Diagnostic assessment of reservoir release policies using LSTM across the continental U.S.

Matthew Chen1, Jonathan D. Herman2

1 Department of Civil & Environmental Engineering, University of California, Davis

2 Department of Civil & Environmental Engineering, University of California, Davis

# Abstract (When we are framing the contribution, let’s try to make it less about the use of LSTMs being new, and more about the diagnostic part – you had written some good research questions to include here).

# Introduction

Ideas for lit review paragraphs:

P1: Importance of reservoirs in hydrologic models in general. Large sample analysis now possible thanks to better datasets. (see refs / reasoning in proposal attached)

P2: Data-driven models of reservoir release policies (explain what types of models have been tested, contrast with other types of release policy models, see proposal section 2.2 especially). Data-driven models of human decisions in water systems more generally: see Liam’s 2021 paper for some references.

Benefits of large sample training – Turner et al 2021

P3: Success of LSTMs in hydrology – what have they been shown to be able to do, and why? Issues of mass conservation, large sample training. By similar logic, LSTMs should also be able to perform well as data-driven models of reservoir releases, because the decisions involve accumulation over time.

LSTMs have been used a bit for reservoir release models, but it is usually ad hoc. There has not been an effort to understand how good the performance is across basins, or to explain the reasons for good/bad performance. (This might fit better in P4)

https://scholar.google.com/scholar?hl=en&as\_sdt=0%2C5&q=Reservoir+Release+Policies+Using+LSTM&btnG=

P4: Issues related to interpretability / diagnostics – both in hydrology (LSTMs) and reservoir release policies (from policy search or other machine learning methods). What is the right balance between flexibility and interpretability? What are some promising methods for analyzing the results of these models?

P5: (not sure, let’s see how it comes together)

P6: This study contributes … incl research questions

\*Note: The attached proposal I wrote with a collaborator at Cornell, Stefano Galelli. Depending which parts of the proposal we use in this paper, I may suggest that we invite him as a coauthor. He is very good and would probably be a big help in framing the results. We can discuss.

# Methods

## Long Short-Term Memory Networks

Here we give a brief introduction to the Long Short-Term Memory (LSTM) architecture (Hochreiter & Schmidhuber, 1997). The LSTM model addresses the problem of unstable gradients in training recurrent neural networks by conserving long term information using memory cells managed by several gating mechanisms, which control the flow of information through element-wise matrix multiplication with gate values ranging between 0 and 1. This allows the model to learn temporal relationships and long-term dependencies. The ability of the LSTM to dynamically accumulate information makes it a good candidate to model dynamical systems (Jordan et al., 2021; Kratzert et al., 2019; Yu Wang, 2017) such as reservoir control. In a LSTM, every timestep has a hidden state and a memory cell state The cell states store and maintain long term information, and information from the cell state is released into the hidden state where it can be used for prediction. This flow of information is managed by the output gate. As new inputs arrive, the model can also save and remove information from the cell state, which are managed by the input gate and forget gate, respectively. For example, in the reservoir control problem, storage states can be modeled by memory cells, where mass accumulation is managed by the input and forget gates, and release decisions can then be modeled based on the accumulated storage and day of the year, as managed by the output gate. Note that a LSTM architecture does not conserve mass unless explicitly tailed to do so.

The gate values at each timestep depend on the previous hidden state and the new input while the sigmoid function enforces that the gates values are between 0 and 1. The forget gate, , parameterized by the weight matrices , , and , controls what information is perpetuated versus forgotten from the previous cell state (Eq. 1).

Meanwhile, the input gate, , controls the information flow from the new input into the cell state. This gate is parameterized by the weight matrices , , and (Eq. 2).

Finally, the output gate, controls information flow from the cell state to the hidden state to make a prediction at the current timestep. This gate is parameterized by the weight matrices , , and (Eq. 3).

After the gate values are computed, a candidate cell state update is computed from the previous hidden state and data input from the current timestep using a *tanh* activation function (Eq. 4).

The cell state is then updated based on the values of the forget and input gates (Eq. 5).

Finally, the hidden state is computed based on the value of the output gate, which is used to derive the final prediction (Eq. 6).

In Figure 1, the operations from Equations 1-6 are presented as a computational graph.

A diagram of a mathematical equation

Description automatically generated

Figure 1. The LSTM architecture represented as a computational graph.

## Data Processing

Reservoir inflow, storage, and release data are drawn from two sources: ResOpsUS (Steyaert et al., 2022), which covers the majority of large reservoirs in the continental U.S. over the period 1980-2020 on a daily timestep; and longer records from the U.S. Bureau of Reclamation (1940s-present) to support more detailed modeling of specific reservoirs in the Western U.S. We filter the ResOpsUS data to only the reservoirs with records that are at least 90% complete record over 1980-2020, which leaves 119 reservoirs including 4 reservoirs from the U.S. Bureau of Reclamation (see Figure 2 map). In general, the reservoir data record including inflow and release timeseries are split into training, validation, and testing portions. The training portion, representing the first 60% of the available timeseries, is used directly for model training, i.e. optimizing model parameters to improve fit. The validation portion, representing the next 20% of the available record, is used for hyperparameter tuning, model selection, and early stopping. Early stopping is a technique that prevents overfitting by interrupting the training process based on the validation data as a proxy for out-of-sample performance (Li et al., 2019). The validation set provides some measure of out-of-sample performance, especially if the validation set is not over utilized in the modeling process (i.e. overfitting to validation or “data leaking”). Finally, the testing portion, representing the last 20% of the available data record, is used solely for the estimation of out-of-sample performance. The testing data is untouched throughout the model building process; however, it is also the furthest away from the training set in time. This may be a challenge if the reservoir operating policy has shifted in the meantime, a challenge we investigate later in the study.

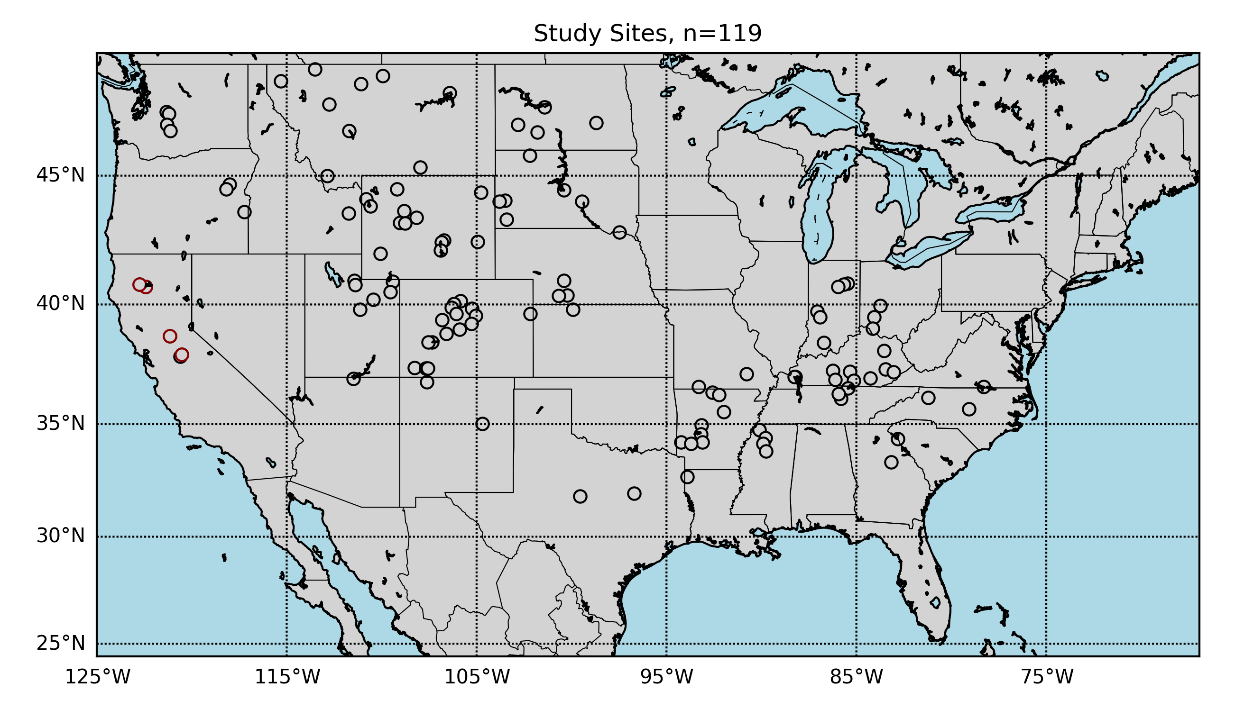


Figure 2. Map of 119 study sites across the continental US. Reservoirs from the U.S. Bureau of Reclamation are highlighted in red, otherwise data is from the ResOpsUS dataset.

Prior to modeling, the data is linearly transformed to be zero mean and unit standard deviation, based on statistics from the training set. In early experiments, data normalization (scaling between 0 and 1) was also tested but provided little benefit over standardization. Additionally, while the majority of records are verified to be complete, missing values are imputed using the training mean. We also split the data into batches of 3 years (the batch sequence length was chosen between 0.5 and 5 years in preliminary testing). Using longer sequences allow the model to capture longer term dependencies but may result in difficulty training due to the vanishing or exploding gradients problem in the training process. That is, gradient magnitudes may become exponentially large or small over many steps of backpropagation through time.

## Model Selection and Hyperparameter Tuning

In the first section of this study, we conduct model selection experiments on an individual reservoir (Shasta Reservoir) to compare model architectures, select optimal hyperparameters, and compare LSTM performance against other machine learning benchmarks. Shasta reservoir was chosen for its long data history and is representative in terms of its degree of regulation (discussed later). It is computationally infeasible to tune hyperparameters using grid search for all 119 reservoirs in the dataset individually, so the model and hyperparameters selection done here are generally considered used throughout the study. While we recognize that optimal hyperparameters for Shasta reservoir may not be optimal for a different reservoir, tuning to a specific reservoir will provide a general idea for the size of model necessary to capture reservoir operations. That is, we assume that reservoir operating policies across the continental US are somewhat similar in terms of their complexity.

We consider four main LSTM architectures (see Figure 3) along with several other machine learning benchmarks. We are interested in the ability of the LSTM to learn to conserve mass and learn reservoir storages implicitly in its cell states; as such, the modeling task is to predict reservoir releases based only on two inputs, the inflow and the day of the year. Model 1 is a standard LSTM model where data is first processed by an LSTM and then by a single layer feed-forward neural network to provide additional non-linear flexibility in learning the operating policy. Model 2 is similar to Model 1, but adopts an autoregressive structure in which the previous output is concatenated as an additional input for the current prediction. For Model 3 and Model 4, we propose two new architectures that conserve mass and accumulate implied storage states internally. These models are inspired by the mass-conserving LSTM (MC-LSTM), which is a modified LSTM architecture where its memory states are true mass accumulators (Hoedt et al., 2021). Specifically, Model 3 explicitly models the mass balance for implied storages as an additional internal state and concatenates them as input so that we do not need to rely on the LSTM gating mechanisms to learn to conserve mass. Model 4 is similar to Model 3; except we discard the LSTM gating mechanisms entirely which results in a model resembling a mass-accumulating recurrent neural network (RNN). However, Model 4 is unable to capture longer-range dependencies through learned cell states beyond storage unlike Model 3. Also note that since mass balance can be expressed explicitly for the reservoir control problem, these models require order of magnitude fewer parameters (more parsimonious) compared to MC-LSTM. All four models are trained using the square error loss function.

For hyperparameter tuning, we tune the model architecture details using exhaustive grid search. Specifically, we select the model with the optimal validation loss over a predefined grid of hyperparameters, averaging over 5 different random seeds to account for stochasticity in the optimization algorithm. We tune the number of LSTM layers (1 or 2), the size of the LSTM hidden layer (between 5 and 50), the hidden size of the feed-forward net (between 5 and 50), and the dropout regularization probability (0.3, 0.5, or 0.7). The LSTM experiments were conducted in the Pytorch deep learning library (Paszke et al., 2019). In training, model parameters were optimized using the Adam algorithm, a first-order stochastic gradient descent algorithm with momentum, and strong performance demonstrated in practice (Kingma & Ba, 2015).

A diagram of a model

Description automatically generated

Figure 3. LSTM model architectures. “FF”: feed-forward neural network; “DOY”: day of the year.

## Machine Learning Benchmarks

As benchmarks for the LSTM model, we consider autoregressive linear and random forest models with 5 lags. Specifically, we model reservoir releases as a function of the current day of the year and inflow, as well as the previous 5 inflow values. We also consider benchmarks where the current observed storage is another data input, since the linear and random forest models cannot learn to preserve information over time unlike the LSTM architecture (these are denoted linear\* and random forest\*, respectively). This modeling problem is represented by Equation 7, where is the predicted target release at time , and are the inflow, storage, and day of year features, respectively.

These benchmark models, trained with square-error loss, are implemented in the open-source *scikit-learn* library (Pedregosa et al., 2011).

Since we are interested in the LSTM learning storage information implicitly, it is useful to also add a benchmark where observed storage is explicitly provided to the LSTM model. As such we also compare a version of Model 1 (denoted as Model 1\*) that is provided inflow, storage and the day of the year rather only inflow and day of the year.

## Model Diagnostics

In the section, we focus on diagnosing the behavior of LSTM reservoir models, including assessing their physical interpretability, analyzing performance in large samples, and testing if learned policies are generalizable. We also explore factors that affect model performance, and how performance changes with time. The goal is to go beyond model selection to gain insight on how release decisions are made, and in what conditions the models perform better than others.

### Inspection of Cell States and Observed Storage

One of the main questions of this study is to test if a LSTM model applied to the reservoir operations problem is capable of learning storage representations in its memory cell states on its own without being given storage data explicitly, with a larger goal to determine if such a model is physically interpretable. This is inspired by the success of LSTM in capturing hydrologic states such as snowpack in its memory cells without requiring snowpack data (Kratzert, Herrnegger, et al., 2019). Similar to Kratzert et al., (2019), we study the correlation coefficients between the memory cell states compared to observed storage, as well as visualizing the timeseries of memory states with stronger relationships. The purpose of such an analysis is to uncover if the LSTM manages to learn storage information (therefore conserving mass) in an interpretable way; the ideal would be a model that aligns with physical understanding of reservoir storage and release decisions.

### Large Sample Individual Training

Next, we ask the question of how well the LSTM model and the benchmarks perform when trained individually to a large sample of reservoirs across the continental United States. Beyond understanding performance in a large sample, these results support analyses that explain conditions of where and how the models perform well. Here we use inflow and release records from the ResOps dataset (Steyaert et al., 2022), and filter for reservoirs where the record is at least 90% complete. For each reservoir selected, we conduct data processing as before, selecting 60% of the available record for training, 20% for validation, and the last 20% for testing.

### Model Performance vs. Degree of Regulation

Research has shown that LSTM rainfall-runoff models perform worse on managed basins (Ouyang et al., 2021). Consequently, we hypothesize that the degree of regulation in a reservoir adversely affects performance. Specifically, we compare performance from the large sample of individually trained reservoirs against the ratio of mean inflow to maximum storage (a proxy for capacity), which represents the degree of regulation. We then compute Pearson’s correlation coefficient between the LSTM performance and the degree of regulation. Statistical inference is done using randomization testing and Monte Carlo resampling, i.e. via permutation test, to determine the p-value against the null hypothesis of no correlation.

### Model Performance Over Time

We also study the trend of model performance over time to gain a more robust understanding of out-of-sample performance. While overfitting can lead to a downward shift between training set and out-of-sample performance, reservoir policies themselves may also change over time. Any difference between the out-of-sample and test distributions may cause a declining trend in performance. To understand this problem, we first train a new “initial” LSTM model for several example reservoirs with different degrees of regulation based on the first 30 years and validate on the next 10 years, and then analyze performance on rolling and sliding 20-year windows to capture how performance changes over time. Notably, this experiment is challenged by limited record lengths: the length of the initial training window is chosen so that the model can learn a reasonable representation of the operating policy while the moving window size is chosen to balance signal and noise. For this reason, we select four example reservoirs for this analysis with longer records available from the U.S. Bureau of Reclamation. We also analyze the entire prediction timeseries for the select example reservoirs with different degrees of regulation and plot predicted releases against observed releases for the entire record length. The goal is to further understand the influence of the degree of regulation, as well as to gain insight into the prediction behavior of the LSTM model.

### Large Sample Pooled Training

Finally, we test a top-down modeling approach, that is, learning a general model by training on all available data. In rainfall-runoff modeling, studies suggest that training on more than one basin outperforms training on individual basins, i.e. (Kratzert et al., 2024). However, hydrological models capture physical processes that are likely to be generalizable across basins, even if they are out-of-sample. Reservoirs, in contrast, may be operated according to unique operating policies, such that they do not represent independently and identically distributed samples from which to infer generalizable operating behavior. To access the ability of a simultaneously trained LSTM reservoir policy to generalize, we randomly select 80% of ResOps reservoirs (where at least 80% of the data record is complete), pool and train on them simultaneously, and test out-of-sample performance using the remaining 20% of reservoirs. This is not to be confused with data splitting in time, where we train, validate, and test on the same reservoir. Here, out-of-sample testing is done on held out reservoirs, not held out time.

We also compare the test performance after fine-tuning the pooled model on individual reservoirs. Finetuning in this context refers to calibrating a pre-trained model to a specific reservoir by running additional training iterations from data unique to the reservoir of interest. This is related to the concept of transfer learning in the machine learning literature in which a pre-trained model trained on a large dataset can be adapted to improve performance for a potentially different task on a smaller dataset (Tan et al., 2018). The idea of “knowledge transfer” has shown to be successful in a variety of domains including image recognition (Iorga & Neagoe, 2019) and natural language processing (Ruder et al., 2019). In this case, we can train and validate (on a 75% training and 25% validation split, respectively) using 5-30 year subsets of the complete data record for the held-out reservoirs as the finetuning data, and finally test using the last 20% of the complete record so that results between finetuning, individual training, and the pooled training model are comparable. Note that validation scores here are not directly comparable since they vary in length depending on the amount of fine-tuning data used.

# Results

## Model Selection and Comparison to Benchmarks

Table 1 summarizes performance results for Models 1-4 trained on Shasta Reservoir, as well as its linear and random forest benchmarks, and models where storage is explicitly provided.

Table 1. Train, validation, and test scores for Models 1-4 and benchmark ML models. Models with observed storage as input are denoted with \*.



From a model selection perspective, we are interested in comparing validation scores so that the test data is withheld from the model building process; the test data can later be used for further analysis of model behavior (such as the behavior of cell states) and provide a final estimate for out-of-sample performance. Between Models 1-4, Model 1 (validation ) and Model 2 (validation ) both perform reasonably well and are essentially the identical, however, we select Model 1 as the main LSTM architecture of interest going forward since Model 1 is more parsimonious and efficient to train. After running the hyperparameter tuning process with exhaustive grid search, we find that 1 LSTM layer, 30 LSTM hidden units, 15 feed-forward hidden units, and a dropout probability of 0.3 is optimal for Model 1. Figure 4 provides a visualization of the hyperparameter tuning results. Note that smaller architectures, particularly with 5 LSTM or feed-forward hidden units, and higher dropout (0.7) are associated with higher validation error. Beyond these cases, tuning results are more uniform.

A grid of lines and dots

Description automatically generated with medium confidence

Figure 4. Hyperparameter tuning of Model 1 on Shasta Reservoir

Model 1 out-performs the linear\* (validation ) and random forest\* (validation ) machine learning benchmarks even when observed storage is provided but underperforms itself when observed storage is provided in Model 1\* (validation ). These results highlight the advantage of LSTM to learn non-stationarities, long-term dependencies and non-linear temporal relationships compared to other architectures such as the linear or random forest models. However, while Model 1 may have learned to maintain some internal information about storage over time, the gap in performance compared to Model 1\* suggests that even if storage is learned, it may be used suboptimally. For example, the implied storages for Model 3 (validation ) match observed storages reasonably well and are directly inputted into the model, yet performance is lower than Model 1 or Model 1\* (Figure 5).

Implied storages for Model 4 (validation ) also match observed storages reasonably well, however, Model 4 performs almost as well as Model 1 but with fewer parameters since Model 4 does not need to parameterize gating behavior. Specifically, after tuning, Model 1 has 4561 trainable parameters and Model 4 has 2751, or 39.7% fewer parameters. This result highlights the potential for physics-informed machine learning to promote model parsimony and interpretability (De La Fuente et al., 2024).

A group of graphs showing different types of graphs

Description automatically generated with medium confidence

Figure 5. Implied storage states plotted against observed storage for Model 3 and 4

Across all models, we observe severe declines in performance in the test period compared to the validation or training periods. For example, Models 1, 2, and 4 each show declines of 0.26 in between the validation and testing periods. Interestingly, some storage-driven models including Model 3, Model 1\*, and Random Forest\* have much lower declines in and perform better in the test period, which suggest that the ability to base release decisions on storage improves the robustness of the model under changes in policy.

## Comparison of Cell States and Observed Storage

In this section, we compare the memory cell states of Model 1 trained on Shasta reservoir with observed storages to see if the model learns physically interpretable states internally. In the test data, six different cell states have correlation coefficients with observed storage that are greater than or equal to in absolute value to 0.40, an arbitrarily chosen threshold, with values of 0.52, 0.50, 0.48, 0.46, 0.43, and -0.45, respectively. These states are scaled and plotted against observed storage for the testing dataset in Figure 6. We observe that in many cases, the analysis reveals memory cell states contain information about the seasonality of storages, but not the values of storages themselves. It is possible that the storage values are represented through complex interactions of multiple different states, though unfortunately such a situation would not be directly interpretable. These results are consistent with the previous finding that performance is lower when storage is withheld from the LSTM.

A group of graphs showing different types of cell phones

Description automatically generated with medium confidence

Figure 6. Comparison of cell states with high correlation (|| > 0.4) to observed storage.

## Performance for Many Reservoirs

Next, we extend our results to evaluate Model 1 performance when fitted individually to large sample of ResOps reservoirs (n=116). Figure 7 shows the distribution of scores across the training, validation, and test sets. We find that in both validation and testing, Model 1 (validation median = 0.618) performs similarly to a random forest with observed storage (validation median = 0.612), while both significantly outperform the linear models (validation median = 0.376). Model 1\*, with observed storage inputs, has the highest performance (validation median = 0.712). These results are consistent with the model selection findings for Shasta reservoir. We also observe large spreads of scores for each model, with high and low extreme values. For example, the interquartile range for Model 1 in validation is 0.357 with a maximum score of 0.978 and a minimum score of -1.656.

A chart of different colored squares

Description automatically generated

Figure 7. Distribution of scores for Model 1 and selected benchmarks for n=116 reservoirs. Models with observed storage as input are denoted with \*.

Figure 8 shows with respect to geographic location, and we find no apparent spatial patterns, Location, which represents climate and hydrologic factors, alone do not appear to be a strong indicator of model performance.

A map of the united states

Description automatically generated

Figure 8. Map of train, validation, and test scores for individually trained LSTMs

## Degree of Regulation and Model Performance

Previously, we found large variances in performance across a large sample of reservoirs (Figure 7). Here, we investigate whether the degree of regulation adversely affects model performance. A lower degree of regulation may indicate shorter lag times between inflow and release, i.e., release predictions are more directly sensitive to inflow and therefore more accurate. Figure 9 shows these results, plotting performances against the log ratio between mean inflow and max storage (a higher value of this ratio indicates a lower degree of regulation). We find that the Pearson correlation between scores and the degree of regulation to be 0.6, 0.59, and 0.49 for the training, validation, and test scores, respectively. Randomization testing reveals that the correlation coefficients are significant at the 0.05 level, rejecting the null hypothesis of no correlation. These results provide strong evidence that model performance is adversely associated with increased degree of regulation. We also find no clear connection between geographic location and model performance.

A graph of a model performance

Description automatically generated with medium confidence

Figure 9. Train, validation, and test plotted against the log mean-inflow-max-storage ratio for individually trained LSTMs

Four specific sites, Folsom (FOL), Shasta (SHA), Trinity (TRI), and New Melones (NM), are selected from the points in Figure 9 to represent different degrees of regulation, as measured by the log mean-inflow-max-storage ratio. Folsom represents the lowest degree of regulation, while Trinity and New Melones have higher degrees of regulation. Shasta reservoir falls in between. An additional factor in this choice of reservoirs is the availability of longer inflow records from the U.S. Bureau of Reclamation dating back to the construction of the reservoir, providing several additional decades prior to the ResOpsUS dataset. Figure 10 plots the predicted and observed releases for these four selected reservoirs using LSTM Model 1.

Both Shasta and Folsom capture peak releases reasonably well, however, Shasta is more prone to false-positive peaks, which can be resolved by inputting observed storage (see Supplementary Materials Figure S1 for Model 1\* timeseries). This suggests that adding observed storage allows the Shasta model to learn better thresholding behavior and improve performance when predicting larger lags between inflow and release. Importantly, the model was unable to learn this optimally on its own. Folsom has a lower degree of regulation making it more sensitive to inflow patterns directly, which corresponds to higher performance. In contrast, New Melones and Trinity reservoirs have much lower performance corresponding to their high degree of regulation. While both learn reasonable seasonal releases, the models have largely ignored peak releases especially for Trinity reservoir. Adding observed storage to the model does not alleviate this behavior, which highlights peak releases that are heavily regulated. In all, these results complement the finding that the degree of regulation adversely affects model performance.

A graph of a graph

Description automatically generated with medium confidence

Figure 10. Timeseries plots for predicted and observed releases for Shasta, Folsom, New Melones, and Trinity using Model 1

## Performance Over Time

The drop in performance between the train and test period may be explained partly by changes in the reservoir operating policy during that time. We investigate this question using an “initial model” trained on the first 30 years of record and validated on the next 10 years for the four selected reservoirs (Folsom, Shasta, New Melones, and Trinity). Figure 11 shows performance in 20-year rolling and sliding windows for these selected reservoirs. The initial model for Folsom shows an initial drop in performance apparent in the rolling windows, but then stabilizes. This behavior is expected with some degree of overfitting; although note that the full Folsom model is not prone to overfitting from Figure 10. In contrast, performance for Shasta continues to decline and does not stabilize. This is also consistent with declining train, validation, and test for the full Shasta model in Figure 10. In the sliding window plots, there is a clear decrease in slope in the out-of-sample region. This behavior is consistent with changing distributions between testing windows and the initial training window – supporting the hypothesis that changing operating policies are resulting in declining performance. The trends for New Melones and Trinity are more difficult to interpret. New Melones has a much shorter record length, so it is difficult to distinguish between overfitting and shifting operating policy. The initial model for Trinity appears to be underfit, with very poor training performance but higher out-of-sample performance.

A graph of a graph

Description automatically generated with medium confidence

Figure 11. 20-year rolling and sliding performance for initial models trained for Shasta, Folsom, New Melones, and Trinity

Shasta, New Melones, and Trinity have higher degrees of regulation and may be more sensitive to changes in policy, while Folsom releases are more sensitive to inflows directly which make them less affected by changes to the operating policy. This conjecture is supported by a weak negative correlation () between the difference in train and test and the log mean-inflow-max-storage ratio (Figure 12). The result is statistically significant (, indicating that a higher degree of regulation corresponds to larger declines between the train and test performance.

A red line with blue dots and numbers

Description automatically generated

Figure 12. Decline in in training and test versus log mean-inflow-max-storage ratio.

Overall, we find that declines in performance over time due to policy changes are location specific and related to the degree of regulation of the specific site. We provide an approach to analyze this problem, provided that there is long enough of a data record to track performance over time meaningfully.

## Pooled Training and Finetuning

After training models to reservoirs individually, we answer the question of whether stronger results can be achieved by training on a pool of reservoirs simultaneously, and if transfer learning or finetuning can be leveraged to further improve performance. Figure 13 compares scores on the last 20% of record for out-of-sample (OOS) reservoirs, comparing individually trained models, the pooled model, as well as finetuning the pooled model with 5-30 years of data. Recall that the training and validation periods for each fine-tuning process do not align, although we can compare performance on the same testing period. We find that the pooled model (median score of 0.343) performs significantly worse than training individually (median score of 0.567). This result confirms that given the feature space, we are unable to find a strong reservoir policy that generalizes across reservoirs. Introducing finetuning improves performance, although it does not improve on the individually-trained models. This suggests that finetuning on the pooled model provides little additional knowledge compared to individual training. It may be possible to improve this result by pooling according to other reservoir characteristics, such as the operating purpose or hydrologic region. However, this experiment suggests that the generalization ability of LSTMs observed in rainfall-runoff modeling may not extend to models of reservoir release policies, as these tend to be location-specific.

A graph of a graph

Description automatically generated with medium confidence

Figure 13.

# Discussion and Conclusion

Think about key takeaways from the results figures, and how they fit with previous literature cited in the introduction.

Some ideas:

* LSTMs show the ability to model reservoir release policies as well or better than other data-driven methods. However, this does not appear to be due to their inherent ability to accumulate storage over time; cell states show only modest correlations with observed storage (and they do not conserve mass).
* The accuracy does not appear to relate to the reservoir location (climate/hydrologic factors). Instead, it is strongly related to the degree of regulation. The models are more accurate for reservoirs with a lower degree of regulation, where the release is more directly related to the inflow and does not depend on longer-timescale accumulation.
* Pooled training does not show an improvement on out-of-sample reservoirs, especially in the absence of fine-tuning. This suggests that the transferability of trained LSTMs for rainfall-runoff modeling may not hold for modeling reservoir release policies.
* The decline in performance between the training and test periods is in part due to overfitting, but may also be influenced by changes in the operating policies over time. This is difficult to resolve because the long data records are needed to support training.

Possible applications of this approach (or data-driven reservoir models in general): more efficient approximations to embed in hydrologic models? How would this compare to ad hoc models used by agencies? See refs in proposal about diagnostic assessment of national water model (has issues with reservoir representation).

Limitations/future work

# Data Availability Statement

All code corresponding to methods and figure generation can be found in the repository: https://github.com/Matt2371/DL-reservoir-modeling.

# Acknowledgments

Remind me to fill in later with funding info

# References

Hochreiter, S., & Urgen Schmidhuber, J. ¨. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/https://doi.org/10.1162/neco.1997.9.8.1735

Hoedt, P.-J., Kratzert, F., Klotz, D., Halmich, C., Holzleitner, M., Nearing, G., et al. (2021). MC-LSTM: Mass-Conserving LSTM. *Proceedings of the 38th International Conference on Machine Learning*, *139*, 4275–4286. Retrieved from http://arxiv.org/abs/2101.05186

Iorga, C., & Neagoe, V.-E. (2019). A Deep CNN Approach with Transfer Learning for Image Recognition. In *2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)* (pp. 1–6). IEEE. https://doi.org/10.1109/ECAI46879.2019.9042173

Jordan, I. D., Sokół, P. A., & Park, I. M. (2021). Gated Recurrent Units Viewed Through the Lens of Continuous Time Dynamical Systems. *Frontiers in Computational Neuroscience*, *15*. https://doi.org/10.3389/fncom.2021.678158

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)* (pp. 1–15). Retrieved from http://arxiv.org/abs/1412.6980

Kratzert, F., Herrnegger, M., Klotz, D., Hochreiter, S., & Klambauer, G. (2019). NeuralHydrology – Interpreting LSTMs in Hydrology. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Vol. 11700, pp. 347–362). https://doi.org/10.1007/978-3-030-28954-6\_19

Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., & Nearing, G. (2019). Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. *Hydrology and Earth System Sciences*, *23*(12), 5089–5110. https://doi.org/10.5194/hess-23-5089-2019

Kratzert, F., Gauch, M., Klotz, D., & Nearing, G. (2024). HESS Opinions: Never train an LSTM on a single basin. *Hydrol. Earth Syst. Sci. Discuss. [Preprint]*, 1–19. https://doi.org/10.5194/hess-2023-275

De La Fuente, L. A., Ehsani, M. R., Gupta, H. V., & Condon, L. E. (2024). Toward interpretable LSTM-based modeling of hydrological systems. *Hydrology and Earth System Sciences*, *28*(4), 945–971. https://doi.org/10.5194/hess-28-945-2024

Li, M., Soltanolkotabi, M., & Oymak, S. (2019). Gradient Descent with Early Stopping is Provably Robust to Label Noise for Overparameterized Neural Networks. In *Proceedings of Machine Learning Research* (Vol. 108, pp. 4313–4324). Retrieved from https://proceedings.mlr.press/v108/li20j.html

Ouyang, W., Lawson, K., Feng, D., Ye, L., Zhang, C., & Shen, C. (2021). Continental-scale streamflow modeling of basins with reservoirs: Towards a coherent deep-learning-based strategy. *Journal of Hydrology*, *599*. https://doi.org/10.1016/j.jhydrol.2021.126455

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)* (pp. 1–12). Retrieved from http://arxiv.org/abs/1912.01703

Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., et al. (2011). *Scikit-learn: Machine Learning in Python*. *Journal of Machine Learning Research* (Vol. 12). Retrieved from http://scikit-learn.sourceforge.net.

Ruder, S., Peters, M. E., Swayamdipta, S., & Wolf, T. (2019). Transfer Learning in Natural Language Processing. In *Proceedings of the 2019 Conference of the North* (pp. 15–18). Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-5004

Steyaert, J. C., Condon, L. E., W.D. Turner, S., & Voisin, N. (2022). ResOpsUS, a dataset of historical reservoir operations in the contiguous United States. *Scientific Data*, *9*(1), 34. https://doi.org/10.1038/s41597-022-01134-7

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A survey on deep transfer learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 11141 LNCS, pp. 270–279). Springer Verlag. https://doi.org/10.1007/978-3-030-01424-7\_27

Yu Wang. (2017). A new concept using LSTM Neural Networks for dynamic system identification. In *2017 American Control Conference (ACC)* (pp. 5324–5329). IEEE. https://doi.org/10.23919/ACC.2017.7963782

# Supplementary Materials

A graph of data on a white background

Description automatically generated with medium confidence

Figure S1. Timeseries plots for predicted and observed releases for Shasta, Folsom, New Melones, and Trinity using Model 1\*