Systems & Control Transactions

Original Research Article
Peer Reviewed Conference Proceeding
ESCAPE 36 - European Symposium on Computer Aided Process Engineering
Sheffield, UK. 21-24 June 2026

PSE PRESS

Relating Loss Geometry to Empirical Generalization in Recurrent Neural Net Surrogates: Three Tanks Case Study

Ricardo M. Roxas II[a] and Karl Ezra Pilario[a]*

[a] Process Systems Engineering Laboratory, Department of Chemical Engineering, University of the Philippines Diliman, Quezon City, 1101, Philippines
* Corresponding Author: kspilario@up.edu.ph.

ABSTRACT

Recurrent neural nets (RNNs) are now commonly used for the surrogate modeling of process systems, leading to better control and faster real-time optimization. However, when trained with small training data sets, most experiments show that RNNs exhibit poor generalization abilities outside the range of the training data space. Nonetheless, recent advances in deep learning research have shown that certain characteristics of the loss landscape of trained models, such as the flatness around the local minimum, tend to relate to generalization ability. This paper investigates this phenomenon for the case of RNN surrogates of the well-known Three Tanks case study, which is representative of many continuous processes. We trained a total of 200 LSTMs (long short-term memory networks) differing in initialization, architecture, and training dynamics on the same data of 500 samples. The number of model parameters ranges from 238 to 11, 353. We estimated the loss curvature of each trained model using the Hessian-vector products method and then evaluated their test accuracy on 100 pre-generated data sets with 50% larger input amplitude than the training data to force extrapolation. We found that the estimate of the top eigenvalue of the Hessian matrix is 75% rank-correlated to the one-step-ahead prediction accuracy averaged on all test data sets. We report other insights on the impact of other Hessian-based metrics to generalization ability. Our study can potentially inform how to control the training dynamics of RNN surrogates to improve their extrapolation ability in the absence of first-principles knowledge.

Keywords: System Identification, Machine Learning, Artificial Intelligence, Derivative Free Optimization, Dynamic Modelling, Hessian vector products, Generalization

INTRODUCTION

Nonlinear system identification tools are now used for the surrogate modelling of continuous industrial processes. Common tools include Koopman models, kernel subspace methods, and neural nets [1]. The recurrent neural net (RNN) is a prominent type of deep learning model suitable for multivariate time series data. Previous works already used RNNs to achieve nonlinear model predictive control (MPC) [2], fault detection [3-4], and process optimization [5] in various case studies.

Despite these applications, the generalizability of RNNs is still poorly understood. In cases where industrial data is scarce or has poor quality, achieving a high validation accuracy does not guarantee good performance during deployment [2]. When the model encounters out-of-distribution data or if it is forced to extrapolate from the training data space, model errors may amplify as it makes future predictions. Yet researchers aim to train valid RNN models from start-up to shutdown [6].

One way to understand the validity of RNNs is to derive generalization bounds. For instance, based on the Rademacher complexity approach [7], RNNs were proven to generalize better with more training samples and simpler architectures. Meanwhile, vanilla RNNs with ReLU (rectified linear unit) activation were shown to generalize better with lower matrix 1-norm and Fisher-Rao norm of the weights, which is consistent with improvements via weight decay and gradient clipping. Recently, [8] obtained tighter bounds on the RNN generalization error using the Frobenius norms of the weights, still pointing to the importance of weight decay in practice.

While theoretical results make clear steps forward in analyzing the validity of RNNs, empirical results are also

helpful especially for finding RNN architectures that generalize well. A recent work validated LSTM surrogates for a boiler, a debutanizer, and two nonlinear reactors [9]. In a polymerization case study, [10] recommends the use of partially input convex RNNs for better accuracy and computational efficiency under MPC. To find good LSTM architectures, [11] advocates for the use of state-of-the-art hyper-parameter optimization methods. This idea is now widely accepted in many LSTM surrogate modeling studies [6], [12].

Aside from architectural choices, there is a growing interest in relating the geometry of the training loss landscape of neural nets to their generalizability. As early as [13], the flatness of the local minima of the training loss was already tied to better generalization (the authors are also the originators of the LSTM), i.e. models that converged on flat minima generalize better than those on sharp minima. In the past, many flatness or sharpness metrics have been defined based on measuring the loss curvature at the minimum, such as the Hessian eigenspectrum [14], Hessian spectral radius, Hessian trace, top Hessian eigenvalue [15], and ϵ-sharpness [16]. Some definitions were motivated by empirical observations while others are more theoretical, such as those based on PAC-Bayes (Probably Approximately Correct-Bayesian) generalization bounds [17]. Some sharpness metrics, however, can be arbitrarily changed by scaling the weights without changing the neural net behavior such as the Hessian eigenspectrum [15]. Hence, scale-invariant metrics are more favorable such as the normalized sharpness [17] or the effective dimensionality (effective rank) of the Hessian of the loss [18], [19]. Due to these works, sharpness has been used to control neural net training to give models that generalize better [20-22].

In terms of regularization, weight decay is known to improve generalization, but the most common L^2 regularization is only one approach. Spectral norm regularization of weight matrices [23] is another approach based on the notion that models generalize better if they are insensitive to large input perturbations. For the case of RNNs, penalizing the largest singular value of the recurrent weight matrix was found to provide smoother state transitions [24] which is another form of regularization.

In this paper, we are interested in relating the training loss curvature to generalization in RNNs for surrogate modeling in continuous industrial processes. Our belief that generalization is related to some form of smoothness in RNNs is motivated by our previous work that pitted RNNs against a kernel method for MPC, namely mixed kernel canonical variate analysis (MKCVA) [25]. MKCVA was found to generalize better in the extrapolation regime, despite being at par in terms of median accuracy with LSTMs at the interpolation regime. Hence, MKCVA gave better control than LSTMs at various setpoints. We attributed this superior performance to the use of kernels that encode smoothness properties. Knowing this, our present aim is to attempt to find some geometry-based indicators of test-time error in trained RNNs. Consistent with [18], we do not view kernel methods and neural nets to be competing, rather, we are interested in finding elements of any representation learner that can improve generalization in surrogate modeling.

To achieve this aim, we take a representative industrial case study, namely the Three Tanks system [25], and train numerous LSTMs on a single training data set from a noisy simulation of this system. These LSTMs differ in architecture, training dynamics, and weight initialization. We then compute loss curvature metrics such as the top Hessian eigenvalue, the anisotropy ratio, the effective rank of the Hessian, and including the spectral norm of the recurrent weights of each LSTM, then connect them to 1-step ahead prediction error averaged over 100 pre-generated test data sets from the same noisy simulator but with 50% larger input signals. We hope that our results can help inform the control of LSTM training towards models with better generalization.

Preliminaries

We consider the vanilla LSTM network defined by:

$$\boldsymbol{z}_k = \sigma_{io}(\boldsymbol{W}_z \boldsymbol{x}_k^* + \boldsymbol{R}_z \boldsymbol{y}_{k-1}^* + \boldsymbol{b}_z) \tag{1}$$

$$\boldsymbol{i}_k = \sigma_{rec}(\boldsymbol{W}_i \boldsymbol{x}_k^* + \boldsymbol{R}_i \boldsymbol{y}_{k-1}^* + \boldsymbol{b}_i) \tag{2}$$

$$\boldsymbol{f}_k = \sigma_{rec}(\boldsymbol{W}_f \boldsymbol{x}_k^* + \boldsymbol{R}_f \boldsymbol{y}_{k-1}^* + \boldsymbol{b}_f) \tag{3}$$

$$\boldsymbol{c}_k = \boldsymbol{z}_k \odot \boldsymbol{i}_k + \boldsymbol{c}_{k-1} \odot \boldsymbol{f}_k \tag{4}$$

$$\boldsymbol{o}_k = \sigma_{rec}(\boldsymbol{W}_o \boldsymbol{x}_k^* + \boldsymbol{R}_o \boldsymbol{y}_{k-1}^* + \boldsymbol{b}_o) \tag{5}$$

$$\boldsymbol{y}_k^* = \sigma_{io}(\boldsymbol{c}_k) \odot \boldsymbol{o}_k \tag{6}$$

where $\boldsymbol{x}_k^* = \{[\boldsymbol{u}_t^T, \boldsymbol{y}_{t-1}^T]^T, t \in [k - \text{lookback}, \dots, k]\}$ are the LSTM inputs, $\boldsymbol{y}_k^* = \{\boldsymbol{y}_t, t \in [k - \text{lookback}, \dots, k]\}$ are the LSTM outputs, $\boldsymbol{u}_t$ and $\boldsymbol{y}_t$ are the inputs and outputs of the process, $\boldsymbol{W}_z, \boldsymbol{W}_i, \boldsymbol{W}_f, \boldsymbol{W}_o$ are input weights, $\boldsymbol{R}_z, \boldsymbol{R}_i, \boldsymbol{R}_f, \boldsymbol{R}_o$ are the recurrent weights, $\boldsymbol{b}_z, \boldsymbol{b}_i, \boldsymbol{b}_f, \boldsymbol{b}_o$ are the biases, $\boldsymbol{c}_k, \boldsymbol{i}_k, \boldsymbol{f}_k, \boldsymbol{o}_k$ are the cell state and gate outputs, σ_{io} is the input-output activation, σ_{rec} is the recurrent activation, and $\odot$ is element-wise multiplication.

For the metrics derived for LSTMs, we denote $\boldsymbol{H}$ as the local Hessian matrix of the loss, $tr(\boldsymbol{H})$ is the trace, and λ is an eigenvalue of $\boldsymbol{H}$. We also measure the singular values of weights matrix in a trained LSTM, which we denote as σ with no subscript, to distinguish them from activation functions σ_{io} and σ_{rec} in the LSTM equations.

METHODOLOGY

To find the best LSTM surrogate for the process system while also encouraging diversity in the set of models from which to reason about curvature effects, we

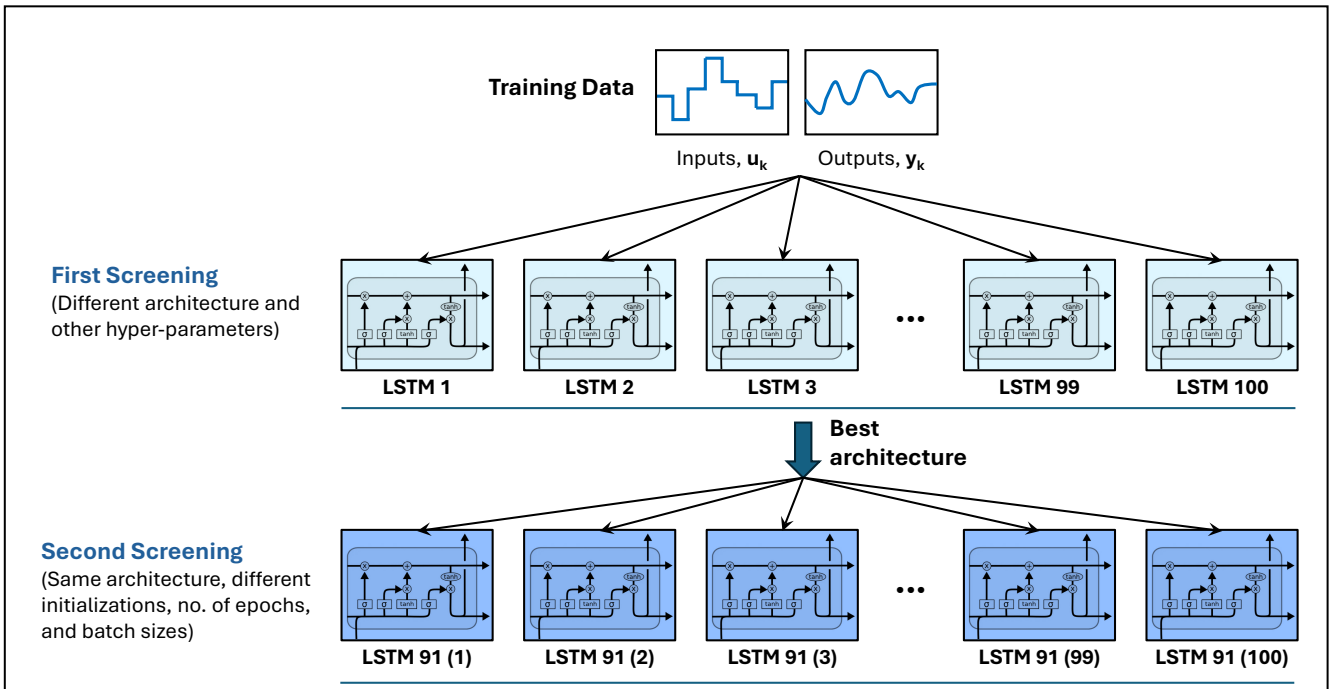

Figure 1. Methodology for investigating the relationship of LSTM architecture, training dynamics, and training loss curvature to model generalizability in this work.

designed an experiment that includes two screening stages (see Fig. 1). In the first stage, we train 100 LSTMs all with hyper-parameter tuning via Optuna [11] on the same fixed training data of 500 samples. Table 1 lists the hyper-parameters and their search ranges. To enforce more diversity, we deliberately narrowed the search into various regions of the search range for some LSTMs. For example, some LSTMs were forced to have ReLU input activations only, some were tuned up to 20 Optuna trials only rather than 50, some were stopped at 50 epochs rather than 200, some were forced to search within 30 to 40 hidden units rather than 5 to 45, and so on. Note that all the LSTMs in this study used *sigmoid* as the recurrent activation to maintain the notion that gates must output a value only from 0 to 1. All LSTMs also have some form regularization by using dropout and L^2 penalty.

For the second screening stage, we then picked the best LSTM from the first screening and kept its architecture. This stage now trains 100 additional LSTMs on the same training data of 500 samples, all of which have the same architecture (i.e. number of model parameters, activation, and regularization) but with different number of training epochs, training batch size, and initial weights.

The purpose of this stage is to remove the effects of model size and architecture choice and instead focus on sampling the training loss landscape diversely. Table 2 shows the range of number of epochs and batch sizes tried in this stage. Note that even though the batch size differs between LSTMs during training, we evaluated all curvature metrics using the entire batch of 500 samples. All generated data sets start with the same known initial state to remove any errors due to different initial conditions. The lookback is also fixed at 23, as with [25].

Table 1: Hyper-parameter search ranges in 1st screening.

Name	Search Range
No. of LSTM units	5, 6, 7, ..., 45
Input-output activation, σ_{io}	Tanh, ReLU, Linear
Dropout rate	0.0, 0.1, ..., 0.5
Recurrent dropout rate	0.0, 0.1, ..., 0.5
Optimizer	Adam, SGD, RMSProp, Adadelta
Batch Size	2^3, 2^4, ..., 2^8
Validation split	0.1 or None
L^2 regularization	0.001 to 1
No. of training epochs	50 to 200
No. of Optuna trials	20 to 50

Table 2: Ranges used for training LSTMs in 2nd screening.

Name	Search Range
Batch Size	2^4, 2^5, ..., 2^8
No. of training epochs	100, 200, 300

Curvature and Smoothness Metrics

We measure the following curvature and local state transition smoothness metrics of all trained LSTMs.

Top Hessian Eigenvalue, $\lambda_{\max}$

A measure of the local curvature of the loss in the parameter direction where it is sharpest.

Hessian Trace, $\mathbf{tr}(H)$

The sum of the eigenvalues of the Hessian.

Anisotropy Ratio, $\lambda_{\max}/\mathbf{tr}(H)$

Top Hessian eigenvalue divided by the Hessian trace, indicating the fraction of the total curvature concentrated in the sharpest direction.

Effective Rank of the Hessian, R_{eff}

Indicates the true dimensionality of the loss curvature, which we compute as the ratio of the squared Frobenius norm to the squared spectral norm:

$$R_{\text{eff}} = (\Sigma\lambda_i)^2/\Sigma\lambda_i^2 \tag{7}$$

Recurrent Kernel Spectral Norm, $\sigma_{\max}(R)$

The largest singular value of the recurrent kernel matrix $[\boldsymbol{R}_z, \boldsymbol{R}_i, \boldsymbol{R}_f, \boldsymbol{R}_o]$ from (1), (2), (3), (5).

Our hypotheses are as follows. If the top Hessian eigenvalue of the loss landscape is low, then the loss is flat along the worst-case (sharpest) direction. The Hessian trace measures the mean overall curvature on all directions, so a low trace also indicates flatness. An anisotropy ratio near 0 means that the curvature is more *spread* out on all directions, and if it is near 1 then the minima is likely a narrow ridge that is flat except for a single sharp direction. Hence, if the anisotropy ratio is near 0 and λ_{max} is also low, then the minima is a flat and wide basin. Meanwhile, the R_{eff} measures the effective number of dominant directions in the curvature. Lower R_{eff} can also mean flatter minima since the *number* of dominant (sharp) directions is low while the rest of the directions are flat. A lower R_{eff} can also indicate that the model is more overparametrized, possibly admitting a simplicity bias [18]. By Occam's Razor, simpler models tend to generalize well.

We include the recurrent kernel spectral norm [23] in the analysis since we consider it to be a smoothness metric. The lower the maximum singular value of the recurrent weights matrix, the more insensitive the model is to large changes in the hidden vector $\boldsymbol{y}_k^*$, which is the data being fed back to the LSTM. This means that upon making multi-step-ahead predictions, smoother trajectories are expected.

For the case of different architectures, as in the first screening stage (see Fig. 1), the Hessian trace and the effective rank must be normalized by the number of model parameters to be comparable.

Approximating the Hessian Metrics

The time and memory cost of computing the full Hessian is quadratic with the model size, making it infeasible for large LSTMs. To address this, most literature use the Hessian-vector products (HVP) method which computes unbiased estimates of the product of the Hessian and an arbitrary vector $\boldsymbol{Hv}$ in linear time [26].

The full set of algorithms are not explained here, but we refer readers to the implementation in PyHessian [27]. In particular, the $\lambda_{\max}$ is calculated using the power iteration method, while the Hessian trace and effective rank use the Hutchinson method, all of which use HVP. We arbitrarily set 30 iterations in the power iteration method and 10 samples in the Hutchinson method.

Generalization Metrics

To measure generalization, 100 additional data sets each having 500 samples were generated from the simulation of the process that differ in the input signals and noise. Inputs in both training and testing sets are amplitude modulated pseudo-random binary signals, but the test data input amplitudes are 50% larger in range than in the training data to test for extrapolation ability. All test data sets are fixed in both screening stages in Fig. 1.

We calculate the mean squared error (MSE) and R^2 accuracies in the 1-step-ahead and 500-step-ahead (or infinite-step) prediction scenarios, as follows:

$$\text{MSE} = \frac{1}{N}\Sigma\left(y_{\text{true}} - y_{\text{pred}}\right)^2 \tag{8}$$

$$\text{R}^2 = 1 - \Sigma\left(y_{\text{true}} - y_{\text{pred}}\right)^2/\Sigma(y_{\text{true}} - \mu_{\text{true}})^2 \tag{9}$$

which are averaged on all outputs, then averaged on all 100 test data sets for a single LSTM. An MSE near 0 and an R^2 near 1 means that a model generalizes well.

Three Tanks Case Study

The system is governed by same differential equations as those written in [25]. Similarly, this paper uses the same two inputs and three outputs. The exact training data set used in this work is shown in Fig. 2.

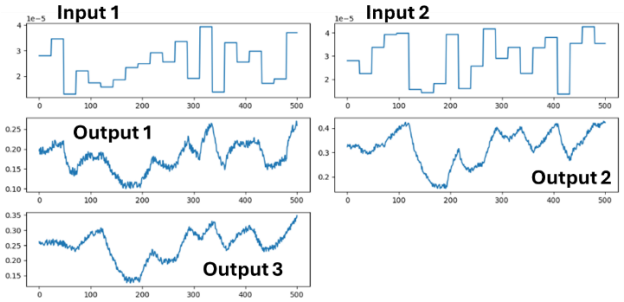

Figure 2. The training data set of 500 samples used to train all 200 LSTMs in this work.

RESULTS AND DISCUSSION

Correlations between Metrics

Quantifying the correlation between metrics already brings many insights. Fig. 3 shows the Spearman rank correlations between the choices during LSTM training,

curvature and smoothness metrics, and their generalizability for the 100 LSTMs with different architectures in the first screening stage. We chose rank-based correlation rather than Pearson because: (i) some accuracy metrics are too worse that they appear as outliers, hence, we analyzed more about the directions of the interaction rather than their magnitudes, and (ii) we wish to find possibly nonlinear interactions rather than linear ones.

First, we find that a lower top Hessian eigenvalue appears in LSTMs that generalize well, but the normalized trace, anisotropy, and normalized effective rank of the Hessian do not indicate the same. The rest of the Hessian metrics are weakly correlated with the mean MSE and mean R^2 for both 1-step-ahead and Inf-step-ahead accuracy. This means that for LSTMs with different architectures, the flatness at the direction of sharpest curvature dictates extrapolation behavior more effectively than the rest of the Hessian metrics. We further checked that the ranges of normalized $tr(\boldsymbol{H})$ (0-0.12) and normalized R_{eff} (0-0.03) are both close to 0. Hence, we attribute their weak correlation with accuracy to either: (i) possible saturation at already low values for trained LSTMs, or (ii) poor choice of normalizing constant, i.e. the number of model parameters. This needs further study.

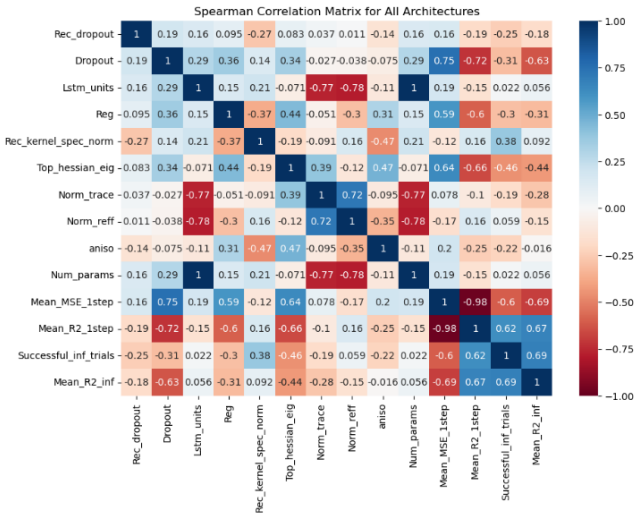

Figure 3. Spearman correlation matrix between metrics for LSTMs in the 1st screening (different architectures).

In terms of architecture, Fig. 3 shows that the number of model parameters is also weakly correlated with the test accuracy, meaning that there are extrapolative LSTMs from both small and large networks. We also find that the larger the network, the lower the normalized trace and normalized effective Hessian rank (correlation=-0.77). This means that bigger LSTMs tend to be more over-parametrized after training, i.e. with larger networks, fewer weights matter against the total number.

We chose a particular LSTM with the best accuracy for inf-step-ahead prediction, namely LSTM-91, in the first screening stage. LSTM-91 gained a mean R^2 of 82.7% and 85.67% for inf-step-ahead and 1-step-ahead prediction, respectively. Table 3 gives more details on this LSTM. Then, we trained 100 more LSTMs with the same hyper-parameters as LSTM-91, except with different initial weights, no. of epochs, and batch sizes.

Table 3: Details of the best LSTM from the 1st screening.

Hyper-parameter	**Value**
No. of LSTM units	31
No. of model parameters	4684
No. of training epochs	100 (Adam)
Input-output activation	Linear
Batch size	2^4
Recurrent dropout rate	0.1
Dropout rate	0.0
L^2 regularization	0.001126
No. of Optuna trials	50

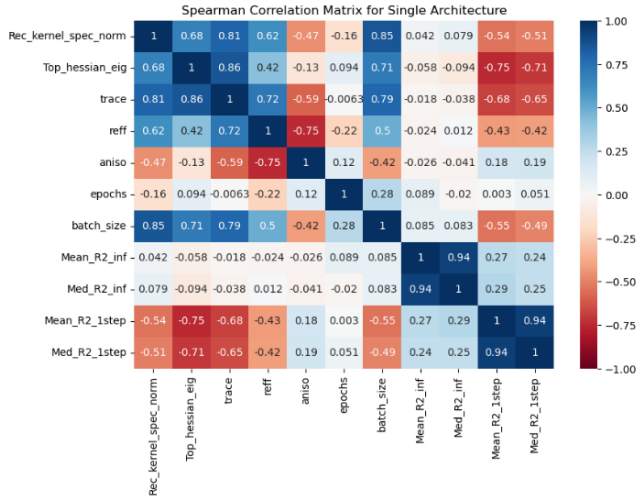

Figure 4. Spearman correlation matrix between metrics for LSTMs in the 2nd screening (same architecture).

Fig. 4 shows the Spearman rank correlation matrix for the 2nd screening stage. For the same architecture, we see stronger correlations between curvature, smoothness, and generalizability. Low values of $\sigma_{\max}(\boldsymbol{R}), \lambda_{\max}$, and $tr(\boldsymbol{H})$ all indicate better 1-step-ahead prediction accuracy (in terms of mean and median R^2). These metrics are also found to be correlated among each other: smoother model dynamics (low $\sigma_{\max}(\boldsymbol{R})$) correspond with flatter minima (low $\lambda_{\max}$). However, we find that anisotropy and effective rank still do not indicate generalizability. To understand why, we looked at the relationship between the metrics more closely in Fig. 5.

Studying 100 LSTMs with Same Architecture

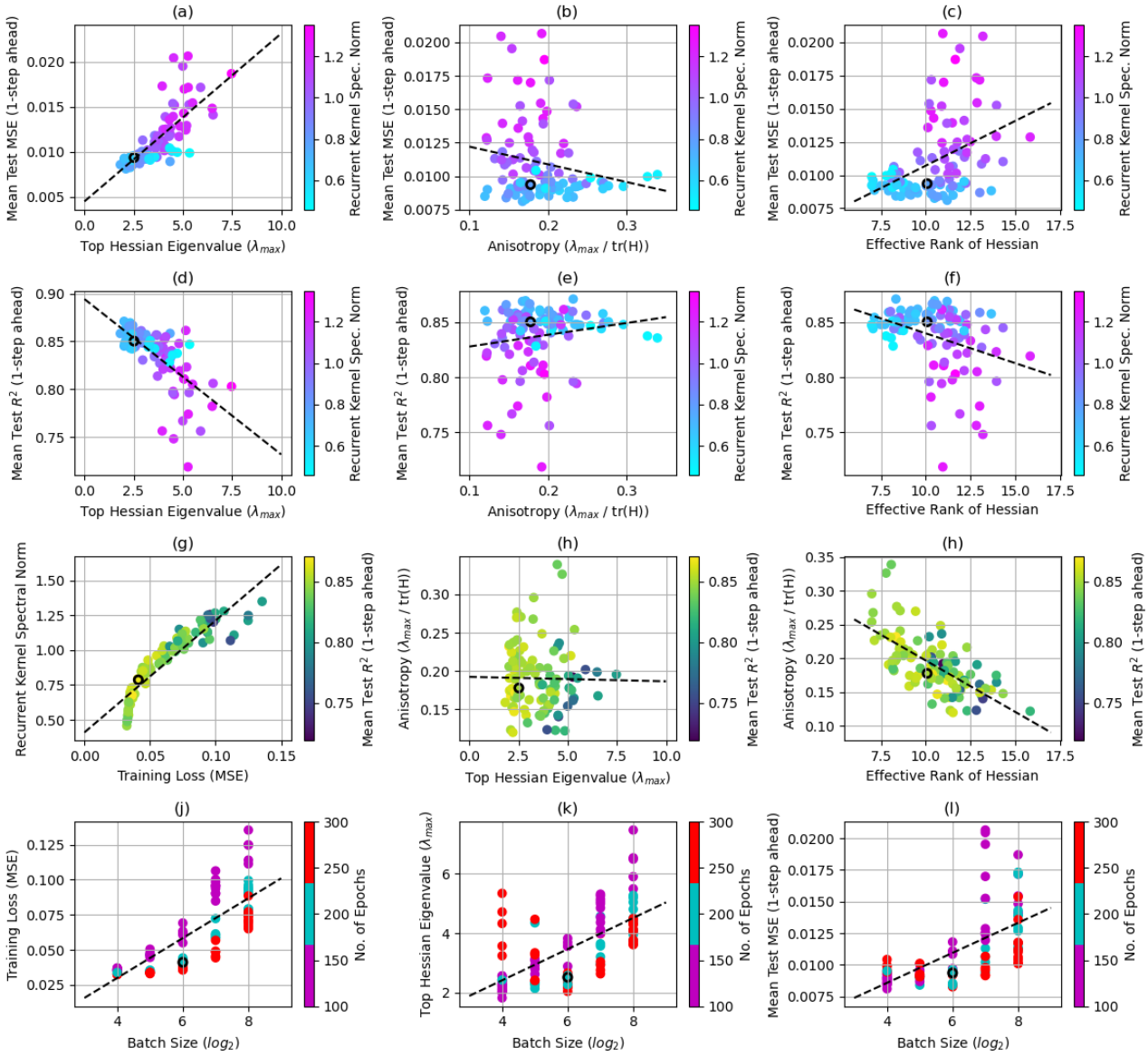

Figure 5. Second Screening Results: Scatter plots relating Hessian metrics, recurrent kernel spectral norm, training loss, batch size, and number of epochs to the 1-step-ahead accuracies averaged on 50 test data sets. Dashed line: Best-fit line; Black circle: LSTM with best inf-step-ahead accuracy.

In Fig. 5(a, d), we see that LSTMs with $\lambda_{\max} < 2.5$ and $\sigma_{\max}(\boldsymbol{R}) < 0.8$ are good extrapolators, and the linear correlation with test accuracy is clearer at lower values of $\lambda_{\max}$. Fig. 5(b, c, e, f) show that there are models that do not generalize well despite having lower anisotropy ratio and lower effective rank. Upon inspecting the trend of the recurrent kernel spectral norm in these plots, the reason for weak correlation of $tr(\boldsymbol{H})$ and R_{eff} with accuracy is the presence of low trace, low rank Hessian models with simultaneously high recurrent kernel spectral norm (pink).

Fig. 5(g) shows that as the model reaches lower minima in the training loss, the recurrent kernel spectral norm also tends to decrease, leading to more accurate models. Fig. 5(h) tells us that anisotropy is not correlated with $\lambda_{\max}$, meaning that the $\lambda_{\max}$ and Hessian trace can be independent in fully trained LSTMs. Next, Fig. 5(i) shows that anisotropy is anti-correlated with R_{eff}, which means that the trained LSTMs converge to either higher effective-dimensional but relatively uniform curvature minima or lower effective-dimensional but non-uniform curvature minima. Still, neither of these ends are indicative of generalizability for 1-step-ahead prediction.

Lastly, we tested some notions from literature about the relationship of batch size, flatness, and test accuracy.

According to [16], large batches are known to drive models to sharper minima and smaller batches lead to flatter minima. Fig. 5(j, k, l) shows that smaller batch sizes tend to converge to lower training loss, but not necessarily flat. We found 3 LSTMs, in particular, each trained with a batch size of 16 using 300 epochs that reached sharper minima ($\lambda_{\max} > 4$) but with roughly the same training loss (~0.03) as those on flat minima ($\lambda_{\max} < 2.5$). These 3 also had the lowest recurrent kernel spectral norms among all LSTMs ($\sigma_{max}(\boldsymbol{R}) < 0.5$, lightest blue dots in Fig. 5(a)). Consistent with [15], these are the sharp-minima LSTMs that still generalize well, but this time owing to low recurrent kernel spectral norms. On the other hand, models with large batch sizes indeed tend to converge to sharper minima, especially those trained with only 100 epochs.

Because we have 100 LSTMs with the same number of model parameters, we can investigate which models are similar in the weight space. We projected the 4, 684 weights of each LSTM in 2-D using t-distributed stochastic neighborhood embedding (t-SNE), a widely used data visualization method. Fig. 6 shows that there is a cluster of LSTMs with roughly equal training loss and similar weights (meaning that they may be "close" to each other in the loss landscape). The other more isolated LSTMs can each be thought of having very different weights distributions. All LSTMs that clustered were found to generalize well in terms of 1-step-ahead test accuracy, but not all of them have flat minima since the 3 sharp-minima LSTMs are also in this cluster. By finding several similar LSTMs clustered together (37 out of 100) despite starting with various initial weights, we believe that there likely exists a low-lying mostly flat basin (with a few sharp minima) in the loss landscape that is relatively easy to find via Adam when training the LSTM-91 architecture (see Table 3) given the fixed training data set.

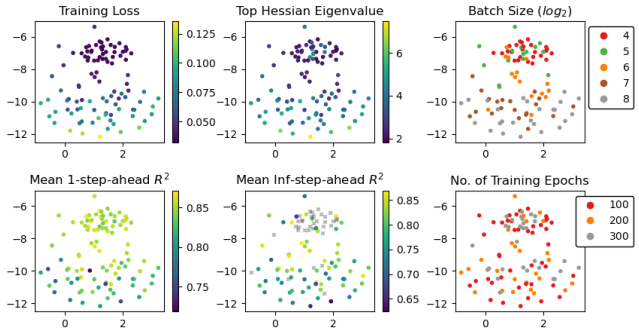

Figure 6. A 2-D projection of 4, 684 weights of the LSTMs in the 2nd screening stage via t-SNE, colored by the titles.

We also found that most of the clustered LSTMs did not perform well for multi-step-ahead prediction (see gray x's in the middle bottom box denoting models with negative mean R^2 for more than 10 out of 100 data sets). This means that steering LSTMs to be accurate 1-step-ahead predictors still does not guarantee stability.

CONCLUSION

This work aims to empirically relate the flatness of the local minima of the loss and smoothness of the model local dynamics to generalizability in LSTMs for the Three Tanks case study. We summarize our findings into the following, subject to the limitations of our experiments:

LSTMs that converged to flatter minima tend to generalize better, but mostly when flatness is measured as the top Hessian eigenvalue. This metric rank-correlates with the mean test R^2 (at -0.66) regardless of architecture. Better rank-correlation (-0.75) was found when analyzing the loss landscape of a single architecture.

Lower Hessian trace (overall loss curvature) also indicates better 1-step-ahead prediction accuracy (rank-correlation=-0.68), but only among LSTMs with the same architecture.

LSTMs with smooth local dynamics marked by a low recurrent kernel spectral norm also tend to generalize better given a fixed architecture.

Based on these results, it might then be worthwhile to steer the training of LSTM surrogates while being aware of both sharpness and local transition smoothness in terms of the metrics that we found useful in this work. Nonetheless, future work must also validate our results in other processes and with more diverse RNNs.

ACKNOWLEDGEMENTS

Authors are grateful for the support of the Philippine Department of Science and Technology, Engineering Research and Development for Technology (DOST-ERDT).

AUTHOR IDENTIFIERS

Author ORCIDs:
Roxas RM II: 0000-0002-1675-3926
Pilario KE: 0000-0001-5448-0909

REFERENCES

1. Pilario KES, Cao Y, Shafiee M. A kernel design approach to improve kernel subspace identification. IEEE Trans. Ind. Electron. 68:6171-6180 (2021). https://doi.org/10.1109/tie.2020.2996142
2. Wu Z, Christofides PD, Wu W, Wang Y, Abdullah F, Alnajdi A, Kadakia Y. A tutorial review of machine learning-based model predictive control methods. Reviews in Chemical Engineering 41:359-400 (2024). https://doi.org/10.1515/revce-2024-0055

3. Sun W, Paiva ARC, Xu P, Sundaram A, Braatz RD. Fault detection and identification using bayesian recurrent neural networks. Computers & Chemical Engineering 141:106991 (2020). https://doi.org/10.1016/j.compchemeng.2020.106991
4. Chen H, Cen J, Yang Z, Si W, Cheng H. Fault diagnosis of the dynamic chemical process based on the optimized CNN-LSTM network. ACS Omega 7:34389-34400 (2022). https://doi.org/10.1021/acsomega.2c04017
5. Zhu W, Chebeir J, Webb Z, Romagnoli J. A Deep Learning Approach on Surrogate Model Optimization of a Cryogenic NGL Recovery Unit Operation. *Computer Aided Chemical Engineering*, 48:1285–1290 (2020)
6. Esche E, Weigert J, Brand Rihm G, Göbel J, Repke JU. Architectures for neural networks as surrogates for dynamic systems in chemical engineering. Chemical Engineering Research and Design 177:184-199 (2022). https://doi.org/10.1016/j.cherd.2021.10.042
7. Alhajeri MS, Alnajdi A, Abdullah F, Christofides PD. On generalization error of neural network models and its application to predictive control of nonlinear processes. *Chem. Eng. Res. Des.* 189:664–679 (2023) https://doi.org.10.1016/j.cherd.2022.12.001
8. Cheng X, Huang K, Ma S. Generalization and risk bounds for recurrent neural networks. Neurocomputing 616:128825 (2025). https://doi.org/10.1016/j.neucom.2024.128825
9. Li J, Qin SJ. Applying and dissecting LSTM neural networks and regularized learning for dynamic inferential modeling. Computers & Chemical Engineering 175:108264 (2023). https://doi.org/10.1016/j.compchemeng.2023.108264
10. ?awry?czuk M. Input convex neural networks in nonlinear predictive control: a multi-model approach. Neurocomputing 513:273-293 (2022). https://doi.org/10.1016/j.neucom.2022.09.108
11. Pravin PS, Tan JZM, Yap KS, Wu Z. Hyperparameter optimization strategies for machine learning-based stochastic energy efficient scheduling in cyber-physical production systems. Digital Chemical Engineering 4:100047 (2022). https://doi.org/10.1016/j.dche.2022.100047
12. Shakouri B, Murthy S, De Clippeleir H, Lesnik K, Massoudieh A. Surrogate modeling of ASM1 using feedforward and LSTM neural networks. Journal of Water Process Engineering 81:109246 (2026). https://doi.org/10.1016/j.jwpe.2025.109246
13. Hochreiter S, Schmidhuber J. Flat minima. Neural Computation 9:1-42 (1997). https://doi.org/10.1162/neco.1997.9.1.1
14. Chaudhari P, Choromanska A, Soatto S, LeCun Y, Baldassi C, Borgs C, Chayes J, Sagun L, Zecchina R. Entropy-SGD: Biasing Gradient Descent Into Wide Valleys. (2017) http://arxiv.org/abs/1611.01838
15. Dinh L, Pascanu R, Bengio S, Bengio Y. Sharp Minima Can Generalize For Deep Nets. (2017) http://arxiv.org/abs/1703.04933
16. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. (2017) http://arxiv.org/abs/1609.04836
17. Tsuzuku Y, Sato I, Sugiyama M. Normalized Flat Minima: Exploring Scale Invariant Definition of Flat Minima for Neural Networks Using PAC-Bayesian Analysis. *Proceedings of the 37th International Conference on Machine Learning* 119 (2020)
18. Wilson AG. Deep Learning is Not So Mysterious or Different. (2025) http://arxiv.org/abs/2503.02113
19. Shoham N, Mor-Yosef L, Avron H. Flatness After All? (2025) http://arxiv.org/abs/2506.17809
20. Foret P, Kleiner A, Mobahi H, Neyshabur B. Sharpness-Aware Minimization for Efficiently Improving Generalization. (2021) http://arxiv.org/abs/2010.01412
21. Zhang X, Xu R, Yu H, Dong Y, Tian P, Cu P. Flatness-Aware Minimization for Domain Generalization. (2023) http://arxiv.org/abs/2307.11108
22. Lee S, He C, Avestimehr S. Achieving small-batch accuracy with large-batch scalability via hessian-aware learning rate adjustment. Neural Networks 158:1-14 (2023). https://doi.org/10.1016/j.neunet.2022.11.007
23. Yoshida Y, Miyato T. Spectral Norm Regularization for Improving the Generalizability of Deep Learning. (2017) http://arxiv.org/abs/1705.10941
24. Saab S Jr, Fu Y, Ray A, Hauser M. A dynamically stabilized recurrent neural network. Neural Process Lett 54:1195-1209 (2021). https://doi.org/10.1007/s11063-021-10676-7
25. Pilario KE, Wu Z. Fast mixed kernel canonical variate analysis for learning-based nonlinear model predictive control. Chemical Engineering Research and Design 219:19-33 (2025). https://doi.org/10.1016/j.cherd.2025.05.050
26. Pearlmutter BA. Fast exact multiplication by the hessian. Neural Computation 6:147-160 (1994). https://doi.org/10.1162/neco.1994.6.1.147
27. Yao Z, Gholami A, Keutzer K, Mahoney MW. Pyhessian: neural networks through the lens of the hessian. 2020 IEEE International Conference on Big Data (Big Data) :581-590 (2020). https://doi.org/10.1109/bigdata50022.2020.9378171