AI for Trade Global Challenge — Method Summary

SABLE: A Four-Step Forecasting Pipeline for Monthly Trade

Method name: SABLE — **Team name:** Blue bird

Overview

We introduce SABLE, a hybrid approach designed for high volume of time series. SABLE is conceptually inspired by innovative deep learning architectures such as N-BEATS/NBEATSx and the Temporal Fusion Transformer (TFT). Although SABLE's implementation diverges significantly from these end-to-end neural network models, it draws inspiration from key principles they popularized, including:

- The explicit handling or decomposition of time series into constituent components (e.g., seasonality and trend), a core concept in N-BEATS's block architecture and often a focus in TFT's analysis.
- The effective leveraging of information across numerous related series, a strength of both N-BEATS and TFT which are designed for multi-variate and high-cardinality forecasting.
- The use of learning mechanisms to refine initial forecasts or model residuals, conceptually similar to N-BEATS's residual connections between blocks.
- The strategic incorporation of relevant exogenous variables and temporal features, a key capability emphasized in both NBEATSx and TFT for improving forecasting accuracy.

SABLE employs a multi-stage process: generating robust base forecasts from an ensemble of naive candidates and seasonal-trend decomposition, selecting relevant lagged exogenous variables based on global correlation, constructing a comprehensive feature set, and finally applying a robust Huber regression to predict and correct the base forecast in log-space. This hybrid architecture offers a balance of interpretability, computational efficiency, and robustness, presenting a practical alternative for large-scale forecasting challenges where the full complexity of deep learning models may not be desirable.

To strengthen the training set, we extracted trade data from UN Comtrade on monthly trade flows for the U.S. and China (2021–2022and 2025) on the same HS4 products. We also extracted macroeconomic covariates from the IMF National Economic Accounts, converted from quarterly to monthly frequency using interpolation and lags to capture GDP, consumption, and investment trends.

Problem and Notation

We forecast monthly trade values at the product×country×flow level:

$$s = (p, c, f), \quad p = \text{HS4}, \; c = \text{partner}, \; f \in \{\text{imports}, \; \text{exports}\}.$$

The initial database has the following variable : `product_id_hs4`, `country_id`, `trade_flow_name`, `month_id` (`YYYYMM`), `trade_value`, plus numeric exogenous candidates. Observations are $\{(t_1, y_1), \ldots, (t_T, y_T)\}$ with $y_t \geq 0$; the month index is $M_t \in \{1, \ldots, 12\}$. We train on Jan 2021–July 2025 and hold out a specified month for testing. **Pipeline (4 steps):** (1) Naïve Estimator (anchor), (2) Feature Engineering, (3) Residual Learning, (4) Reconstruction & Guardrails.

Step 1 — Naïve Estimator (Anchor)

We build a robust seasonal anchor from multiple simple forecasters and take their median to obtain b_{T+1}.

1.1 Candidate forecasts

$$\text{last} = y_T, \quad \text{seas12} = y_{T-12} \text{ (if available)}, \quad \text{MA}_3 = \frac{1}{3}\sum_{i=T-2}^{T} y_i, \quad \text{MA}_6 = \frac{1}{6}\sum_{i=T-5}^{T} y_i.$$

1.2 Drift on log-values (last 6 months)

$$\beta = \frac{\sum_{i=T-5}^{T}(i-\bar{i})\left(\log(1+y_i) - \overline{\log(1+y)}\right)}{\sum_{i=T-5}^{T}(i-\bar{i})^2}, \qquad \text{drift\_last} = y_T\, e^{\beta}.$$

1.3 Seasonal–trend candidate

We adopt a multiplicative structure with robust seasonal factors:

$$y_t \approx g\, S_{M_t} \cdot \text{trend}_t, \qquad g = \text{median}\{y_t : \; y_t > 0\}.$$

Monthly seasonal factors are clipped:

$$S_m = \text{clip}\left(\frac{\text{median}\{y_t : \; M_t = m, \; y_t > 0\}}{g}, \; 0.2, \; 5.0\right).$$

Deseasonalize and extrapolate with a short log-slope ($w \in \{6, 12, 24\}$):

$$d_t = \frac{y_t}{S_{M_t}}, \qquad \hat{d}_{T+1} = d_T\, e^{\beta}, \qquad \text{seasonal\_ST} = \hat{d}_{T+1}\, S_{M_{T+1}}.$$

We stabilize via the median with the last observed value and the same-month rolling median (last 3 years).

1.4 Anchor (median of candidates)

$$b_{T+1} = \text{median}\Big\{\text{last},\ \text{seas12},\ \text{MA}_3,\ \text{MA}_6,\ \text{drift\_last},\ \text{seasonal\_ST},\ \text{same\_month\_med3}\Big\}.$$

Step 2 — Feature Engineering (Causal)

All features are strictly *causal* (lagged only), preventing look-ahead leakage.

2.1 Calendar and seasonality

We use the prediction month M_{T+1}, indicators `is_jan`/`is_dec`, and the seasonal factor $S_{M_{T+1}}$ estimated in Step 1.3.

2.2 Lagged target signals

$$y_T,\quad y_{T-1},\quad y_{T-2},\quad y_{T-3},\quad y_{T-6},\quad y_{T-12}.$$

Moving averages (3, 6) are already captured via the anchor candidates.

2.3 Target encodings (train-only)

We compute mean trade values by product, country, and flow on the training slice (fallback to the global mean), and use these as continuous encodings for high-cardinality identifiers.

2.4 Exogenous lags (Spearman selection)

Numeric exogenous columns are auto-detected (excluding IDs and target). For each x and lags $L \in \{1, 3, 6\}$, we pool (x_{t-L}, y_t) across series and compute Spearman ρ; we select the top-k (x, L) pairs by $|\rho|$. At prediction time, each retained pair contributes x_{T+1-L} (or the last available non-missing value under causality).

Step 3 — Residual Learning (Global Huber)

We correct the anchor via a global regression model learned on log-residuals.

3.1 Short one-step backtest for training

Per series, we run a short backtest on the last k points (default k=3). For each fold, we recompute the anchor b_t on the truncated history, build the causal features X_t, and define the log-ratio residual target:

$$z_t = \log(1 + y_t) - \log(1 + b_t).$$

This formulation stabilizes variance and expresses relative deviation to the seasonal anchor.

3.2 Huber regression

We fit a global Huber regressor $f_{\text{Huber}}(X)$ to predict z_t across all series. The Huber loss is quadratic near zero and linear in the tails, providing robustness to outliers and zero-inflated patterns while preserving efficiency.

Step 4 — Reconstruction and Guardrails

4.1 Recompose to original scale

$$\hat{y}_{\text{corr}} = \exp\big(\log(1 + b_{T+1}) + \hat{z}_{T+1}\big) - 1.$$

4.2 Rare-series safeguards

If the historical nonzero fraction is below a threshold, we apply: (i) a **floor** based on the median of recent nonzero values (scaled), and (ii) a **growth cap** limiting $\hat{y}_{\text{corr}}$ to a multiple of the last observation y_T:

$$y_{\text{final}} = \max\big(y_{\text{floor}},\ \min(\hat{y}_{\text{corr}},\ y_T \cdot g_{\text{cap}})\big).$$

4.3 Blend with the anchor

For stability, we blend the corrected value with the anchor:

$$\hat{y}_{T+1} = w\, y_{\text{final}} + (1 - w)\, b_{T+1}, \qquad w = 0.6.$$

Implementation Notes

Series consistency. Series IDs are concatenations of (p, c, f); only series present in both train and test are scored; duplicate test rows are safely deduplicated.
Safety defaults. If an anchor component is unavailable, we back off to the last valid observation. Feature order is frozen during training to ensure consistent matrices.