Time Series Forecasting with TimeGPT

Mariana Menchero

mariana@nixtla.io

About me

Senior forecaster at Nixtla

Developer of the nixtlar package

Previously, senior forecaster at Ritter Dragon, a consulting company in Mexico City

MIT Micromasters in Supply Chain Management credential holder

Forecasting enthusiast since 2019 ♥

Overview of Foundation Models

What are Foundation Models?

In the context of time series forecasting, a foundation model is a machine learning model that has been trained on a very large dataset and that is capable of predicting series not seen during training.

Some foundations models are also capable of perform other time series related tasks, such as anomaly detection or classification.

Types of Foundation Models

There are two main **categories**:

1. Foundation models that are specifically designed to handle time series data.
    - TimeGPT
    - Lag-LLaMa
    - Chronos
    - TimesFM
    - Moirai

2. LLM that have been extended to handle time series problems.
    - TimeLLM
    - PromptCast

Why Do Foundation Models work?

Foundation models are trained on very large amounts of data. These models often weight many gigabytes and are trained for days.

A single foundation model can predict series with different properties, such as frequencies, trends, or seasonal periods. The idea is that even though these series have never seen your data, they have seen other series that are “similar enough”.

Fine-tuning vs Zero-shot

Foundation models can also be **fine-tuned**, meaning that certain parts of the model are trained on your specific data. In this process, only a subset of the model's parameters is updated, rather than the entire model, since full retraining would take very long.

The **zero-shot** version of a foundation model refers to its original state, without any fine-tuning.

Beware of overfitting!

How Do Foundation Models Work?

Most foundation models, including TimeGPT, are built on the **transformer architecture**.

1. **Embedding Layer**: Converts the time series into a format the model can understand.
2. **Positional Encoding**: Adds temporal information so that the order of the time steps is recognized.
3. **Encoder**: Uses self-attention to learn patterns and dependencies across time steps.
4. **Decoder**: Predicts step-by-step, feeding each output back into the model.

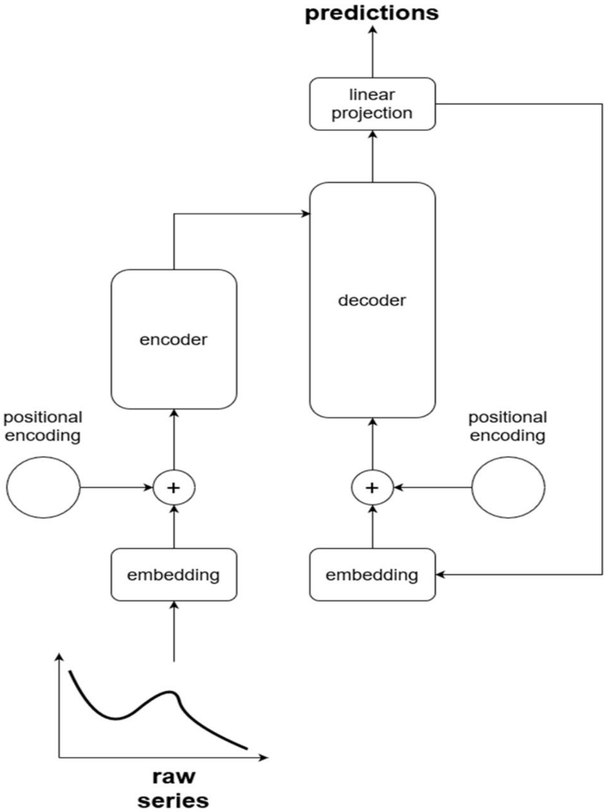


Image taken from Peixeiro (2025)

Advantages	Disadvantages
Requires minimal code and forecasting expertise.	Your specific case might not be well-represented.
Can generate forecasts even with few observations.	No direct access to model internals for customization and black-box model.
Applicable to many forecasting scenarios.	If accessed via API, service outages can disrupt availability. Check http://status.nixtla.io/

nixtlar

Introduction to TimeGPT

The first foundation model for time series forecasting.

Released by Azul Garza, Cristian Challu and Max Mergenthaler-Canseco on May 2024 (research paper)

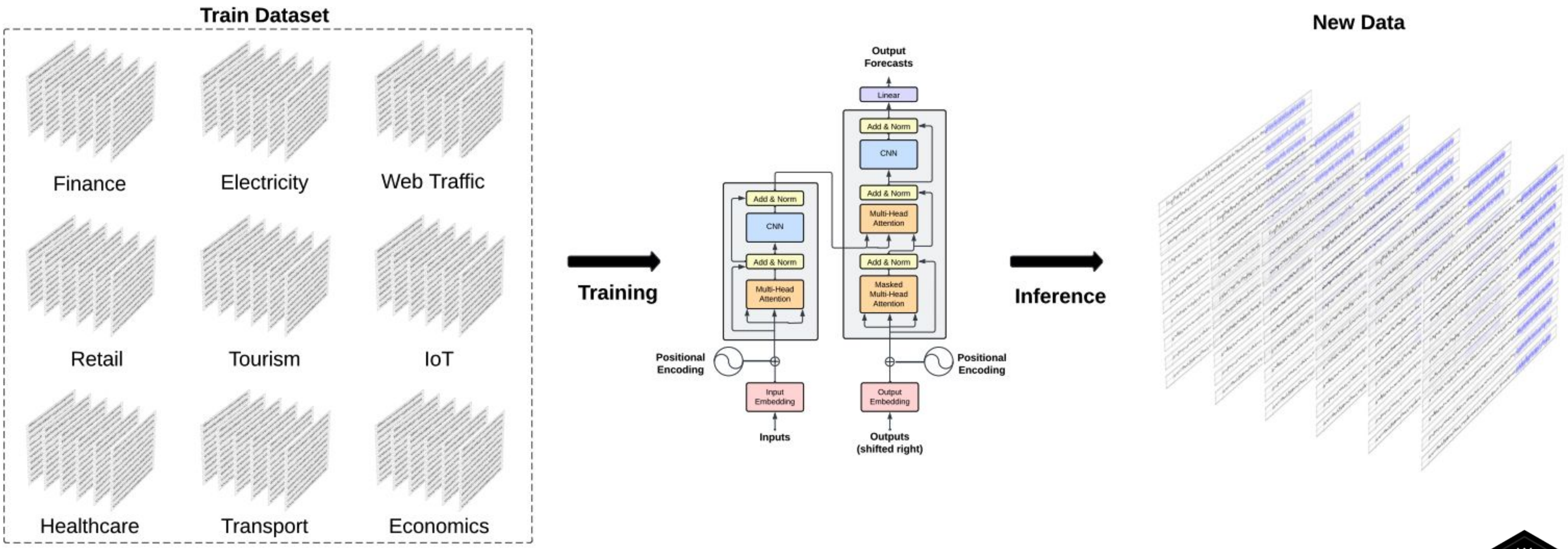
TimeGPT model

TimeGPT is a **generative pre-trained transformer**, built on the transformer architecture described in the previous section.

It is **generative** because it is trained to take an input sequence and generate an output sequence in an **autoregressive** manner, so that each predicted element is fed back into the model to generate the next step.

Additionally, it has been **pre-trained** on a very large dataset - *more on this later*

TimeGPT model



TimeGPT Training Data

- **Large-scale dataset**: The training set contains over 100 billion data points.
- **Diverse domains**: Time series come from multiple industries and applications.
- **Varied series**: The dataset includes time series with different scales, lengths, and temporal patterns (e.g., multiple seasonalities and various types of trends).
- **Minimal preprocessing**: Only missing values were interpolated using a proprietary method. Hence, it has seen data with noise and outliers.
- **Computationally intensive training** – Training took several days on a cluster of NVIDIA A10G GPUs.

TimeGPT Capabilities

Supports the following features:

- Multiple series concurrently
- Fine-tuning
- Exogenous variables (historic and future)
- Cross-validation
- Anomaly detection
- Prediction intervals

All of these features are available through *nixtla*, the Python SDK.

The nixtlar Package

Available on CRAN and on GitHub (development version)

THE NIXTLAVERSE

The R ecosystem for time series forecasting

forecast

Fable

modeltime

Business Science

smooth

nixtlar

VN1 Forecasting Competition

September 2024 - October 2024

Weekly retail dataset with 15,053 series and a forecast horizon of 13 weeks

Organized by Nicolas Vandeput on the datasource.ia platform and sponsored by Flieber, SupChains and Syrup

Evaluation metric

$$score = \frac{\sum_{i=1}^{N} |\hat{y} - y|}{\sum_{i=1}^{N} \hat{y} - y}$$

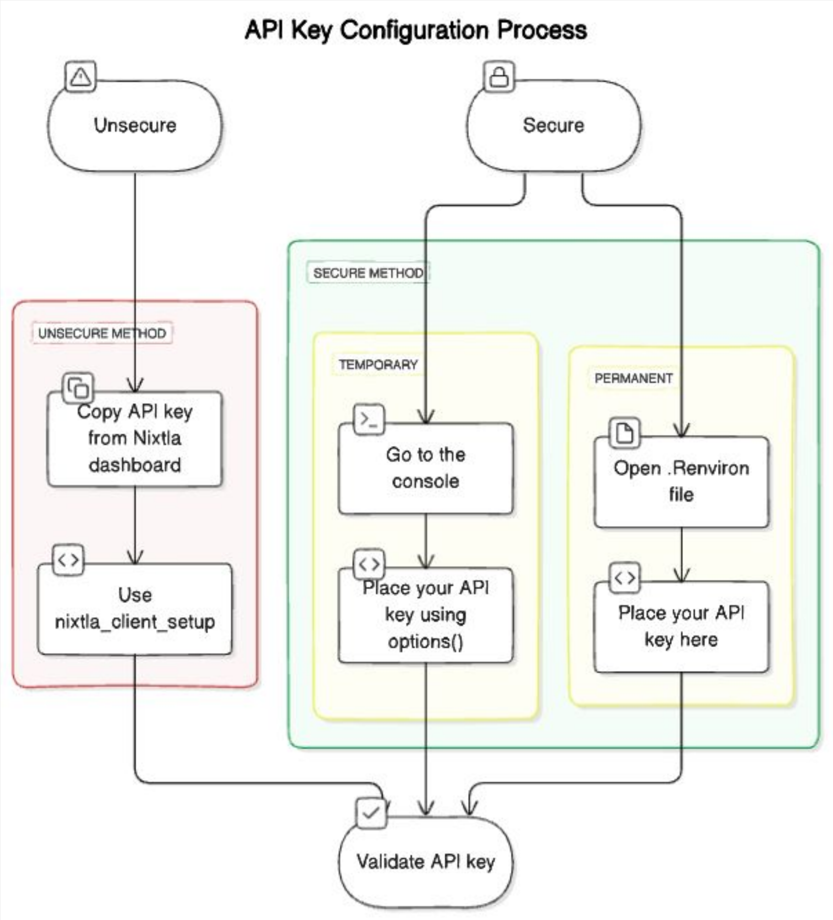
What’s Next

In this section, I will explain:

- How to set up your API key
- Data requirements and required pre-processing steps
- How to use TimeGPT
- How to leverage TimeGPT’s features in nixtlar

Set up your API key

Get yours at:
https://dashboard.nixtla.io/sign_in



Takeaways

Foundation models are a new tool for time series forecasting. They may or may not be the best fit for all your series, but they’re worth experimenting with.

Forecasting is not an end in itself, decision-making is. Don’t focus just on KPIs and think about how your forecasts impact the decisions you make.

Get started, experiment, and keep iterating!

Takeaways

- ✔ Visualize your data – *always the first step*
- ✔ Implement TimeGPT (setting up your API key & data requirements)
- ✔ Build long-horizon forecasts with TimeGPT
- ✔ Fine-tune your models with several parameters or via the AutoTimeGPT
- ✔ Incorporate exogenous variables (historic and future)
- ✔ Generate probabilistic forecasts to estimate uncertainty
- ✔ Detect anomalies

Q&A

Stay in touch: https://www.linkedin.com/in/marianamencherogarcia/

Open an issue in nixtlar: https://github.com/Nixtla/nixtlar/issues