Evaluation of state-space models and widely used architectures in deep learning for heterogenous tasks

By

Ankush Paul

MT22190

Under the Supervision of Dr. Vibhor Kumar

Indraprastha Institute of Information Technology, Delhi

August, 2024

Evaluation of state-space models and widely used architectures in deep learning for heterogenous tasks

By

Ankush Paul

MT22190

Submitted

in partial fulfilment of the requirements for the degree of Master of

Technology

To

Indraprastha Institute of Information Technology Delhi August, 2024

CERTIFICATE

This is to certify that the thesis titled " **Evaluation of state-space models and widely used architectures in deep learning for heterogenous tasks**" being submitted by Ankush Paul (MT22190) to the Indraprastha Institute of Information Technology, Delhi, for the award of the Master of Technology, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

August, 2024, Dr. Vibhor Kumar

Department of Computational Biology

Indraprastha Institute of Information Technology,

Delhi New Delhi 110020

Acknowledgments

I would like to express my heartfelt gratitude to my supervisor, Dr. Vibhor Kumar, for his consistent support and motivation throughout this project. Their invaluable guidance and encouragement have been instrumental in the completion of this work. Also I would like to thanks my all batchmates and Professors for encouraging me.

The Linear and Non-linear state-space models like Kalman filter and Recurrent Neural Networks (RNN) have been used widely for prediction of signals and response of the system. Such as Kalman filter have been used for tracking movement of objects and RNN has been used in time series prediction. Recent trend in Deep learning field has lead to extensions of RNN for text processing and text based Response system. For the same purpose transformer model also been heavily used. Here we benchmark different state space model and transformers in predicting future signals and text which can be applied in various domain. We benchmarked different models for noisy signals as well as large text-based responses. Finally, we have developed our own simplified architecture which could perform tasks meant for deep learning model with lower number of parameters for applications in healthcare and biological sciences.

Table Of Contents

Introduction to Neural Network and State Space Model

1.1 Introduction to Neural Network **1**

1.2 Introduction To State Space Model **2**

1.3 Differences and Similarities between State Space Model and Neural Network. **4**

Comparing Neural Network and State Space Model for Continuous Signal

2.1 Dataset Used **5**

2.3 Introduction to noise **6**

2.3 Implementations **10**

2.4 Results **12**

2.5 Discussion **13**

Neural Network/Deep Learning Architecture and State Space Model for Categorical Data

3.1 Dataset Used **14**

3.2 Implementation **14**

3.3 Results **14**

Conclusions **17**

References **18**

List Of Figures

Figure 1. Equations representing forward propagation in neural networks.	**1**
Figure 2. Equations representing backward propagation in neural networks.	**1**
Figure 3. Architectures of deep learning models expanding from recurrent neural networks (RNN), Long short-term memory (LSTM), Gated recurrent unit (GRU) and Transformers.	**2**
Figure 4. Basic workflow of Kalman filter.	**2**
Figure 5. Different states of Kalman filter	**3**
Figure 6. Weather forecasting data representing temperature as a feature for 1ooooo data points	**5**
Figure 7. Additive noise incorporation in weather forecasting data.	**6**
Figure 8. Multiplicative noise incorporation in weather forecasting data.	**7**
Figure 9. RMSE vs variance graph for RNN, LSTM, Transformer and Kalman filter on weather forecasting additive noisy data.	**8**
Figure 10. RMSE vs variance graph for RNN, LSTM, Transformer and Kalman filter on weather forecasting multiplicative noisy data.	**8**
Figure 11. Stock market pricing statistics data from Jan 2012 to May 2024.	**9**
Figure 12. RMSE vs variance graph for RNN, LSTM, Transformer and Kalman filter on stock market pricing additive noisy data.	**10**
Figure 13. RMSE vs variance graph for RNN, LSTM, Transformer and Kalman filter on stock market pricing multiplicative noisy data.	**11**
Figure 14. Bar plot comparison of all DL models using 4 scores BLEU, ROUGE, METEOR and cosine similarity for text generation of text seed "A long lens shot of a far distant metallic object hovering".	**16**
Figure 15. Bar plot comparison of all DL models using 4 scores BLEU, ROUGE, METEOR and cosine similarity for text generation of text seed "Do not Try to bend the spoon that's impossible".	**17**

Lists Of Tables

Table 1. *Output Result with Seen Text for all the models, RNN, LSTM, Transformers and Kalman filters on text data for the seed sentence "A long lens shot of a far distant metallic object hovering*	**15**
Table 2. *Output Result with Seen Text for all the models, RNN, LSTM, Transformers and Kalman filters on text data for the seed sentence "Do not Try to bend the spoon that's impossible.*	**16**

Chapter 1

1.1 Introduction To Neural Network

Neural networks are computational models designed to simulate the way the human brain processes information, enabling pattern recognition and predictive capabilities. These models consist of interconnected layers of artificial neurons that process input data through a series of transformations. Activation functions, such as RELU, Sigmoid, and Tanh, are applied to introduce non-linearity into the network, enhancing its ability to learn complex patterns. Key neural network architectures include Feedforward Neural Networks (FNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), and Transformers, each tailored to different types of data and learning tasks

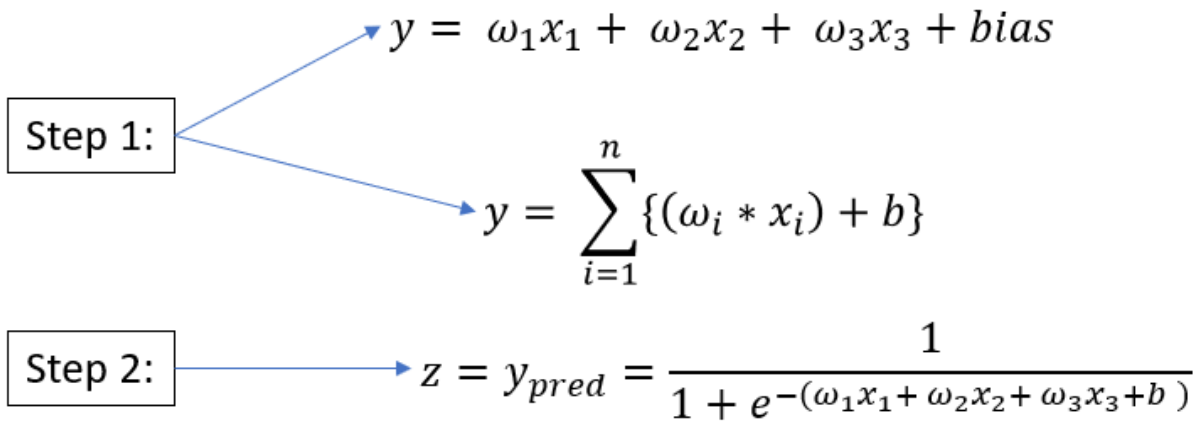

***Figure 1**. Equations representing forward propagation in neural networks.*

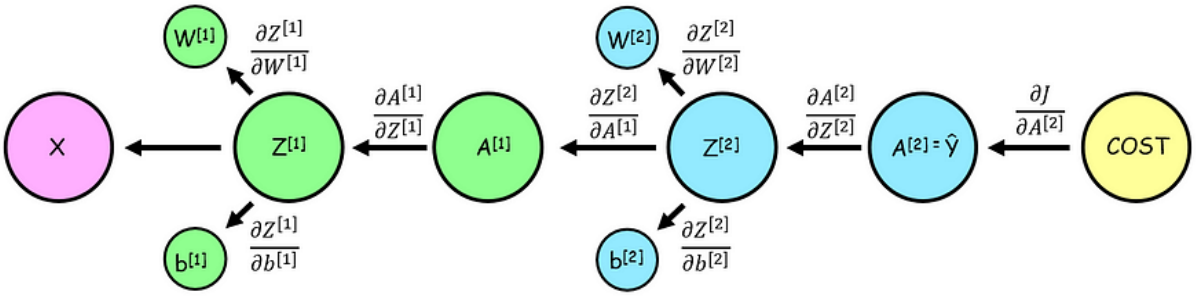

***Figure 2**. Equations representing backward propogation in neural networks.*

So, Back Propagation is basically Updating the Weights and Minimize the Loss Function.

Current Neural Network Used

The deep learning models being used for benchmarking are RNN, LSTM, and Transformer.

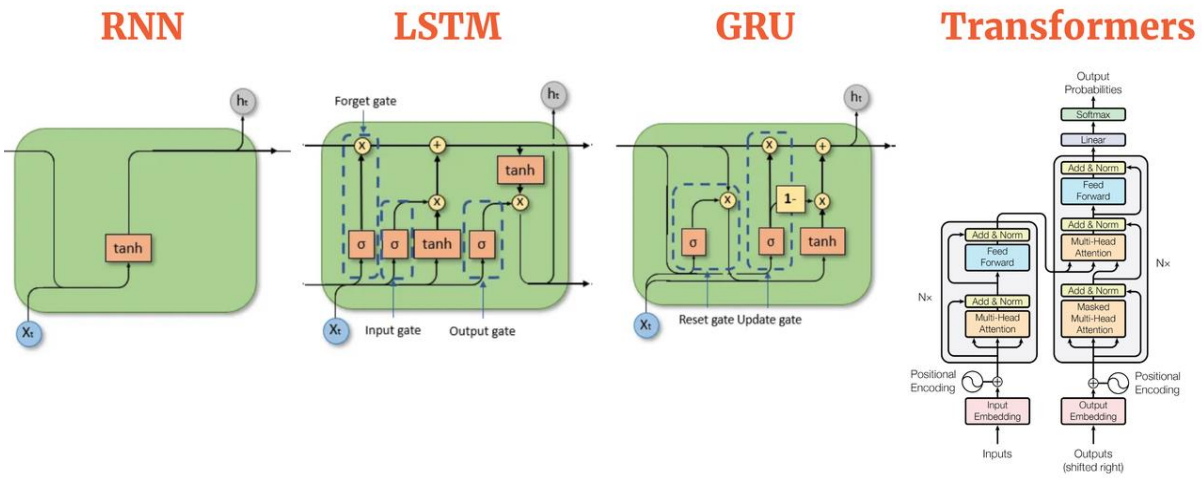

***Figure 3**. Architectures of deep learning models expanding from recurrent neural networks (RNN), Long short term memory (LSTM), Gated recurrent unit (GRU) and Transformers.*

1.2 Introduction to State Space Models

State space models provide a framework for representing and analysing dynamic systems through a combination of state transitions and observations. These models describe how the internal state of a system evolves over time based on inputs and prior states. The Kalman Filter, a key tool in state space modelling, estimates the internal state of a linear system by processing noisy observations. It operates in a two-step process: first, it predicts the future state based on the current state and known dynamics, and second, it updates this prediction using new measurements to estimate the state with reduced uncertainty. This approach allows for accurate

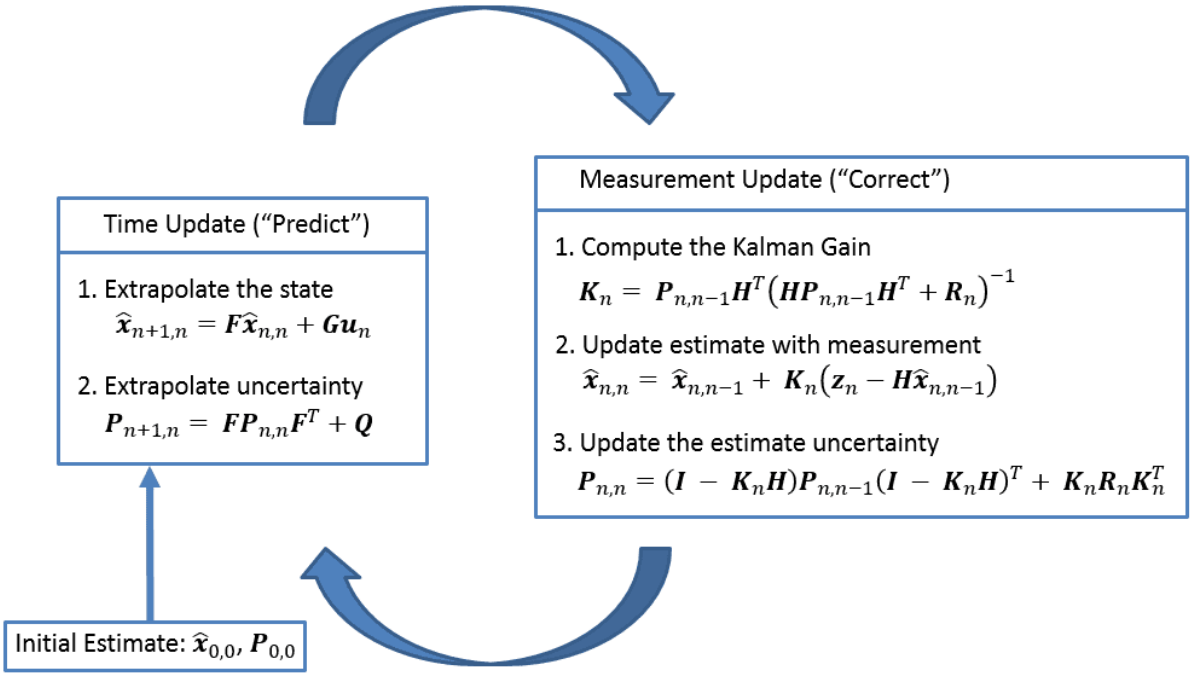

***Figure 4.** Basic workflow of Kalman filter.*

tracking and forecasting even when the system is subject to random disturbances and measurement errors. By integrating predictions and corrections, the Kalman Filter effectively manages and mitigates the impact of noise, making it a powerful method for robust system.

States In Kalman Filter

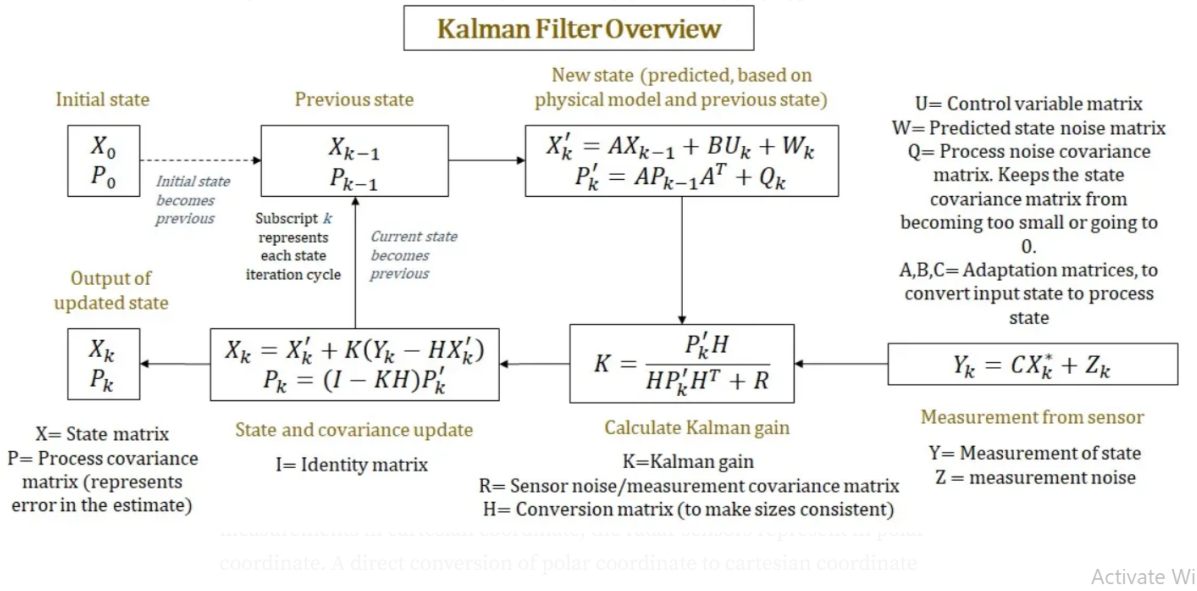

Figure 5. *Different states of Kalman filter.*

1.3 Differences and Similarities Between State Space Model and Neural Network

Differences: Kalman Filters and neural networks approach prediction and estimation in fundamentally different ways. Kalman Filters operate within a structured mathematical framework designed for linear systems, updating the system's state based on noisy measurements and predictions. This iterative process involves a prediction step, where the system's future state is estimated, and a correction step, where this prediction is adjusted using new data to improve accuracy. In contrast, neural networks leverage a learning-based approach where they refine their predictive capabilities through training on large datasets. They use complex, non-linear functions to model intricate patterns and relationships within the data. While Kalman Filters are particularly effective for systems with well-defined linear dynamics and noise characteristics, neural networks are versatile tools capable of capturing more complex, non-linear patterns across various data types.

Similarities: Despite their different methodologies, Kalman Filters and neural networks share common objectives in prediction and estimation. Both techniques aim to refine their outputs based on incoming data, with Kalman Filters updating system states and neural networks adjusting weights and biases during training. They handle sequential data effectively, with Kalman Filters incorporating temporal information into its state updates, and neural networks using structures like recurrent layers to process sequences. Both methods work to minimize

errors, with Kalman Filters achieving this through iterative prediction and correction, while neural networks rely on backpropagation and optimization algorithms to reduce prediction errors over multiple training epochs. Their ability to manage and improve predictions based on continuous data input highlights their shared focus on enhancing accuracy and performance in dynamic environments.

Chapter 2

Comparing Neural Network and State Space Model for Continuous Signal

What is continuous Signal?

A continuous signal is one that changes smoothly over time and can assume any value within a specific range. It is defined at every moment, without gaps or jumps, and is typically represented by continuous functions that allow for precise measurement at any instant.

2.1 Dataset Used

- Weather Forecasting Data
- Dataset consists more than 20 features and more than 100000 rows

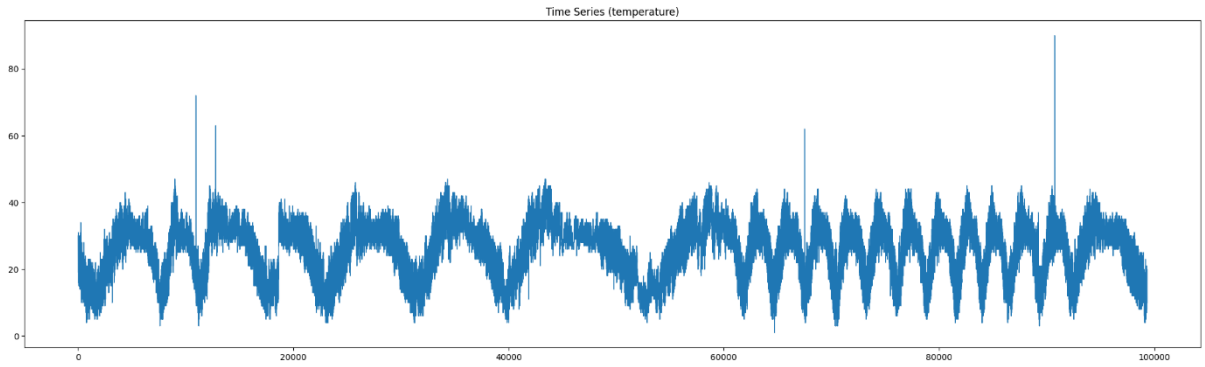

***Figure 6**. Weather forecasting data representing temperature as a feature for 1ooooo data points.*

Data Preparation: The weather forecasting dataset undergoes comprehensive preparation to ensure it is suitable for model training. This involves cleaning the data by addressing missing values, normalizing features, and splitting it into training and testing sets. Additional steps may include feature selection or extraction to focus on the most relevant variables for accurate forecasting, Univariate Analysis.

Training with LSTM: The Long Short-Term Memory (LSTM) network is trained on the weather forecasting dataset to capture temporal dependencies. LSTMs are adept at learning from sequential data and handling long-term dependencies, making them suitable for forecasting future weather conditions based on historical patterns.

Training with RNN: A Recurrent Neural Network (RNN) is also trained using the weather dataset. While RNNs can capture sequential dependencies, they are generally less effective at managing long-term dependencies compared to LSTMs. The RNN model focuses on identifying shorter-term trends and patterns in the weather data.

Training with Transformer: The Transformer model is employed to handle the weather forecasting task. Transformers utilize self-attention mechanisms to process and weigh different time steps in the sequence, allowing them to capture complex dependencies in the data. The Transformer model is trained to predict future weather conditions by analyzing both short-term and long-term patterns.

Training with Kalman Filter: The Kalman Filter is used for forecasting by estimating the internal state of the weather system through noisy measurements. It operates in a two-step process: prediction and update. In the prediction phase, it forecasts future states based on current estimates, while in the update phase, it refines these predictions using new observational data.

Evaluation: Each model's performance is assessed using metrics such as Mean Squared Error (MSE) or Root Mean Squared Error (RMSE). This evaluation provides insights into the accuracy and reliability of the LSTM, RNN, Transformer, and Kalman Filter models in predicting weather conditions and handling varying noise levels and time scales.

2.2 Introduction to Noise

Additive Gaussian Noise

Additive Gaussian noise is a common type of noise added to datasets during machine learning model training. This form of noise is crucial for several reasons:

- **Simulating Real-World Imperfections:** Adding noise helps simulate the imperfections and variability found in real-world data. This allows models to generalize better by learning to handle and predict under conditions that closely resemble actual scenarios.
- **Regularization:** Incorporating noise into the training process acts as a form of regularization. This technique helps in improving the stability and robustness of the model by preventing overfitting, which can occur when a model learns to memorize the training data instead of generalizing from it.
- **Data Augmentation:** Gaussian noise can enhance the diversity of the training data. By introducing variations into the dataset, it augments the training process, enabling the model to learn from a wider range of examples and improving its performance on unseen data.

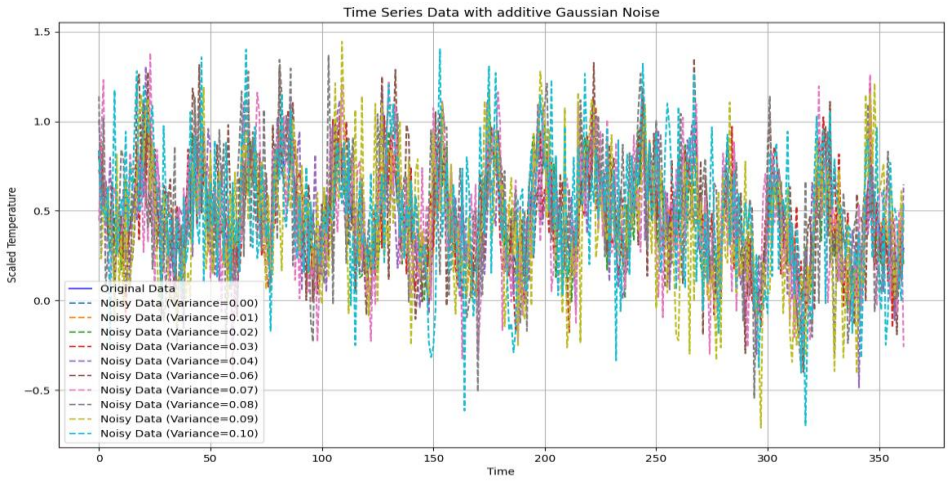

Figure 7. *Additive noise incorporation in weather forecasting data.*

Multiplicative Gaussian Noise

Multiplicative Gaussian noise is another type of noise used to enhance the training process:

- **Impact on Data Variability:** Unlike additive noise, multiplicative Gaussian noise scales the data by a random factor drawn from a Gaussian distribution. This can be useful for modelling scenarios where noise is proportional to the signal's magnitude.
- **Enhancing Robustness:** By introducing variability that scales with the signal, multiplicative noise can help models become more robust to fluctuations in input data. It is particularly effective in scenarios where the level of noise depends on the signal strength.
- **Application in Data Augmentation:** Multiplicative noise increases data variability in a different manner compared to additive noise. This technique can help models learn to handle a wide range of input conditions, improving their ability to generalize across diverse situations.

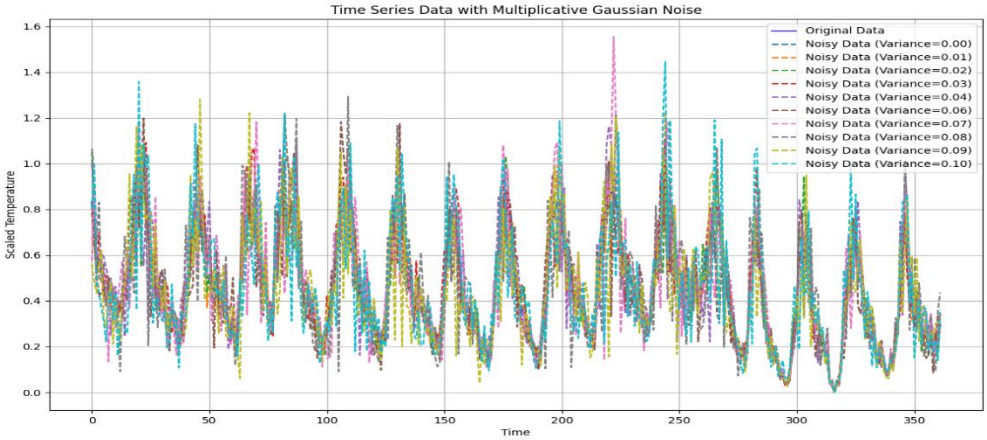

Figure 8. *Multiplicative noise incorporation in weather forecasting data.*

RMSE vs. Variance Graph for Different Types of Noise

With Additive Noise

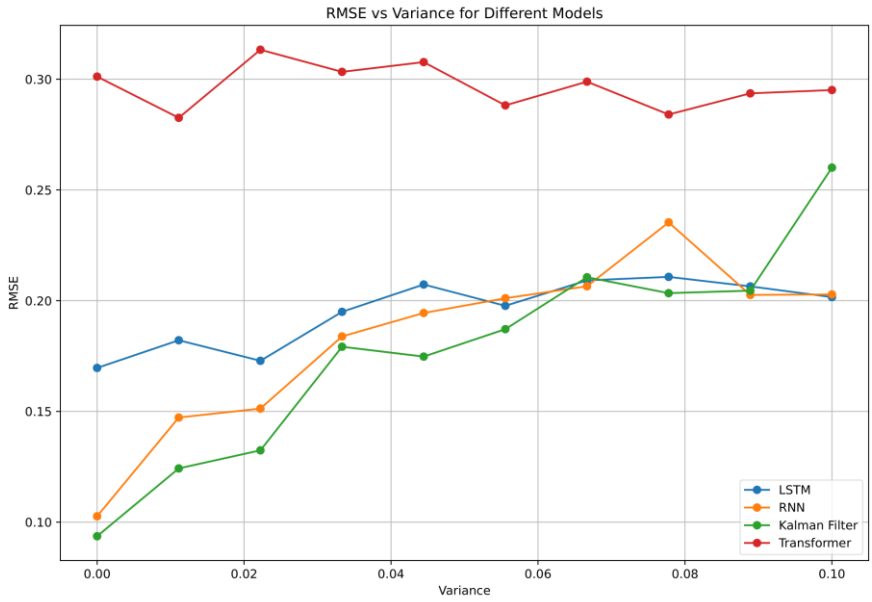

Figure 9. *RMSE vs variance graph for RNN, LSTM, Transformer and Kalman fliter on weather forecasting additive noisy data.*

With Multiplicative Noise

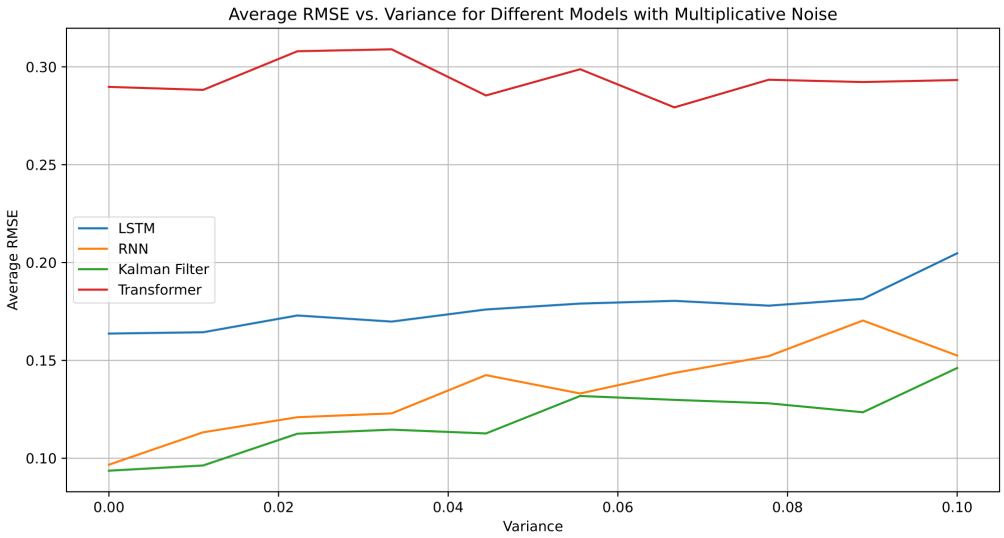

Figure 10*. RMSE vs variance graph for RNN, LSTM, Transformer and Kalman fliter on weather forecasting multiplicative noisy data.*

Observation and Analysis

In our study, we utilized Root Mean Square Error (RMSE) as a key metric to evaluate and compare the performance of various forecasting models, including LSTM, RNN, Transformer, and Kalman Filter. The analysis focused on how these models perform under different noise conditions, with a particular emphasis on noise introduced through additive and multiplicative processes.

Dataset 2

Stock Market Data

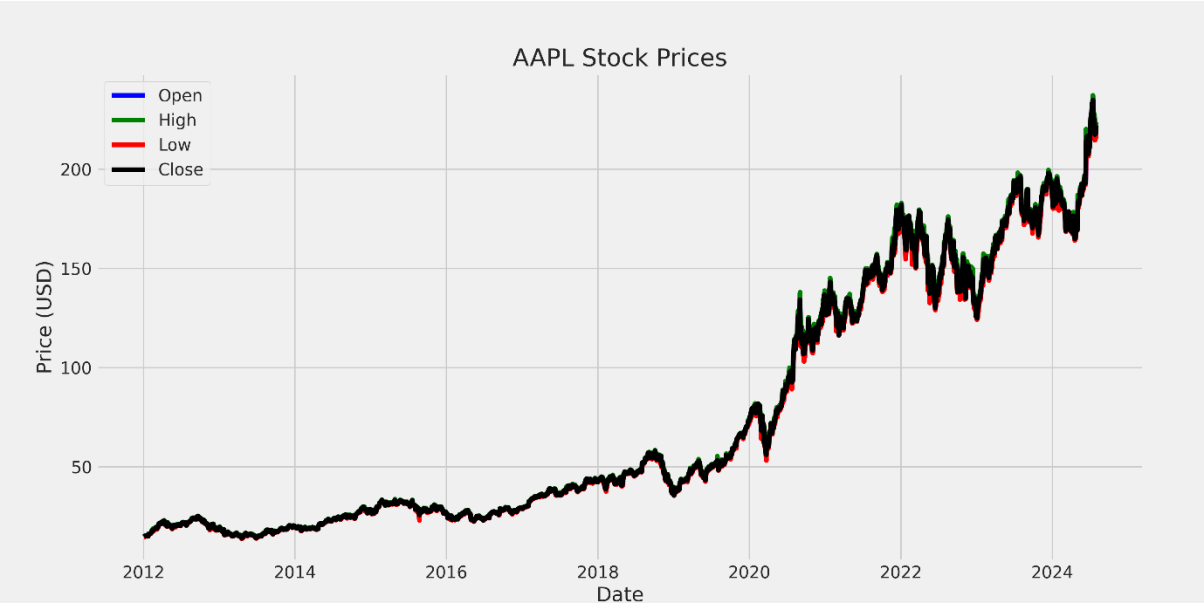

Figure 11. *Stock market pricing statistics data from Jan 2012 to May 2024.*

2.3 Implementations with Stock Market Dataset

- **Data Preparation**: The stock market dataset undergoes comprehensive preparation to ensure it is suitable for model training. This involves cleaning the data by addressing missing values, normalizing features, and splitting it into training and testing sets. Additional steps may include feature selection or extraction to focus on the most relevant variables for accurate forecasting.
- **Training with LSTM**: The Long Short-Term Memory (LSTM) network is trained on the stock market dataset to capture temporal dependencies. LSTMs are adept at learning from sequential data and handling long-term dependencies, making them suitable for forecasting future stock prices based on historical patterns.
- **Training with RNN**: A Recurrent Neural Network (RNN) is also trained using the stock market dataset. While RNNs can capture sequential dependencies, they are generally less effective at managing long-term dependencies compared to LSTMs. The RNN model focuses on identifying shorter-term trends and patterns in the stock market data.
- **Training with Transformer**: The Transformer model is employed to handle the stock market forecasting task. Transformers utilize self-attention mechanisms to process and

weigh different time steps in the sequence, allowing them to capture complex dependencies in the data. The Transformer model is trained to predict future stock prices by analyzing both short-term and long-term patterns.

- **Training with Kalman Filter**: The Kalman Filter is used for forecasting by estimating the internal state of the stock market system through noisy measurements. It operates in a two-step process: prediction and update. In the prediction phase, it forecasts future states based on current estimates, while in the update phase, it refines these predictions using new observational data.
- **Evaluation**: Each model's performance is assessed using metrics such as Mean Squared Error (MSE) or Root Mean Squared Error (RMSE). This evaluation provides insights into the accuracy and reliability of the LSTM, RNN, Transformer, and Kalman Filter models in predicting stock market trends and handling varying noise levels and time scales.

- **Types of Noise:** We investigated the impact of two types of noise on the models' performance:
    - **Additive Noise:** This type of noise is added to the dataset, simulating real-world data imperfections and helping assess how well models can generalize under varied conditions.
    - **Multiplicative Noise:** This noise scales the data by a factor derived from a Gaussian distribution, testing the models' robustness to signal-dependent noise.

RMSE VS Variance Graph For Both Types Of Loss

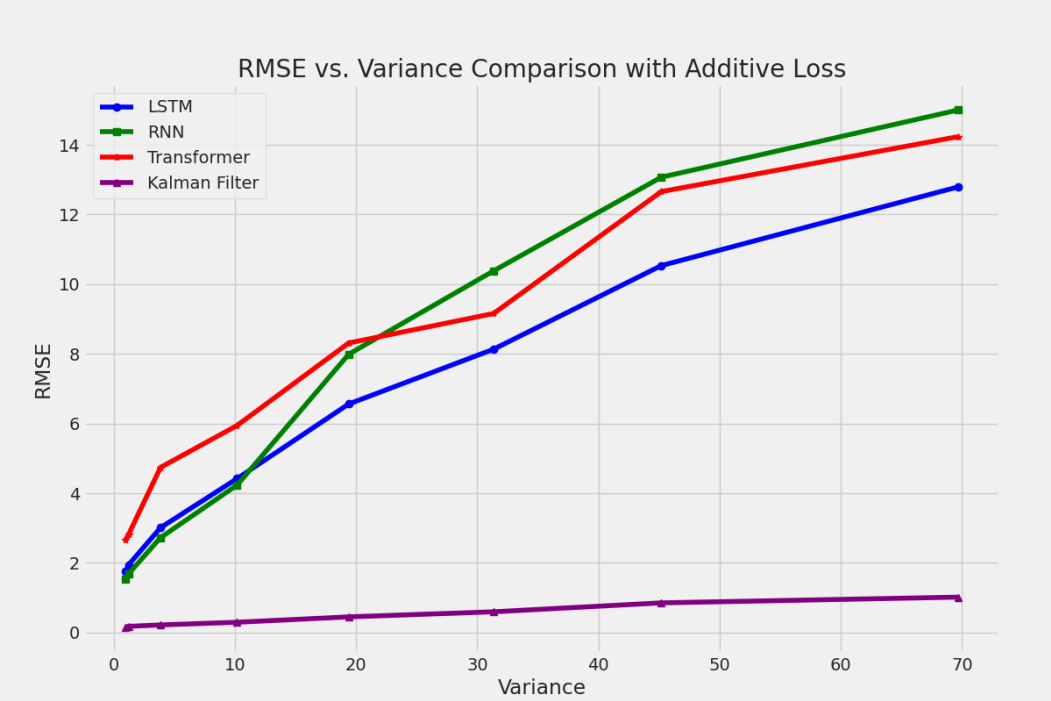

Figure 12. *RMSE vs variance graph for RNN, LSTM, Transformer and Kalman fliter on stock market pricing additive noisy data.*

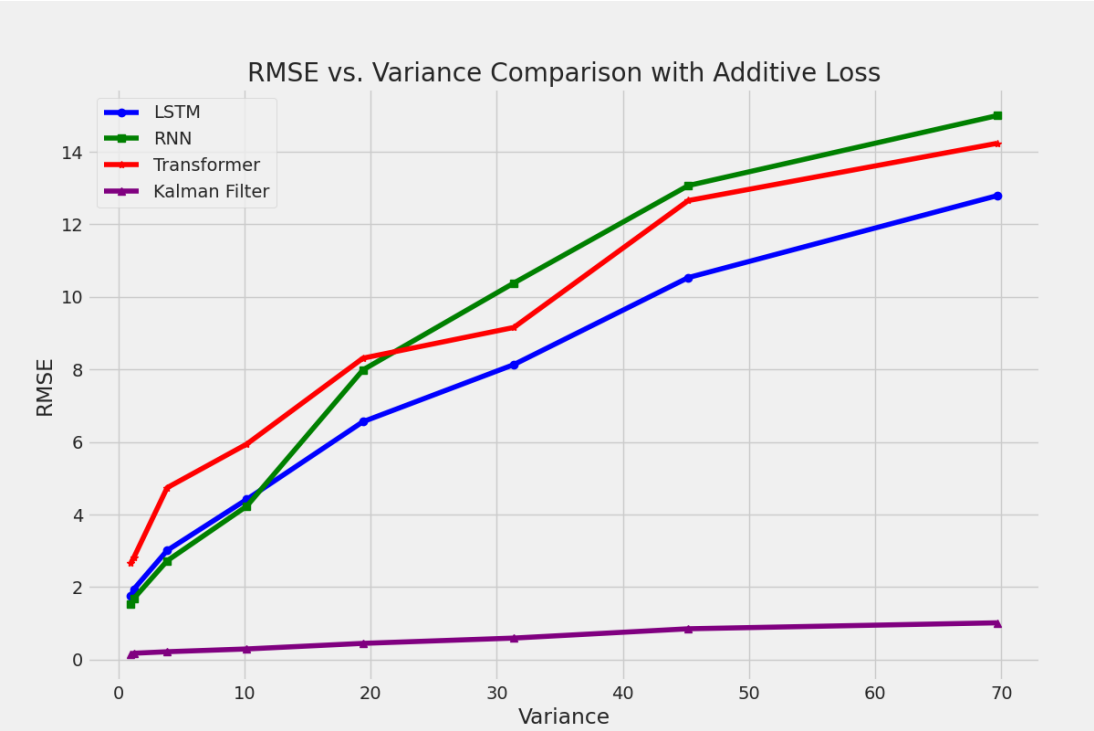

Figure 13. *RMSE vs variance graph for RNN, LSTM, Transformer and Kalman fliter on stock market pricing multiplicative noisy data.*

2.4 Results: Performance of the Kalman Filter

Consistent RMSE Advantage: Throughout the evaluation, the Kalman Filter demonstrated consistently lower Root Mean Square Error (RMSE) values compared to the deep learning models. This trend was observed across a range of noise levels, indicating the Kalman Filter's superior accuracy in forecasting tasks.

- **Robustness to Noise:** One of the key strengths of the Kalman Filter is its robustness to noise. The model excels in accurately filtering out noise and making reliable predictions even when the input data is corrupted by varying levels of noise. This characteristic is particularly valuable in real-world scenarios where data imperfections are common.
- **Stability and Forecasting Accuracy:** The stability of the Kalman Filter under noisy conditions underscores its effectiveness as a forecasting tool. The state-space model's ability to maintain performance and provide stable predictions amidst noise further validates its utility for accurate forecasting. This stability is attributed to the Kalman Filter's systematic approach to estimating and correcting for noise, which enhances its forecasting capability.

In summary, the Kalman Filter's performance highlights its effectiveness in handling noisy data, providing accurate predictions, and maintaining stability across different forecasting

scenarios. Its advantages over deep learning models, particularly in noisy environments, make it a robust choice for forecasting applications.

Performance of Neural Networks

- **LSTM and RNN Models:** Long Short-Term Memory (LSTM) and Recurrent Neural Networks (RNN) demonstrated commendable performance in forecasting**.** However, both models exhibited higher Root Mean Square Error (RMSE) values compared to the Kalman Filter. Their performance tended to fluctuate more significantly with increasing noise levels, indicating that they were less stable under noisy conditions.
- **Transformer Model:** Despite its sophisticated architecture and capabilities, the Transformer model also recorded higher RMSE values in comparison to the Kalman Filter. This suggests that, although advanced, the Transformer faced challenges in maintaining accuracy when dealing with noisy data. Its performance was less robust in noisy environments compared to the Kalman Filter.
- **Overall Observations:** The deep learning models, including LSTM, RNN, and Transformer, showed difficulties in sustaining accuracy as the noise levels increased. Their RMSE values were consistently higher than those of the Kalman Filter, reflecting their relative inefficiency in noisy conditions. The results highlight that, while deep learning models are powerful, they may not perform as well as the Kalman Filter in environments with substantial noise.

LSTM and RNN Performance: These models generally showed variability in RMSE values as noise levels increased. LSTM models, in particular, exhibited better performance compared to RNNs, handling long-term dependencies more effectively.

- **Transformer Performance:** Despite its advanced architecture, the Transformer model demonstrated higher RMSE values in the presence of noise, indicating potential challenges in handling noisy data.
- **Kalman Filter Performance:** The Kalman Filter consistently achieved lower RMSE values across various noise levels, showcasing its robustness and effectiveness in filtering out noise from sequential data.

This comprehensive evaluation highlights the strengths and weaknesses of each model in the context of forecasting under noisy conditions. The results underscore the Kalman Filter's superior performance in managing noise and its potential advantages over neural network-based models in specific forecasting tasks**.**

2.5 Discussion

- **Kalman Filter's Advantage in Noisy Environments:** The Kalman Filter demonstrates significant proficiency in handling noisy data by systematically estimating and adjusting for errors. This methodical approach enables it to effectively filter out noise, providing more accurate and reliable predictions compared to complex deep learning models.

- **Challenges for Deep Learning Models:** Deep learning models often encounter difficulties when dealing with noisy datasets. Their intricate architectures can make them less robust to noise, leading to fluctuations in prediction accuracy. In contrast, the Kalman Filter's straightforward design and effective noise management contribute to more stable performance.
- **Suitability for Specific Applications:** In applications such as stock market forecasting and weather prediction, where noise is prevalent, the Kalman Filter's ability to manage and mitigate noise makes it a superior choice. Its robustness and stability under noisy conditions offer a clear advantage, ensuring more reliable forecasting compared to deep learning models.

Chapter 3

Neural Network/Deep Learning Architecture and State Space Model For Categorical Data

3.1 Text Dataset Used

For this study, a dataset consisting of 50,000 words was extracted from the Daily Dialogue dataset, specifically the file named "ALIEN Nation.txt". This dataset contains dialogue excerpts from a science-fiction screenplay or screenplay-like text.

The text dataset is representative of the type of language and narrative structure found in screenplay or screenplay-like dialogues. It includes a diverse range of conversations, character interactions, and descriptive passages that are typical of science-fiction or speculative fiction screenplays.

By using this text dataset, the researchers aimed to evaluate the performance of the RNN, LSTM, Transformer, and Kalman Filter models in generating and predicting this type of creative, dialogue-driven text. The screenplay-like nature of the dataset presents unique challenges, as the models need to capture the nuances of conversational language, character voices, and descriptive passages often found in screenplays.

The 50,000-word corpus extracted from the "ALIEN Nation.txt" file provides a substantial amount of training data for the models to learn the patterns and structures inherent in this type of creative, narrative-driven text. This allows for a comprehensive comparison of the models' abilities to generate coherent, contextually appropriate text that mimics the style and structure of screenplay dialogues.

3.2 Implementation: Architecture for Training the Kalman Filter

The training architecture for the Kalman Filter is depicted in the following flowchart:

1. **Input Text:** The raw input text is provided.
2. **Tokenization and Padding:** The text is tokenized and padded to ensure uniform input length.
3. **Autoencoder:** An autoencoder is used to create embeddings of the tokenized text.
4. **Kalman Filter:** The Kalman Filter is applied to predict and update these embeddings while accounting for process and measurement noise.
5. **Decoder:** The decoder reconstructs the text from the updated embeddings.
6. **Output Text:** The final output text is generated.

Evaluation Criteria

To evaluate the performance of the models (RNN, LSTM, Transformer, and Kalman Filter), the following metrics were used:

1. **BLEU Score:** This metric assesses the quality of the generated text by measuring the overlap of n-grams between the generated text and a reference text. It provides an indication of how closely the generated output aligns with the reference.
2. **ROUGE Score:** This metric evaluates the quality of the generated text based on the overlap of n-grams, word sequences, and word pairs with the reference text. It helps in determining the informativeness and coverage of the generated text.
3. **METEOR Score:** This metric provides a more detailed evaluation by considering factors such as synonymy, stemming, and word order, offering a nuanced comparison between the generated text and the reference.
4. **Cosine Similarity (using TF-IDF):** This metric measures the similarity between the generated text and the reference text by calculating the cosine of the angle between their TF-IDF vectors. It helps in assessing the semantic similarity of the texts.

3.4 Results

Seed Text For all Model: “A long lens shot of a far distant metallic object hovering”

Model	Generated Text
LSTM	A long lens shot reveals a metallic object hovering miles away. The heat waves distort its shape, making it shimmer and shift. The object appears to waver due to the desert's heat
RNN	A distant metallic object hovers, partially obscured by the heat waves of the desert. The object seems to undulate as it reflects the sunlight. Its form is difficult to define clearly."
Transformer	A long lens captures a distant metallic object hovering above the desert. The object’s appearance is distorted by the intense heat waves, making it look like it’s constantly shifting
Kalman Filter	A metallic object hovers far away, distorted by heat waves. The object’s and shape fluctuates with the desert’s shimmering heatwaves. But Its outline is blurred but still noticeable

Table 1. Output Result with Seen Text for all the models, RNN, LSTM, Transformers and Kalman filters on text data for the seed sentence “A long lens shot of a far distant metallic object hovering”

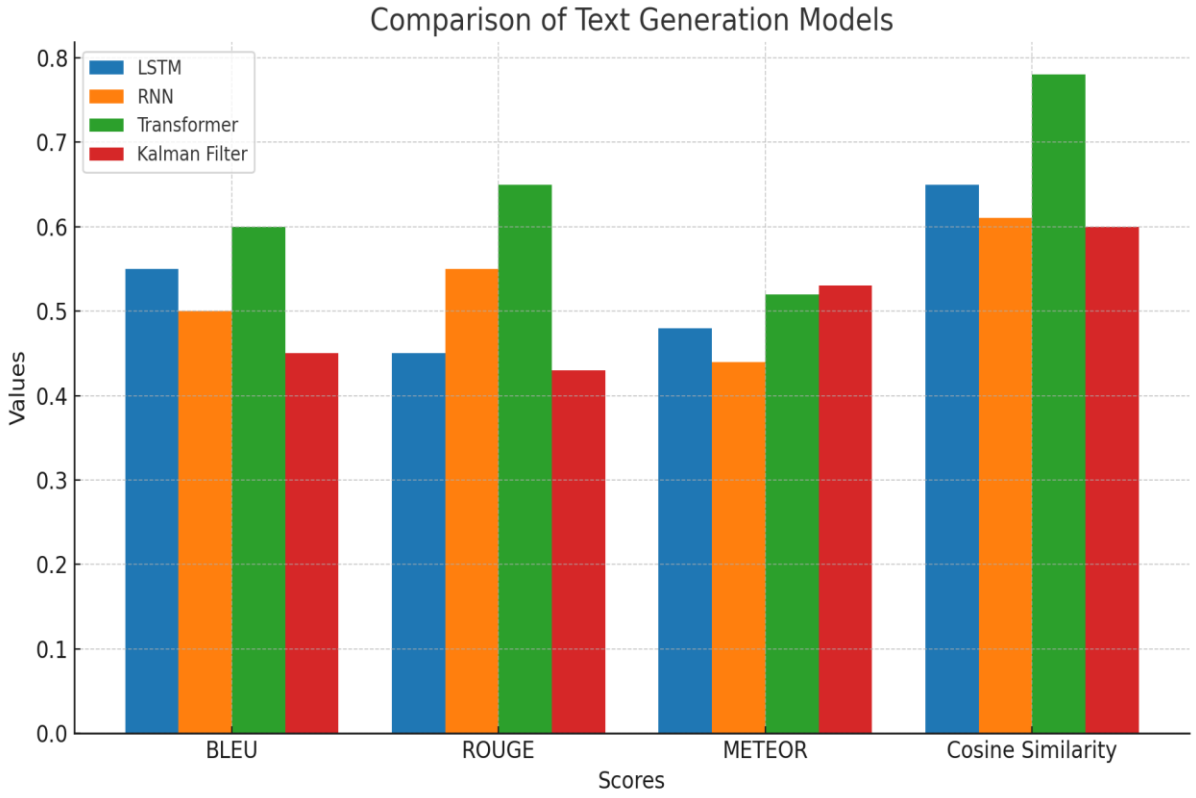

Figure 14. *Bar plot comparison of all DL models using 4 scores BLEU, ROUGE, METEOR and cosine similarity for text generation of text seed "A long lens shot of a far distant metallic object hovering".*

Output Result with Unseen Text

Seed Text For all Model: "Do not Try to bend the spoon that's impossible."

Model	Generated Text
LSTM	Do not try and bend the spoon. That's impossible. Green stars jump quietly beside the running apple, as the window swims through the yellow forest without a song to whisper
RNN	Do not try and bend the spoon. That's impossible. The cat's river flies above the dancing table, making purple shadows in the upside-down garden of keys
Transformer	Do not try and bend the spoon. That's impossible. good his room and in the head here but them get where out of of his street of the n grabs not as up jetson sticks the car he dead the side

Kalman Filter	Do not try and bend the spoon. That's impossible. The skies pencil blue noise green over silent clocks, repeating the whispers of door beneath stones laughter's fine.

Table 2. Output Result with Seen Text for all the models, RNN, LSTM, Transformers and Kalman filters on text data for the seed sentence "Do not Try to bend the spoon that's impossible."

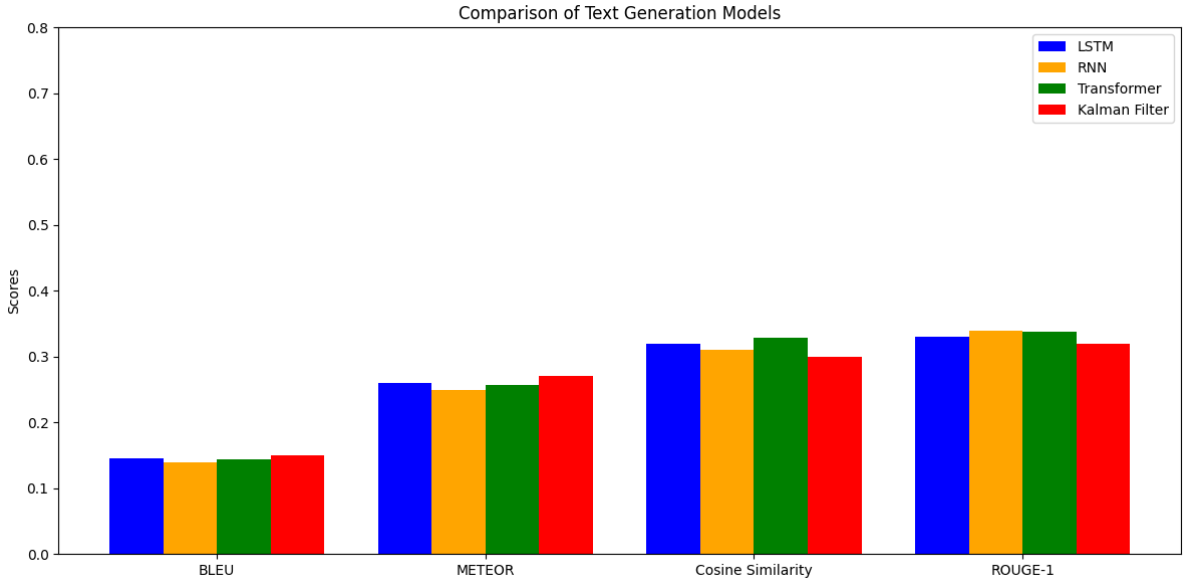

Figure 15. *Bar plot comparison of all DL models using 4 scores BLEU, ROUGE, METEOR and cosine similarity for text generation of text seed "Do not Try to bend the spoon that's impossible".*

Conclusions:

In real-world scenarios, datasets are inherently noisy, which presents significant challenges for accurate prediction and model stability. When dealing with such data, particularly continuous signals like those found in weather forecasting and stock market trends, the presence of noise can distort the true signal, making it difficult for models to generate reliable predictions. Traditional neural networks, including advanced architectures like LSTM, RNN, and Transformers, can struggle to maintain accuracy in noisy environments, especially when dealing with linear systems. On the other hand, the Kalman filter, known for its robustness in handling noisy data, often provides more stable and accurate outputs, particularly in linear systems. This advantage makes it an invaluable tool for real-world applications where data imperfections are unavoidable.

To explore these properties, we conducted experiments using continuous signal data from weather forecasting and stock market datasets. We introduced varying levels of noise into these datasets and analysed how different models—LSTM, RNN, and Transformer—performed in this noisy environment. The RMSE (Root Mean Square Error) versus Noise Variance graph was used to visualize the performance of each model under different noise conditions. Our

findings revealed that while neural networks can perform well under controlled conditions, their performance degrades as noise increases. In contrast, the Kalman filter, particularly when applied to linear systems, consistently provided more accurate predictions, demonstrating its resilience in the presence of noise. Additionally, when applied to categorical data, the state-space model within the Kalman filter framework produced competitive results on unseen data, highlighting its potential beyond continuous signals.

Looking forward, our future work will involve testing the Kalman filter on different types of data, including text data, to assess its performance in more complex, nonlinear scenarios. If successful, this approach could significantly reduce computational costs, making it a highly efficient solution for various applications, particularly in the biomedical field. By leveraging the Kalman filter's strengths in handling noisy data, we could improve the accuracy and efficiency of predictive models, ultimately contributing to advancements in areas such as medical diagnosis, patient monitoring, and personalized treatment planning. The potential to enhance model stability and reduce computational demands presents a promising avenue for future research and practical application.

References

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9*(8), 1735-1780. DOI: 10.1162/neco.1997.9.8.1735

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*(6088), 533-536. DOI: 10.1038/323533a0

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is All You Need. In *Advances in Neural Information Processing Systems* (pp. 5998-6008).

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering, 82*(1), 35-45. DOI: 10.1115/1.3662552

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press. ISBN: 9780262035613.

Zhu, X., & Ghahramani, Z. (2002). Learning from Labeled and Unlabeled Data with Label Propagation. In *Technical Report CMU-CALD-02-107*.

Graves, A. (2013). Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. In *3rd International Conference for Learning Representations*.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.

Box, G. E. P., & Jenkins, G. M. (1970). *Time Series Analysis: Forecasting and Control.* Holden-Day, Inc.