Proof of Concept Portfolio for Predictive Maintenance using TimesFM

Key Objectives

I began by exploring how Large Language Models (LLMs) could be applied to predictive maintenance in real-world industrial setups. Initially, I studied the traditional approaches to predictive maintenance and investigated the direction in which recent AI developments are heading.

It became clear that general-purpose LLMs like ChatGPT or LLaMA aren't directly suitable for handling time series data. I then shifted focus to transformer-based architectures, which form the backbone of LLMs, and identified **TimesFM**—a foundation model pretrained by Google specifically on time series data—as a strong candidate for this task.

Dataset for Analysis

To begin, I chose the **NASA Bearing dataset** because it presents raw signals in a relatively simple form. This helped me understand fundamental concepts such as signal windowing, feature engineering, and degradation modeling.

Using this dataset, I built a **custom health index** from features like mean, standard deviation, and kurtosis, and mapped it to Remaining Useful Life (RUL) using threshold-based logic.

However, for more robust modeling and demonstration, I plan to use the **CMAPSS Turbofan dataset**, which provides structured labels and standardized RUL calculations—allowing me to focus more on model design and less on preprocessing challenges.

What I've Built So Far

1. Data Preprocessing on NASA Bearing Data

- Developed a pipeline to extract sliding-window features (mean, std, kurtosis) from raw vibration signals.
- Engineered timestamps and a synthetic health index to simulate degradation.
- Computed RUL dynamically by associating the health index with a failure threshold (RUL = 0 when health index crosses it).
- Simulated various degradation patterns and visualized them.

2. Forecasting with TimesFM (Zero-Shot)

- Used **TimesFM in inference mode** (no fine-tuning) to forecast future values of the custom health index.
- Estimated RUL from the forecasted trajectory.
- Demonstrated TimesFM's ability to operate without training on the same asset, showcasing its foundation model strength.

3. TimesFM Fine-Tuning

- Adapted the official TimesFM finetuning framework with enhancements:
    - **Distributed training support** via PyTorch DDP.
    - **Quantile loss** for capturing uncertainty in forecasts.
    - **WandB integration** for tracking experiments.
- Prepared patch-level data formatting to align with TimesFM's input-output specification.

4. Model Evaluation Strategy

- Built visual diagnostics to track degradation and predicted RUL over time.
- Compared model forecasts with baseline trends.
- Evaluated using standard metrics such as **MSE**.

What's Ready for Full Implementation

- Modular data loader utilities for sliding windows and patch alignment.
- Preprocessing pipeline for CMAPSS or similar datasets.
- Working zero-shot inference with TimesFM.
- Fine-tuning scaffold that supports custom loss functions and multi-GPU training.
- Evaluation framework to compare predicted vs. actual RUL using RMSE, MAE, and custom degradation metrics.

Tools & Technologies Used

- **LLM for Time Series**: TimesFM (via Hugging Face), PyTorch

- **Preprocessing & Modeling**: pandas, numpy, scipy, sklearn
- **Visualization**: matplotlib, seaborn
- **Experiment Logging**: Weights & Biases (WandB)
- **Training Infrastructure**: PyTorch DDP (Distributed), Custom Fine-Tuner
- **Problem Focus**: RUL Estimation, Health Index Forecasting, Failure Prediction

Next Steps

- I can wrap up the POC phase and present the results based on current NASA bearing data.
- Or, I can extend this work to a **full implementation on the CMAPSS dataset**, which better mirrors real-world predictive maintenance settings and can be a compelling demonstration of TimesFM's capabilities.