Interpreting Structured State Space Model

Yang Su, Yin Li, Lancaster Wu

1. Abstract

The structured state space model(SSM) is a novel model that is good at modeling long-range continuous signals. This was a hard or even intractable task and taking in long input makes many SOTA models not working. Therefore, we are interested in interpreting how the SSM-based model is good at long-range tasks.

Since we are a group of three, we worked on two SSM-based models on multiple tasks. First, in the recent series of work per SSM, the S4 model is an improved version for efficiency. Secondly, the SPACETIME model is a variant for predicting the time series autoregressively. We train them on our selected datasets, including time series, text, and images. And we interpret the trained model using SHapley Additive exPlanations (SHAP) and surrogate model. Finally, we discuss the trustworthiness of the S4 model by conducting a qualitative user study.

2. Introduction

(Cite (YYY, 2023))

2.1. Understanding S4 model

Structured State Space Model(S4) (Gu et al., 2021a;b; 2020) is one of the most influential state-of-the-art papers in modeling long-range dependencies (LRD). Compared with the dominant transformer-based family of models, it has impressive performance in both scalability and efficiency. Moreover, the hidden states in the model are a kind of latent variable, which might lead to some interpretability.

Modeling long-range sequences is a challenging task and one of the bottlenecks of the performance of transformers in many tasks including natural language processing (NLP), computer vision (CV), and audio. In other words, for example, ChatGPT (OpenAI, 2022) is huge recently but it is not able to read input sequences longer than a length of about 3000 words. It is potentially because of the gradients scaling exponentially in the sequence length. At a high level, all these types of data are continuous signals as a function of time. Thus, many existing works use machine learning to approximate the distribution with a set of pre-defined simple functions. More specifically, S4 uses the family of Legendre polynomial functions as the basis, with customized accelerations introduced by the following series of works.

In brief, they first develop the Hippo theory framework (Gu et al., 2020). It projects the problem of learning coefficiencies for the Legendre polynomial basis into a format of state space ODE. As shown in equation 2, A, B, C are the model parameters; u is input, y is output, x is state. They discretize and scale the fundamental state space matrices(SSM) to long dependencies, so this equation is now a sequence-to-sequence map. But it is a kind

† Team name: YYY.
Email: {ys724, yl3243, jw2555}@cornell.edu.

of recurrent memory, which makes the computation expensive.

$$x_k = \bar{A}x_{k-1} + \bar{B}u_k \tag{1}$$

$$y_k = \bar{C}x_k \tag{2}$$

Secondly, in the follow-up paper (Gu et al., 2021b), they propose a new representation to condition the SSM and dramatically improve the computation and memory efficiency. To elaborate, with a Linear State Space Layer as $y_k = \bar{C}\bar{A}^k\bar{B}u_0 + \bar{C}\bar{A}^{k-1}\bar{B}u_1 + ...\bar{C}\bar{A}\bar{B}u_{k-1} + \bar{C}\bar{B}u_k$ and $y = \bar{K} * u$. In other words, S4 can operate as a convolutional neural network (CNN) for training, while then convert to an efficient recurrent neural network (RNN) at the testing time. Therefore, it has the ability of both modeling the continuous data and easy optimization plus parallelizable training.

Finally, in the S4 paper (Gu et al., 2021a), they further improve the computational efficiency by conditioning the A^k matrix by conditioning on a low-rank correction, allowing it to be diagonalized stably and reducing the SSM to the well-studied computation of a Cauchy kernel. The extensive benchmarking results show that S4 substantially closes the gap to transformers on image and language modeling tasks while performing generation 60× faster. It is the SoTA on every task from the Long Range Arena benchmark, including solving the challenging Path-X task of length 16k that all prior work fails on.

2.2. SPACETIME model: the variant for time series

By further extending the literature survey of our proposal, we found the latest variant of the deep state-space model, SPACETIME (Zhang et al., 2023). It proves that the previous SSM models (Gu et al., 2021a; Gupta et al., 2022; Smith et al., 2022; Alcaraz and Strodthoff, 2022) can not express autoregressive(AR) processes. Their adapted SSM parameterization based on a new representation of latent matrix fix this issue and outperform the previous work on discrete time series forecasting task by a large margin. In general, it has three advantages including the expressivity of traditional time series processes, a closed-loop single SSM layer that avoids sequential input and output over the entire network, and subquadratic time and space complexity. In brief, it models the discrete time series with multiple SSM layers where each dimension is $y_{k+1} = u_{k+1} = C(Ax_k + Bu_k)$. A is a relatively simpler companion matrix compared with previous specific classes of matrices in (Gu et al., 2021b;a).

3. SHAP for SPACETIME

3.1. A demo dataset for time series prediction using SPACETIME

We replace the dataset in the original paper with a new demo dataset. The original SPACETIME paper uses a dataset about electricity transformers. As shown in our midterm report, the data is the temperature of the electricity transformer and they intend to use it for electricity production planning. Instead, for the final presentation, we replace the dataset with another one, a Yahoo finance dataset. The reason why is that we think this one is more comprehensive, close to daily life common knowledge, and would be a better example used for interpreting machine learning models.

In detail, it has the daily stock price of GSPC/S&P 500 from Jan 4th, 1993 to May 5th, 2023. Excluding the weekends, it is 30 years for 7k data points in total. The columns include Date, Open, High, Low, Close, Volume,

Dividends, and Stock Splits. We chose the close price as the target and the training data.

Preprocessing Sliding window slices the raw data into equal-length segments. Each segment is 104 long, where the first 84 days, i.e. the prior 12 calendar weeks, as inputs. And then we try to predict out the next 20 days, i.e. around 4 working weeks. The step of the sliding window is one, i.e. overlapping is 103. From the perspective of time series prediction, the horizon and lag are 84 and 20. We will train the model that can forecast a horizon of future terms given the past historical lag terms.

3.2. SPACETIME model training

We train the model on a GPU server with one GTX 3090 for 5000 epochs. The dataset is split into training, validation, and testing sets. The first 28ish years are used for training and 10 percent of them are used as validation set. The last 1.5 years are used for testing. Not that we do not predict the entire 1.5 years, but slice them into windows and then split each window into horizon and lag.

Evaluation: As shown in Fig 1, the model performs well on training, validation, and testing set. The mean absolute errors (MAE) are 28, 89, and 125, which is well for stock price prediction at around a few thousand.

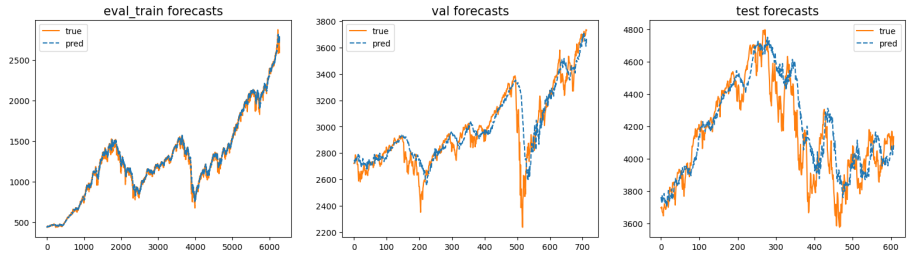


Figure 1: Evaluation of the training: a visualization of the true value and the predicted value on training, validation, and testing set.

3.3. The linear surrogate model

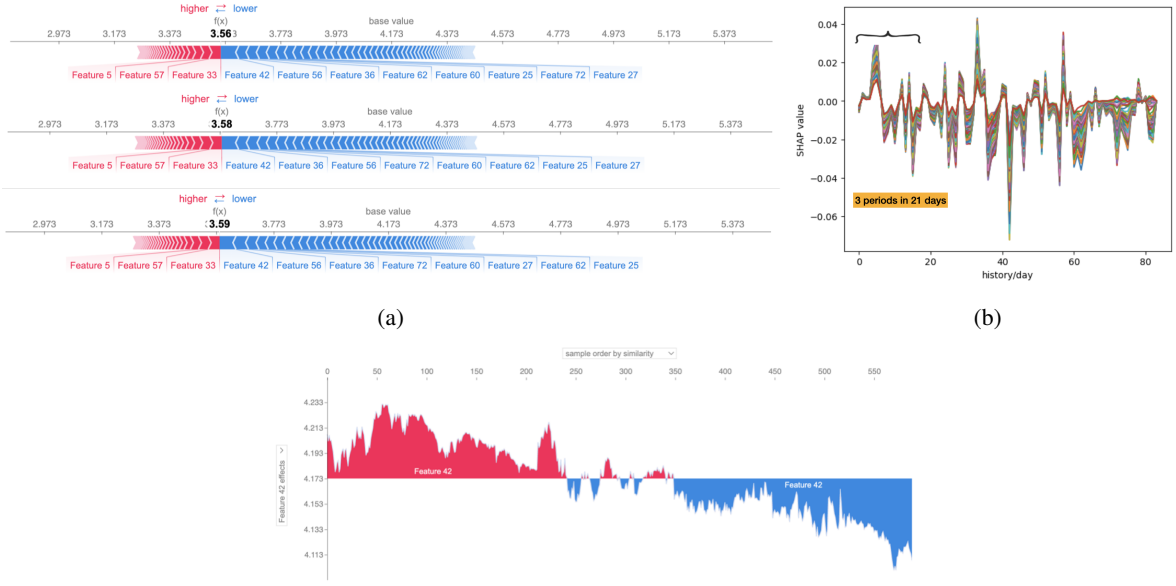
To use the implementation of SHAP (Lundberg and Lee, 2017), we surrogate the SPACETIME model to make it compatible to input to SHAP. So, we overfit the test set on a linear model with three layers using the original input x and the output y_pred by our trained SPACETIME model. The output dimension of the three linear layers is set to 64, 32, and 20 consecutively. And we use Adam as the optimizer and MSELoss. After training the surrogate model for 100k epochs to overfit the test set, it yields an MSE of 215, which is roughly 2 times better than the training loss of the original SPACETIME model whose MAE is 28. This ensures the surrogate represents the original model equivalently.

3.4. Analysis of the SHAP values

By inputting the trained surrogate model and the test set into the SHAP, we visualize the results as shown in Fig. 2. Feature X is the name of the Xth day in the horizon of 84. And we select the first day in the 20 lag of days for visualization without considering multi-variate. Fig. 2a are the SHAP value of three data samples, where feature 42 turns out to have a strong impact in all of them. So, we plot the SHAP value of feature 42 on all samples as

in Fig. 2c, which shows no consistency in the positiveness or negativeness. Furthermore, we find the index of strong impact features' difference are mostly multiples of 7, so we guess there might be a certain weekday that is constantly important, for example, Monday when the market opens. But it is inaccurate because the datasets consist of weekdays only. Therefore, we plot the SHAP values of all samples and all features in Fig. 2b and are still able to see that there are roughly weekly periodic peaks. For example, the first 21 days have around 3 peaks.

Discussion: Besides, we would like to discuss that the result could be further improved because of two inefficiencies in the current setup. First, the kernalSHAP uses a linear model to fit the perturbations of the same long-range data, which might be incapable and not autoregressive. Secondly, the consecutive dependence of time series is hard to capture using the same method as other tabular features.



(a) (b)

(c)

Figure 2: SHAP results of SPACETIME model.

4. SHAP for S4

4.1. Text Generation Explainability with Permutation SHAP

We first try to reproduce the result of S4 trained on the WikiText-103 dataset, one of the original datasets the author used. This data consists of over 100 million tokens extracted from a subset of Wikipedia articles. It includes a wide variety of text genres, ranging from scientific articles to movie reviews, and contains many rare words and long-range dependencies, making it a challenging dataset for language modeling. We then compare the performance of S4 and GPT-2 small and medium by letting the model generate with a threshold on their generation diversity (confidence) and analyze the influence of input sentences by Shapley values. We specifically choose GPT-2 as the model to compare against because they are of similar parameter sizes as S4 (parameter sizes - S4: 259 Million, GPT2-Small: 117 Million, GPT2-Mid: 345 Million), and they are all trained on WikiText-103 (GPTs are trained on a super-set of it).

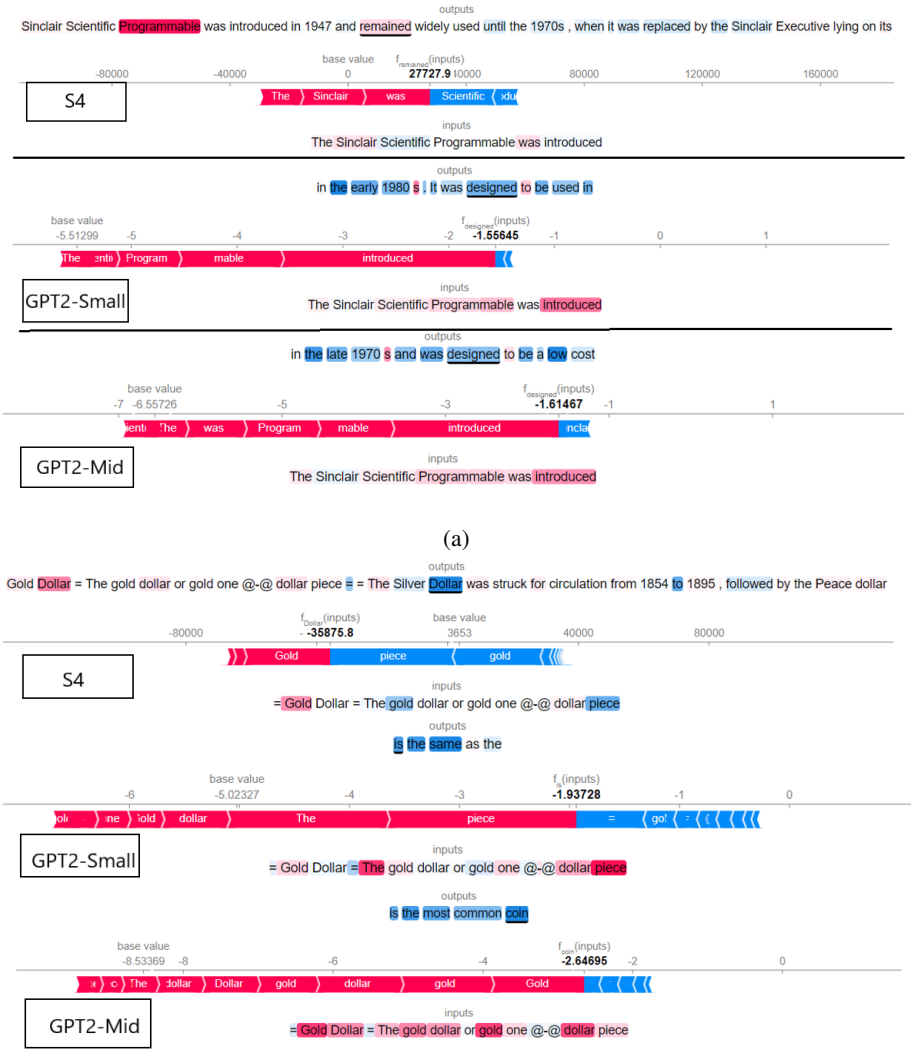


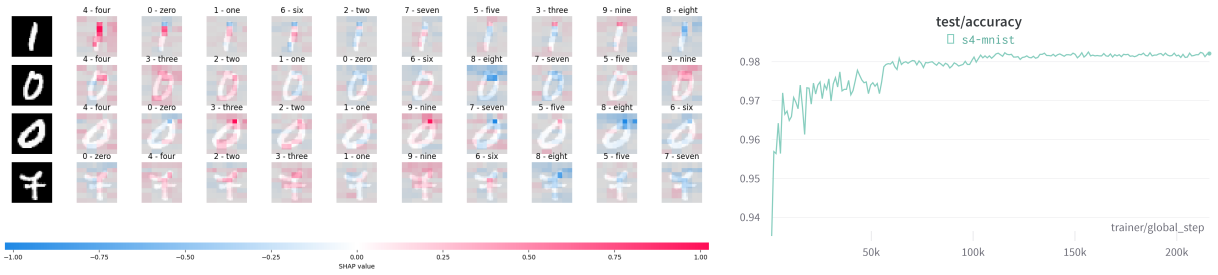
Figure 3: SHAP results of S4 and GPT2 model on WikiText-103 with 1D masking.

Random Sampling We first randomly sample a prefix sentence of an article from WikiText-103 and let the model generate autoregressively. We set use top-k generation to $k = 50$, so the model always keeps the top-50 words as candidate input for the next token generation; we then set all model's temperatures to 0.9 to encourage non-determinism; finally, we set the least tolerable repeating n-gram size to 2, so the model would stop generating if some of their prediction candidates are the same as other 2-gram tokens that have appeared before. By doing this, we aim to compare whether the S4 model produces more diverse (confident) and longer outputs than the transformer-based model. As we can see in Figure 3 (a), S4 produces much longer output with more rare words: "Sinclair" from the input prefix sentence, and "Executive" from its own generation; whereas GPT2s stop generating (i.e., they want to generate repeating words and gives higher perplexity) sooner. From the explanation of permutation SHAP, we can also see that the token "remained" is influenced most by "Sinclair Scientific", which appears at the beginning of the input sentence; contrastively, the similar verb "designed" generated from

GPT2s gained more influence from closer tokens (”introduced”), and the influence vanished as the distance from the generated word and the input word gets longer. This shows the ability of S4 to generate longer and rare contexts confidently and has longer-range dependencies.

Adversarial Sampling It is possible that S4 was trained on WikiText-103 only, hence causing its superiority in long-context generation. Hence we would like to sample specifically from the training examples that S4 did poorly on. We rank 50 example input-output pairs (we essentially do it by taking in a batch of size 50) by their perplexity (confidence or likelihood in generation) and pick the one sample that S4 is mostly unsure about. As in Figure 3 (b), the input is an article about the ”Gold Dollar” and does not have a humanly readable sentence structure, and GPT2s can only generate a few tokens mostly based on the closest input ”dollar piece”. We can see that even under this condition, S4 can still generate a long piece of text more confidently than GPT2s.

From the experiments above, we find that S4 is good at capturing long-range dependencies from its state space structure, which is likely the reason that it can excel at the Path-X dataset that previous transformer-based models failed on. Note that the Path-X dataset consists of 2D images with trajectories drawn on them, and the models' goal is to find the path that connects the beginning and end points of the trajectories. We then would like to conduct an experiment to show that S4 does not find these trajectories through the 2D structure of the images; instead, they really capture the pixel relations through extremely long-range dependencies.



(a) (b)

Figure 4: SHAP results of S4 on MNIST with 2D masking.

4.2. Image Classification Explainability with Partition SHAP

We first train S4 on the MNIST dataset and easily achieve a test accuracy of 98% (Figure 4 (b)), then switch the masking strategy from the 1D permutation SHAP to 2D partition SHAP. By doing this, we hope to see almost none of the results are explainable, as the 2D structure of the images is flattened as the input to S4; hence, doing 2D masking would completely break its structure. Note that 1D permutation SHAP masks out input tokens randomly while always keeping some input unmasked, and calculate the average influence of those unmasked words; whereas 2D partition SHAP splits the image into some 2D subsets of filters (or kernels in convolution-based architectures), and calculates the average influence of each kernel. If S4 is capturing 1D structures, then masking out some continuous ”lines” of input would devastatingly break the prediction ability of the model. As we can see in Figure 4 (a), we randomly choose 4 input images (by row) and plot the shapely values on each prediction class (by column), then sorted the influence by the average of each pixel decreasingly from left to right, and the pixels that contribute the most to the outputs do not produce meaningful explanations

(the written digits structure, style, etc.), which supports our assumption.

5. Trustworthiness

5.1. Computational Evaluation: Perturbation

To ensure the S4 model is trustworthy, we used two different approaches from both computational perspective and human perspective. The first approach applied a perturbation-based evaluation with metrics for correctness. The similar strategy has been used in previous research (Yeh et al., 2019), but with AUC for ROC curve.

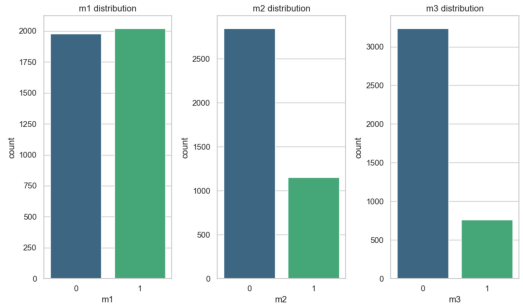


Figure 5: Perturbation-based evaluation: ReMF.

The figure shows the performance comparison of three models: m1, m2, and m3, where 1 denotes correct and 0 incorrect predictions. m1, using all features, has the highest accuracy, demonstrating the importance of leveraging all available features. m2, lacking the most relevant feature, sees reduced accuracy, highlighting the significant role of key features. m3, missing the top two features, performs the worst, reinforcing the necessity of relevant features.

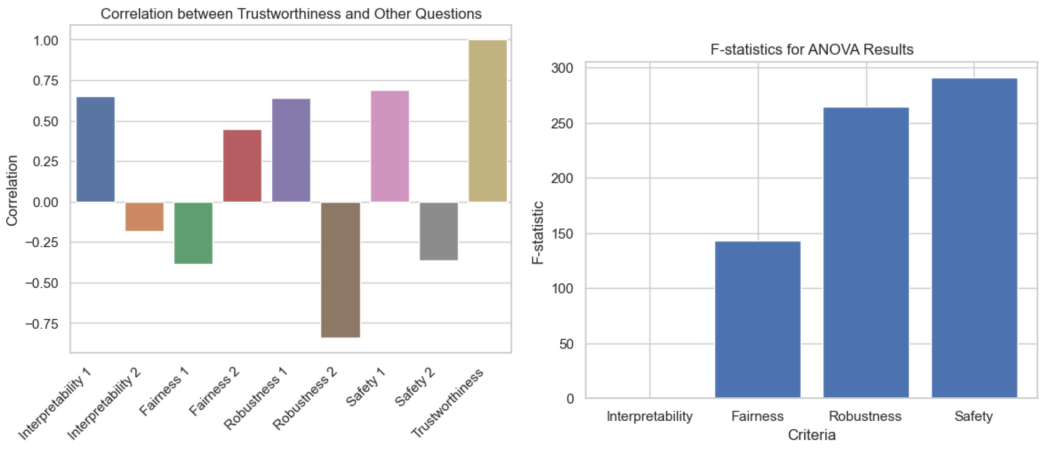
In essence, the figure provides compelling evidence of the trustworthiness of our model - the use of relevant features directly contributes to improved prediction accuracy. Consequently, the model's performance aligns with our expectations, confirming that the model is effectively doing what it's designed to do.

5.2. Human Evaluation

To validate the computational trustworthiness, it is also important to analyze human evaluation. We conducted a study with 13 participants, 6 of them are Cornell Tech graduate students, and 7 of them are from industry with ML related background. We want to make sure all participants have domain expertise to gauge their trust in the model.

We demonstrated the S4 model's text generation task in the mixed format of in-person and remote. Then asked participants to answer 9 survey questions through google form. These questions were categorized into 5 sections, Interpretability, Fairness, Robustness, Safety, and Overall trustworthiness. Similar to how trustworthiness is studied in information science(Ma et al., 2017), we can validate the trustworthiness by categorizing it into smaller subsections and comparing the correlations. The strong correlation between these subsections and overall measurement can increase our confidence in the trustworthiness of the model. We applied F-statistics and ANOVA

to further analyze the data we collected based on the survey in Appendix1.



(a) Correlation

(b) F-statistics for ANOVA results

Figure 6: Statistical Analysis of Survey Responses

The results of the regression analysis reveal an R-squared value of 0.862, suggesting that approximately 86.2% of the variation in Trustworthiness can be explained by the independent variables in the model. However, the adjusted R-squared value is relatively lower at 0.587, indicating a possible overfitting of the model, which is expected since we are having 2 similar questions for one subcategory.

The F-statistic of the model is 3.133 with an associated p-value of 0.142. As the p-value is greater than 0.05, we do not reject the null hypothesis at the 5% significance level, suggesting that the explanatory variables do not provide significant explanatory power as a group. Interestingly, none of the predictors are statistically significant at a 5% level, suggesting that the model doesn't provide strong evidence of a linear relationship between these predictors and Trustworthiness. Furthermore, the Durbin-Watson statistic of 2.846 indicates that there is no autocorrelation in the residuals.

The results in Figure 6a shows expected positive and negative correlations between subcategories and the overall trustworthiness besides the Interpretability2. The results from the ANOVA tests provide valuable insights into the differences in means among the pairs of categories for the four criteria: Interpretability, Fairness, Robustness, and Safety. For Interpretability, the p-value is 1.0, which is much larger than the standard significance level of 0.05. This indicates that there is no significant difference between the means of the two Interpretability groups. Conversely, the p-values for Fairness, Robustness, and Safety are all much smaller than 0.05, suggesting that there are significant differences between the means of the respective groups for these criteria. As depicted in Figure 6b, the F-statistics for Fairness, Robustness, and Safety are notably higher compared to Interpretability, further illustrating the significant differences between their group means, showing that Interpretability2 might be an abstract question that created some biases. Although the human study on S4 is limited by the sample size, we still can conclude with some confidence of the model's trustworthiness, which validates the computational evaluation.

6. Code

- SPACETIME Github Repo
- S4 Interpretability Github Repo
- Human Study Data and Analysis Github Repo

7. Appendix

In the following table of survey questions, I1, I2, F2, R1, S1 and O1 are Likert-scale questions from 1 to 5 and F1, R2, S2 are Yes/No questions.

I1.	Were the generated text samples easy to understand and interpret?
I2.	If the model provided explanations for its text generation decisions, how helpful do you think these explanations would be in understanding the generated text samples?
F1.	Did you notice any instances of biased or discriminatory language in the generated text samples?
F2.	In terms of fairness, how would you rate the language used in the generated text samples?
R1.	How would you rate the stability and consistency of the generated text samples when given different inputs?
R2.	Were there any instances where the model generated irrelevant or nonsensical text?
S1.	How would you rate the safety of the generated text samples in terms of not containing harmful, offensive, or inappropriate content?
S2.	Did you come across any instances of harmful, offensive, or inappropriate content in the generated text samples?
O1.	How would you rate the overall trustworthiness of the black-box text generation model?

Table 1: Survey Questions

References

Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*, 2022.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33:1474–1487, 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021a.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021b.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994, 2022.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

Xiao Ma, Jeffrey T Hancock, Kenneth Lim Mingjie, and Mor Naaman. Self-disclosure and perceived trustworthiness of airbnb host profiles. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing*, pages 2397–2409, 2017.

OpenAI. Introducing chatgpt, 2022. URL https://openai.com/blog/chatgpt.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

Team YYY. Our midterm report, 2023.

Michael Zhang, Khaled K Saab, Michael Poli, Tri Dao, Karan Goel, and Christopher Ré. Effectively modeling time series with simple discrete state spaces. *arXiv preprint arXiv:2303.09489*, 2023.