.0
Hyperopt Version	0.2.7
TensorFlow Version	2.11.0
Statsforecast Version	1.7.4

Table 5: Versions of software employed in the script

Hyperparameter	Searched Values	Best Value
Input window length	16, 20, 24, 28	20
Number of fully-connected layers	1, 2	2
Number of nodes n	8, 16, 32, 64	16
Optimizer	adam, rmsprop, sgd, nadam	nadam
Kernel size	3, 5, 7, 9	3
Learning rate	0.0001 - 0.0010 (increments of 0.0001)	0.0005

Table 6: Hyperparameter Values and Best Parameters of CNN Model

Hyperparameter	Searched Values	Best Value
Input window length	16, 20, 24, 28	20
Number of LSTM layers	1, 2, 3	1
LSTM units	16, 32, 64, 128	128
Optimizer	adam, rmsprop, sgd, nadam	adam
Learning rate	0.0001 - 0.0010 (increments of 0.0001)	0.0005

Table 7: Hyperparameter Values and Best Parameters of LSTM Model

Hyperparameter	Searched Values	Best Value
Input window length	16, 20, 24, 28	20
Number of LSTM layers	1, 2	1
Number of filters	8, 16, 32, 64, 128	64
Optimizer	adam, rmsprop, sgd, nadam	adam
Learning rate	0.0001 - 0.0010 (increments of 0.0001)	0.0007
Kernel size	3, 5, 7, 9	5

Table 8: Hyperparameter Values and Best Parameters of CNN-LSTM Model

APPENDIX B

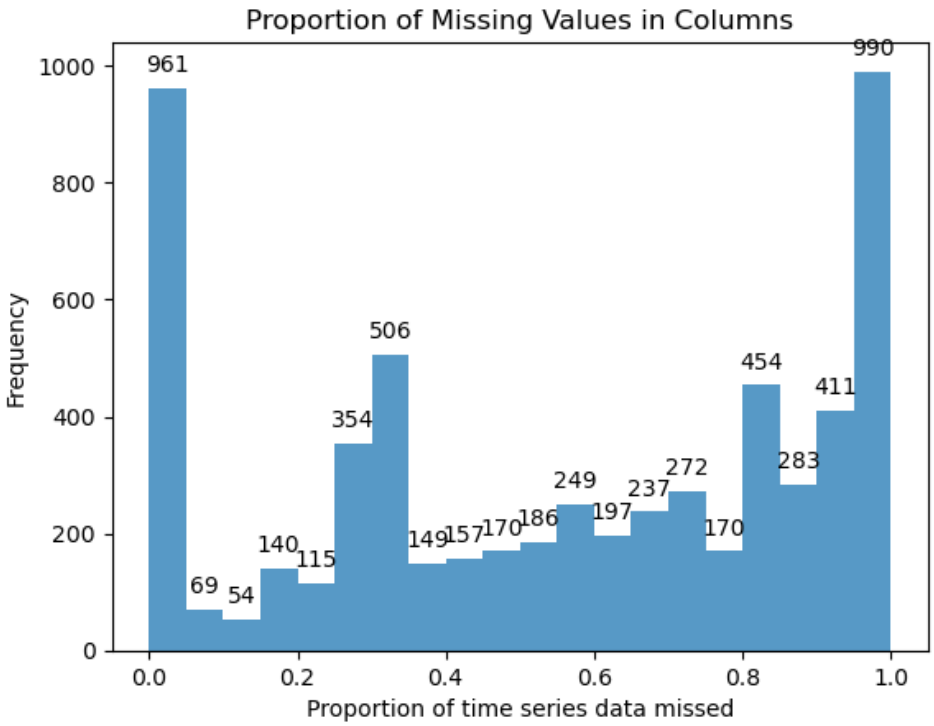


Figure 11: Missing Data Distribution of the Time-Series

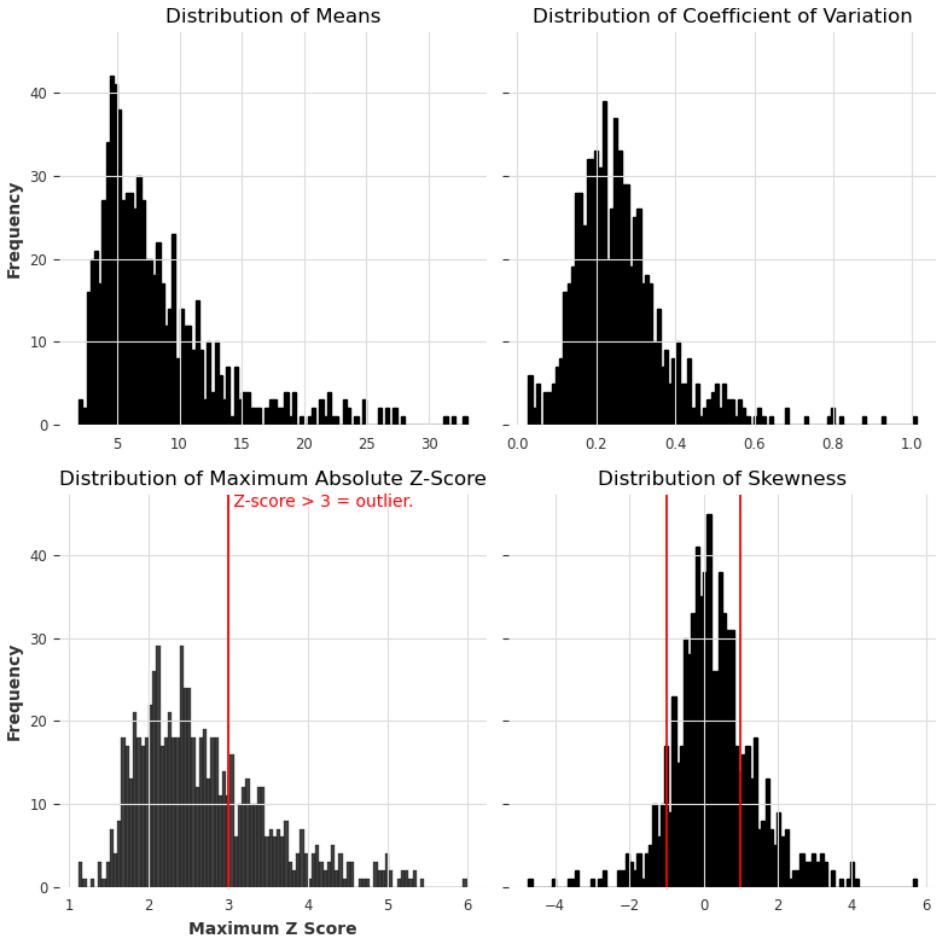


Figure 12: The distribution of mean, coefficient of variation, z-score and skewness values of all the series

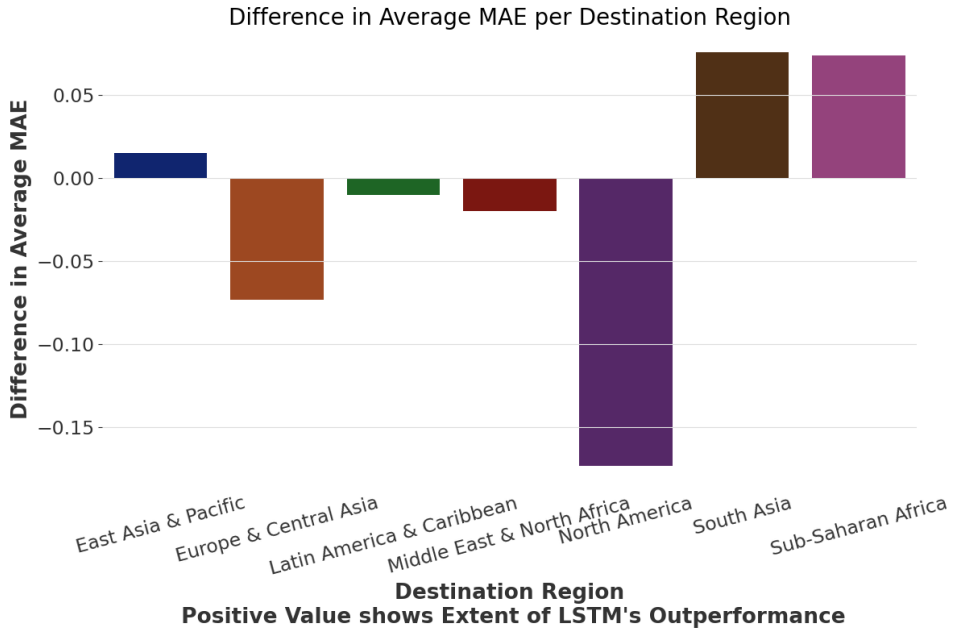


Figure 13: Difference in MAE between LSTM and Naive Last Value across remittance destination regions

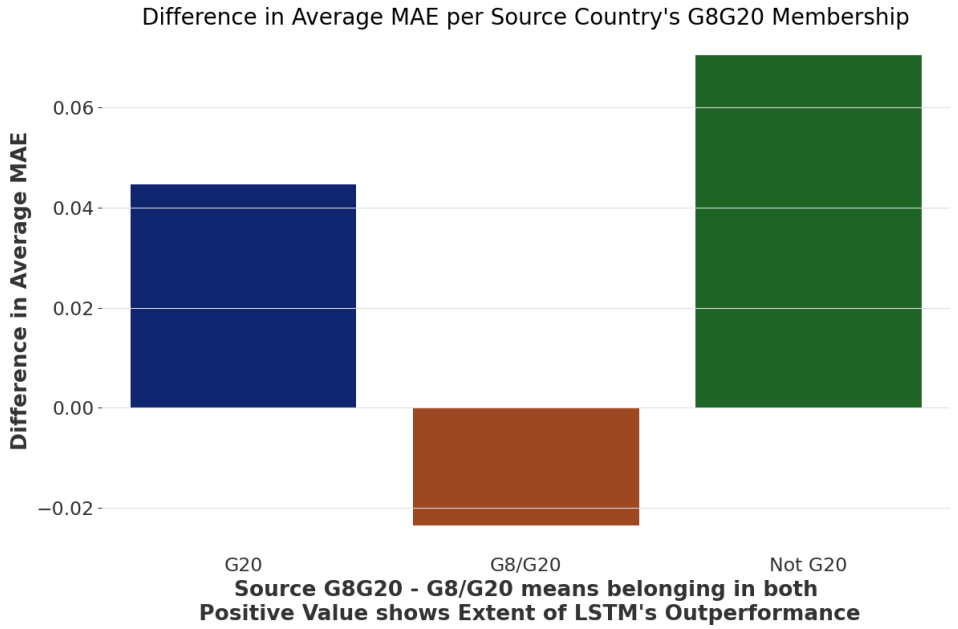


Figure 14: Difference in MAE between LSTM and Naive Last Value per remittance source's economic status

Series 1 - Original	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Series 2 - Shuffling all but the Last	4	7	16	5	2	9	19	3	1	10	13	17	11	14	18	6	12	15	8	20
Series 3 - Shuffling the most recent 10%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	20	19
Series 4 - Shuffling the most recent 20%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	19	20	18	17
Series 5 - Shuffling the most recent 30%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	19	20	17	18	15	16
Series 6 - Shuffling the most recent 40%	1	2	3	4	5	6	7	8	9	10	11	12	20	16	19	15	17	18	13	14
Series 7 - Shuffling the most recent 50%	1	2	3	4	5	6	7	8	9	10	19	20	12	11	15	17	18	16	13	14
Series 8 - Shuffling the most recent 60%	1	2	3	4	5	6	7	8	14	9	19	20	12	11	15	17	18	16	13	10
.....										.....										

Figure 15: Data Shuffling Example - Highlighted numbers have been shuffled

APPENDIX C

Mathematics behind ARIMA

This section introduces the mathematics of ARIMA model, following the official documentation of the library used (Garza et al., 2022). More concretely, ARIMA(p, d, q) is expressed as follows:

The Autoregressive (AR) part looks at the past p values of the time-series to forecast the next value by combining the past values in a weighted manner plus a constant:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

where:

- c is a constant.
- y_t is the actual value at time t.
- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients.
- $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$ are the previous p values of the time-series at times $t-1, t-2, \ldots, t-p$.
- ϵ_t is the random error term.
- p is the number of past values being used.

The Moving average (MA), in contrast, expresses the value for prediction as a weighted combination of past q:

$$y_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

where:

- y_t is the actual value at time t.
- ϵ_t is the random error term at time t and it is recursively calculated as $\varepsilon_t = y_t + \theta \varepsilon_{t-1}$.
- $\theta_1, \theta_2, \ldots, \theta_q$ are the moving average coefficients.
- $\epsilon_{t-1}, \epsilon_{t-2}, \ldots, \epsilon_{t-q}$ are the previous q random error terms at times $t-1, t-2, \ldots, t-q$.
- q is the number of past values' error terms being used.

In the general form of the ARIMA(p, d, q) model, p and q are parameters used to define how many past values and errors one wants to use, respectively and d means the series is differenced d times - differencing is the replacement of each time point's value by the difference between the current value and previous one. For ARIMA(p, 1, q), the equation is:

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{1}$$

where additionally:

- y'_t is the value after differencing, and the series can be difference more thann once.
- $y_t = y_{t-1} + y'_t$ is how the current value is forecast.
- The rest remains the same.

Tuning ARIMA Manually

First and foremost, ARIMA's assumption requires stationarity testing with Augmented Dickey-Fuller (ADF) test (Dickey & Fuller, 1979). In case of a non-stationary series as in Graph 16, the p-value, larger than 0.05, shows statistically significant non-stationarity, accompanied by an autocorrelation plot decreasing slowly. To make it stationary, differencing is applied until autocorrelation drastically drop from one and revolve around zero. Graph 16 shows the effect of differencing (d = 1) creating a stationary sequence.

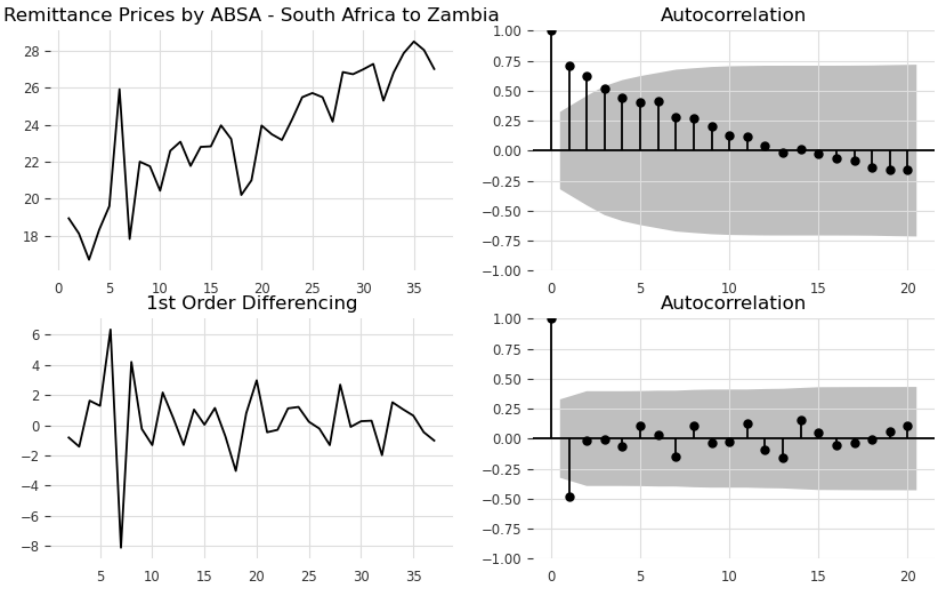


Figure 16: Nonstationary Time-series before and after Differencing with ACF plot (bottom right)

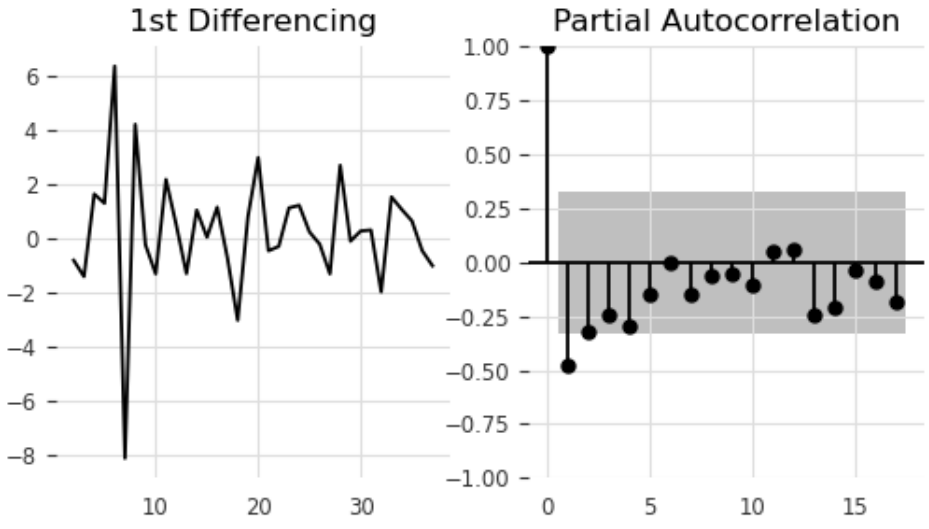


Figure 17: PACF plot

After determining the d, p should be chosen by examining the partial autocorrelation function (PACF) plot. p should be set to the lag at which the PACF drops off sharply to capture the strongest linear dependencies in the data. In Figure 17 it is set to 1. Similarly, the parameter q is determined from the autocorrelation (ACF) plot - as shown in 16, q should also be 1. Thus, so far, the ARIMA model for ABSA (South Africa - Zambia) is set down as ARIMA(1, 1, 1). Nonetheless, tuning p and q can be challenging due to the potential presence of multiple statistically significant lags, and testing and tuning series by series for a large dataset is time-consuming.

LSTM

At each time step t, the cell state passes the previous time step's hidden state to three gates, which decide how to process the past information. These gates are the Forget Gate, the Input Gate, and the Output Gate (Figure 2).

The first gate is the **Forget Gate**. This gate determines how much of the previous cell state c_{t-1} should be forgotten using a sigmoid function. The previous hidden state h_{t-1} and the current input x_t are transformed into f_t, a value between 0 and 1. If f_t is 0, the previous cell state is completely forgotten; if f_t is 1, it is entirely retained. Mathematically, this is expressed as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

- f_t: Forget gate output at time step t
- W_f: Weight matrix for the forget gate
- h_{t-1}: Hidden state at time step $t-1$
- x_t: Input at time step t
- b_f: Bias vector for the forget gate
- σ: Sigmoid activation function

Next, the previous hidden state h_{t-1} and the current input x_t pass through the **Input Gate**. This gate has a sigmoid layer that determines which values to update, i_t, and a tanh layer that creates the new cell state $\tilde{c}_t$ containing this time step's information. The product of i_t and $\tilde{c}_t$ via weights and biases updates the current cell's state value. The process is described by the following equations:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

where:

- i_t: Input gate output at time step t
- W_i: Weight matrix for the input gate
- $\tilde{c}_t$: Candidate cell state at time step t
- W_c: Weight matrix for the candidate cell state
- b_i: Bias vector for the input gate
- b_c: Bias vector for the candidate cell state
- tanh: Hyperbolic tangent activation function

Lastly, the **Output Gate** filters the cell state and calculates what part of the current cell state c_t should be output as the current hidden state h_t. The cell state c_t is updated by combining parts of the previous cell state c_{t-1} (based on the forget gate f_t) and parts of the new candidate values $\tilde{c}_t$ (decided by the input gate i_t). This can be mathematically expressed as:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(c_t)$$

where:

- c_t: Cell state at time step t
- c_{t-1}: Cell state at time step $t-1$
- o_t: Output gate output at time step t
- W_o: Weight matrix for the output gate
- b_o: Bias vector for the output gate
- h_t: Hidden state at time step t