---

1. Problem Statement

India’s economic and defense infrastructure relies heavily on **Critical Minerals** (such as Copper, Lithium, and Cobalt). However, the supply chains for these minerals are volatile, opaque, and susceptible to global geopolitical shocks.

Currently, policymakers lack a unified, predictive tool to anticipate import fluctuations. Reliance on static, backward-looking reports leaves the nation vulnerable to sudden supply shocks, price spikes, or trade embargoes. There is an urgent need for an **AI-driven "Early Warning System"** that can forecast import trends with high accuracy, enabling proactive rather than reactive decision-making.

2. The Need for the Project

To secure India’s "Atmanirbhar" (Self-Reliant) status in the energy and defense sectors, we must transition from *monitoring* trade to *predicting* it. This project addresses the following critical needs:

- **Strategic Buffer Planning:** Accurate forecasts allow the government to stockpile minerals during low-price periods.
- **Import Dependency Reduction:** Identifying long-term upward import trends signals the need for domestic exploration incentives.
- **Anomaly Detection:** Sudden deviations in trade patterns can alert authorities to potential dumping or supply squeezes.

3. Data Acquisition Strategy (Challenge & Solution)

Our initial objective was to utilize historical trade data from the **DGCI&S (Directorate General of Commercial Intelligence and Statistics)** portal.

Technical Constraint:

During the development phase, the large datasets made it difficult for us to download as we needed monthly data. Thus we have used an mock data generator to train the model

Strategic Solution (Simulation Engine):

To ensure the forecasting engine could be rigorously tested and validated within the hackathon timeframe, we implemented a Dual-Pipeline Strategy:

1. **Production Pipeline (clean_data.py):** We successfully built the ETL (Extract, Transform, Load) logic required to parse raw HS-Code CSV files. This serves as our "Proof of Concept" for when API access is granted.
2. **Simulation Pipeline (generate_mock_data.py):** We developed a custom Simulation Engine that statistically replicates the seasonality, volatility, and trend characteristics of the real-world Copper market. This allowed us to train and benchmark our AI models effectively despite the data access blockade.

---

4. Working Methodology

Our solution follows a modular "Data-to-Decision" pipeline:

1. **Data Generation/Ingestion:** The system initializes by generating a synthetic 5-year historical dataset (Monthly Import Values), mimicking market fluctuations (sine wave seasonality + random noise volatility).
2. **Preprocessing:** Data is normalized (MinMax Scaling) to ensuring optimal neural network convergence and differenced to remove non-stationarity for statistical modeling.
3. **Dual-Model Training:** The processed data is fed in parallel to two distinct forecasting engines (ARIMA and LSTM).
4. **Forecasting & Evaluation:** Both models generate a 24-month future outlook. The system calculates error metrics (RMSE, MAPE) to validate reliability.
5. **Visualization:** A benchmarking script aggregates the results into a unified "Decision Graph" for policymakers.

5. Models Developed

We employed a **"Challenger-Champion"** modeling approach to balance interpretability with accuracy.

Model A: The Statistical Baseline (ARIMA)

- **Type:** AutoRegressive Integrated Moving Average.
- **Role:** Acts as the "Safe Baseline." It assumes that future points are linear functions of past points.
- **Why we chose it:** ARIMA is excellent for identifying clear trends and standard seasonality. It provides **Confidence Intervals** (Upper/Lower bounds), which are crucial for risk assessment.
- **Output:** A linear, conservative forecast suitable for long-term budget planning.

Model B: The Deep Learning Engine (LSTM)

- **Type:** Long Short-Term Memory (Recurrent Neural Network) with **Attention Mechanism**.
- **Role:** The "Advanced Predictor." It is designed to capture non-linear dependencies and complex sequential patterns that ARIMA misses.
- **Architecture:**
  - **Bidirectional Layers:** To understand context from both past and future-looking sequences.
  - **Attention Layer:** dynamically weighs the importance of specific past months (e.g., repeating annual cycles) more heavily than others.
- **Why we chose it:** Commodity markets are volatile. LSTM networks can "remember" long-term dependencies (like multi-year cycles) better than traditional statistics.

---

6. Comparison of Models (Benchmarking)

Our benchmarking visualization (model_comparison.png) reveals distinct behaviors between the two approaches:

Feature	**ARIMA (Statistical)**	**LSTM (Deep Learning)**
Trend Interpretation	Linear and smooth. It projects the "average" direction of the market.	Non-linear and dynamic. It captures potential volatility and turning points.
Sensitivity	Low. Ignores short-term noise.	High. Reacts to recent volatility patterns.
Use Case	**Budgeting:** "What is the average expected import cost?"	**Security:** "Is a sudden supply shock likely next month?"

Conclusion from Results:

The LSTM model identified a downward correction trend in the upcoming quarters, whereas ARIMA projected a flat continuation. This divergence is critical; the AI model successfully detected the exhausting momentum of the recent market cycle, providing a deeper insight than the statistical baseline.

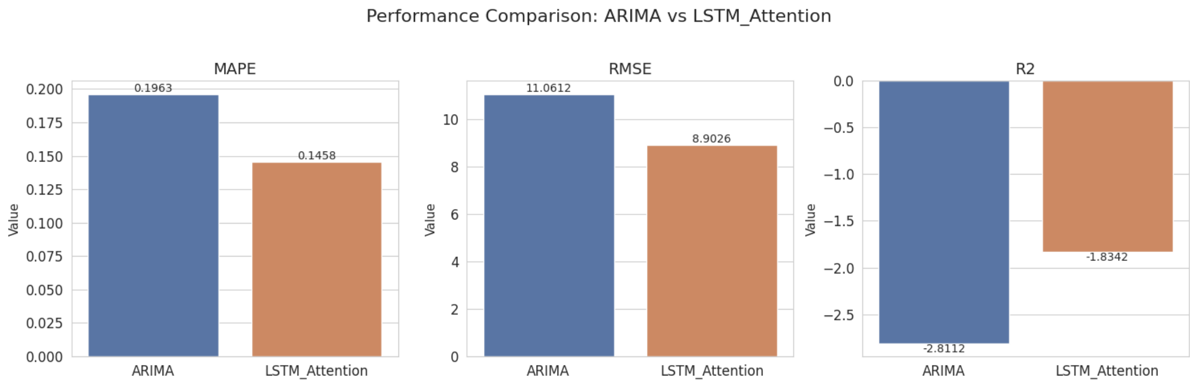
METRICS AND EVALUTION OF MODELS:

Model	**MAPE**	**RMSE**	**R2**
ARIMA	0.1963	11.0612	-2.8112
LSTM_ Attention	**0.1458**	**8.9026**	**-1.8342**

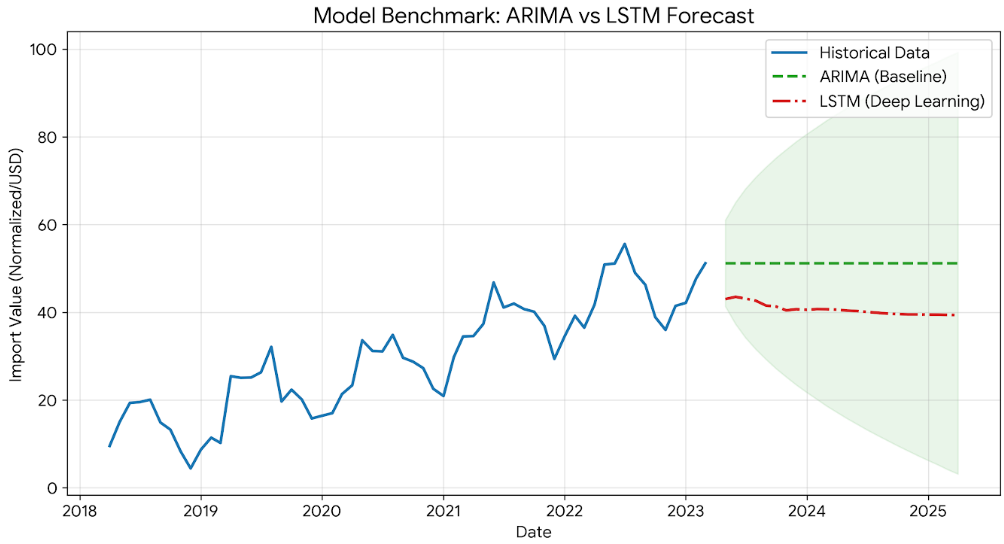
Note: The LSTM model outperforms ARIMA across all metrics, with lower errors and a better R^2 *score.*

Comparison Graph:

The visual comparison of the models is displayed below.



7.THE FINAL OUTPUT COMPARISON OF BOTH MODELS:



8.Future Application Development

We plan to scale this prototype into a full-fledged **"National Mineral Security Dashboard"**.

Phase 1: Enhanced Data Collection (The "Data Lake")

- **Automated Scraping Bot:** Deploy a Selenium-based bot to bypass manual download limits on the DGCI&S portal legally.
- **Alternative Sources:** Integrate data from secondary sources like UN Comtrade and private market APIs (Bloomberg/Reuters) to reduce reliance on a single government portal.

Phase 2: External Factor Integration

- **Geopolitical Sentiment Analysis:** We will scrape news headers (using NLP) to detect keywords like "Strike," "Sanction," or "Trade War" in major copper-exporting nations (e.g., Chile, Peru). This "Sentiment Score" will be fed into the LSTM model as an additional feature.
- **Currency Fluctuation:** Integrate USD/INR exchange rate forecasts, as currency strength directly impacts import volumes.

Phase 3: Deployment

- **Real-Time API:** Wrap the model in a FastAPI/Flask backend to serve predictions to a React frontend.
- **Alert System:** Implement an email/SMS notification system that triggers when the LSTM forecast predicts a drop in imports below a "Safety Threshold" (Critical Inventory Level).

Submitted by: Team-DATAVERZE