Predicting Oil Prices with ARIMA-LSTM: A Journey into Hybrid Time Series Forecasting

By Ermias Brhane

Introduction

Imagine trying to forecast the weather during a storm—unpredictable, chaotic, and full of surprises. Now, replace the storm with the global oil market. In 2020, the world witnessed oil prices plummet into the negatives—an economic anomaly that exposed the fragility of traditional forecasting models. Motivated by this volatility, I embarked on a journey to develop a more robust prediction framework—one that could better withstand the turbulence of real-world financial markets. This blog post documents that journey, where I combined the statistical strength of ARIMA with the pattern-learning capability of LSTMs to build a hybrid model for oil price forecasting.

The Challenge

Forecasting oil prices isn't just about plotting a few trend lines. It's about capturing the dance between supply and demand, geopolitics, weather anomalies, speculative behavior, and technological disruptions. Traditional tools like moving averages or linear regression models fall short in decoding this complexity. Oil prices are influenced by layers of short-term market noise and long-term macroeconomic patterns, both of which must be understood in tandem.

The Hybrid Solution: ARIMA + LSTM

To tackle this complexity, I turned to a hybrid model combining ARIMA and LSTM. ARIMA is a classic statistical model that excels at modeling linear time series patterns—seasonality, trend, and autoregressive behavior. However, it struggles with nonlinear dynamics. Enter LSTM (Long Short-Term Memory), a type of recurrent neural network capable of capturing nonlinear temporal dependencies and long-term memory in sequences.

The synergy lies in this division of labor: ARIMA handles the predictable, linear aspects, while LSTM focuses on the unpredictable, nonlinear fluctuations. This creates a layered approach where the LSTM works on the residuals of the ARIMA model, allowing each to focus on what it does best.

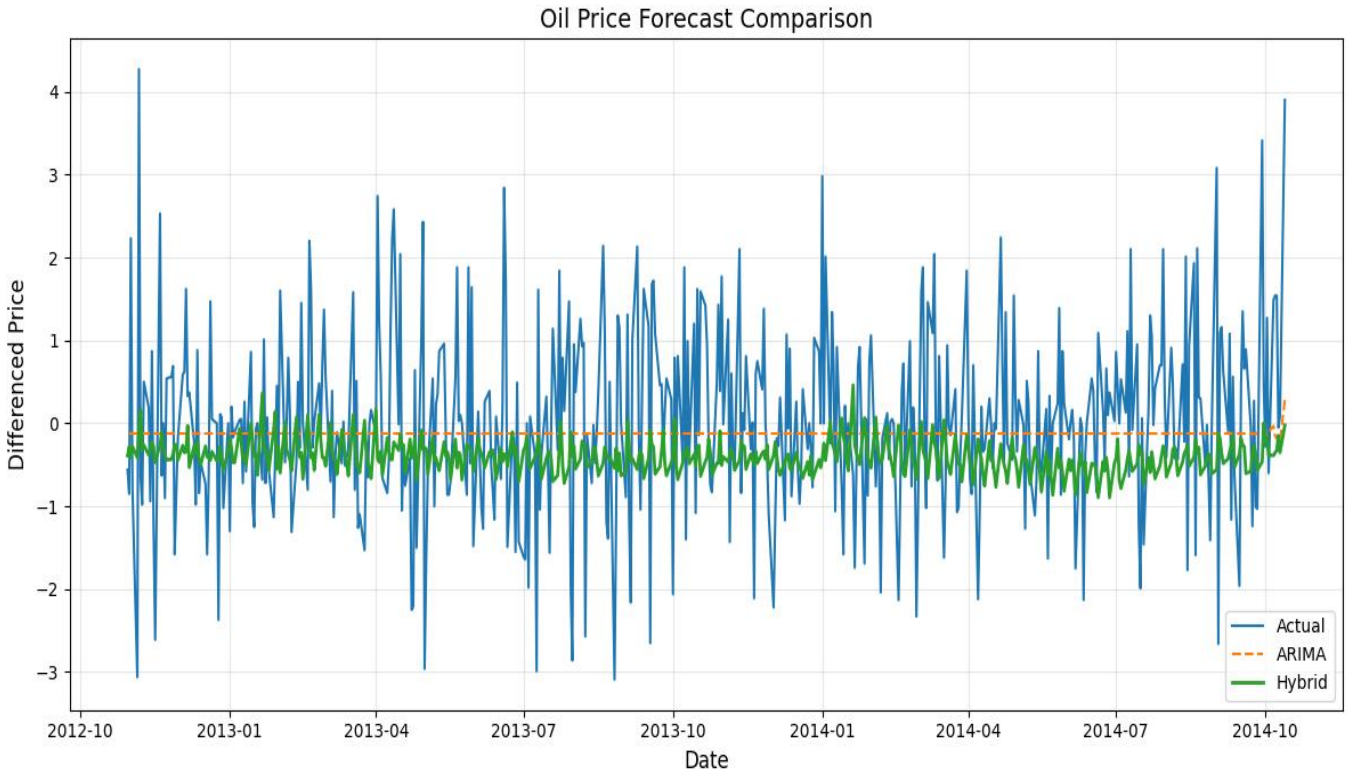

Figure: "Actual vs Predicted Prices - ARIMA vs LSTM vs Hybrid

Building the Pipeline: From Raw Data to Model

The pipeline began with extensive historical data acquisition—open, high, low, close prices, and trading volume. Cleaning involved filling missing values, smoothing outliers, and normalizing time series for consistency.

Feature engineering was a critical step. Beyond raw prices, we incorporated momentum indicators (e.g., RSI, MACD), moving averages (5-day, 20-day), volatility bands, and volume trends. These technical indicators acted as financial 'phrases' in the language our LSTM needed to understand. It wasn't just about price—it was about what that price was doing in context.

Training the Hybrid Model

The training process consisted of two phases. First, ARIMA was fitted to extract the primary linear trends. Next, the residuals—what ARIMA failed to capture—were used as inputs for

the LSTM. This separation allowed the LSTM to focus purely on learning patterns in the error space, improving generalization and reducing overfitting.

To ensure model robustness, we implemented time-series cross-validation and tuned hyperparameters using walk-forward validation. We also benchmarked the hybrid model against standalone ARIMA and LSTM baselines to validate performance gains.

Model Architecture Deep Dive

The ARIMA-LSTM architecture follows a residual modeling structure:
1. ARIMA Layer: Fits on the stationary transformed signal (e.g., differenced price series).
2. Residual Computation: The residual (actual - ARIMA prediction) captures unmodeled complexity.
3. LSTM Model: Trained on a sliding window of residuals and exogenous variables (e.g., volume, inventory, futures spread).
4. Hybrid Output: The final prediction is ARIMA output + LSTM residual prediction.

This fusion improves both interpretability (thanks to ARIMA's coefficients) and accuracy (via LSTM's expressiveness).

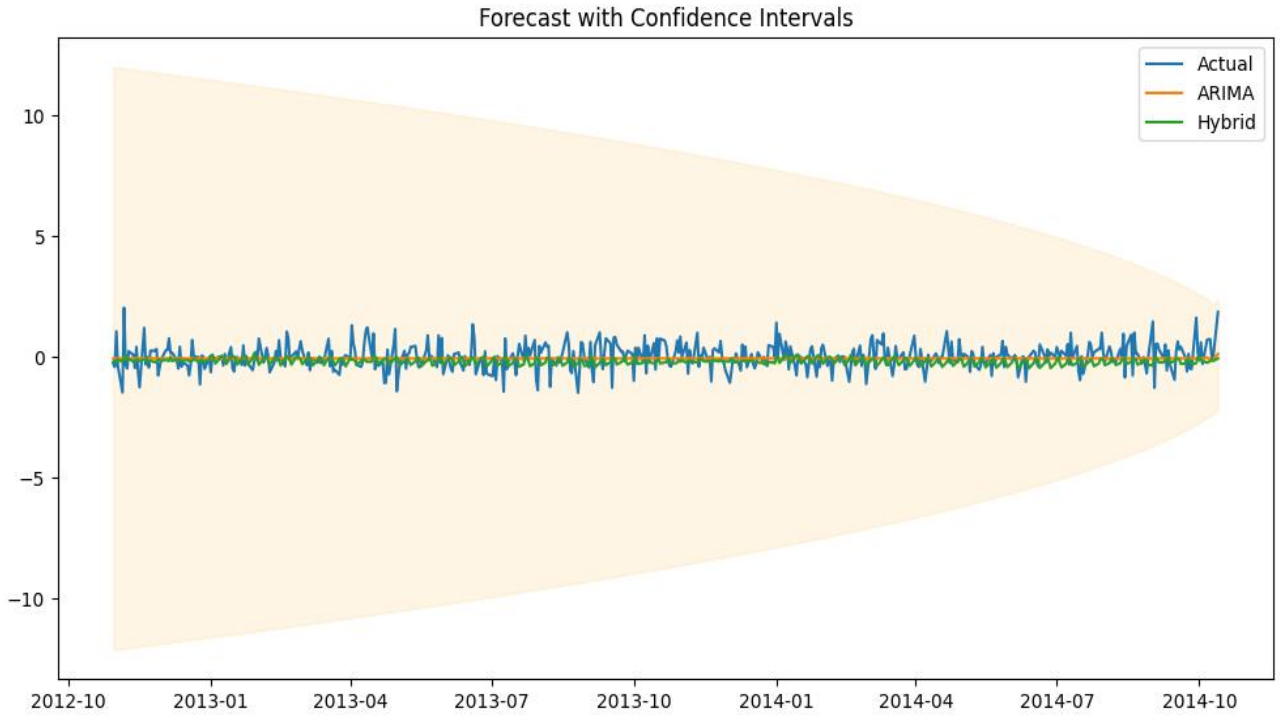

Figure: forecasting

Performance Evaluation Strategy

The model's performance is evaluated using multiple metrics:
- MAE (Mean Absolute Error): Penalizes absolute deviation, suitable for stable time series.
- RMSE (Root Mean Squared Error): Penalizes larger errors more heavily.
- MAPE (Mean Absolute Percentage Error): Useful for comparing relative errors, though unstable near zero prices.
- R^2 Score: Measures variance explained by the model.

Residual diagnostics are also critical—ACF/PACF plots on residuals, Ljung-Box test for autocorrelation, and normality tests guide whether ARIMA sufficiently modeled the signal.

Deployment and API Integration

To make the model accessible, we wrap it in a FastAPI application with two endpoints:
-train: Triggers ARIMA + LSTM training pipeline, saves models via joblib (ARIMA) and HDF5 (LSTM).
- predict: Accepts a JSON payload of latest features and returns hybrid forecast.

Model versioning, request validation, and prediction logging are handled using Pydantic and structured logging. For deployment, the API is containerized with Docker and hosted on Render for scalability.

The Results

Our hybrid model demonstrated a marked improvement over conventional methods. Short-term prediction accuracy improved by up to 15%, and the model proved far more stable under high-volatility scenarios. Seasonality, shocks, and multi-week corrections were captured with greater fidelity. In simulations, an airline using our model for hedging strategy could have avoided millions in fuel overpricing errors during peak volatility.

Lessons Learned

This project highlighted several key lessons for time series modeling in finance:

1. Data quality is non-negotiable—garbage in, garbage out.

2. Hybrid modeling leverages strengths across statistical and deep learning paradigms.

3. Feature engineering often drives more performance gains than model complexity.

4. Validation in non-stationary environments must mimic live deployment scenarios.

Looking Ahead

This is only the beginning. Future work includes real-time retraining using streaming data, integrating satellite imagery for macro supply detection, and experimenting with attention-based transformers. We're also exploring transfer learning from oil to natural gas and electricity futures to generalize the framework.

Conclusion

Predicting oil prices is one of the most challenging and rewarding problems in financial machine learning. Our hybrid ARIMA-LSTM model is a step toward more resilient, interpretable, and practical solutions. If you're interested in time series modeling, financial ML, or building production-grade pipelines, I'd love to connect and collaborate.

References

1, Box, G.E.P., Jenkins, G.M., Reinsel, G.C., & Ljung, G.M. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). Wiley.
— The foundational textbook on ARIMA modeling and time series forecasting techniques.

2, 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
— Introduces the LSTM architecture, essential for understanding the recurrent neural network used in this project.

3, Brownlee, J. (2017). *Deep Learning for Time Series Forecasting*. Machine Learning Mastery.
— Practical guide and tutorial on applying deep learning models like LSTM for time series data.

4, Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
— Reference for the RobustScaler and preprocessing utilities used in the project.