The "best" algorithm for time series forecasting isn't universal and highly depends on the specific characteristics of your data and the forecasting task. However, a comprehensive list of algorithms considered effective and widely used includes:

Classical/Statistical Methods:

- **ARIMA (Autoregressive Integrated Moving Average):** A classic and widely used model that combines autoregressive (AR) terms, integrated (I) differencing for stationarity, and moving average (MA) terms to model linear dependencies in the data. It's suitable for stationary time series data.
- **SARIMA (Seasonal ARIMA):** An extension of ARIMA that explicitly models the seasonal component in the time series data. Useful for data exhibiting both trend and seasonality.
- **Exponential Smoothing (ES):** A family of methods (including Simple Exponential Smoothing, Holt's Linear Trend, and Holt-Winters' Seasonal Methods) that use weighted averages of past observations, with weights decreasing exponentially. Effective for data with trend and/or seasonality.
- **Moving Average:** A simple technique that smooths out noise by averaging data points over a specific window. Useful for identifying trends but doesn't handle seasonality well.
- **STL (Seasonal Decomposition of Time Series):** Decomposes a time series into trend, seasonal, and remainder components, making it easier to analyze and forecast each component separately.
- **Prophet:** Developed by Facebook, this is an additive model that excels with time series data having strong seasonality and holiday effects. It's robust to missing data and outliers and handles trend changes well.
- **Vector Autoregression (VAR):** Used for multivariate time series data, where multiple interdependent time series are modeled together.

Machine Learning Methods:

- **Regression Models (Linear Regression, Polynomial Regression):** Can be adapted for time series forecasting by using lagged values of the target variable and other relevant features as predictors.
- **Support Vector Regression (SVR):** A versatile machine learning algorithm that can be used for non-linear time series forecasting by mapping the data into a higher-dimensional space.
- **Decision Tree-Based Models (Random Forest, Gradient Boosting - XGBoost, LightGBM, CatBoost):** These ensemble methods can capture complex non-linear relationships in time series data when engineered with lagged features.
- **Multi-Layer Perceptron (MLP):** A basic type of neural network that can learn non-linear patterns when past values are used as input features.

Deep Learning Methods:

- **Recurrent Neural Networks (RNNs):** Designed to process sequential data, making them suitable for time series. They have memory capabilities to capture dependencies over time.
- **Long Short-Term Memory (LSTM) Networks:** A type of RNN that excels at learning long-range dependencies in sequential data, often outperforming traditional RNNs for longer time series.
- **Gated Recurrent Units (GRUs):** Similar to LSTMs but with a simpler architecture, often providing comparable performance with fewer computational resources.
- **Convolutional Neural Networks (CNNs):** While traditionally used for image processing, 1D CNNs can be effective for extracting local patterns from time series data.
- **Transformer Networks:** With their attention mechanisms, Transformers can model long-range dependencies effectively and have shown promising results in time series forecasting, especially for long sequences.
- **DeepAR:** A probabilistic forecasting model by Amazon that uses autoregressive recurrent neural networks to generate probabilistic forecasts.
- **N-BEATS:** A deep learning model with a purely additive architecture based on backward and forward residual links and a rich set of basis functions.
- **NeuralProphet:** An extension of Facebook's Prophet that incorporates neural networks to capture complex patterns.

Hybrid Methods:

- **ARIMA-ANN:** Combines the linear modeling capabilities of ARIMA with the non-linear learning of Artificial Neural Networks (ANNs) by using ARIMA residuals as input for the ANN.

Key Considerations When Choosing an Algorithm:

- **Data Characteristics:**
  - **Stationarity:** Is the statistical properties of the time series constant over time? ARIMA and SARIMA often require stationary data.
  - **Trend:** Is there a long-term increasing or decreasing pattern?
  - **Seasonality:** Are there regular, repeating patterns at fixed intervals?
  - **Cyclicity:** Are there longer-term fluctuations that are not strictly periodic?
  - **Noise:** How much random variation is present in the data?
  - **Linearity vs. Non-linearity:** Are the relationships in the data linear or non-linear?
  - **Multivariate vs. Univariate:** Are you forecasting a single time series or multiple interdependent ones?
  - **Length of the Time Series:** Deep learning methods often require large amounts of data to train effectively.
  - **Presence of Outliers and Missing Values:** Some models are more robust to these issues than others (e.g., Prophet).
- **Forecasting Horizon:** Are you forecasting short-term or long-term? Some models perform better at different horizons.

- **Interpretability:** How important is it to understand the reasons behind the forecasts? Classical methods like ARIMA are generally more interpretable than deep learning models.
- **Computational Resources:** Deep learning models can be computationally expensive to train and may require specialized hardware.
- **Complexity and Ease of Implementation:** Some models are simpler to implement and tune than others.
- **Performance Requirements:** How accurate do the forecasts need to be?

Practical Tips:

- **Visualize your data:** Understanding the patterns in your time series is crucial for selecting an appropriate model.
- **Preprocess your data:** Techniques like handling missing values, outlier detection, and scaling can significantly impact model performance.
- **Feature engineering:** For machine learning models, creating relevant features (e.g., lagged values, time-based features) is essential.
- **Split your data:** Divide your data into training, validation, and testing sets to properly evaluate model performance and avoid overfitting.
- **Evaluate multiple models:** It's often beneficial to try several different algorithms and compare their performance on your validation set using appropriate metrics (e.g., MAE, MSE, RMSE).
- **Tune hyperparameters:** Most models have parameters that need to be optimized for your specific data.
- **Consider ensembling:** Combining the predictions of multiple models can sometimes lead to improved accuracy.

In conclusion, there is no single "best" algorithm for all time series forecasting tasks. The optimal choice depends on a careful analysis of your data's characteristics and the specific requirements of your forecasting problem. Experimentation and thorough evaluation are key to finding the most suitable approach.