COMP 4107 - Project Report
Corn-Oracle: A Corn Futures Prediction Model

Abdullah Aswad - 101228852

Saad Sheikh - 101209067

Owen Lucas - 101226679

Statement of contributions: Did each group member make a significant contribution: Yes

Did each group member make an equal contribution: Yes

Which aspects did each member contribute to? All members participated in some way to dataset gathering, pre-processing, model training and optimizing, and the project report

Background & Motivation

Corn is one of the most widely cultivated crops globally. It is widely renowned for its usage not only as a source of food for humans and animals alike, but also as biofuel as it is the main material used to make ethanol. (Nebraska CornBoard, 2022) Therefore, it should come as no surprise that corn is a primary driver in the agricultural market worldwide (especially in the United States, the most corn producing nation in the world) and is becoming increasingly valuable as research into biofuel-based energy sources proliferates. As such, forecasting the trading prices for the purposes of arbitrage and contract trading could be extremely advantageous to any prospective trader in agricultural markets.
Market predictors aren't particularly novel in today's world, with quantitative trading firms developing automated strategies to gain an edge over the market when it comes to equities and options. However, futures trading, particularly for agricultural commodities like corn, remains relatively underexplored in machine learning-driven finance. For this reason, we see a lot of potential in gaining an advantage, or "alpha" over other traders. Therefore, we've decided to analyze corn futures while also observing weather data from Meteo's weather API and historical US corn prices using yahoo's yfinance library.

1 Prior Works

As previously explained, research in financial forecasting, particularly when it comes to machine learning, is innumerably wide. Therefore, analyzing prior works pertinent to our project's field is essential to figuring out what works best for our model when it comes to creating the dataset, training and optimizing the model, as well as choosing what parameters yield the greatest performance. Here is what we found from delving into some of these articles.

1.1 Regression Methods on Crop Financial and Weather Data.

The close price of corn futures, an agricultural product, is heavily influenced by the climate of the land the plant grows in. As such, a study published for Frontiers Research Topics (*V. Konduri et al. 2020*) addresses key gaps in understanding the impact of weather on crop yield. It introduces non-linear and ridge regression models to capture and reveal the significance of weather when it comes to predicting crop yield. From their results, the researchers find that non-linear models generally outperform linear approaches in said predictive accuracy. Coincidentally, this study also uses corn data to demonstrate the crop performance's dependence on weather by using historical weather information as a source of data to train the model on.

1.2 Neural Network Methods.

For the sake of building the best model possible, it is important to understand what sort of research and models created are already out there. Accordingly, a paper was written to provide a comprehensive review of several machine learning approaches for agricultural price prediction by analyzing 27 recent and credible papers (*N.Q. Tran et al, 2023*). Utilizing the PRISMA methodology, the authors identify neural network based methods as the most prevalent. Mostly, LSTM is the most used type of neural network for time series forecasting. This paper underscores the increasing adoption of hybrid techniques, and the incorporation of weather data and market trends into the trained dataset.

1.3 Hybrid Models.

Another study was written to analyze the effectiveness of different types of models in price forecasting through performing comprehensive comparative analysis of using four machine learning models (MLP, ELM, NNAR, and XGBoost), four time series models (ARIMA, AR, NP-AR, SES), and a hybrid model based on simple averaging (*M. Qureshi et al, 2024*). For this study specifically, these models are trained on and instructed to predict Bitcoin (BTC-USD) prices. After conducting three train-test splits (70/30, 80/20, 90/10) to evaluate model performance, the authors found that the hybrid model consistently outperformed the other individual models on multiple error metrics and directional accuracy.

1.4 Comparing Traditional to ML Methods

Finally, we reviewed one last study which presents a comparison between three different machine learning-based time series models for forecasting oil production in shale reservoirs. (*Y. Ning et al, 2022*) Namely, ARIMA, LSTM, and Prophet were the three models being compared. The authors address the limitations of traditional methods to time-series forecasting, and in-turn, highlight the strengths of these models in handling fluctuating production trends. As a result of their analysis, the authors found that the Prophet model captures seasonal effects like winter impacts the best, while LSTM handles the non-linearity of data well. Finally, ARIMA seems to show the most robust results for short-term forecasts.

Overall, while our implementations had their own stylings and usages, a lot of what was created and tested came from studies like these. Particularly when it comes to our best performing model, utilizing the power of TS2Vec (*Z. Yue et al, 2022*) in order to predict the close price of corn futures. We will delve into our different methods later on in this report.

2 Objectives

- Finding a dataset of daily weather features
- Finding a dataset of historical US Corn Features
- Aggregating data and applying some pre-processing
- Tuning Hyperparameters to optimize the models
- Training and validating the models
- Developing a machine learning model that predicts the corn futures market prices given a specific day or span of days.

3 Datasets

Predicting the future market for an agricultural product such as corn requires data on both corn prices and the main factors that affect corn production, such as weather data. For the sake of our model, we intended to use a mix of weather and financial data and combine them together. As most forecasting models and non-intraday traders operate on a time resolution over days, we also chose that level of granularity for our data, so our predictions and training data points exist in the space of days. All data go up to March 31st, 2025.

Our primary features are derived from historical US corn prices using yahoo's yfinance library. Unsurprisingly, the model needs previous corn market data to predict future corn market data. As the furthest Yfinance can go back for corn is August 1st, 2000, all other datasets do not go further back than this. Dataset origin: `https://finance.yahoo.com/quote/ZC%3DF/`

Our secondary dataset is weather data from Meteo's weather API. We are using this data because the climate of a land dictates what plants the land can produce, and as such, controls the supply available and in turn, the demand as well. We specifically picked Iowa's weather as it is the state that produces the most corn in the USA. Dataset origin: `https://open-meteo.com/`

4 Methods

4.1 RNN Model

Firstly, as this is a time series forecasting task, a many-to-one LSTM model was implemented as seen in prior works (*Y. Ning et al, 2022*) (*N.Q. Tran et al., 2023*), taking in data over a fixed window of days and predicting a future data point to forecast a number of days into the future, a common autoregression formulation. With multiple stacked LSTMs, the motivation was for it to capture temporal correlations between features and be trained on the 1-day forecast, predicting all prices and weather features to forecast further than one day. The architecture is a simple stacked LSTM with a single MLP layer (Appendix C) with the same output dimensions as the number of input features. In terms of drawbacks, RNN architectures are slower as a result of the recursive nature limiting parallelizability, however, the vanishing gradient problem is amended by the LSTM layer. The exploding gradient problem still remains a potential issue with the architecture. The main benefit with this model is its ability to capture long term causal dependencies in the data, as well as the relations between different features such as weather and close prices.

4.2 TS2Vec with Linear Regression.

The second approach used TS2Vec (a convolutional encoder model using a self supervised, contrastive loss objective) to encode a fixed window of days into a single vector representing the entire sequence and condensing information into a smaller vector space. Using ridge regression as per the original paper, we fit a linear model using the entire vector to each feature of the 1-day targets independently, assuming no multicollinearity. (using an MLP model may improve performance, as our features are highly correlated.) To forecast, the outputs of the ridge regressor are appended to the fixed window, pushing the 1st day out and encoding the window again and predicting the next day using the encoding vector. The main drawback here is that the linear model lacks the ability to capture nonlinear relations within the encoding and 1-day forecast targets, which could be amended with an MLP processing the encodings instead. In terms of benefits, the TS2Vec model is well suited to capturing causal relations between features and fit models are readily transferable to any decoder architecture, whether it's an RNN or MLP. Additionally, the model can be extended to make either a full-series encoding or a per-sample sequence of encodings, the latter being a format usable for RNNs and transformers. The simplicity and robustness (due to L2 regularization) of the ridge regressor decoder makes training much faster and simpler, without the need to specify learning rate, epochs and so on that a neural network decoder would require.

4.3 Triple Exponential Smoothing Model

Holt-Winters Triple Exponential Model is essentially a filtering algorithm (think signal processing) with 3 components approximating 3 exponentially smoothed values (with respect to a univariate

time series) over time: level, trend, and seasonality. Using just the close price of corn, level is the smoothed current value of the close price, while trend is the change in close prices between days. Level can be thought of as the underlying variable, while trend is the 1st order derivative. Seasonality introduces a variable that modulates the level, while being influenced by past level and trend values. It could be thought of as a first order linear differential equation. Unlike the previous methods, there is no fixed window, and involves very few hyperparameters to configure. The parameters it learns are simply 3 smoothing factors: gamma, beta and alpha which correspond to seasonality, trend, and level respectively. The model was acquired through the statsmodels library and training was handled within the function call. The model is fit using SSE and is much simpler than the previous methods. Since corn prices are a univariate time series with trends and literal seasonality as it pertains to growing seasons, this model is straightforward and well suited to our task of predicting corn prices. However, with the lack of complexity and capability to capture correlations in multivariate data, the model may struggle as historical corn prices are not the sole predictor of future corn prices.

5 Validation Strategy & Experiments

Since this is a time series forecasting problem, we did not use the traditional train-test split shuffling method, which would break the causality between data points. Thus, to validate, we kept each split contiguous in terms of time and tested on periods of time after the train data to avoid data leakage. We defined the period from January 1st 2025 until March 31st 2025 as the test period. We used the span of time from August 1st 2000 to December 31st 2024 as the train/validate split, with varying proportionality. For training, we used the MSE between forecasted values and actual values as a performance metric, but since the objective of this project is predicting corn prices, we used close price MSE as the metric during testing and validation.

5.1 Triple Exponential Smoothing Model

As this model optimizes using the statsmodels library employing BFGS for optimization, consisting of a single call of a fitting function, validating different models of Holt-Winters proved difficult as the BFGS iteration was being done within the library. Besides changing the smoothing hyperparameters and evaluating it on MAE, optimizing this model was simple and experiments were limited.

5.2 RNN Model

Using the TimeSeriesSplit method from scikit-learn, we first validated the performance of the LSTM model using 2 stacked LSTMs and a single MLP layer, which resulted in very poor performance with MSE train and validate loss in the hundreds of millions over all sequences per epoch, only marginally decreasing as training went on. Graphing the results showed a near constant flat line in the 2025 autoregression task (see in results), with the predictions being nowhere near the mean of the 2025 data. This result may have been the cause of unnormalized data, as the network must learn the scale and shift of the distribution of financial and weather data that range from 0 to 1e4. Even after standardizing the training data, the forecasting task produced a line marginally closer to the mean of the test data, but a constant line with no slope nonetheless. By the suggestion of Prof. Holden, the lack of decreasing train loss indicated underfitting, which necessitated increasing the network depth and layer sizes (see appendix E and C). However, this only improved the proximity of this predicted line to the actual mean of the 2025 data, and required an abnormally high learning rate to achieve within a reasonable training budget.

5.3 TS2Vec Linear/Kernel Regression Model

To find good hyperparameters, we validated on the last 10% of 2000 to 2024, fitting TS2Vec on the train data, then encoding a sliding window of 30 days across the entire training set in order to fit the encodings to the actual 1-day target values with a linear model. 20 Epochs with 2500 iterations at a learning rate of 0.001 were found to be good hyperparameters for training. The initial hyperparameters (Appendix D) served as a baseline and were configured based on the original TS2Vec paper and our intuitions about the nature of our multivariate sequence data. During validation, this configuration resulted in an MSE of around 1000 across the prediction of all features (on solely financial data), which was not indicative of over or underfitting even when graphed. Similar to the original paper

(Yue et al, 2021), we found that identifying the epoch with the best representational learning for downstream tasks was difficult, and could not identify over/underfitting in evaluation, only the absence of it through test loss metrics. Since TS2Vec's training objective produces a model capable of embedding similar sequences closely in latent space, and would theoretically lead to more linearly separable representations than the full feature space with increasing separability and expressiveness as encoding dimensions increase, 2 other models were attempted with opposing hypotheses: firstly, decrease the encoding dimensions from 128 to 16 and attempt to create encodings that were less expressive, in order to reduce overfitting of encodings to train data. If this reduced validation loss, then it would provide evidence for the overfitting hypothesis. The second configuration raised encoding dimensions from 128 to 256, and reduction in validation loss would prove the underfitting hypothesis to be likely, as a larger vector space would accommodate the learning of complex data correlations and greater separation of encodings in latent space. The smaller encoding size reduced the validation loss somewhat, while the larger encoding size significantly reduced it down to 700. Both settings are evaluated in the test on 2025 data, including a kernel regression version for the 256 dim encoding, to see if any further improvement could be gained through introducing nonlinear approximation (see results)

5.4 Data Features

We started off by acquiring all the features we felt would directly impact the corn price, from both the yfinance and weather api (Appendix A.1). These features would make up the dataset for the initial linear TS2Vec model. The model resulted in an MAE of 36.3 between the forecast predictions and the actual closing price over the course of 2 months. Relative to our other attempts, our results were quite accurate. However, due to its erratic fluctuations while forecasting, it could very much be improved. An approach we used to address these volatile predictions was to reduce redundant features which in turn would reduce multicollinearity, which would stabilize our predictions as the features would be less correlated and prevent overfitting as parameters and noise in accumulated gradients are reduced. Examples of such redundant features would be the "rain_sum" and "snowfall_sum", when features such as "precipitation sum" would already capture both; potentially leading to inflated predictions. After training on different configurations of features, we arrived at one which was significantly less erratic, at the expense of performing much worse on its predictions relative to the actual closing price with a MAE of 80.2 (Appendix A.2. To provide historical context beyond the current input sequence, we introduced a 30-day moving average. To avoid losing responsiveness to recent fluctuations and maintain prediction accuracy over shorter time frames, we also added a 7-day moving average. This combination aimed to help the model balance long-term trend detection with short-term precision. Through iterations of train-validate-test splits, we found that a set of combined weather and financial data with additional moving averages for certain features provides the best results for forecasting with a MAE of 25.7. A specific list of the finalized features employed can be viewed in Appendix A.3. The addition of moving averages gives a smoothed and less erratic measure of price and weather, and also enables the model to take in data past its fixed window. As an example, the moving average of price on the 1st day of a month implicitly includes price data from the last month.

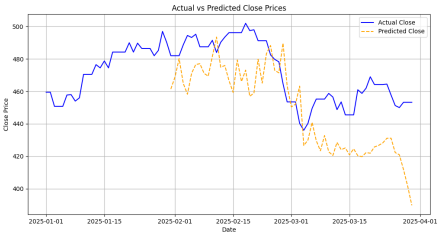
5.5 Results

The following results on forecasting prices from 2025-01-01 to 2025-03-31 only start forecasting from 2025-01-30 onwards as all models besides Holt-Winters required at least a window of 30 days. All models forecast for 59 days, and the MAE values are calculated from that span of time. We introduce a measure of smoothness/erraticity for predictions and errors (henceforth referred to as MPV and MEV, respectively) (appendix F.1 and F.2) during final evaluation not used during validation and training, since all neural network models were trained on one-step predictions. **Note: the mean variation for the close prices in this test data is 3.728.**

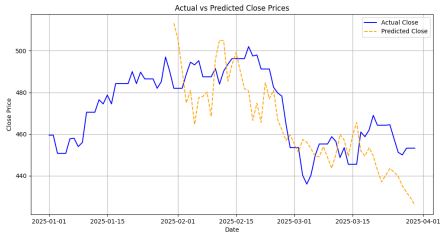
Variant	Initial Linear Model	Encoding(-) w/Linear	Encoding(+) w/Linear	Encoding(+) w/Kernel
MAE	24.285	**13.729**	19.956	28.385
MPV	8.661	7.031	5.746	**5.271**
MEV	10.203	8.633	7.358	**6.787**

(a) LSTM and TES metrics excluded from comparison to lack of viability.

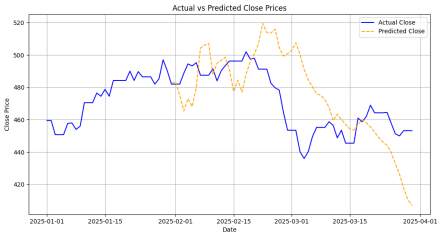
Figure 1: Test Criteria for TS2Vec Models. Best values in Bold.



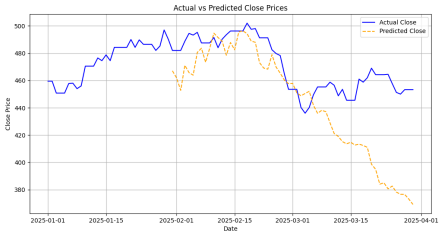
(a) Initial Model Forecast (Appendix D.1)



(b) Reduced Encoding (Appendix D.2)

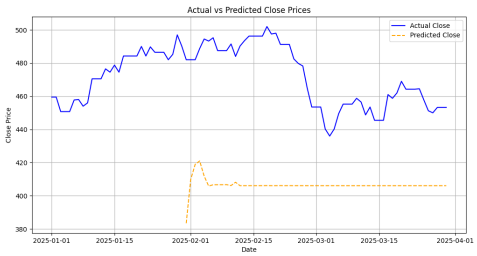


(c) Increased Encoding (Appendix D.3)

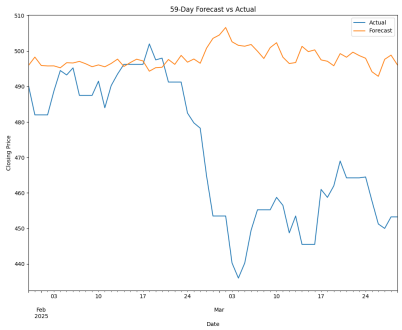


(d) Increased Encoding + Kernel Regression (Appendix D.3)

While mean absolute error is a standard measure used in assessing deviation from ground truth, we stress that it is not the defining factor in the quality of a forecast. Though the reduced linear encoding scheme produced the lowest MAE of all variants of this model, the variance in its predictions and errors far outpace those of all other models, save for the initial model which used reduced network size. This indicates, quantitatively, the erraticity and lack of viability of this forecasting model. In comparison, we show that the linear and kernel variations of the increased encoding models are much less erratic, with the kernel variation producing qualitatively much more accurate forecasts in the 30-day region than any other model with the lowest variation in both error, prediction, and difference in actual data variation vs prediction variation, which could be used as a heuristic in determining if the predictions are similarly distributed to the ground truth. In general, all of these models far outperform the LSTM and Holt-Winters models, either due to potential training bugs, or lack of predictors/features available to the model.



(a) Improved LSTM Model (Appendix E)



(b) Univariate Holt-Winters Model

Variant	Holt-Winters TES	Improved LSTM
MAE	27.075	64.338
MPV	Not Calculated	1.009
MEV	Not Calculated	4.591

Figure 4: Test Criteria for TES and LSTM.

Unsurprisingly, as seen in validation, the LSTM model (did not perform well at all, even when using a deeper model with increased parameter count. Though the prediction is smooth, qualitatively, the viability of the LSTM model as a predictor is poor, supported quantitatively by the high MAE value.

The Holt-Winters model unfortunately provides a forecast too conservative in terms of volatility, achieving large errors when faced with sudden market events like in the test data here. Another potential shortcoming is that the Holt-Winters model is univariate, and does not have access to the same amount of data as the TS2Vec models. We suspect that a weather predictor such as precipitation may have provided a strong basis for the TS2Vec models' ability to forecast a sudden dip, outperforming univariate methods such as Holt-Winters. This warrants further investigation, but performing ablation studies on features for specific forecasting abilities such as this would be far too time-consuming.

6 Conclusion

Though the project did not set clear parameters for the forecasting length of our objective, we can say that we achieved a model incorporating both weather and financial data that predicts 30-day future corn prices with a moderate level of accuracy. Through testing, incremental improvements and several quantitative and qualitative measures, we found the TS2Vec family of models to be competent in the forecasting task, with the kernel variant accurate and stable to 30 days out and the linear variants to be accurate but erratic to around 50 days. However, our other models did not perform at this same level and we can definitively say that our LSTM model failed at the task. As this is a relatively novel and specific task, we could not find a plug-and-play baseline to compare our method against to calculate an advantage value over, but going forward, it would be prudent to set up baselines ourselves using transformer-based pre-trained forecasting models, as they are widely available (Xu et al., 2024) as well as the methods outlined in the work of Tran et al. (2023). We freely admit this is a limitation in our project and research. Additionally, the performance of our LSTM, after validation and various architectural and hyperparameter alterations, could not come close to a simple linear regression model using TS2Vec. This is not indicative of a tuning error, but rather an implementational error and greater care should be taken next time in regards to the programming of training routines for such models. As for future work improving on current methods, a modification to the training setup and objective could be made: we found that an important criterion (that is, variation) could be quantified at test time but not during training time. This is due to a discrepancy in how the model trains on one-step predictions and the forecasting evaluation is done using multiple forward passes. Going forward, it could be beneficial to train on n-step predictions, which would complicate gradients and would require a semi-gradient method due to bootstrapping off of past predictions, but could provide a

stronger gradient signal and would allow us to add a regularization/penalty term using error variation to encourage smooth predictions (see appendix G for a potential objective function). In conclusion, we accomplished our objectives to a satisfactory degree and see many paths of improvement in the future.

References

How is corn used around the world? (2024, January 22). *Nebraska Corn Board*. https://nebraskacorn.gov/cornstalk/food/how-corn-is-used-domestically-and-internationally/

Konduri, V. S., Vandal, T. J., Ganguly, S., & Ganguly, A. R. (2020, April 7). Data Science for weather impacts on crop yield. *Frontiers*. https://www.frontiersin.org/journals/sustainable-food-systems/articles/10.3389/fsufs.2020.00052/full

Ning, Y., Kazemi, H., & Tahmasebi, P. (2022, May 6). A comparative machine learning study for time series oil production forecasting: Arima, LSTM, and prophet. *Computers & Geosciences*. https://www.sciencedirect.com/science/article/abs/pii/S009830042200084X?via%3Dihub

Qureshi, M., Iftikhar, H., Rodrigues, P. C., Rehman, M. Z., & Salar, S. A. A. (2024). Statistical Modeling to Improve Time Series Forecasting Using Machine Learning, Time Series, and Hybrid Models: A Case Study of Bitcoin Price Forecasting. *Mathematics, 12*(23), 3666. https://doi.org/10.3390/math12233666

Tran, N.-Q., Felipe, A., Ngoc, T. N., Huynh, T., Tran, Q., Tang, A., & Nguyen, T. (2023, October 28). Predicting agricultural commodities prices with Machine Learning: A Review of Current Research. *arXiv.org*. https://arxiv.org/abs/2310.18646

Xu, Y., Liu, A., Hao, J., Li, Z., Meng, S., & Zhang, G. (2024, August 20). Plutus: A well pre-trained large unified transformer can unveil financial time series regularities. *arXiv.org*. https://arxiv.org/abs/2408.10111

Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y., & Xu, B. (2021, June 19). TS2VEC: Towards Universal Representation of Time Series. *arXiv.org*. https://arxiv.org/abs/2106.10466

Appendix A. Feature Sets

Appendix A.1 Full Features

- **Close price (Yfinance)**: Primary forecasting objective.
- **Yfinance features**: Date, High, Low, Open, Volume.
- **Open-Meteo weather features**:
  - `weather_code`
  - `shortwave_radiation_sum`
  - `temperature_2m_mean`
  - `temperature_2m_max`
  - `temperature_2m_min`
  - `sunshine_duration`
  - `precipitation_sum`
  - `precipitation_hours`
  - `rain_sum`

  - daylight_duration
  - snowfall_sum

Appendix A.2 Reduced Features

- **Close price (Yfinance)**: Primary forecasting objective.
- **Open-Meteo selected features**:
  - shortwave_radiation_sum
  - temperature_2m_mean
  - sunshine_duration
  - precipitation_sum
  - precipitation_hours

Appendix A.3 Reduced Features with Extended Moving Average

- **Close price (Yfinance)**: Primary forecasting objective.
- **Weather features affecting corn production**:
  - shortwave_radiation_sum
  - temperature_2m_mean
  - sunshine_duration
  - precipitation_sum
  - precipitation_hours
- **Derived features (Yfinance)**:
  - 7-day moving average of close price
  - 30-day moving average of close price

Appendix B. Dataset Resources

- **Yfinance corn futures data**: https://finance.yahoo.com/quote/ZC%3DF/
- **Historical weather data**: https://open-meteo.com/

Appendix C. Improved RNN Autoregressor Architecture

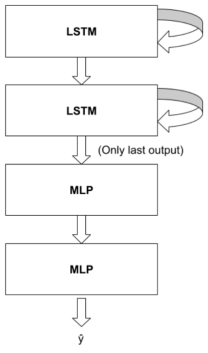


Figure 5: Architecture Diagram

Appendix D. TS2Vec Parameters

Appendix D.1 Initial Parameters

```
sequence_length: 30
```

```
hidden_size: 64
encoding_dims: 128
num_epochs: 20
n_iters: 2500
learning_rate: 0.001
kernel: False
```

Appendix D.2 Reduced Encoding Parameters

Differences from initial only:

```
hidden_size: 256
encoding_dims: 16
num_epochs: 25
learning_rate: 0.0001
```

Appendix D.3 Increased Encoding Parameters

Differences from initial only:

```
hidden_size: 256
encoding_dims: 256
num_epochs: 25
learning_rate: 0.0001
```

Appendix E. LSTM Improved Parameters

```
hidden_size: 128
num_layers: 2
num_epochs: 10
learning_rate: 0.05
n_splits: 3
```

Appendix F. Prediction Variation Metrics

Appendix F.1 Mean Prediction Variation (MPV)

$$\text{MPV} = \frac{1}{N} \sum_{t=1}^{N-1} |y_{t-1} - y_t| \tag{1}$$

Where N is the number of forecast steps and y_t is the prediction at time t.

Appendix F.2 Mean Error Variation (MEV)

$$\text{MEV} = \frac{1}{N} \sum_{t=1}^{N-1} |e_{t-1} - e_t| \tag{2}$$

Where e_t is the prediction error at time t.

Appendix G. Smoothness Constraint Objective

$$\mathcal{L}_\theta(Y, \hat{Y}) = \text{MSE}(Y, \hat{Y}) + \lambda \left(\text{MEV}(\hat{Y})\right)^2 \tag{3}$$

Where Y are the ground truth values and $\hat{Y}$ are the model predictions.

$$\mathcal{L}_\theta(Y, \hat{Y}) = \frac{1}{N} \sum_{t=1}^{N-1} (y_t - \hat{y}_t)^2 + \lambda \frac{1}{N} \sum_{t=1}^{N-1} \left[(\hat{y}_{t-1} - \hat{y}_t) - (\hat{y}_t - \hat{y}_{t+1})\right]^2 \tag{4}$$