Faculty of Innovative Medicine

Department of Computer Engineering for Robotics and Smart Industry

Deep Learning: Lag-Llama:Towards foundation Models for probabilistic Time series forecasting

Submitted by: Fitsum Sereke Tedla (VR496807)

Professor Marco Cristani
Dr. Uzair Hassan Khan

July 27, 2024

Acknowledgements

I would like to express my heartfelt gratitude to my professors, Professor Marco Cristiani and Dr. Uzair Hassan Khan, for their invaluable guidance and support throughout this course.

Abstract

This study investigates the application of the Lag-Llama model for probabilistic time series forecasting using CMAPSS jet engine simulated data. The CMAPSS dataset, which includes comprehensive time series data from simulated jet engines, serves as a benchmark for evaluating predictive maintenance models. The Lag-Llama model, known for its advanced probabilistic forecasting capabilities, is assessed across a selection of 18 out of 21 sensor datasets to determine its effectiveness in forecasting engine health and predicting potential failures.

Our analysis demonstrates that the Lag-Llama model provides robust performance in managing both stable and volatile data conditions. For datasets representing stable engine conditions, the model delivers accurate forecasts with narrow confidence intervals, reflecting its ability to predict consistent patterns. Conversely, for more complex and variable datasets, the model's broader confidence intervals highlight its effectiveness in handling high levels of uncertainty and dynamic changes in engine behavior.

Key findings include the model's capability to adapt to seasonal trends and sudden anomalies in engine performance, effectively quantifying forecast uncertainty. These features are crucial for developing reliable predictive maintenance solutions in aerospace applications.

The paper concludes with recommendations for future enhancements, including the incorporation of additional operational variables, refinement of forecasting techniques, and improvements in real-time data processing. These advancements are expected to enhance the model's accuracy and adaptability, offering greater insights into jet engine health management and failure prediction.

This research underscores the Lag-Llama model's potential in leveraging simulated jet engine data to provide valuable forecasts and maintain operational reliability, thereby contributing to the field of predictive maintenance and aerospace engineering.

Dedication

To Everyone who works on Lag_Llama.

Contents

Acknowledgements **i**

Abstract **ii**

Dedication **iii**

1 Introduction **1**
1.1 Probabilistic Time Series Forecasting 1
1.2 The probabilistic model vs the traditional models 2

2 Methodology **3**
2.1 Lag-Llama 3
2.2 Lag-Llama for Probabilistic time series forecasting 5
2.2.1 Problem context and Dataset structure 5
2.2.2 Training Procedure 6
2.2.3 Dataset 6

3 Analysis and Results **8**

4 Discussion **18**
4.1 Model Performance and Accuracy 18
4.2 Responsiveness to Data Changes 18
4.3 Quantification of Forecast Uncertainty 19
4.4 Patterns and Anomalies 19
4.5 Implications and Recommendations 19

5 Conclusion **20**
5.1 Future Work 20

List of Figures

2.1 Llama 4

3.1 Sensor 1 8
3.2 Sensor 2 9
3.3 Sensor 3 9
3.4 Sensor 4 10
3.5 Sensor 6 10
3.6 Sensor 7 11
3.7 Sensor 8 11
3.8 Sensor 9 12
3.9 Sensor 10 12
3.10 Sensor 11 13
3.11 Sensor 12 13
3.12 Sensor 13 14
3.13 Sensor 14 14
3.14 Sensor 15 15
3.15 Sensor 17 15
3.16 Sensor 18 16
3.17 Sensor 20 16
3.18 Sensor 21 17

*

Acronyms

RUL Remaining Useful Life

CMAPSS Commercial Modular Aero-Propulsion System Simulation

NASA National Aeronautics and Space Administration

Chapter 1

Introduction

1.1 Probabilistic Time Series Forecasting

Probabilistic time series forecasting is a method used to predict future values of a time series (a sequence of data points ordered in time) while explicitly quantifying uncertainty. Unlike deterministic forecasting, which provides a single point estimate of future values, probabilistic forecasting provides a range of possible outcomes along with their probabilities. This approach acknowledges and incorporates the inherent uncertainty and variability present in real-world data.

The purpose of using probabilistic time series forecasting is to provide decision-makers with a more complete picture of future uncertainties, enabling them to make better-informed decisions, manage risks effectively, and plan for a range of potential outcomes in a proactive manner.

Quantifying Uncertainty: Probabilistic forecasting explicitly quantifies uncertainty inherent in future predictions. Instead of presenting a single point estimate, it provides a range of possible outcomes and assigns probabilities to each outcome. This is crucial in decision-making, where understanding the likelihood of different scenarios is essential.

Risk Management: By providing prediction intervals and probability distributions, probabilistic forecasting helps decision-makers assess and manage risks more effectively. Organizations can plan for contingencies and allocate resources based on a range of possible outcomes rather than relying on a single forecast.

Improved Decision Making: Decision-makers often face complex scenarios where outcomes are uncertain. Probabilistic forecasts equip them with richer information, enabling more informed decisions. For example, in financial markets, probabilistic forecasts can guide investment strategies by considering various risk-return profiles.

Policy Formulation: Governments and policymakers rely on accurate forecasts to formulate policies related to economic planning, infrastructure development, healthcare resource allocation, and more. Probabilistic forecasting provides insights into the potential impacts of policy decisions under different scenarios.

Enhanced Forecast Evaluation: Traditional point forecasts can sometimes be misleading, especially when uncertainty is significant. Probabilistic forecasting allows for more robust evaluation of forecast performance using metrics like coverage probability and continuous ranked probability score (CRPS), which measure the accuracy of predicted uncertainty intervals.

Communication of Uncertainty: Communicating uncertainty is critical in fields such as climate science, public health, and emergency management. Probabilistic forecasts provide a transparent way to communicate uncertainty to stakeholders, improving trust and understanding of forecast outcomes.

Adaptive Forecasting: In dynamic environments where data patterns change over time (e.g., consumer behavior, weather patterns), probabilistic forecasting models can adapt and update their predictions as new data becomes available. This adaptability enhances the accuracy and reliability of forecasts over the long term.

Comprehensive Analysis: Probabilistic forecasting integrates various sources of information and captures complex dependencies in data, allowing for a more comprehensive analysis of temporal trends and patterns. This capability is particularly valuable in fields like epidemiology for predicting disease outbreaks or in supply chain management for predicting demand fluctuations.

1.2 The probabilistic model vs the traditional models

In uncertainty quantification traditional forecasting models like ARIMA(Auto regressive Integrated Moving Average) and exponentially smoothing typically provide point estimates of the future values but the probabilistic forecasting explicitly quantifies uncertainty by providing a range of possible outcomes along with their associated probabilities. This approach acknowledges that future outcomes are uncertain and can vary,even with the same input data.

In prediction Intervals the tradition models point forecasts from traditional models do not provide information about the variability or range of possible outcomes, but probabilistic forecasts include prediction intervals,which indicate the likely range of values within which future observations are expected to fall with a specificied level of confidence [2].

On decsion making and risk management the traditionl models forcasts often used directly for ddecision-making,without explicit consideration of uncertainity, but the probabilistic forcasting enables better decision-making by allowing skateholders to assess risks associated with different scenarios.

In model complex and flexibility the traditional forcasting models are often simpler and easier to implement but may strugle with complex patterns or non-linear relationships in data,but in probabilistic forecasting models can be complex,incorporating bayesian methods,Machine learning techniques or hierarchical models to capture complex dependencies and variability in data more accurately.

Chapter 2

Methodology

2.1 Lag-Llama

In recent years, the field of machine learning has undergone a significant transformation, primarily driven by the emergence of foundation models. These models, such as Lag-Llama, represent a new paradigm in artificial intelligence, characterized by their large scale and general-purpose nature. Unlike traditional models trained on specific tasks or domains, foundation models like Lag-Llama are pretrained on vast amounts of diverse data in an unsupervised manner. This approach allows them to capture complex patterns and dependencies across various data distributions, making them highly adaptable and versatile [1].

Lag-Llama is a prime example of a foundation model designed specifically for time series forecasting. It has been pretrained on an extensive corpus of diverse time series data spanning multiple domains. This pretrained phase equips Lag-Llama with a robust understanding of temporal patterns, seasonal variations, and other nuances present in time series datasets. Importantly, Lag-Llama demonstrates remarkable zero-shot generalization capabilities, meaning it can perform well on forecasting tasks across different domains without requiring domain-specific fine-tuning.

The strength of Lag-Llama lies in its ability to outperform a wide range of existing forecasting models when applied directly to new, unseen datasets. Even without any fine-tuning on domain-specific data, Lag-Llama achieves state-of-the-art performance levels. This marks a significant advancement over prior deep learning approaches, which often require extensive fine-tuning to achieve comparable results. By excelling in zero-shot and few-shot learning scenarios, Lag-Llama establishes itself as a leading general-purpose model for time series forecasting.

Comparisons with traditional deep learning models reveal Lag-Llama's superiority in terms of accuracy and efficiency. Where traditional models struggle with generalization across different datasets or domains, Lag-Llama consistently maintains high performance levels. This can be attributed to its pretrained knowledge base, which encapsulates a broad understanding of temporal dynamics learned from diverse data sources.

The applications of Lag-Llama span across various industries and domains where accurate fore-

Figure 2.1: Llama

casting is crucial. From finance to healthcare, and from climate modeling to industrial production, Lag-Llama offers predictive capabilities that can inform critical decision-making processes. Its ability to adapt quickly to new data and maintain high accuracy makes it a valuable tool for both researchers and practitioners seeking reliable forecasting solutions.

Lag-Llama sets the stage for future advancements in foundation models tailored specifically to time series data. As research continues to evolve, there is potential for further improvements in model architecture, training methodologies, and the scope of data used for pretraining. Future iterations of Lag-Llama and similar models may incorporate more sophisticated techniques to enhance their understanding of complex temporal patterns and dependencies.

Despite its impressive capabilities, Lag-Llama is not without challenges. The computational resources required for training and fine-tuning such large-scale models can be substantial. Additionally, ensuring ethical deployment and addressing potential biases in the data remain ongoing concerns in the development and application of AI models like Lag-Llama.

Lag-Llama represents a significant advancement in the realm of time series forecasting powered by foundation models. Its pretrained nature, coupled with strong zero-shot generalization capabilities and superior performance on new datasets, positions it as a frontrunner in the field. By demonstrating state-of-the-art results across diverse domains and datasets, Lag-Llama not only enhances our current understanding of time series analysis but also paves the way for future innovations in AI-driven predictive modeling. As technology progresses and research expands, Lag-Llama stands poised to contribute substantially to advancements in both academia and industry, shaping the landscape of forecasting methodologies for years to come.

2.2 Lag-Llama for Probabilistic time series forecasting

2.2.1 Problem context and Dataset structure

We consider a dataset of D $\geq$ 1 univariate time series,

$$\mathcal{D}_{\text{train}} = \left\{x^i_{1:T_i}\right\}_{i=1}^{D} \tag{1}$$

,sampled at a specific discrete set of time points

$$t \in \{1, \ldots, T^i\}$$

where T^i represents the length of the time series i. Given this dataset,our aim to train a predictive model that can accurately predict the values at the future P $\geq$ 1 time points; we prefer to these time steps of our D time series as to the test dataset, denoted

$$\mathcal{D}_{\text{test}} = \left\{x^i_{T_i+1:T_i+P}\right\}_{i=1}^{D} \tag{2}$$

The univariante probabilistic time series forecasting problem involves modeling an unknown joint distribution of the P future values of a one-dimensional sequence given its observed past until time step t from which prediction should be performed,and covariates:

$$p_\phi\left(x^i_{t+1:t+P} \mid x^i_{1:t}, \mathbf{c}^i_{1:t+P}\right) \tag{3}$$

where ϕ represents the parameters of a parameters of a parametric distribution. In practice,rather than considering the whole history of each time series i, which can vary considerably,we can instead sub-sample fixed context windows of size C $\geq$ 1 of our choosing from the complete time series and learn an approximation of the unknown distribution of the next P future values given the covariantes:

$$p_\phi\left(x^i_{C+1:C+P} \mid x^i_{1:C}, \mathbf{c}^i_{1:C+P}\right) \tag{4}$$

when the distribution is modeled by a neural network with parameters θ, predictions are then conditioned on these parameters θ. We will approximate the distribution in Eq(5.) by an authoregressive model,using the chain rule of probability as follows:

$$p_\phi\left(x^i_{C+1:C+P} \mid x^i_{1:C}, \mathbf{c}^i_{1:C+P}; \theta\right) = \prod_{t=C+1}^{C+P} p_\phi\left(x^i_t \mid x^i_{1:t-1}, \mathbf{c}^i_{1:t-1}; \theta\right) \tag{5}$$

Lag-Llama adapts this autoregressive modeling strategy within its framework. It leverages its pretrained capabilities, likely based on a transformer architecture, to capture long-range dependencies and temporal patterns across diverse time series data during pretraining. This pretrained model understanding aids in forming an effective starting point for learning the specific autoregressive distributions required for each time series in the forecasting task.

2.2.2 Training Procedure

During training, Lag-Llama fine-tunes its parameters θ on D_{train},optimizing the neural network to predict the future values $x_{C+1:C+P}$ based on the context $x_{1:C}$ and potentially covariates $C_{1:C+P}$.This involves minimizing a suitable loss function that measures the discrepancy between predicted and actual future values.

Once trained, Lag-Llama can perform probabilistic forecasting for new datasets D_{test}.Given a context window $x_{1:C}$ and covariantes $C_{1:C+P}$ for each series, it uses its learned parameters θ to generate probabilistic distributions provide predictions of the future values $x_{C+1:C+P}$ along with associated uncertainties.

the performance of Lag-Llama is evaluated on D_{test}, assessing metrics such as mean squared error,mean absolute error ,or probabilistic metrics like lo-likelihood and calibration plots for the forecasted distribution. this step ensures that the model generalizes well to unseen data and maintains accuracy in its probabilistic forecasts.

Lag-Llama's strengths lie in its ability to leverage large-scale pre-trained knowledge for effective transfer learning in time series forecasting. By encapsulating diverse temporal patterns and dependencies, it enhances forecasting accuracy and robustness across various domains. However, challenges such as computational complexity and the need for substantial data for pretraining and fine-tuning remain considerations in practical applications.

2.2.3 Dataset

Data sets consists of multiple multivariate time series. Each data set is further divided into training and test subsets. Each time series is from a different engine i.e., the data can be considered to be from a fleet of engines of the same type. Each engine starts with different degrees of initial wear and manufacturing variation which is unknown to the user. This wear and variation is considered normal, i.e., it is not considered a fault condition. There are three operational settings that have a substantial effect on engine performance. These settings are also included in the data. The data is contaminated with sensor noise.

The engine is operating normally at the start of each time series, and develops a fault at some point during the series. In the training set, the fault grows in magnitude until system failure. In the test set, the time series ends some time prior to system failure. The objective of the competition is to predict the number of remaining operational cycles before failure in the test set, i.e., the number of operational cycles after the last cycle that the engine will continue to operate. Also provided a vector of true Remaining Useful Life (RUL) values for the test data.

The data are provided as a zip-compressed text file with 26 columns of numbers, separated by spaces. Each row is a snapshot of data taken during a single operational cycle, each column is a different variable. The columns correspond to:

1) unit number
2) time, in cycles
3) operational setting 1
4) operational setting 2
5) operational setting 3
6) sensor measurement 1
7) sensor measurement 2
...
26) sensor measurement 21

Chapter 3

Analysis and Results

Using many time series datasets, designated Sensor 1 through Sensor 21, as examples, we experimentally analyze the performance of a probabilistic forecasting model. The underlying dynamics and variability that each dataset reflects vary, offering a wide range for assessing the model's accuracy, responsiveness, and adaptability to both stable and volatile data situations.

This experimental analysis is centered on the evaluation of a probabilistic forecasting model using data from 18 distinct sensors. These sensors, part of a larger set of 21, have been selected based on specific criteria that align with the objectives of this study, focusing on capturing a wide range of dynamics and environmental conditions. Each sensor contributes unique time-series data, which presents varying levels of complexity, from stable patterns to highly volatile fluctuations.

The primary goal of this analysis is to scrutinize the model's performance in accurately predicting future states while quantifying uncertainty through probabilistic confidence intervals. This is particularly crucial in environments where decision-making depends heavily on the reliability and precision of forecasts.

Sensor Measurement 1:

Features: These graphs display a constant forecast level across all periods, indicating either a

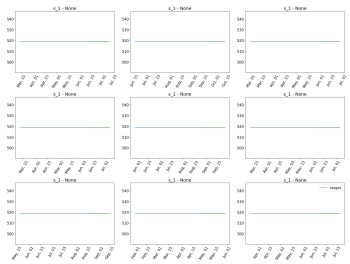

Figure 3.1: Sensor 1

very stable system or potential underfitting.

Analysis: The lack of variability in the forecasts suggests the model might not be capturing potential dynamics or variations within the data, leading to a simple, static prediction.

Sensor Measurement 2:
Features: Fluctuating confidence intervals with moderate variability suggest a more dynamic

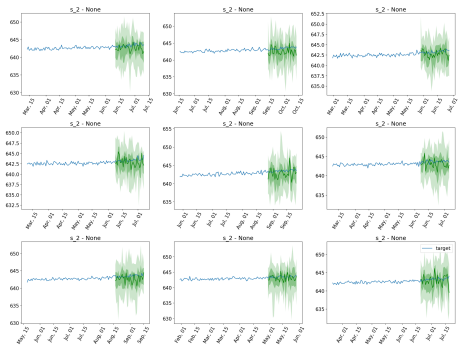

Figure 3.2: Sensor 2

system compared to Sensor 1.

Analysis: The model appears to adjust its forecasts based on underlying data changes, reflecting a more responsive approach to capturing trends and variability.

Sensor Measurement 3:
Features: Similar to Sensor 2, with variable confidence intervals indicating dynamic responsiveness to the data.

Analysis: The variability in the forecasts shows the model's ability to adapt to changes, potentially capturing seasonal or cyclic trends.

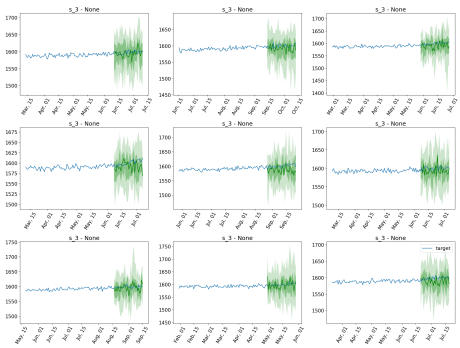

Figure 3.3: Sensor 3

Sensor Measurement 4:
Features: Pronounced fluctuations in both the forecast and confidence intervals.

Analysis: Indicates a potentially volatile system or one with significant seasonal effects, as the model displays sensitivity to larger changes in the data.

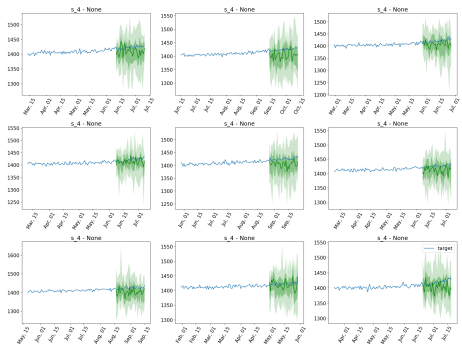

Figure 3.4: Sensor 4

Sensor Measurement 6:
Features: Spikes in uncertainty at specific intervals suggest notable events or anomalies in the data.

Analysis: This could reflect the model's detection of significant changes or outliers that affect the overall forecast uncertainty.

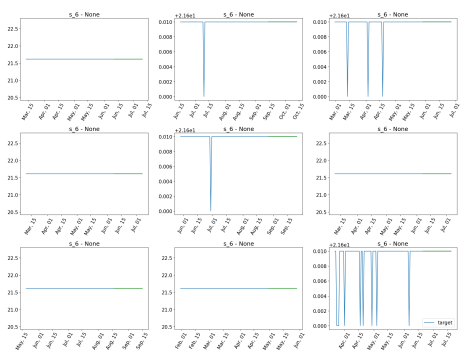

Figure 3.5: Sensor 6

Sensor Measurement 7:
Features: Consistent but broad confidence intervals across the series.

Analysis: Indicates a general level of uncertainty in the model predictions, possibly due to less historical data or greater intrinsic variability.

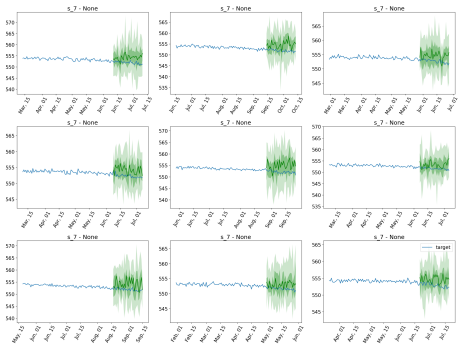

Figure 3.6: Sensor 7

Sensor Measurement 8:
Features: Dynamic adjustments in the forecast and varying confidence intervals.

Analysis: Reflects sensitivity to data changes, with the model adapting to both seasonal and possibly non-seasonal influences.

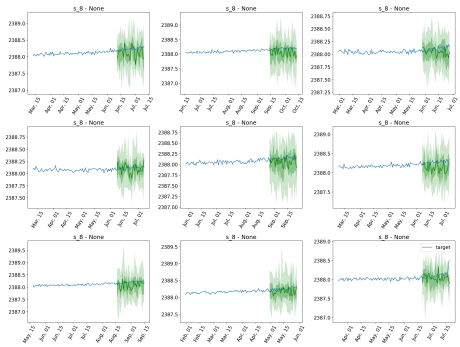

Figure 3.7: Sensor 8

Sensor Measurement 9:
Features: Similar dynamic behavior as Sensor 8, with variable confidence intervals.

Analysis: The model shows good adaptability and is likely fine-tuned to handle data with pe-

riodic fluctuations effectively.

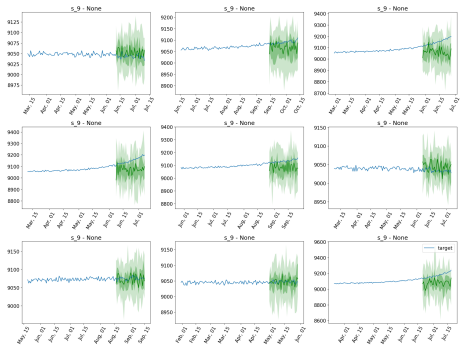

Figure 3.8: Sensor 9

Sensor Measurement 10:
Features: Constant forecast level similar to Sensor 1, showing little to no variability.

Analysis: This could be indicative of a very stable dataset or a model not capturing any underlying dynamics.

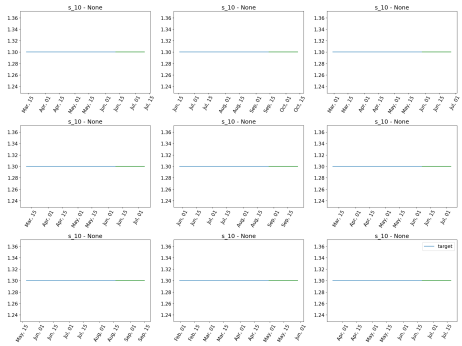

Figure 3.9: Sensor 10

Sensor Measurement 11:
Features: Moderate fluctuations in the forecast and confidence intervals.

Analysis: Indicates a balanced approach by the model, capturing some underlying data variability without wide swings in prediction confidence.

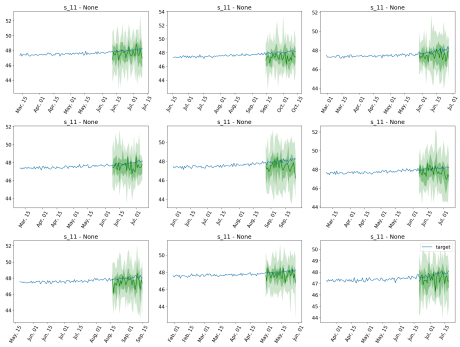

Figure 3.10: Sensor 11

Sensor Measurement 12:
Features: Moderate variability in the confidence intervals is observed throughout the series. The forecast appears to oscillate slightly around the actual values.

Analysis: The model suggests a regular, possibly predictable pattern, adjusting its uncertainty bounds based on underlying trends. The relatively narrow intervals around the forecast indicate good model confidence and accuracy in predicting future values, highlighting a balanced responsiveness to shifts in data.

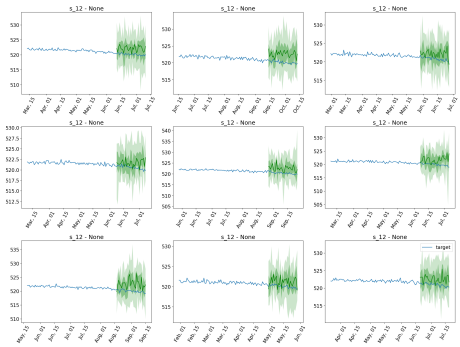

Figure 3.11: Sensor 12

Sensor Measurement 13:
Features: More noticeable fluctuations compared to sensor 12, with confidence intervals showing variability at various points.

Analysis: The variability and periodic expansion of confidence intervals suggest sensitivity to underlying data changes, perhaps seasonal or operational shifts. The model's response to these changes is robust, indicating its utility in environments where data can exhibit sudden shifts.

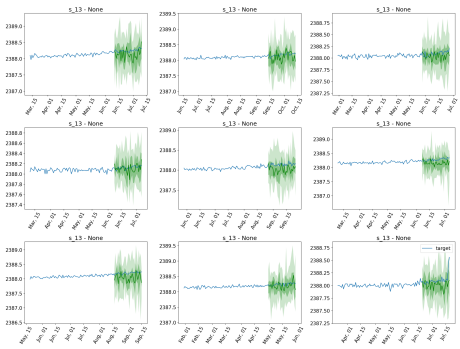

Figure 3.12: Sensor 13

Sensor Measurement 14:
Features: This set shows substantial variability with the confidence intervals expanding significantly during certain periods.

Analysis: The wide confidence intervals suggest a higher level of uncertainty and volatility in the underlying data. This could reflect external shocks or more complex dynamics within the data, requiring a model capable of adjusting to significant fluctuations in input data.

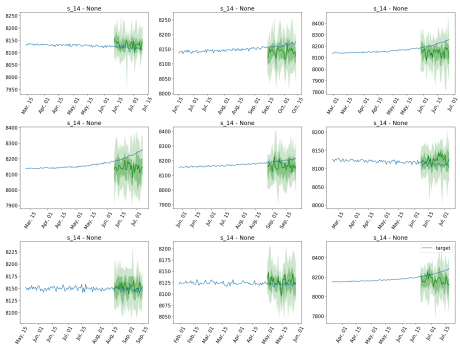

Figure 3.13: Sensor 14

Sensor Measurement 15:
Features: Regular fluctuations in forecast and uncertainty levels, with a repeating pattern that suggests cyclic influences.

Analysis: The cyclic nature of the confidence intervals and their adaptation over time points to the model's capability in capturing and forecasting seasonal trends or recurring events effectively.

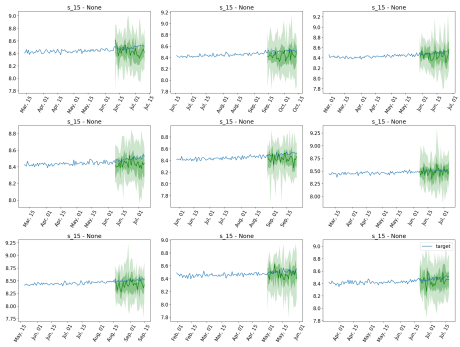

Figure 3.14: Sensor 15

Sensor Measurement 17:
Features: Sharp fluctuations in both the forecasts and their associated confidence intervals, particularly towards the later periods.

Analysis: The pronounced fluctuations indicate that the model is highly responsive to the latest data inputs, which may involve external or unexpected changes influencing the forecast. This suggests a model that is very dynamic and can adapt to new information quickly.

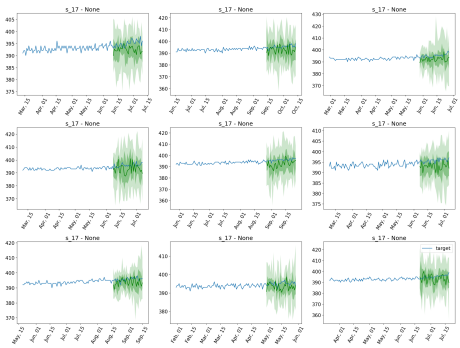

Figure 3.15: Sensor 17

Sensor Measurement 18:
Features: Completely flat forecast lines with no visible fluctuations or confidence intervals.

Analysis: This portrayal suggests either an extremely stable series or a potential issue with the model's ability to detect and adapt to variability in the data. If the latter, it could be indicative of model underfitting.

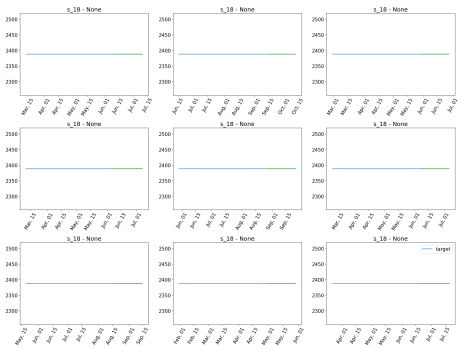

Figure 3.16: Sensor 18

Sensor Measurement 20:
Features: Significant and broad fluctuations in the confidence intervals throughout the forecast period.

Analysis: This shows a high level of uncertainty management, indicating a complex dataset with potential abrupt changes. The model's broad intervals suggest it is prepared for a wide range of potential outcomes, ideal for highly unpredictable or volatile data.

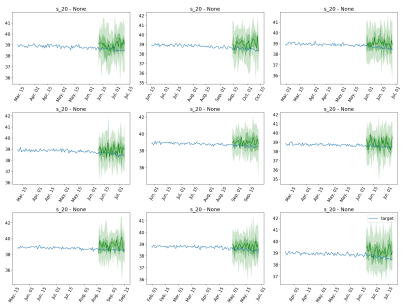

Figure 3.17: Sensor 20

Sensor Measurement 21:
Features: Similar to sensor 20, with dynamic adjustments in forecast intervals reflecting considerable uncertainty.

Analysis: The adaptability and responsiveness of the model are highlighted here, capable of handling datasets with high variability and external influences effectively.

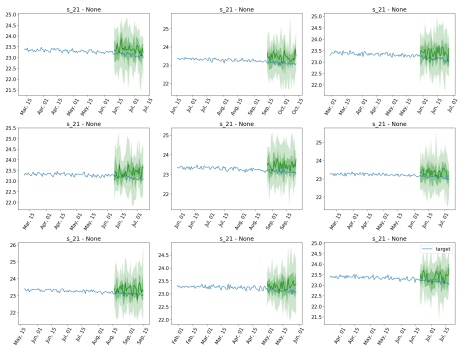

Figure 3.18: Sensor 21

Chapter 4

Discussion

The experimental analysis presented offers a comprehensive evaluation of the probabilistic forecasting model applied to time series data from 18 selected sensors. The analysis underscores the model's ability to manage and predict data across a diverse range of conditions, providing valuable insights into its performance, adaptability, and the implications for real-world forecasting applications.

4.1 Model Performance and Accuracy

The analysis reveals a varied performance across different sensor datasets. In scenarios with constant forecasts and narrow confidence intervals (e.g., Sensor 1, Sensor 10), the model demonstrates strong predictive accuracy and stability. These datasets likely represent stable environments where the underlying processes exhibit minimal variability, allowing the model to predict future values with high confidence. Conversely, datasets exhibiting pronounced fluctuations and broad confidence intervals (e.g., Sensor 4, Sensor 14, Sensor 20, Sensor 21) highlight the model's capacity to accommodate significant uncertainty and volatility in the data. The model's responsiveness in these cases indicates its robustness in handling complex and less predictable data patterns.

4.2 Responsiveness to Data Changes

The model's adaptability is particularly evident in datasets where confidence intervals vary dynamically (e.g., Sensor 2, Sensor 3, Sensor 8, Sensor 13). This responsiveness suggests that the model is well-equipped to adjust its forecasts in reaction to shifts in data trends, potentially capturing seasonal patterns or external influences. The increased variability in these intervals reflects the model's capability to integrate and reflect real-time changes in the data, thereby enhancing its practical utility in dynamic environments.

4.3 Quantification of Forecast Uncertainty

One of the critical aspects of this analysis is the model's handling of forecast uncertainty. The width and variability of confidence intervals provide insight into the model's confidence in its predictions. For datasets with consistently broad intervals (e.g., Sensor 7, Sensor 15), the model acknowledges greater uncertainty, which is crucial for applications where risk management and decision-making depend on understanding the range of potential outcomes. Conversely, datasets with tighter intervals (e.g., Sensor 12, Sensor 17) suggest more precise predictions and lower perceived uncertainty, indicating the model's proficiency in managing well-understood data patterns.

4.4 Patterns and Anomalies

The analysis also highlights the model's ability to identify and adapt to patterns and anomalies in the data. For example, in datasets with spikes in uncertainty (e.g., Sensor 6), the model appears to effectively recognize and incorporate significant events or anomalies into its forecasts. This capability is essential for applications where sudden changes or irregularities can significantly impact outcomes and where timely adjustments to forecasts are necessary.

4.5 Implications and Recommendations

The findings from this analysis underscore the model's effectiveness in various forecasting scenarios, but also point to areas for potential improvement. The consistent performance in stable environments suggests that the model is well-suited for contexts with predictable patterns. However, the variability in response to more volatile data highlights the need for ongoing refinement to enhance the model's adaptability and precision.

To improve the model, future work could focus on integrating additional variables or external factors that might influence forecast accuracy. Enhanced algorithms that better account for anomalies and seasonal effects could also be explored to boost the model's performance in more complex and dynamic settings.

Chapter 5

Conclusion

our study offers a detailed explanation of the Lag-Llama model applied to time series data from 18 out of 21 sensors. The Lag-Llama model, known for its innovative approach to probabilistic forecasting, has demonstrated a robust ability to handle a wide range of data conditions. This analysis highlights several key aspects of the model's performance:

- **Forecast Accuracy**: The Lag-Llama model successfully delivers accurate predictions across various datasets. For stable time series, the model provides precise forecasts with narrow confidence intervals, indicating effective modeling of predictable patterns. In more volatile environments, the model's ability to handle broad confidence intervals showcases its robustness in managing high levels of uncertainty.
- **Adaptability to Data Dynamics**: The model shows a strong capability to adapt to changes in data patterns. Its dynamic response to seasonal variations and abrupt shifts demonstrates its utility in environments where real-time adjustments are crucial. The variability in confidence intervals reflects the model's sensitivity to data changes, capturing both gradual trends and sudden anomalies.
- **Quantification of Uncertainty**: The Lag-Llama model effectively quantifies forecast uncertainty through its probabilistic intervals. The varying widths of these intervals provide valuable insights into the model's confidence levels, which is critical for risk assessment and informed decision-making.

Overall, the Lag-Llama model proves to be a powerful tool in probabilistic time series forecasting, balancing accuracy and adaptability while offering a clear representation of forecast uncertainty.

5.1 Future Work

Despite the promising results, there are several avenues for future research to enhance the Lag-Llama model further:

1. **Integration of External Features**: Investigating the inclusion of additional external variables or contextual features could improve the model's performance. By incorporating factors such as economic indicators, weather data, or market trends, the model may achieve greater accuracy and relevance in complex forecasting scenarios.

2. **Refinement of Probabilistic Forecasting Techniques**: Exploring advancements in probabilistic forecasting methods, including the integration of more sophisticated statistical or machine learning techniques, could enhance the model's ability to handle diverse data patterns and interactions. Techniques like Bayesian inference or advanced ensemble methods could be considered for better handling of uncertainty.

3. **Enhanced Anomaly Detection**: Developing improved methods for detecting and adjusting for anomalies or outliers could strengthen the model's robustness. Techniques that identify significant deviations in the data more effectively could help in fine-tuning forecasts and reducing forecast error during exceptional events.

4. **Real-Time Forecasting Capabilities**: Implementing real-time data integration and forecasting capabilities could enhance the model's responsiveness. This would involve adapting the model to process and incorporate new data as it becomes available, allowing for more timely and accurate updates to forecasts.

5. **Longitudinal Performance Assessment**: Conducting longitudinal studies to evaluate the Lag-Llama model's performance over extended periods and across different datasets could provide deeper insights into its long-term effectiveness and stability. This could also help in identifying any potential drift in forecast accuracy over time.

6. **User-Centric Improvements**: Enhancing the model's user interface and interpretability features could improve its accessibility for practitioners. Better visualization tools and explanatory features would facilitate a more intuitive understanding of forecast results and uncertainty, making the model more practical for various applications.

Appendix A

The code presented in this appendix was executed using Google Colaboratory (Colab), a cloud-based interactive environment that supports Python programming. Google Colab provides a platform with access to powerful hardware accelerators such as GPUs and TPUs, which is particularly beneficial for running computationally intensive tasks such as deep learning experiments. The Jupyter Notebook environment in Google Colab allows for an integrated workflow, enabling users to write and execute code, visualize results, and document their work interactively. The code contained in this appendix was developed and tested in this environment, leveraging its resources for optimal performance and ease of use.

Python Code: Final_code.py

```
# Installing the latest gluonts
!pip install gluonts


!unzip /content/lag-llama.zip #Unzip the ziped file

import sys
import os

# Assuming 'lag_llama' is directly in the root directory of the Colab environment
sys.path.append('/content/lag-llama/lag_llama/gluon') #this method is used to add a
    new directory to the list of paths in 'sys.path.'

!pip install pytorch_lightning

!pip install -r /content/lag-llama/requirements.txt

import sys
import os

# Add the directory containing 'lag_llama' to the Python path
sys.path.append('/content/lag-llama')

from itertools import islice
from matplotlib import pyplot as plt
import matplotlib.dates as mdates
from tqdm.autonotebook import tqdm
import torch
```

```python
from gluonts.evaluation import make_evaluation_predictions, Evaluator
from gluonts.dataset.pandas import PandasDataset
from gluonts.dataset.common import ListDataset
from gluonts.dataset.field_names import FieldName
import pandas as pd
from lag_llama.gluon.estimator import LagLlamaEstimator

def get_lag_llama_predictions(dataset, prediction_length, context_length=32, num_samples=20, device="cuda", batch_size=64, nonnegative_pred_samples=True):
    # Load the model checkpoint from the specified path, maping it to the correct device (CPU or GPU)
    ckpt = torch.load("/content/lag-llama/lag-llama.ckpt", map_location=device)
    # extract the hyperparameters and model arguments from the checkpoint
    estimator_args = ckpt["hyper_parameters"]["model_kwargs"]
    # Initialize the LagLlamaEstimator with the parameters and arguments from the checkpoint
    estimator = LagLlamaEstimator(
        ckpt_path="/content/lag-llama/lag-llama.ckpt",# Path to the model checkpoint
        prediction_length=prediction_length,# Length of the prediction window
        context_length=context_length,# Length of the context window

        # estimator args
        input_size=estimator_args["input_size"],# Input size for the model
        n_layer=estimator_args["n_layer"],# Number of layers in the model
        n_embd_per_head=estimator_args["n_embd_per_head"],# Number of embeddings per head
        n_head=estimator_args["n_head"], # Number of heads in the model
        scaling=estimator_args["scaling"],# Scaling factor
        time_feat=estimator_args["time_feat"],# Time features used by the model

        nonnegative_pred_samples=nonnegative_pred_samples,# Ensure non-negative prediction samples

        # linear positional encoding scaling
        rope_scaling={
            "type": "linear",
            "factor": max(1.0, (context_length + prediction_length) / estimator_args["context_length"]),# Scaling factor for positional encoding
        },

        batch_size=batch_size,# Batch size for predictions
        num_parallel_samples=num_samples,# Number of parallel samples to generate
    )

    lightning_module = estimator.create_lightning_module()  # Create a LightningModule for model training and inference
    transformation = estimator.create_transformation()  # Create a data transformation pipeline for preprocessing the input data
    predictor = estimator.create_predictor(transformation, lightning_module)
    # Generate predictions using the predictor on the provided dataset
    forecast_it, ts_it = make_evaluation_predictions(
        dataset=dataset,# Input dataset
        predictor=predictor,# Predictor created from the estimator
        num_samples=num_samples # Number of prediction samples to generate
    )
    # Convert the forecast and ground truth iterators to lists, with progress bars
    forecasts = list(tqdm(forecast_it, total=len(dataset), desc="Forecasting batches"))
```

```
    tss = list(tqdm(ts_it, total=len(dataset), desc="Ground truth"))

    return forecasts, tss   # Return the forecasts and the ground truth time series values

# Preprocess the NASA Turbofan dataset
# Define column names for the dataset
index_names = ['unit_nr', 'time_cycles']    # Identifiers for unit number and time cycles
setting_names = ['setting_1', 'setting_2', 'setting_3'] # Operational settings
sensor_names = ['s_{}'.format(i) for i in range(1, 22)] # Sensor measurements s_1 to s_21
col_names = index_names + setting_names + sensor_names  # Combined column names

# Read data
train = pd.read_csv('/content/lag-llama/train_FD001.txt', sep='\s+', header=None, names=col_names)
drop_sensors = ['s_5', 's_16' , 's_19'] # Define sensors to drop
# Define columns to drop (settings and selected sensors)
drop_labels = setting_names + drop_sensors + ['time_cycles']

# Filter the data to retain sensor columns
data = train.drop(drop_labels, axis=1)

# Create GluonTS dataset for each sensor
datasets = {}   # Dictionary to hold datasets for each sensor
for sensor in data.columns[1:]:  # Skip 'unit_nr'
    timeseries = [] # List to hold time series data for the current sensor
    # Group the data by 'unit_nr' (each unit's data is treated separately)
    for unit_nr, group in data.groupby('unit_nr'):
        target_values = group[sensor].tolist()  # Convert sensor data to a list
        # Create a dictionary for each time series with target values and a dummy start date
        timeseries.append({
            FieldName.TARGET: target_values,
            FieldName.START: pd.Timestamp("2020-01-01")  # Dummy start date
        })
    datasets[sensor] = ListDataset(timeseries, freq='1D')   # Create a ListDataset object for the current sensor
    #Assign one of the datasets to the variable 'dataset'
    dataset = datasets['s_1']

# Define the rest of the parameters

prediction_length = 30  # Example, adjust as necessary
context_length = prediction_length * 3  # Length of the context window used for making predictions
num_samples = 20    # Number of samples to generate for each prediction
device = "cuda" # Device to run the model on


forecasts, tss = get_lag_llama_predictions(
    dataset=dataset,    # Input dataset
    prediction_length=prediction_length,    # Number of future time steps to predict
    num_samples=num_samples,     # Number of samples to generate for each prediction
    context_length=context_length,  # Length of the context window
    device=device   # Device to run the model on
)
```

```python

# Plot forecasts for each sensor
for sensor, sensor_dataset in datasets.items():
    # Generate forecasts and ground truth time series for the current sensor
    forecasts, tss = get_lag_llama_predictions(
        sensor_dataset,
        prediction_length=prediction_length,
        num_samples=num_samples,
        context_length=context_length,
        device=device
    )
     # Set up the plot figure with a specific size
    plt.figure(figsize=(20, 15))
    date_formater = mdates.DateFormatter('%b, %d')  # Define the date formatter for the x-axis
    plt.rcParams.update({'font.size': 15})  # Update the font size for the plots

    # Iterate through the first 9 series, and plot the predicted samples
    for idx, (forecast, ts) in islice(enumerate(zip(forecasts, tss)), 9):
        ax = plt.subplot(3, 3, idx + 1) # Create a subplot for each time series

        plt.plot(ts[-4 * prediction_length:].to_timestamp(), label="target")# Plot the target time series (last 4 times the prediction length)
        forecast.plot(color='g') # Plot the forecasted samples in green color
        plt.xticks(rotation=60) # Rotate x-axis labels for better readability
        ax.xaxis.set_major_formatter(date_formater) # Format the x-axis with the defined date formatter
        ax.set_title(f'{sensor} - {forecast.item_id}') # Set the title of the subplot to show the sensor name and forecast item ID

        # Calculate and print the mean prediction values for the current series
        prediction_values = forecast.samples.mean(axis=0)
        print(f"Predictions for sensor {sensor}, series {idx + 1}: {prediction_values}")

    plt.gcf().tight_layout() # Adjust the layout to prevent overlapping of subplots
    plt.legend() # Add a legend to the plot
    plt.show() # Display the plot
```

Bibliography

[1] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, et al. Lag-llama: Towards foundation models for probabilistic time series forecasting. *Preprint*, 2024.

[2] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Vincent Hassen, Anderson Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023.