Iowa Initiative for Artificial Intelligence
Final Report

Project title:	Machine-learning-based flood forecasting and control
Principal Investigator:	Shaoping Xiao and Ricardo Mantilla
Prepared by (IIAI):	Avinash Mudireddy
Other investigators:	Faruk Gurbuz
Date:	03/20/2021

Were specific aims fulfilled:	YES
Readiness for extramural proposal?	YES
If yes … Planned submission date	Submitted June 2021
Funding agency	NSF
Grant mechanism	Research grant
If no … Why not? What went wrong?	N/A

Brief summary of accomplished results:

Research report:

Aims (provided by PI):

The aim of this research is to develop a neural-network-based predictive model for flood forecasts. The specific aims include: 1) designing proper recurrent neural network architectures with the state-of-art techniques in the AI and ML community; 2) Comparing the developed models with the existing work in literature for flood forecast; 3) Studying the effects of partially selected input features on flood forecast for scalability reduction.

The larger goal is that Flood control reservoirs are multimillion-dollar investments that typically benefit only a few locations downstream from the reservoir. So, we ask: Can we extend the idea/success of a flood control system to smaller scales? Can we construct a greater number of small reservoirs instead of one big reservoir? How many reservoirs would be needed to achieve comparable flood risk reduction levels? Where should they be placed?

However, planning of such small reservoirs needs an estimation water outflow at various cross-sections of the river network in the catchment. Currently, those outflows are being calculated with physics based mathematical models(mass transport and mass conservation equations) which are computationally intensive to compute at real-time.

Data:

The Turkey river watershed of east Iowa region is consider for our research as a more realistic river network. Across this river network, we chose 5 unique links for USGS gauges to study the cross sectional streamflow at their outlet. The links are identified as 483619, 434365, 434478, 39971 and 434514. The network structure of these links is visible in the Figure 2 below.
Currently, all the data for the models is simulated using software, ASYNCH. In total, we generated 5000 events.
Event description:

- Each event is 481 hours long in duration.
- Precipitation at all the links in the river networks for every 1 hour is simulated.
- Streamflow at all the links' outlets in river network for every 1 hour is generated.

Our deep learning models use the information of Precipitation and Streamflow at each link to predict the Streamflow for future 24 hours.
In total our dataset has 2,405,000 datapoints corresponding to 5000 events. We experimented with various Train and Test splits. But finally got to 80% (4000 events) and 20% (1000 events) respectively. Moreover, the 80% of data used for training used split into 80-20 ratio with shuffling as a part of training and validation respectively for each epoch.

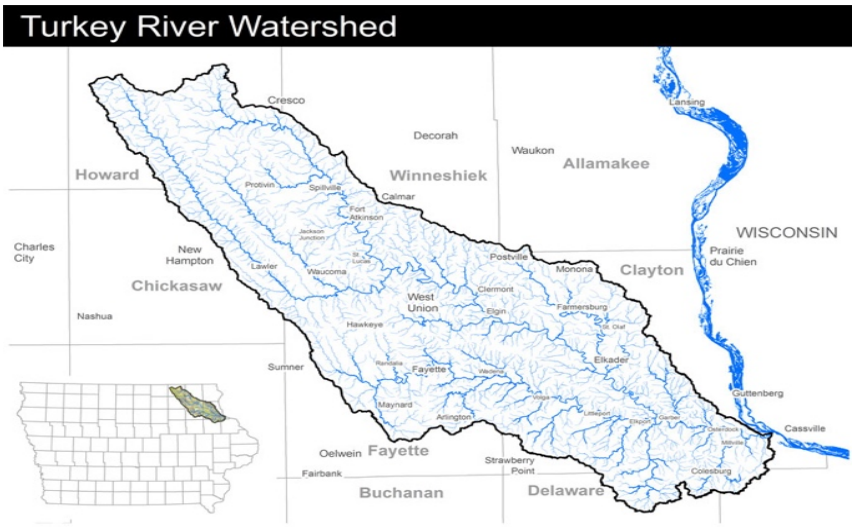


Figure 1. Turkey river watershed.

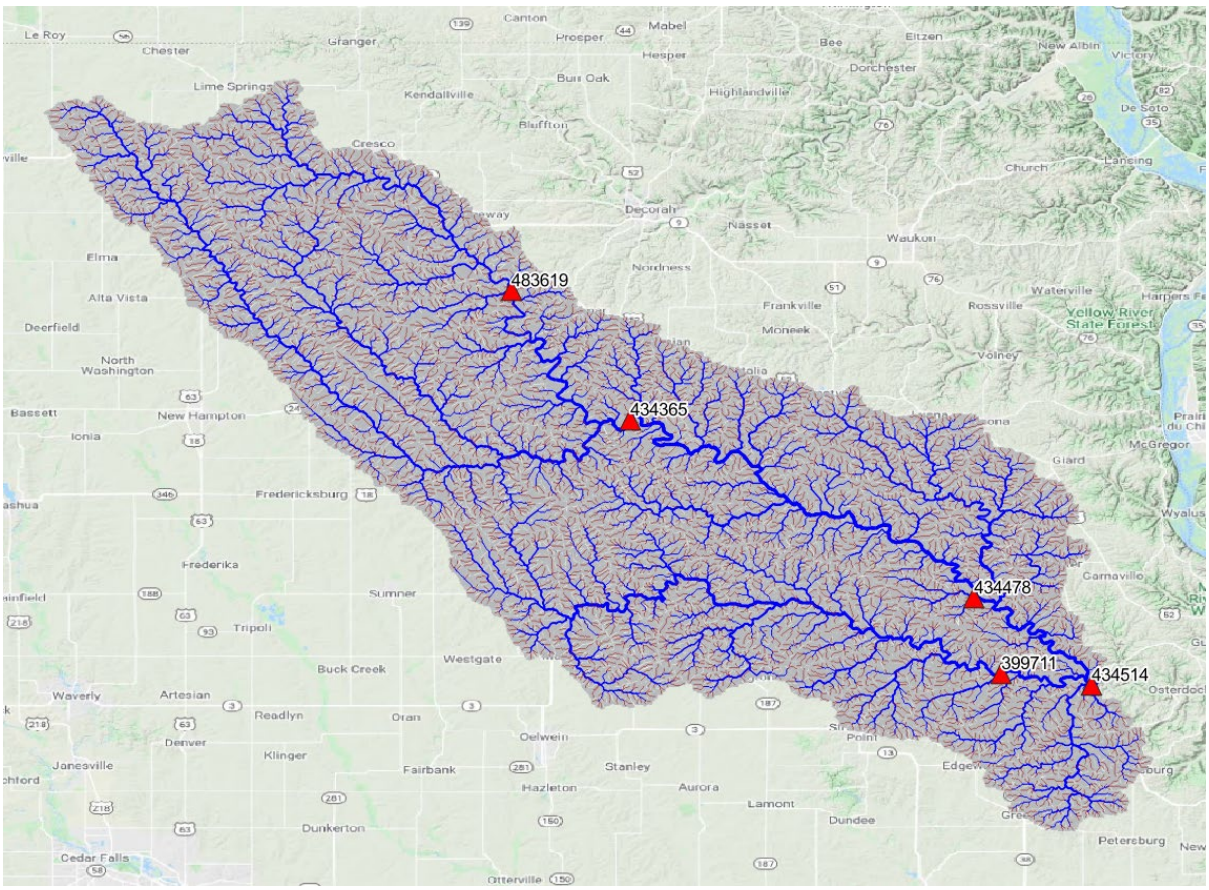


Figure 2. River network structure of preferred locations on Turkey river watershed.

AI/ML Approach:

In deep learning LSTMs(Long short-term memory) are an adapted versions of RNNs proposed by Hochreiter and Schmidhuber to address the problem of exploding/vanishing gradients that RNNs face. These networks are useful to preserve information and learn patterns in them over long periods of times using their unique network design. A basic LSTM unit is composed of a cell, gates (input, output, and forget gate) to control how the information flows in the different layers. Each LSTM cell constitutes,

$$it = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

$$\widetilde{C_t} = tanh(W_c x_t + U_c h_{t-1} + b_c)$$

$$C_t = f_t \otimes C_{t-1} + i_{t-1} \otimes \widetilde{C_t}$$

$$h_t = o_t \otimes tanh(C_{t-1})$$

where,

- $x_t \in \mathcal{R}^d$: input vector to the LSTM unit
- $f_t \in \mathcal{R}^h$: forget gate's activation vector
- $i_t \in \mathcal{R}^h$: input/update gate's activation vector
- $o_t \in \mathcal{R}^h$: output gate's activation vector
- $h_t \in \mathcal{R}^h$: hidden/output state vector
- $\widetilde{C_t} \in \mathcal{R}^h$: cell input activation vector
- $C_t \in \mathcal{R}^h$: cell state vector
- $W \in \mathcal{R}^{h \times d}$, $U \in \mathcal{R}^{h \times h}$ and $b \in \mathcal{R}^h$: weight matrices and bias vector parameters which need to be learned during training

where the superscripts d and h refer to the number of input features and number of hidden units, respectively. σ corresponds to the sigmoid function and $\otimes$ is the Hadamard product.

We also used GRU (gated recurrent units) which are similar to LSTM but with fewer parameters. Each cell constitutes,

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$\widetilde{h_t} = tanh(W_h x_t + U_h(r_t \otimes h_{t-1}) + b_c)$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \widetilde{C_t}$$

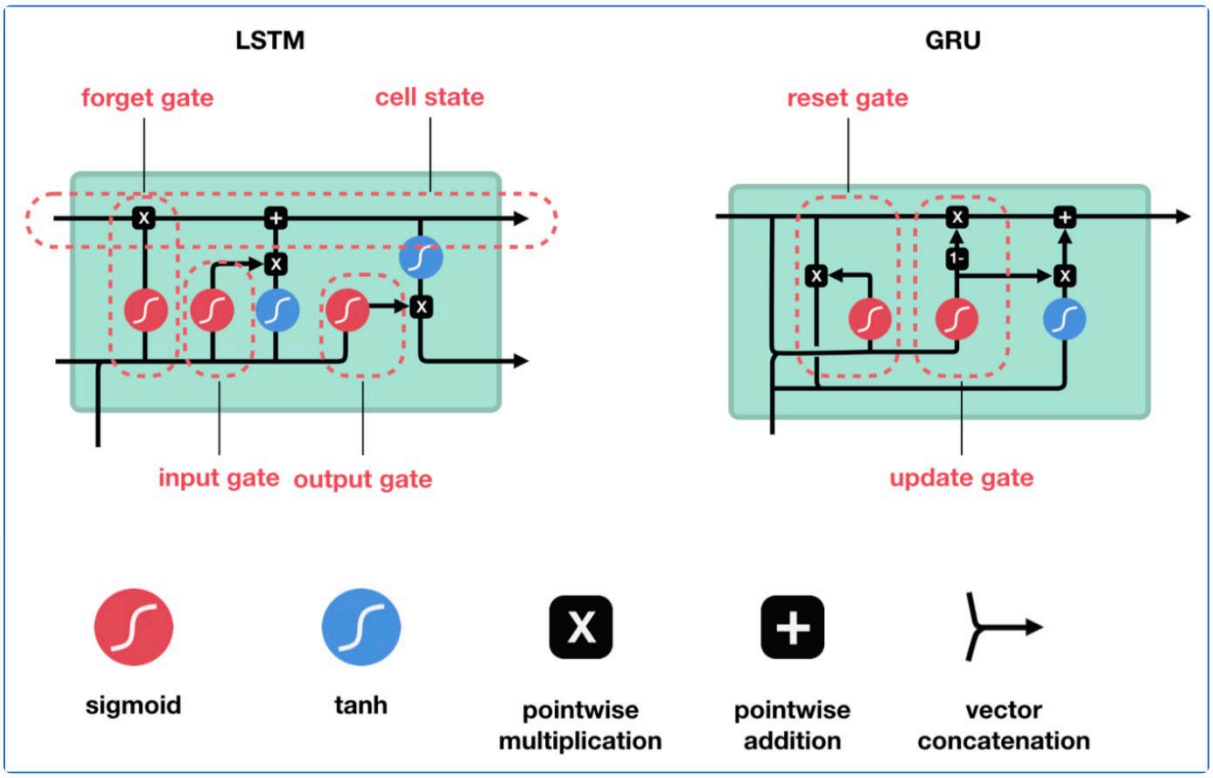
where,

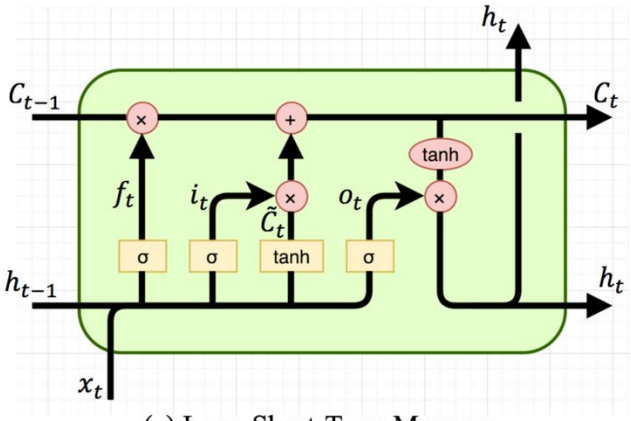
- $x_t \in \mathcal{R}^d$: input vector to the LSTM unit
- $r_t \in \mathcal{R}^h$: reset gate vector
- $z_t \in \mathcal{R}^h$: update gate vector
- $h_t \in \mathcal{R}^h$: hidden/output state vector
- $\widetilde{h_t} \in \mathcal{R}^h$: candidate activation vector

- $W \in \mathcal{R}^{h \times d}$, $U \in \mathcal{R}^{h \times h}$ and $b \in \mathcal{R}^{h}$: weight matrices and bias vector parameters which need to be learned during training

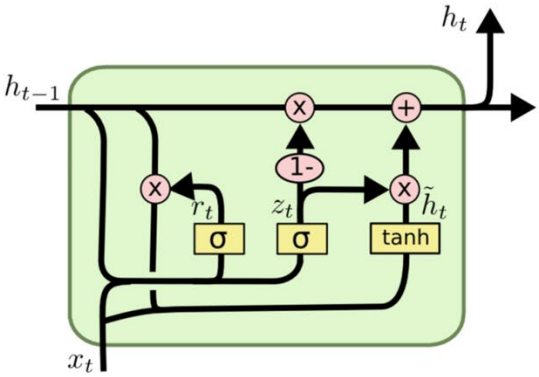
where the superscripts d and h refer to the number of input features and number of hidden units, respectively. σ corresponds to the sigmoid function and $\otimes$ is the Hadamard product.

Both the cells are shown below:
Image source: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
https://i.stack.imgur.com/YzGyL.png





(a) Long Short-Term Memory



(b) Gated Recurrent Unit

Though we used LSTM and GRUs as our basic cells, the we experimented with 2 kinds of architectures.

1. Encoder -Decoder architecture
2. Encoder-Decoder with Luoung Attention architectures

Encoder decoder architectures are designed to address the sequence-to-sequence(seq2seq) prediction problems. In seq2seq, prediction involves forecasting the next values of a sequence for one or more of input sequences. This suits out flood prediction problem as we want to predict future outlet streamflow sequence, given precipitation and/or past streamflow sequences.

- The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.
- The decoder steps through the output time steps while reading from the context vector to generate the output sequence.

However, in the usual encoder decoder model, the decoder tries to learn the context from the last cell. Attention model differs here. It provides a richer context through attention mechanism where decoder learns where to pay attention on the context vector in a weighted manner.

A sample illustration is shown below.

Image reference: https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3
Encoder decoder LSTM

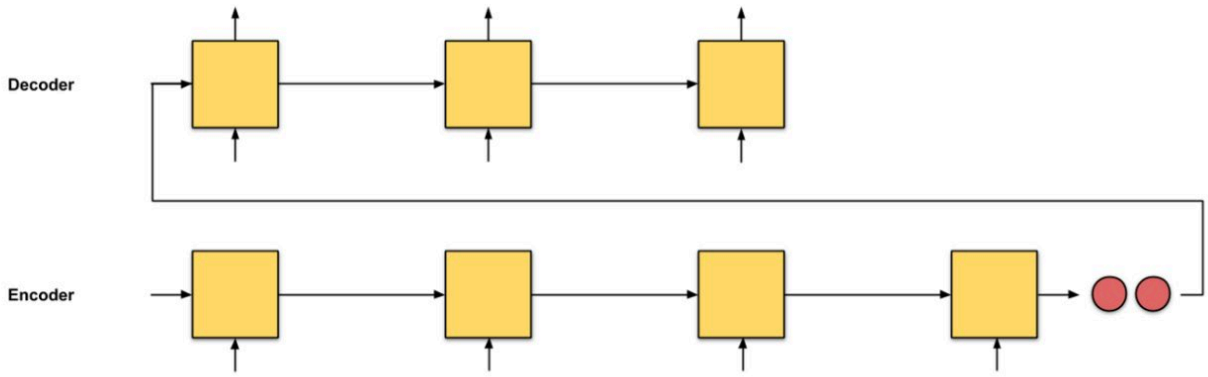
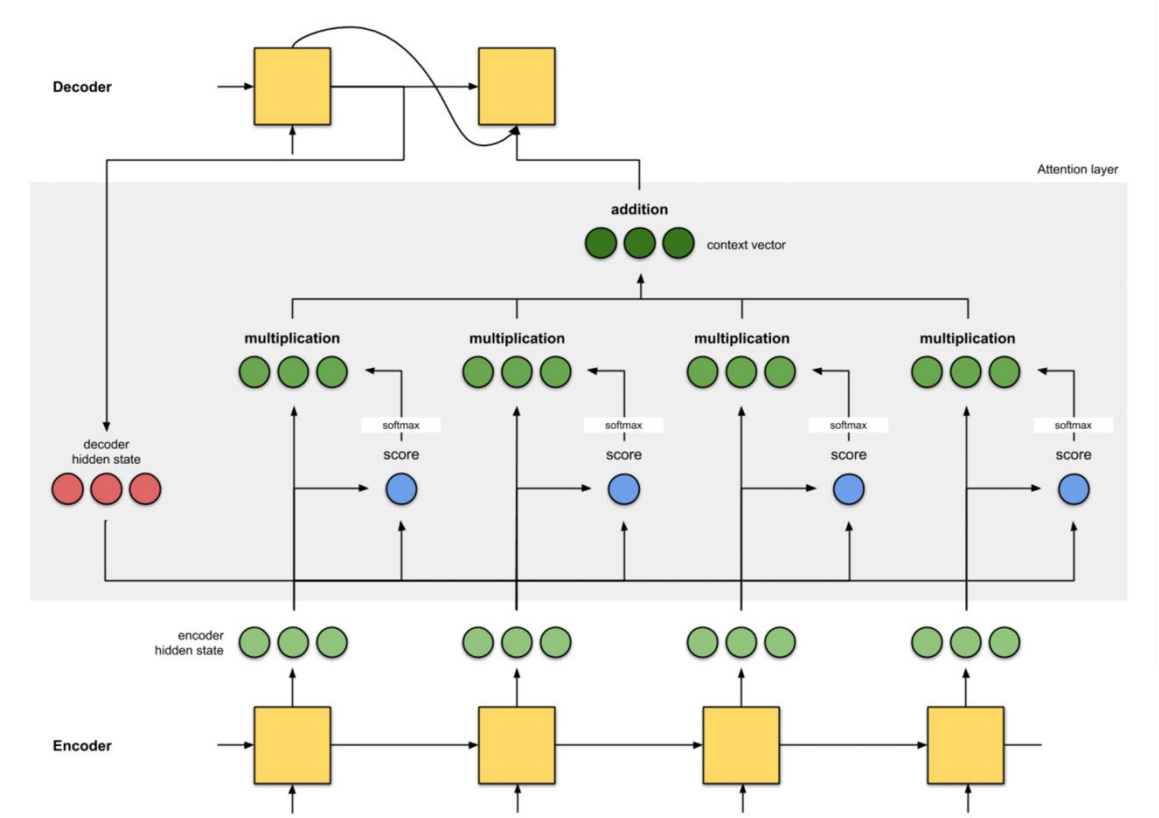


Fig. 0.1: seq2seq with an input sequence of length 4

Arrows in the bottom indicate the input sequences. Arrows in the top indicate output sequences. Each box is a LSTM/GRU cell. Circles represent cell states of context vector. Horizontal arrows indicate information flow.

Encoder decoder with attention



The details on the exact architectures chosen for each model for experiments is described in the next section.

Experimental methods, validation approach:

Data Size:

The first step in our experimentation is to get fixated on the train-test proportions of the dataset. In the real world, acquiring huge amounts of past data is not possible as the collection of flood data has only begun from past few decades. However, for deep learning we may require huge volume of data. Hence, we used a software called ASYNCH to generate simulated data. The description of dataset is discussed in the section "Data" above.

This kind of data generation is a feasible approach for new watersheds too as the simulation is dependent on mathematical models. The idea is to train the model on simulated data and use transfer learning on real world data at a later stage.

In total we have 2,405,000 datapoints corresponding to 5,000 events. We have experimented with various data sizes with an intension to work with smaller data sets closer to what we have in real world. Mostly to strike a balance between, real world small volume datasets and huge volumes of simulated data sets.

We have set up our experiments to try with following data sizes:
$1/10^{th}$, $1/5^{th}$, $1/4^{th}$, $1/3^{rd}$, $1/2^{nd}$ and full sizes of 2,405,000 datapoints with every dataset being 80-20 split for train-test.

The observations that we had were:

- Smaller the data size, higher the training loss and worse the model predictions on validation dataset.
- We have observed the same trend until we hit the full data set

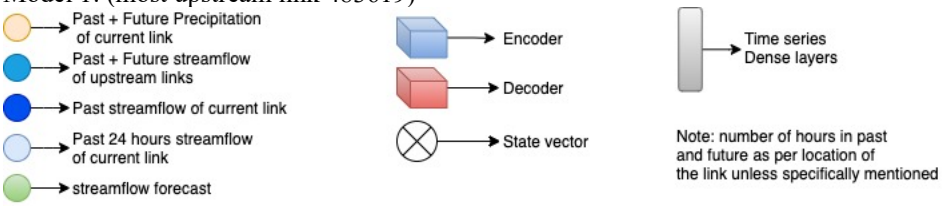
- This is due to the fact that training dataset doesn't have enough rainfall patterns at lower data sizes and hence the performance on validation datasets was also poor.
- We observed that, only at full dataset size there are enough variations that are representative of rainfall patterns.
- The same observations are consistent with statistical patterns of hydrographs (Professor Ricardo can explain well)

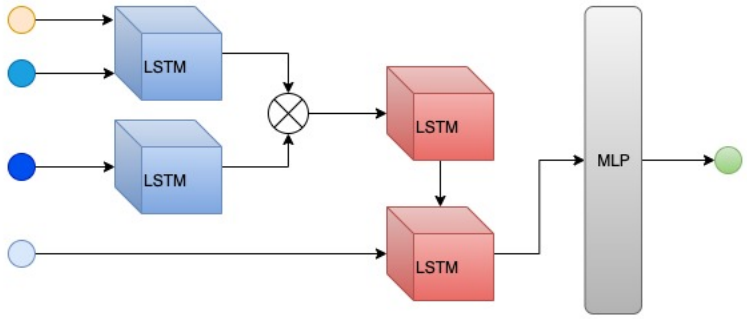
Therefore we decided to go ahead with full dataset for further investigation.

<u>Models of choice:</u>

In total, we developed 7 different models for the upper most link's catchment but only went ahead with 5 models for each of downstream links.

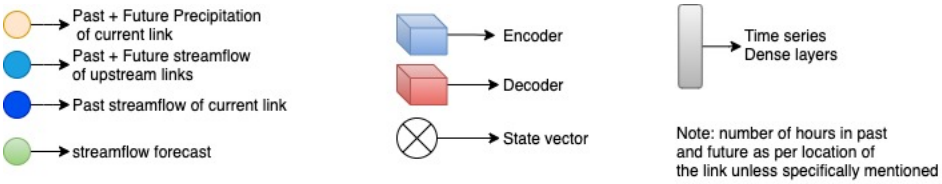
Model 1: (most upstream link 483619)

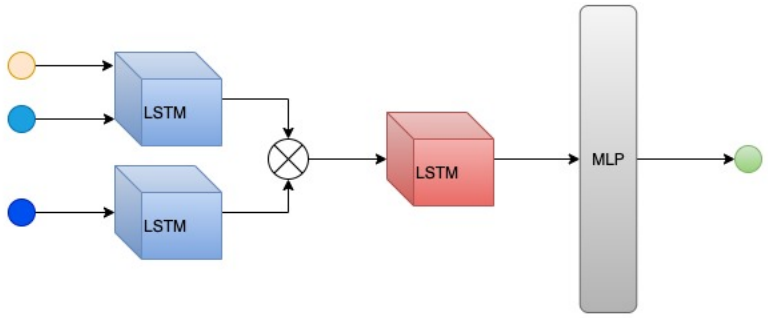




LSTM encoder - decoder with additional inputs to decoder

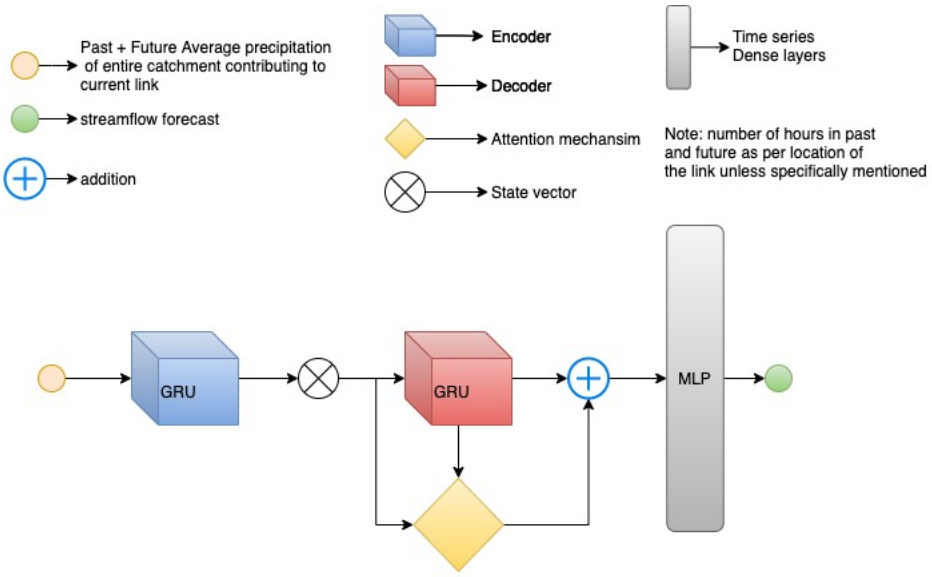
Model 2:





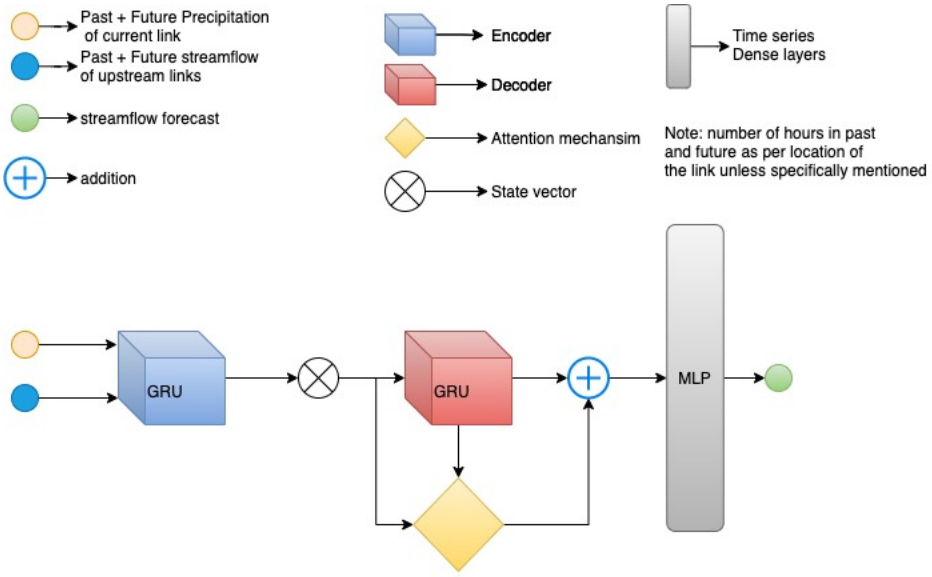
LSTM encoder - decoder

Model 3: Rainfall only



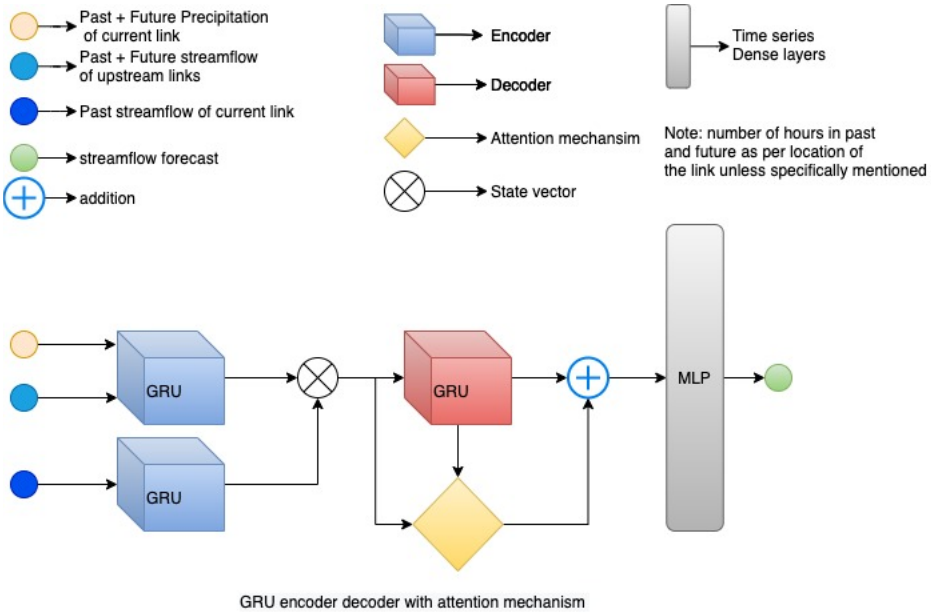
GRU encoder decoder with attention mechanism (Only average rainfall as inputs)

Model 4: Rainfall and upstream



GRU encoder decoder with attention mechanism (rainfall and upstream streamflow)

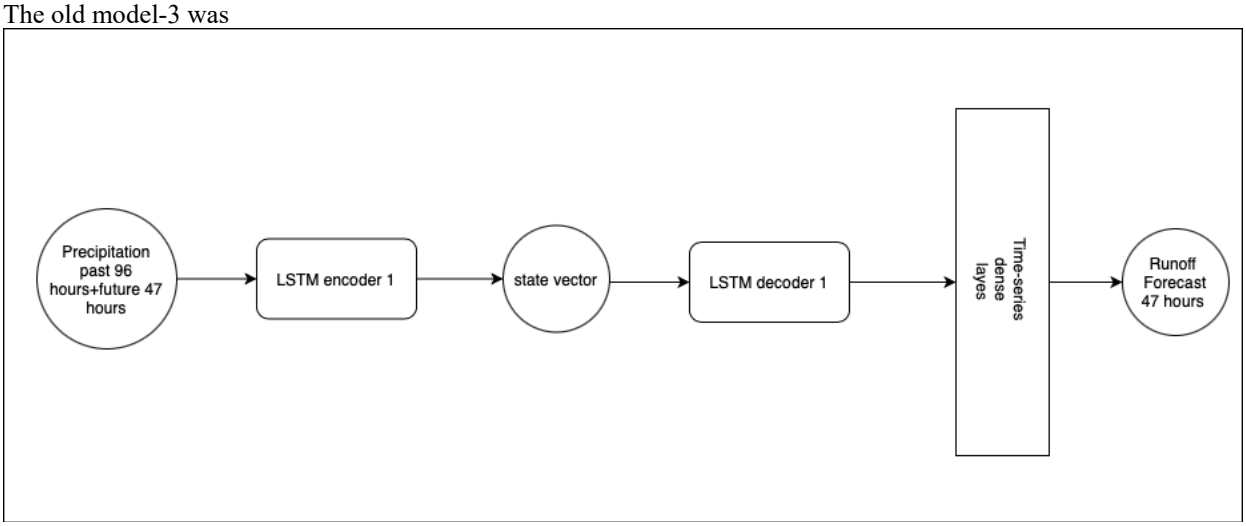
Model 5: Attention model



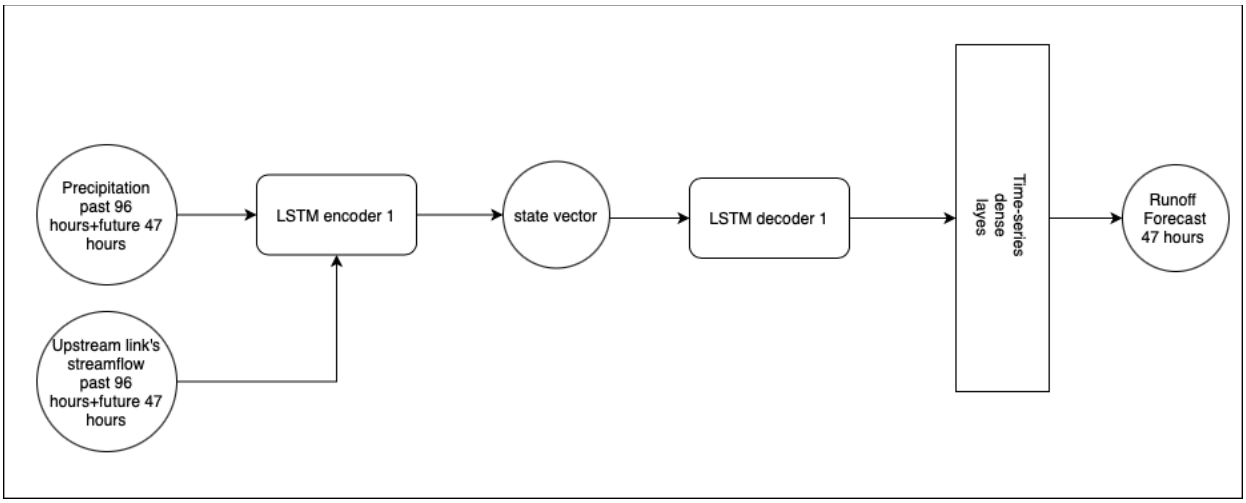
GRU encoder decoder with attention mechanism

Similarly the model structure is same for rest of the links. Only the Runoff forecast hours for input and output change according to the width of the average hydrograph for that particular link. We chose the 47 hours for the first link as mentioned above by simply measuring the average base of the graph for 10 hydrographs for the link 483619 which turned out to be 95 hours. We took the length up to the peak(Approximately half of the base – 47 hours) as out input and output future forecast length.
The average base lengths for the links [483619, 434365, 434478, 39971, 434514] are [95,125, 190, 150, 210] respectively. Hence the future forecast lengths for the links are [47,62, 95, 75, 105] respectively.

Initially, Model 3 and model 4 are built with LSTM encoder decoder networks instead of Attention-GRU networks. The old model-3 was



And old model 4 was



However, they were discarded due to much lower performance when compared to Attention GRU networks.

Model Evaluation Methodology:

We used Kling-Gupta Efficiency (KGE), a widely used metric for hydrological modelling, to evaluate the performance of the models. The equation for KGE is as follows:

$$KGE = 1 - \sqrt{(r-1)^2 + (\beta-1)^2 + (\gamma-1)^2}$$

where r, $\beta = \frac{\mu_S}{\mu_O}$ and $\gamma = \frac{\sigma_S}{\sigma_O}$ denote the linear correlation, mean ratio, and deviaton ratio of the two time series compared, respectively. KGE ranges between 1 and $-\infty$ and higher values indicates better performance. For calculating a single KGE for a particular lead or lag time, we calculated the KGE values of the hydrographs resulting from different rainfall events, separately. Then, we calculated the mean and median KGEs with respect to the lead times. We present an example of the model outputs for 24-hour lag time in Figure 1.

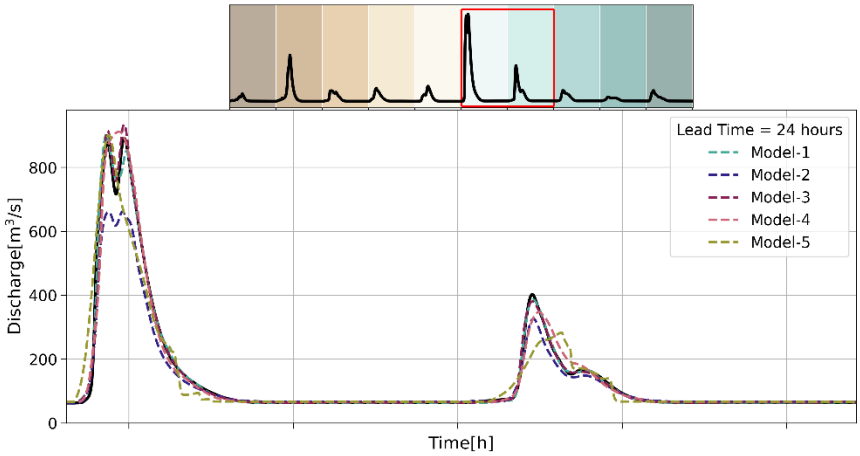


Figure 1. The top plot shows the hydrograph resulting from 10 events out of 1000 in the validation set at the outlet. (Colorful windows represent different events.) the bottom plot shows hydrograph of the two events along with the 24-hour ahead predictions made by the five models.

As the concentration time varies among different watersheds, we used a dimensionless time axis to be able to fairly test the prediction skills of the models across scales. We obtained the dimensionless time axis by dividing lead (i.e., lag) times with a reference time that is based on the time of concentration. The reference time can be calculated as:

$$T_{reference} = \left(\frac{L}{v}\right)^{n}$$

where L is the longest channel length to the point of interest, v is the average flow velocity in the channel and n is the exponent. The parameter of L significantly changes from location to location. For the smallest sub catchment, for example, it is about 56 km while the distance between the most remote point in the watershed and the outlet is about 200 km. We used a constant velocity, 0.75 m/s. We tuned the exponent, n, relying on the notion of river flow persistence and set it as 0.4.

Results:

We present mean and median KGE values for each model as well as persistence in Figure 1. The figure indicate that predictability of streamflow decreases as the fraction of lead time over reference time increases. Simple or local persistence outperform all the five models to a certain point, as expected. However, the models consistently provide KGE values greater than 0.6 even for greater lag times. Specifically, Model-1 and Model-2 showed similar performance with a slight difference. The performance of Model-1 and Model-2 dominate simple persistence significantly for dimensionless time values greater than 1 and 2, respectively. We also observe the effect of removing some input features from Model-2 in the figure. Model-5, for example, has the least information as input (i.e., rainfall) and it underperform all the models for a wide range of fractions of lead time over reference time. Interestingly, Model-5 shows a better performance than Model-3 and Model-4 when the fraction of lead time over reference time is greater than 9. Model-3 and Model-4, on the other hand, show similar performance.

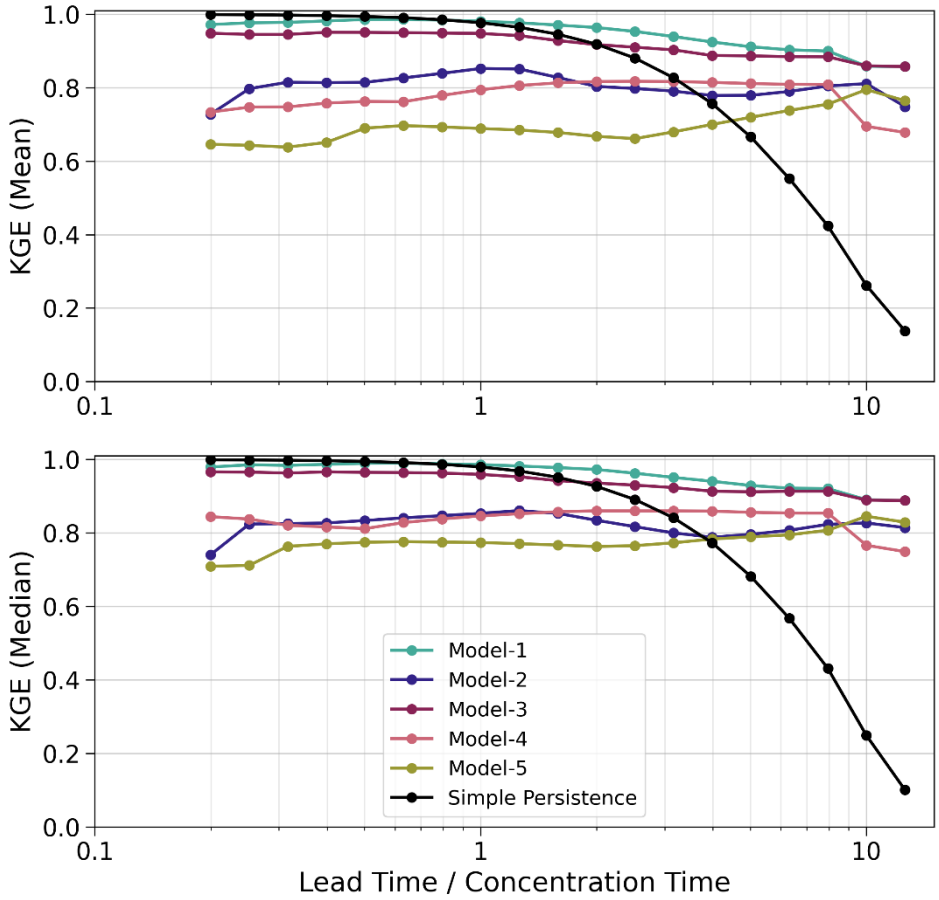


Figure 2. Mean and median KGE values for all five locations corresponding to different models.

We further tested the performance of the models assuming that the future rainfall is zero. As shown in Figure 3, KGE values for all the models decay rapidly and eventually fall under the persistence curve when the fraction of lead time over reference time is about 12. However, all the models outperform simple persistence for some dimensionless time values. Model-1 and Model-2 show a better performance than simple persistence when the fraction of lead time over reference time is between 1-12 and 2-12, respectively.

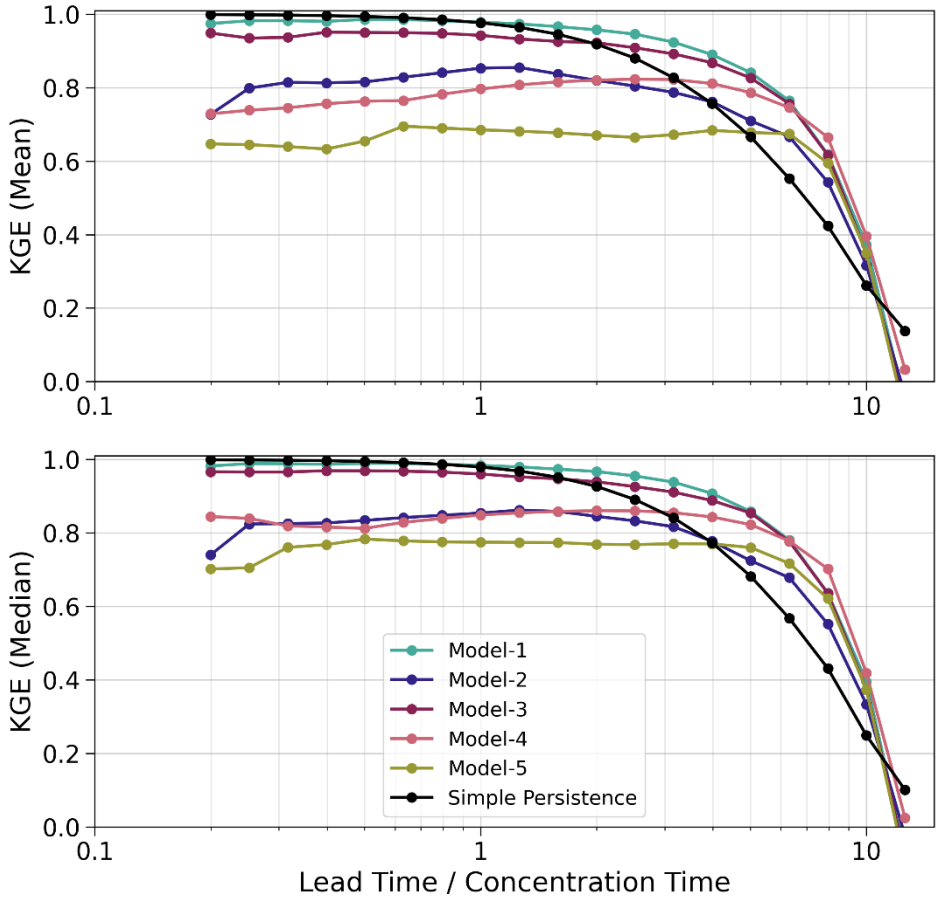


Figure 3. Median KGE values for all five locations corresponding to different models when the future rainfall is assumed zero.

Instead of assuming future rainfall is zero, we removed future rainfall feature from the model inputs and investigated the changes in model performances. Figure 4 illustrates the results for this experiment. The figure shows that the performance of the models does not degrade as rapid as in the previous experiment. The curves for the models get over the curve for the local persistence at a certain dimensionless time values and the decay in the curves is somehow parallel to the simple persistence. This is, indeed, the case when we analyze the median KGE values. We conclude that some of the hydrographs are not well predicted by the models regardless of the fraction of lead time over reference time when we interpret the mean KGE values.

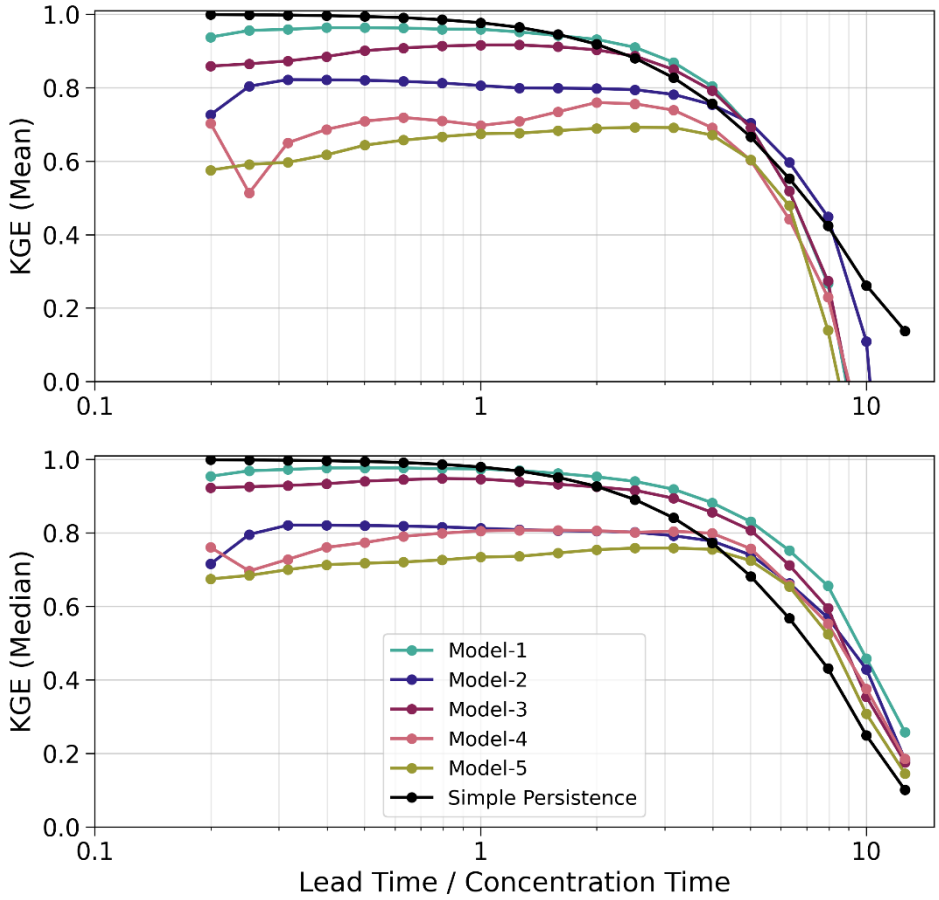
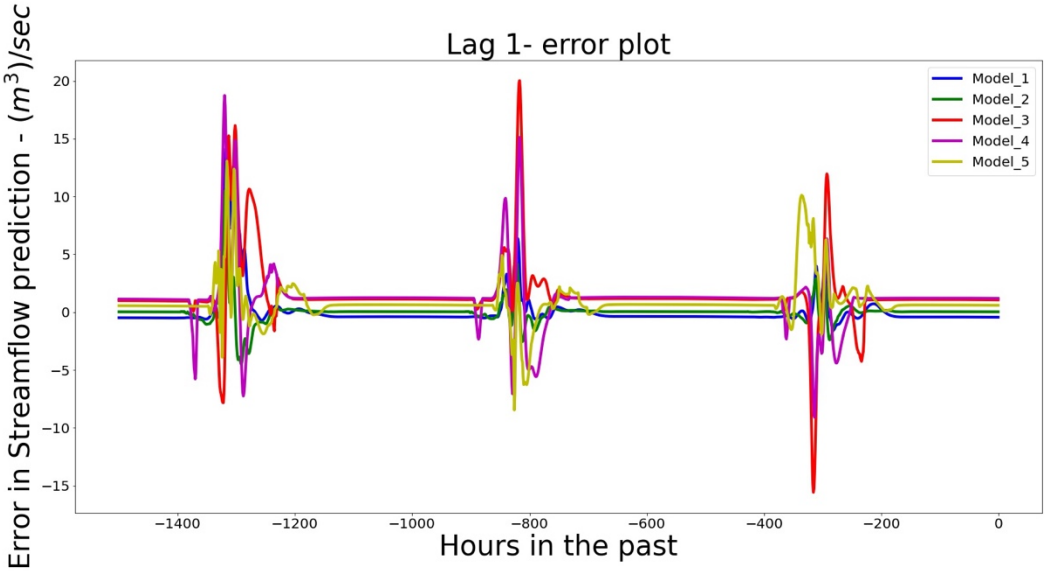
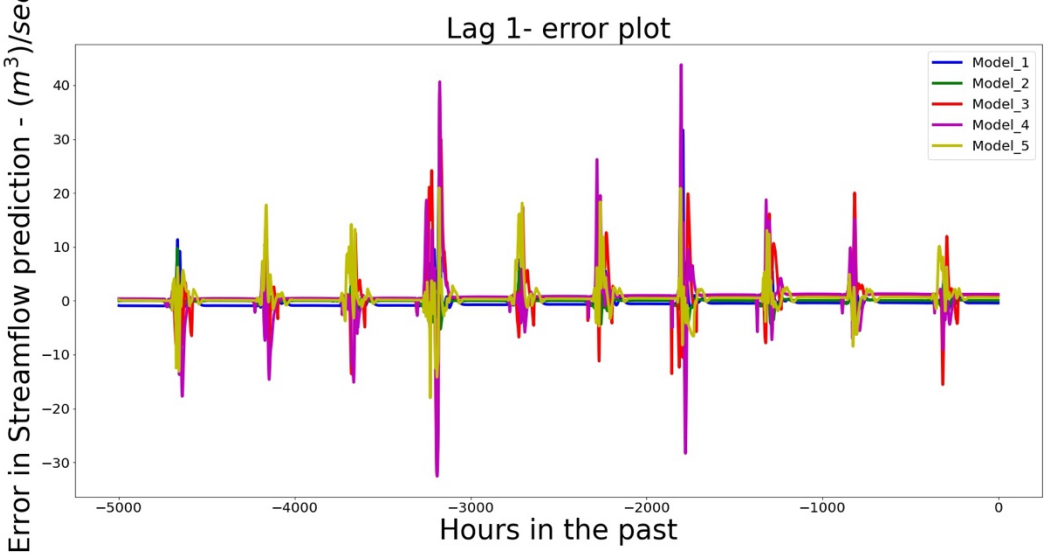


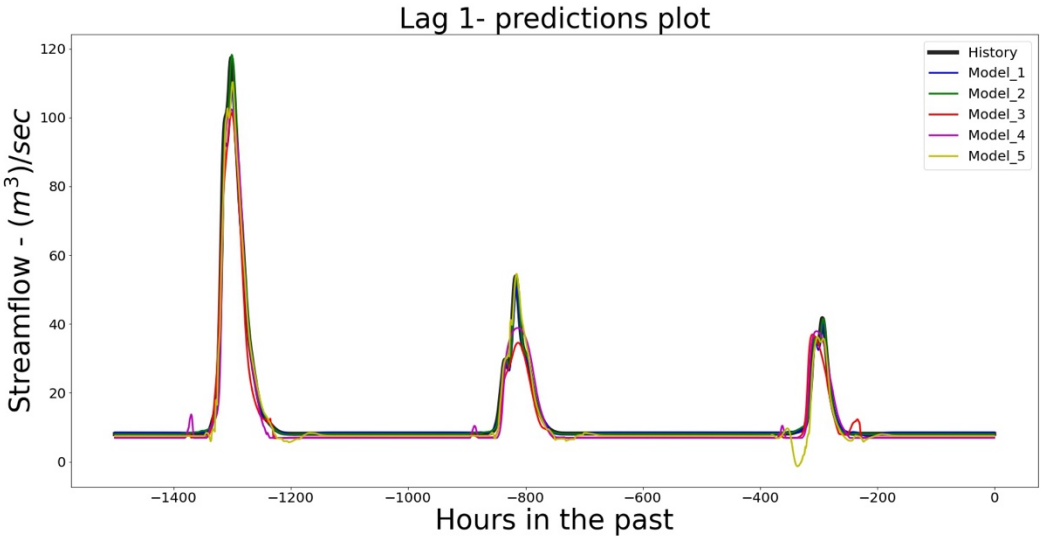
Figure 4. Median KGE values for all five locations corresponding to different models when the future rainfall component is removed from the models' input features and the models are retrained.

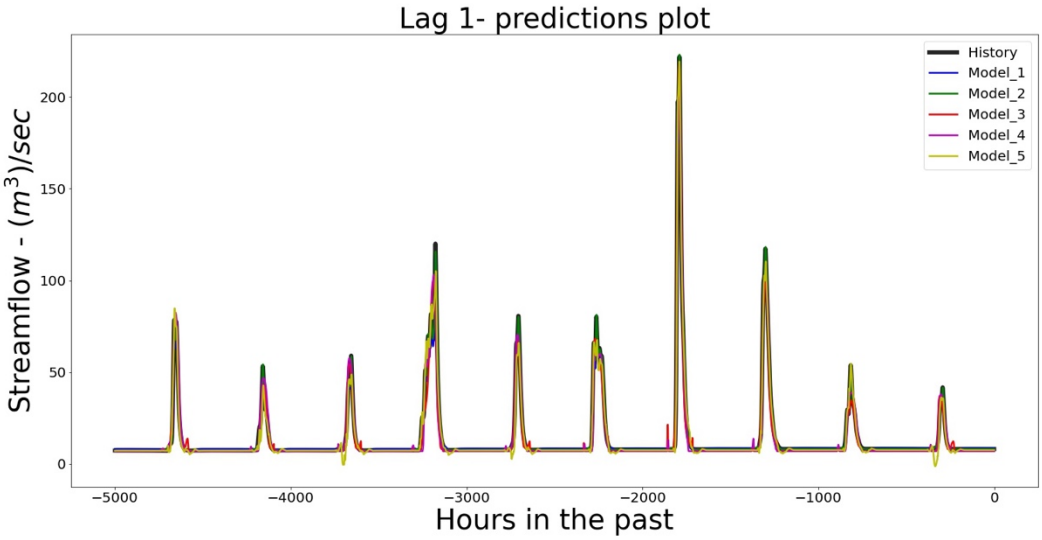
For location 483619: (3 events vs 10 events)

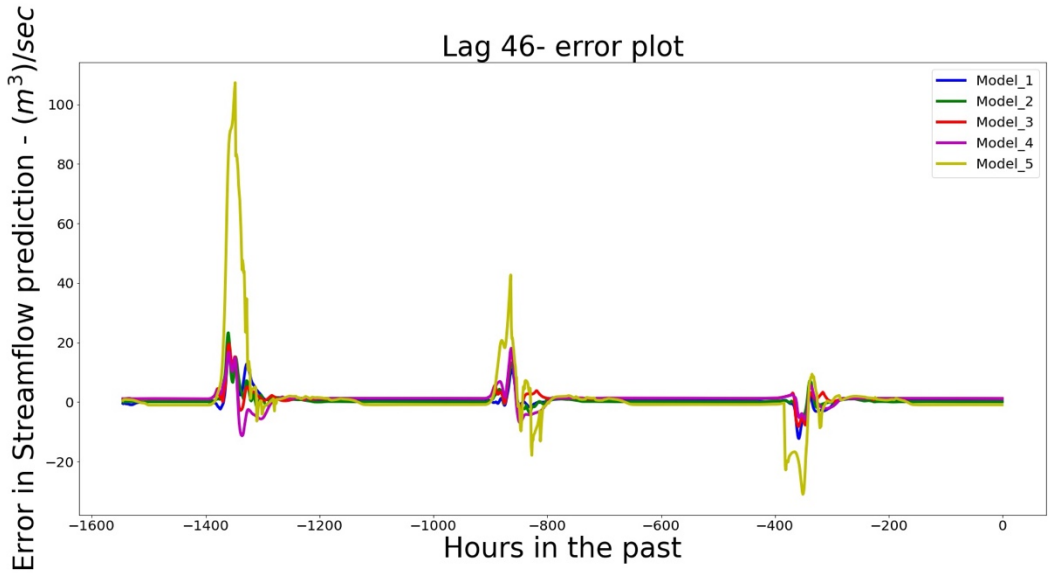
This contains both error plots and actual predictions. We present the first lag and last lag in this report. But we have results for each lag separated by 5 hours.

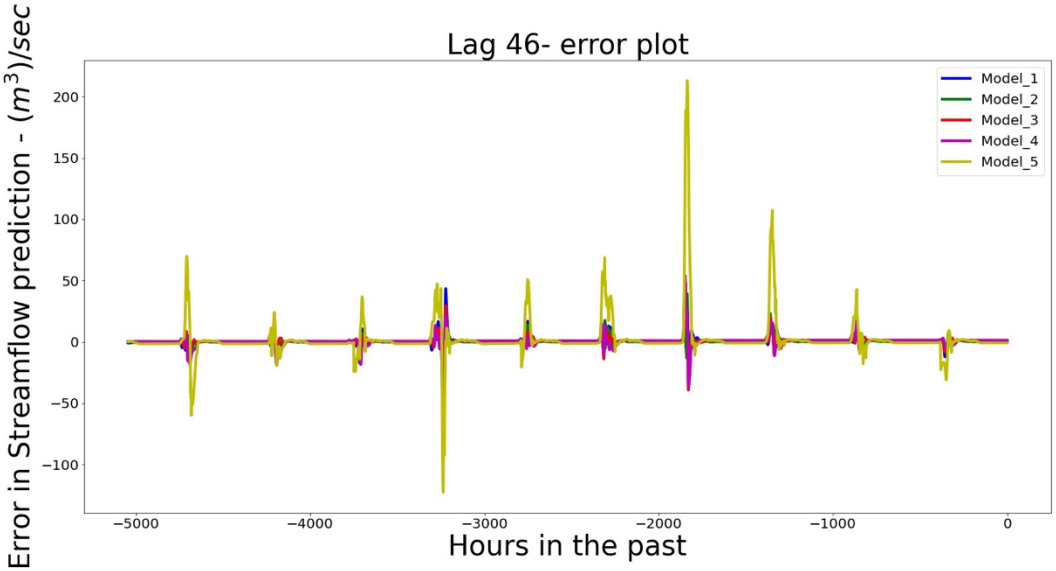

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
20
15
10
5
0
−5
−10
−15
−1400
−1200
−1000
−800
−600
−400
−200
0

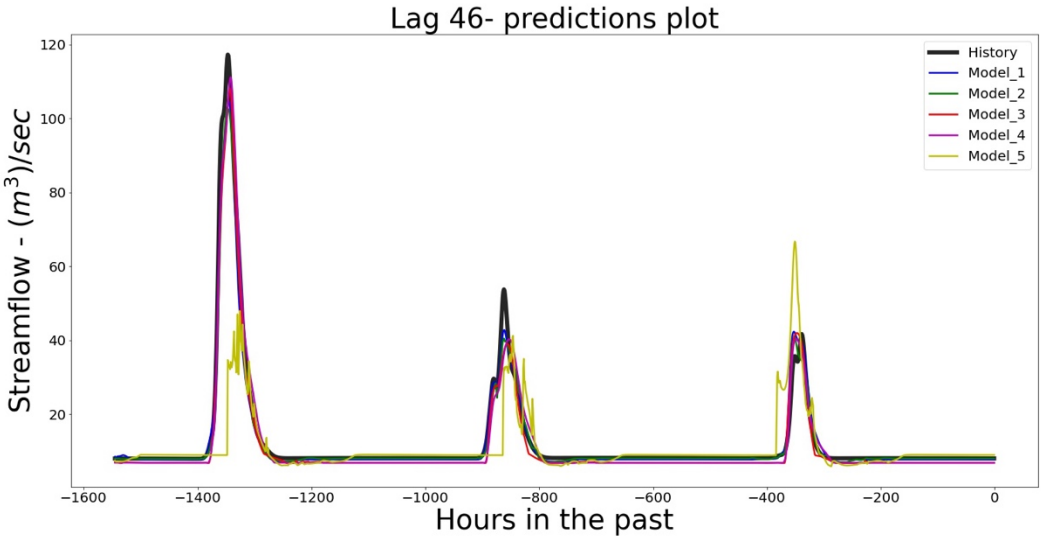

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
40
30
20
10
0
−10
−20
−30
−5000
−4000
−3000
−2000
−1000
0

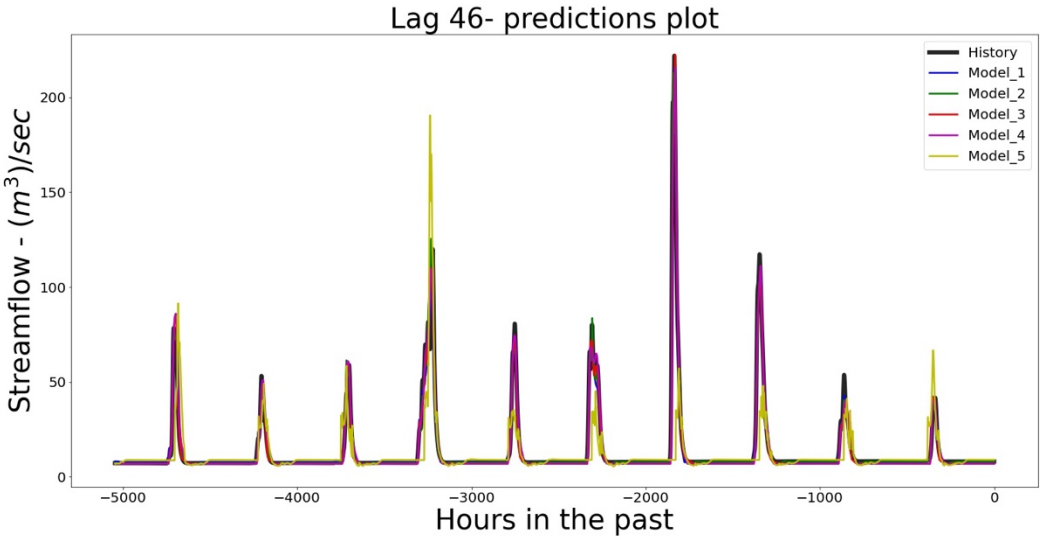

Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
Hours in the past
120
100
80
60
40
20
0
−1400
−1200
−1000
−800
−600
−400
−200
0


Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
Hours in the past
200
150
100
50
0
−5000
−4000
−3000
−2000
−1000
0

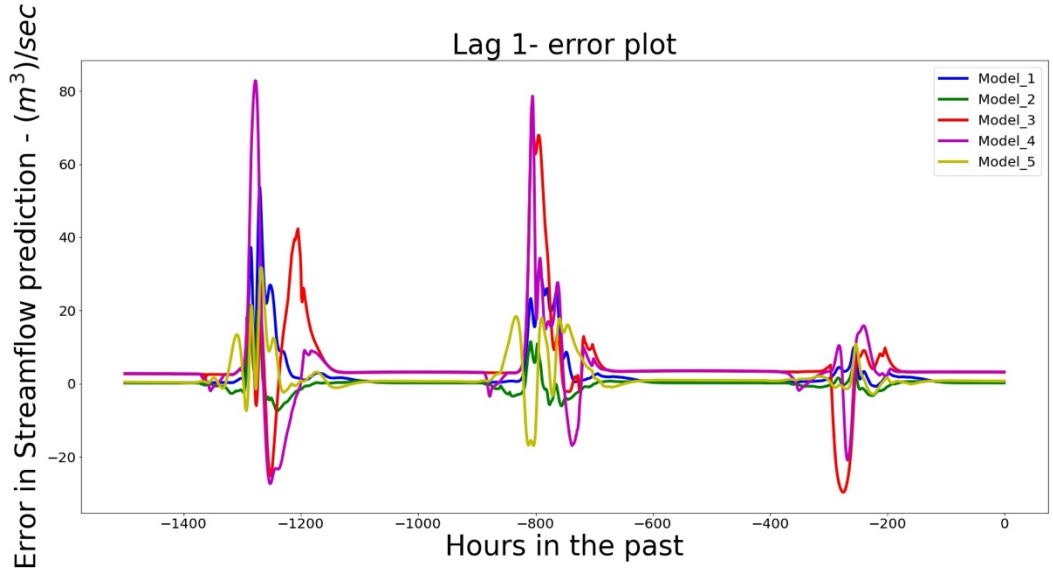

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
100
80
60
40
20
0
−20
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

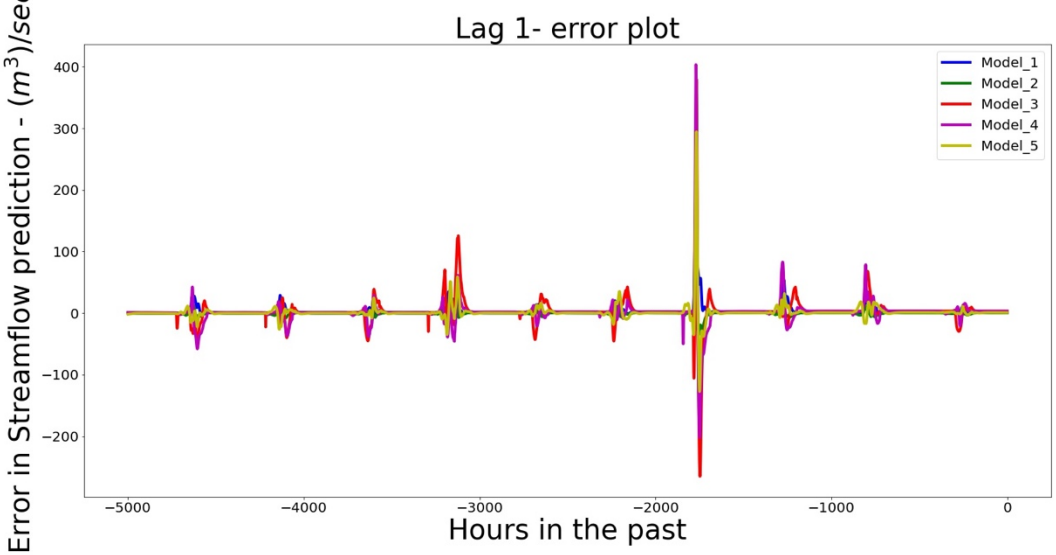

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
200
150
100
50
0
−50
−100
−5000
−4000
−3000
−2000
−1000
0

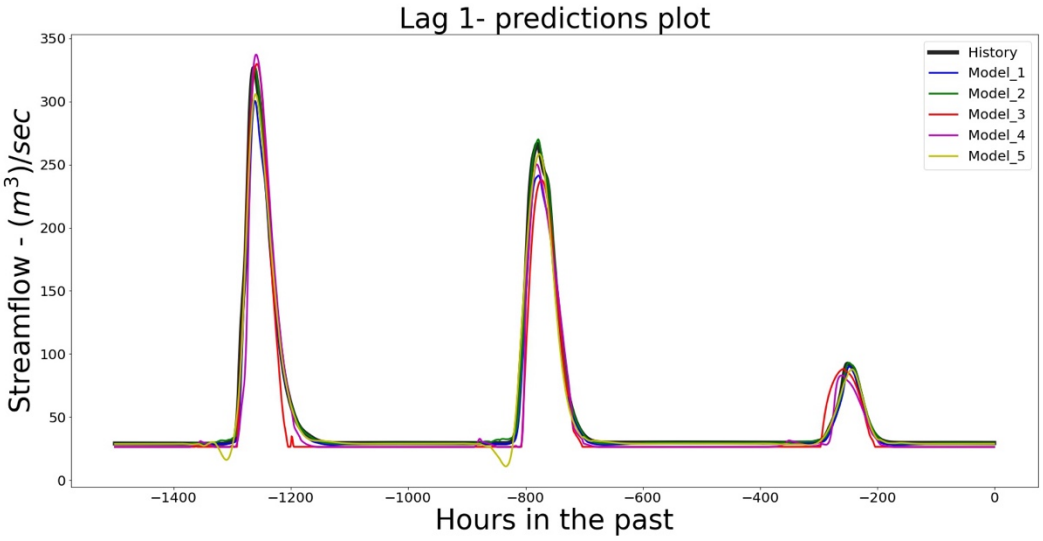


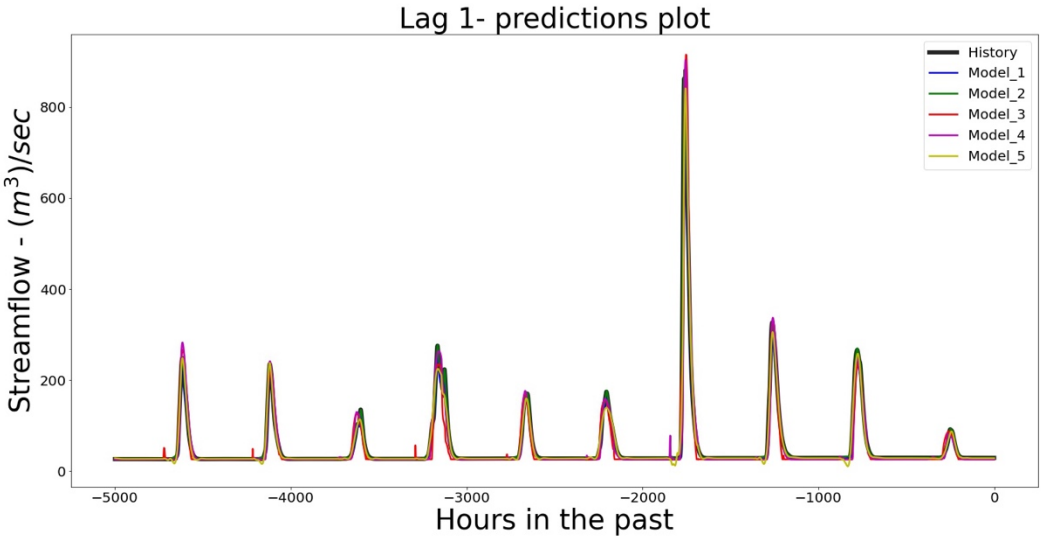


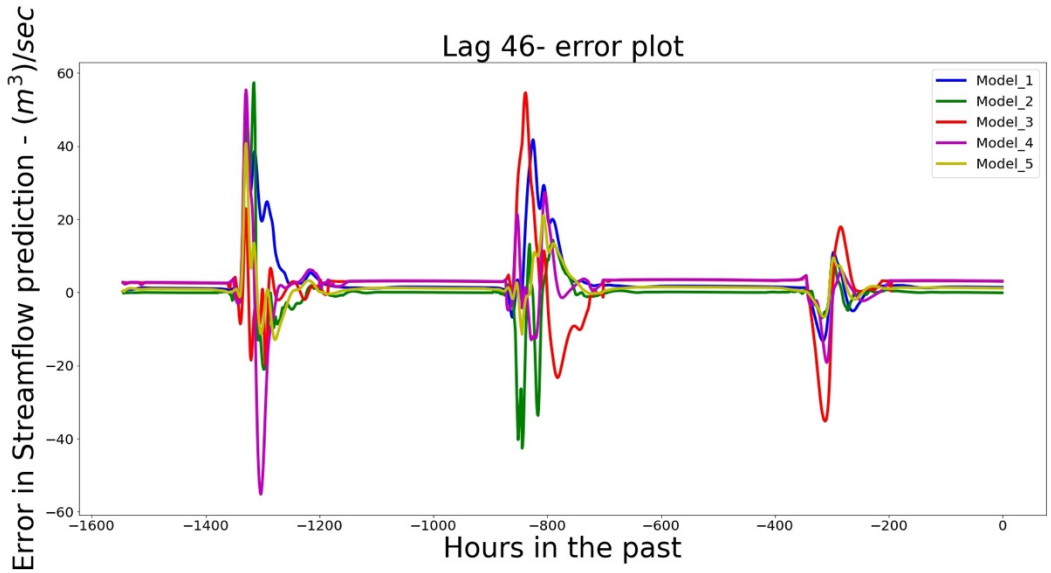
Similarly For location 434365: (3 events vs 10 events)

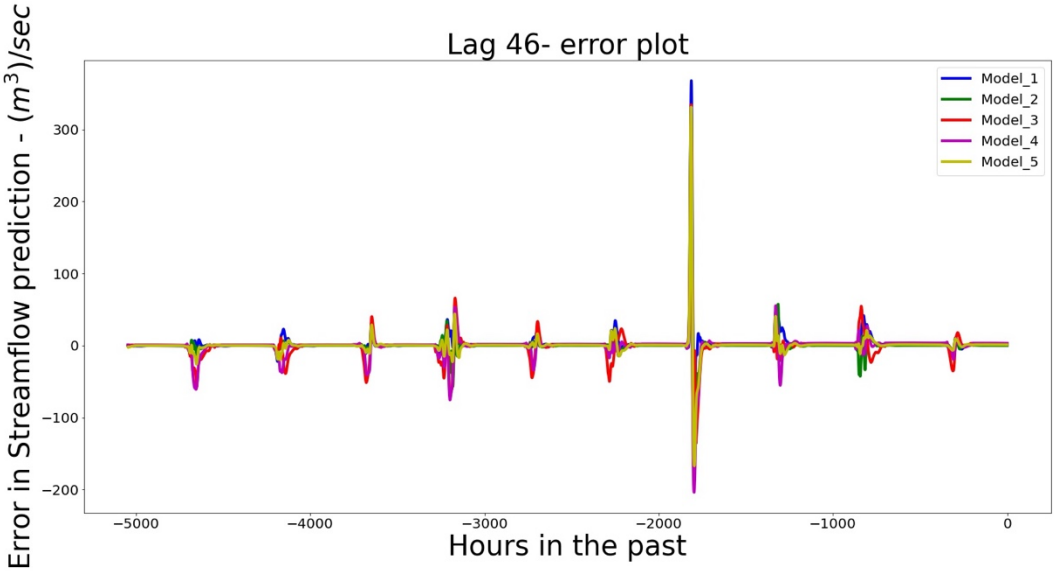

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
80
60
40
20
0
−20
−1400
−1200
−1000
−800
−600
−400
−200
0

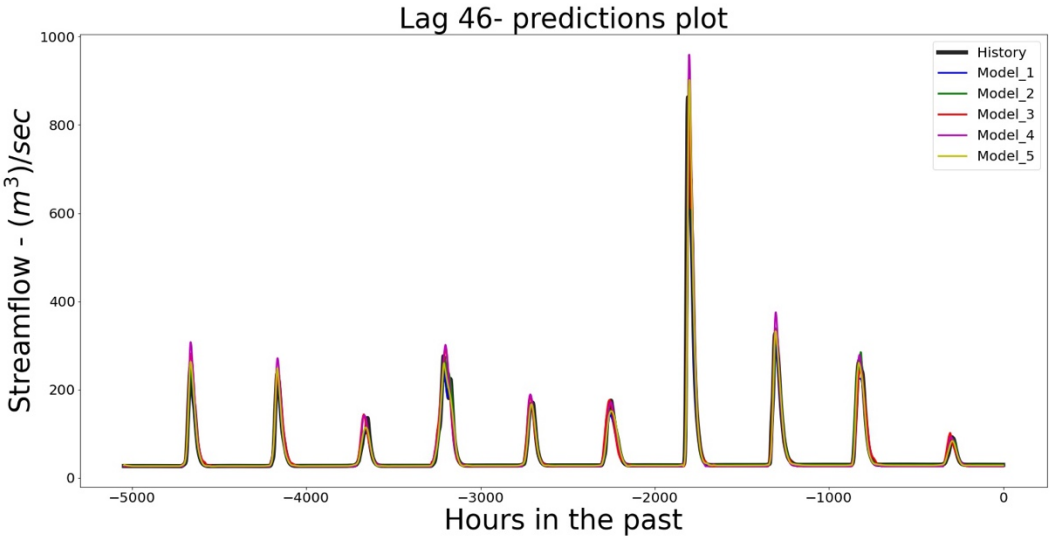

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
400
300
200
100
0
−100
−200
−5000
−4000
−3000
−2000
−1000
0


Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
Hours in the past
350
300
250
200
150
100
50
0
−1400
−1200
−1000
−800
−600
−400
−200
0

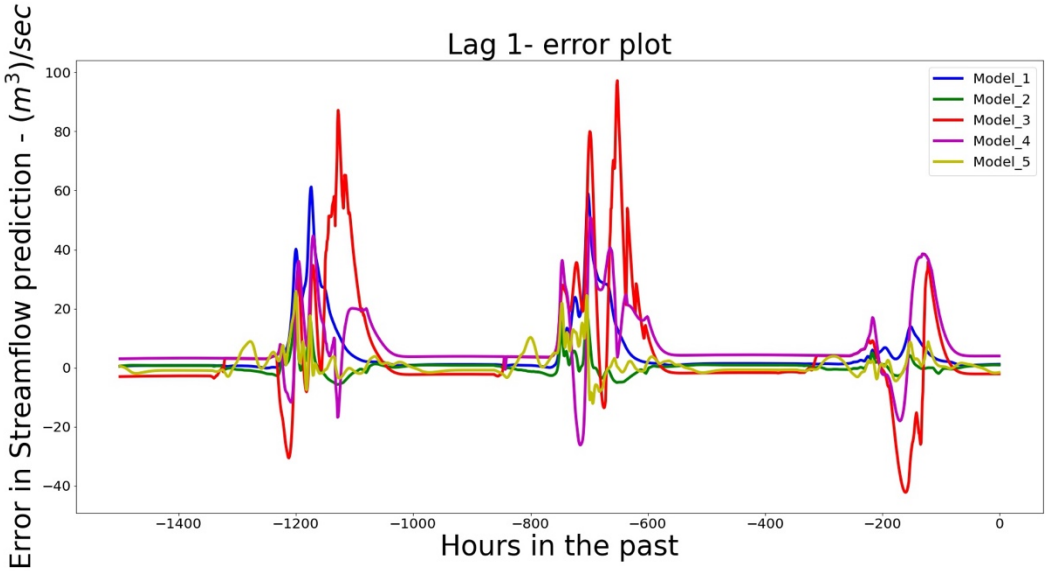

Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
Hours in the past
800
600
400
200
0
−5000
−4000
−3000
−2000
−1000
0

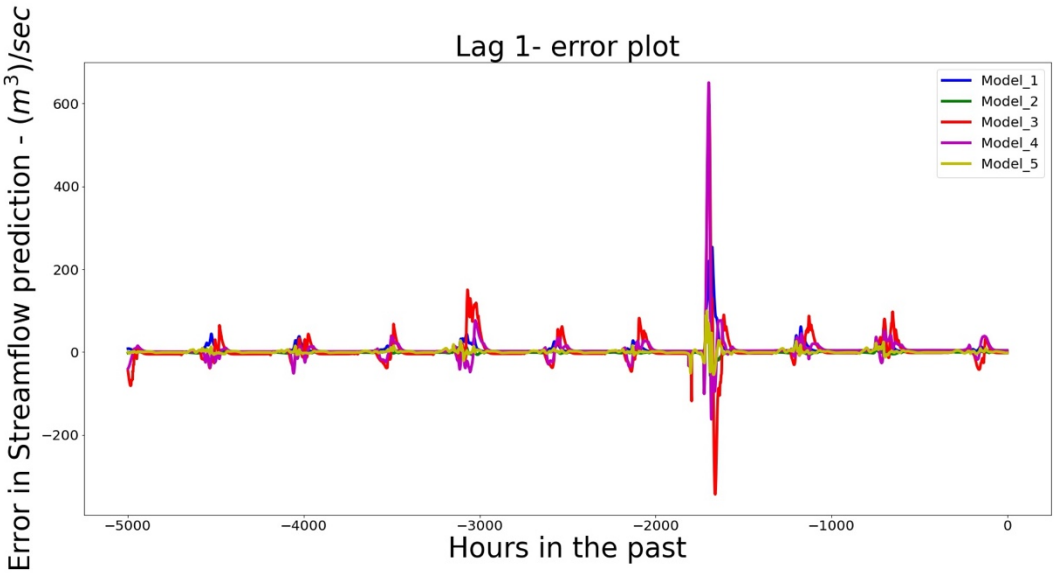

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
60
40
20
0
−20
−40
−60
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

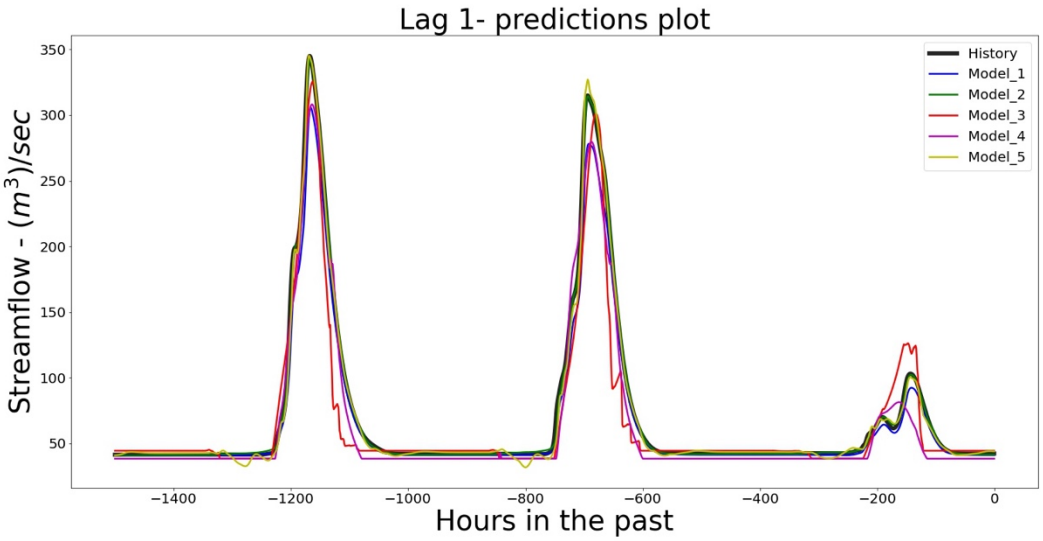

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
300
200
100
0
−100
−200
−5000
−4000
−3000
−2000
−1000
0

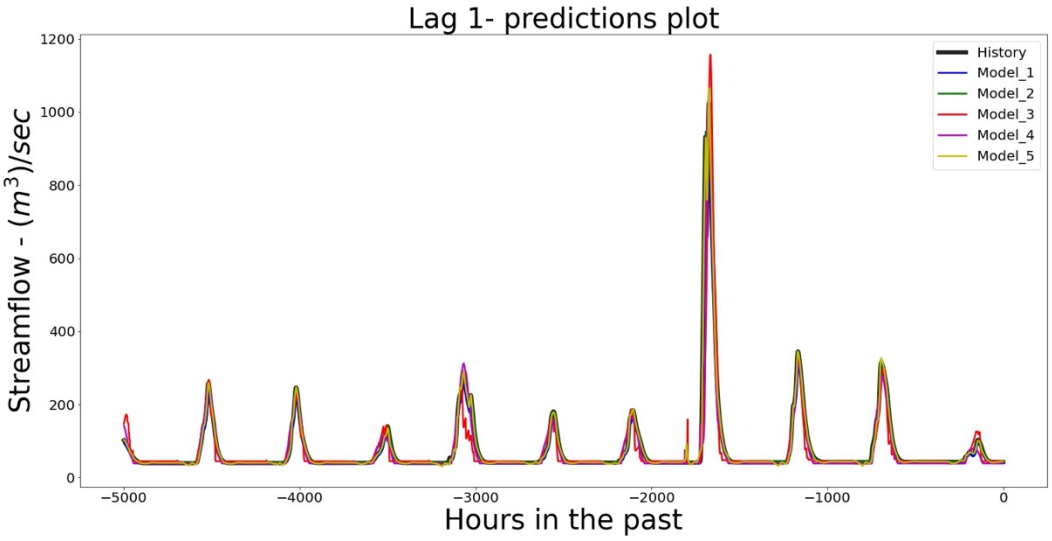


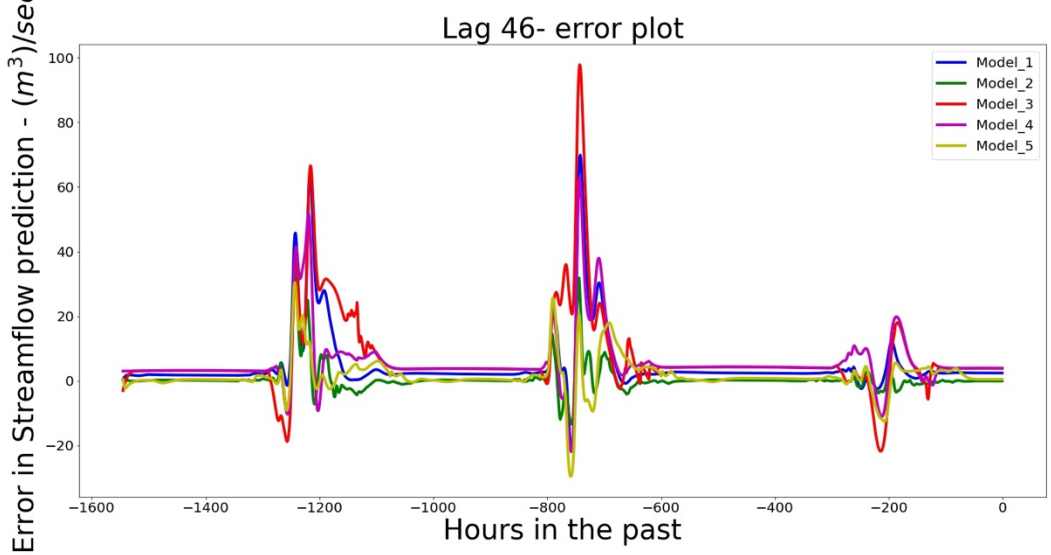
Similarly For location 434478: (3 events vs 10 events)

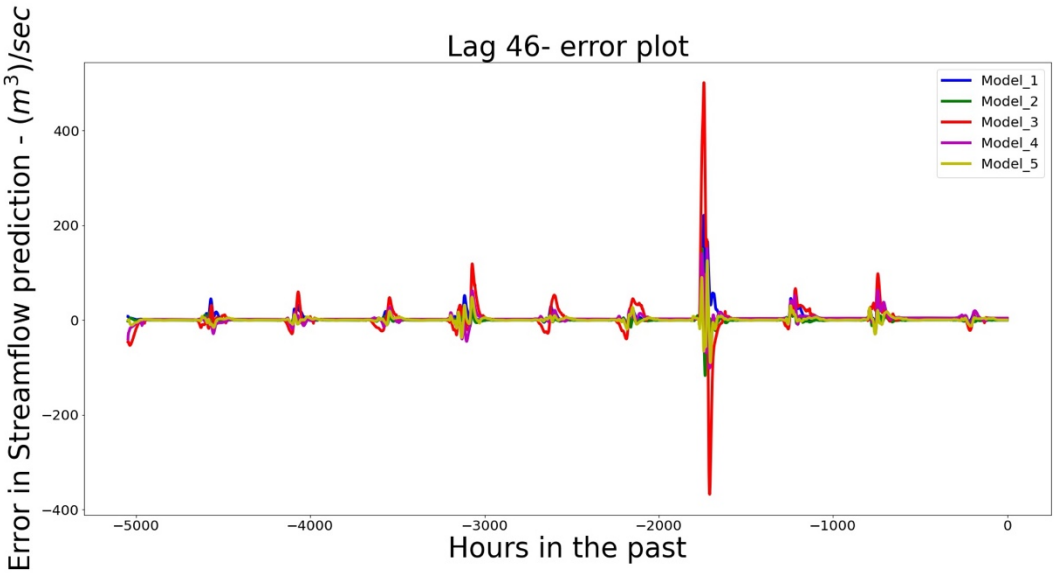


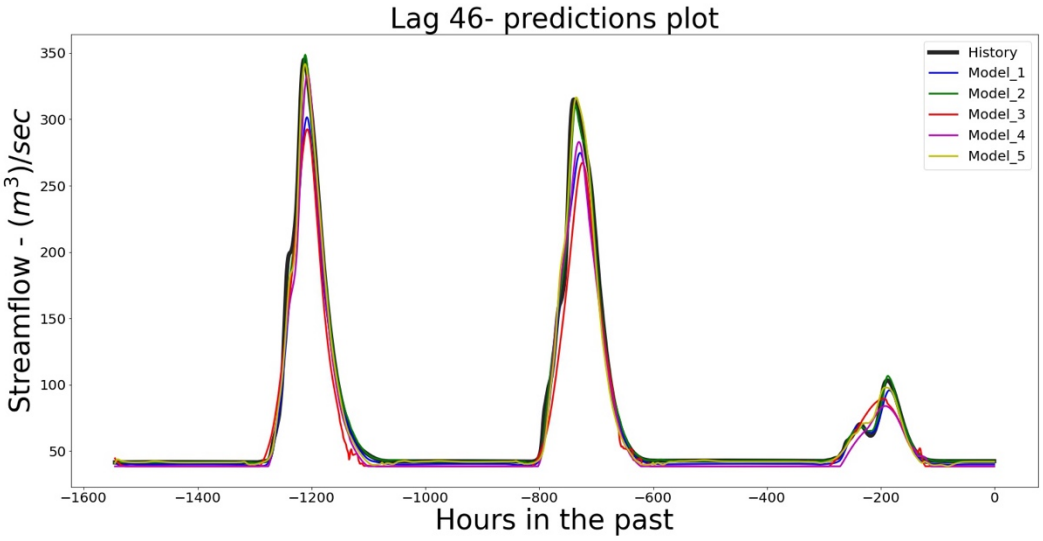

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
600
400
200
0
−200
−5000
−4000
−3000
−2000
−1000
0

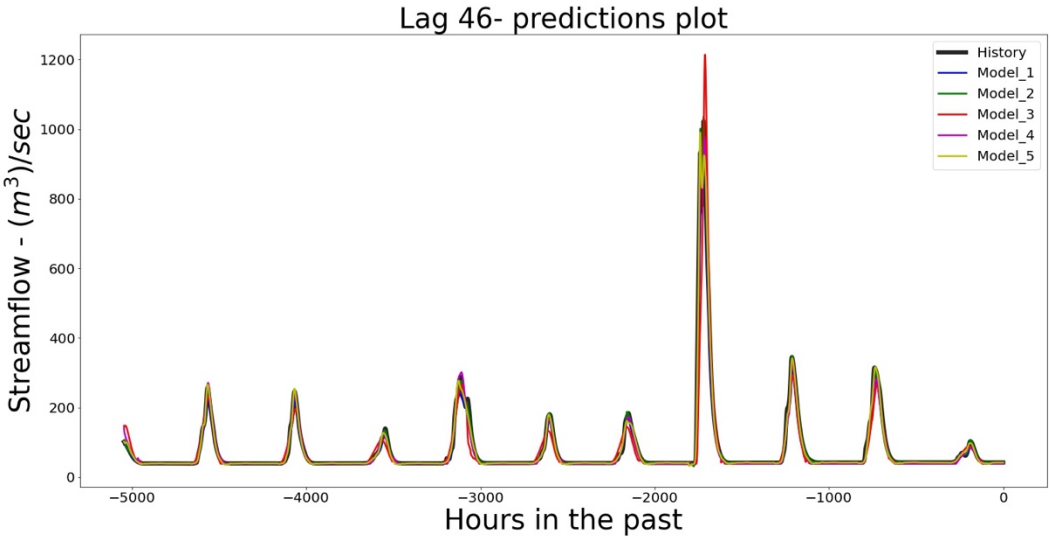

Lag 1- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
350
300
250
200
150
100
50
−1400
−1200
−1000
−800
−600
−400
−200
0


Lag 1- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
1200
1000
800
600
400
200
0
−5000
−4000
−3000
−2000
−1000
0

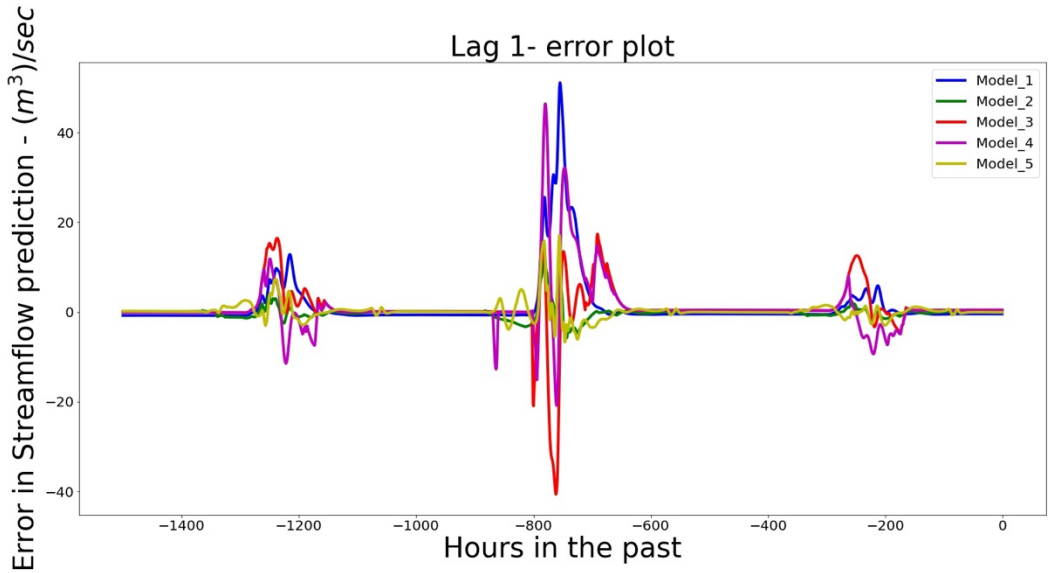

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
100
80
60
40
20
0
−20
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

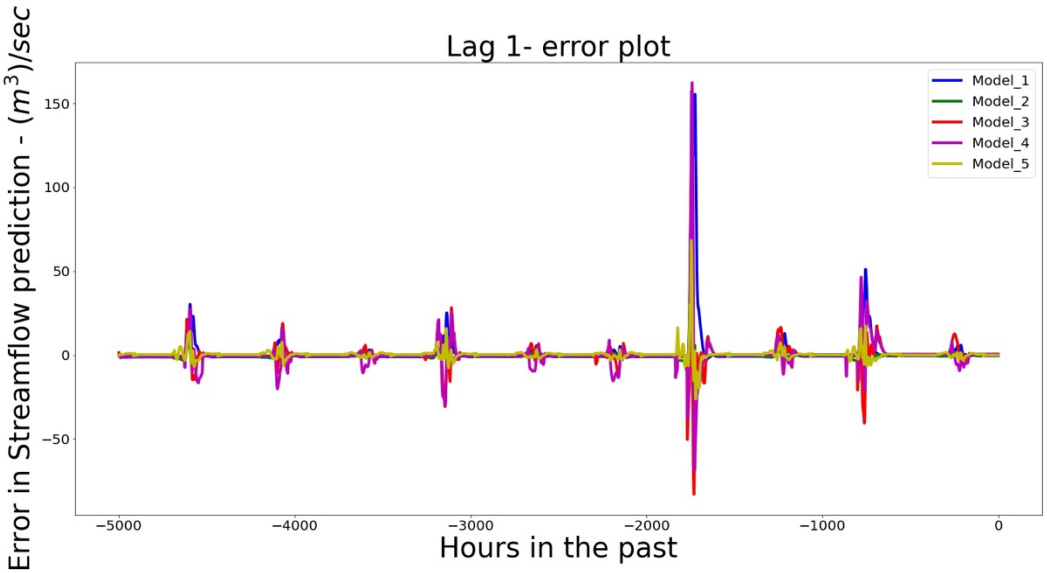

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
400
200
0
−200
−400
−5000
−4000
−3000
−2000
−1000
0

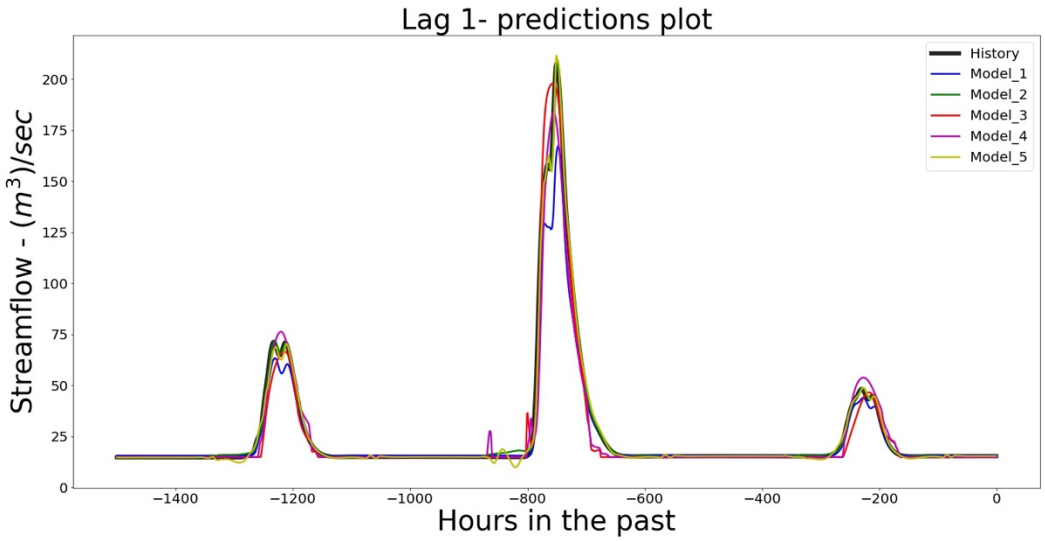

Lag 46- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
350
300
250
200
150
100
50
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

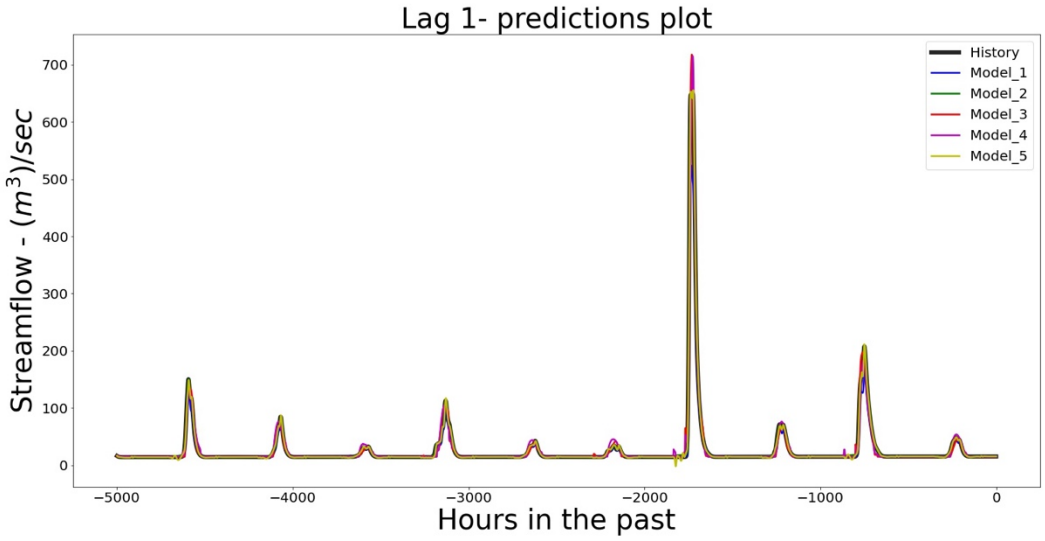


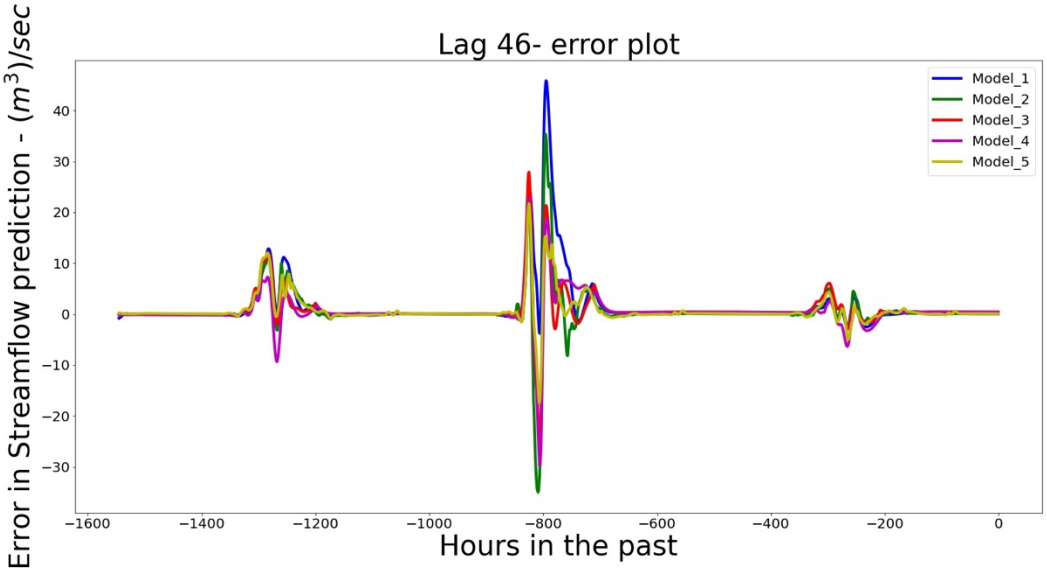
Similarly For location 39971: (3 events vs 10 events)

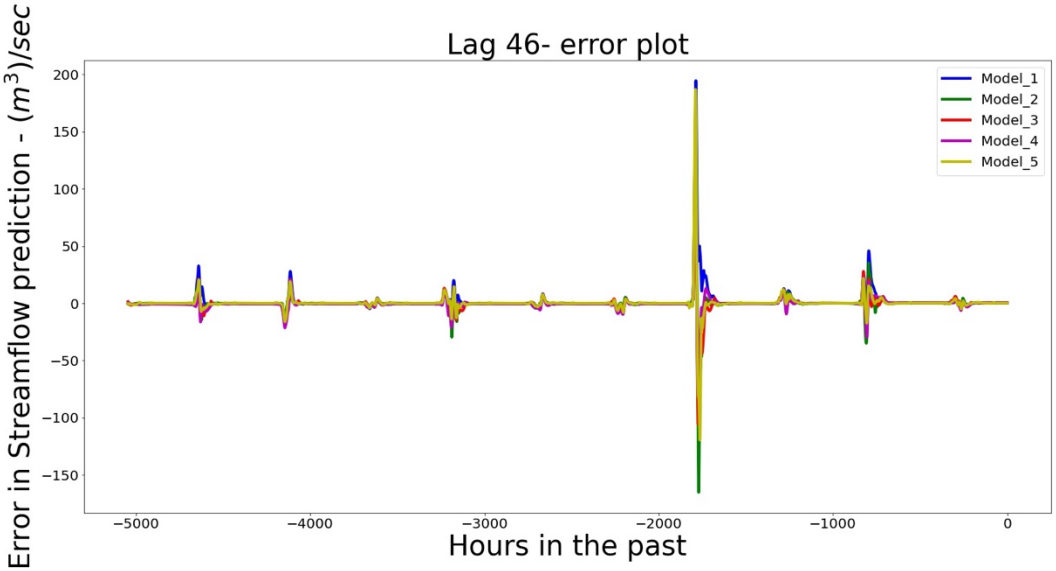


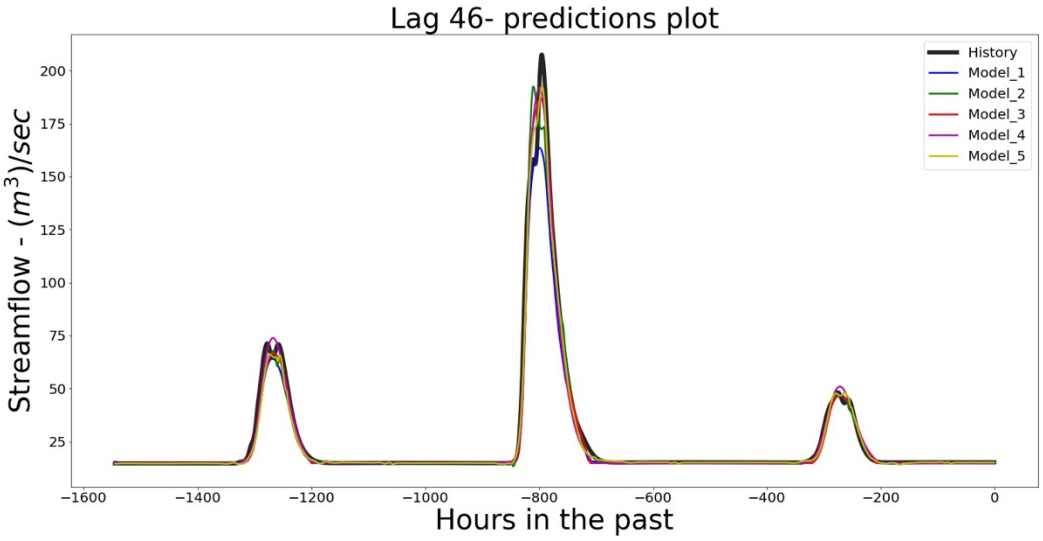

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
150
100
50
0
−50
−5000
−4000
−3000
−2000
−1000
0

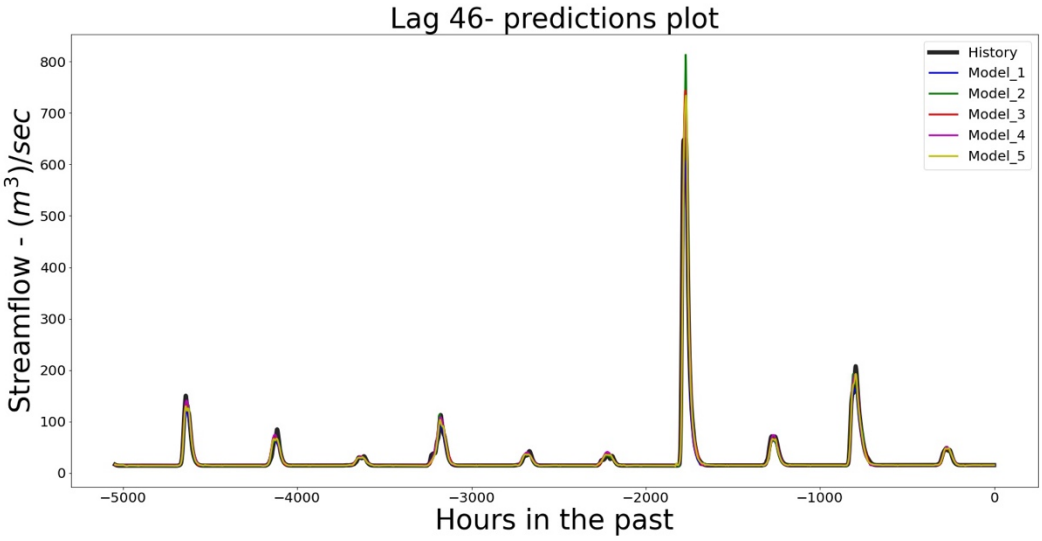

Lag 1- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
200
175
150
125
100
75
50
25
0
−1400
−1200
−1000
−800
−600
−400
−200
0


Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
700
600
500
400
300
200
100
0
−5000
−4000
−3000
−2000
−1000
0
Hours in the past

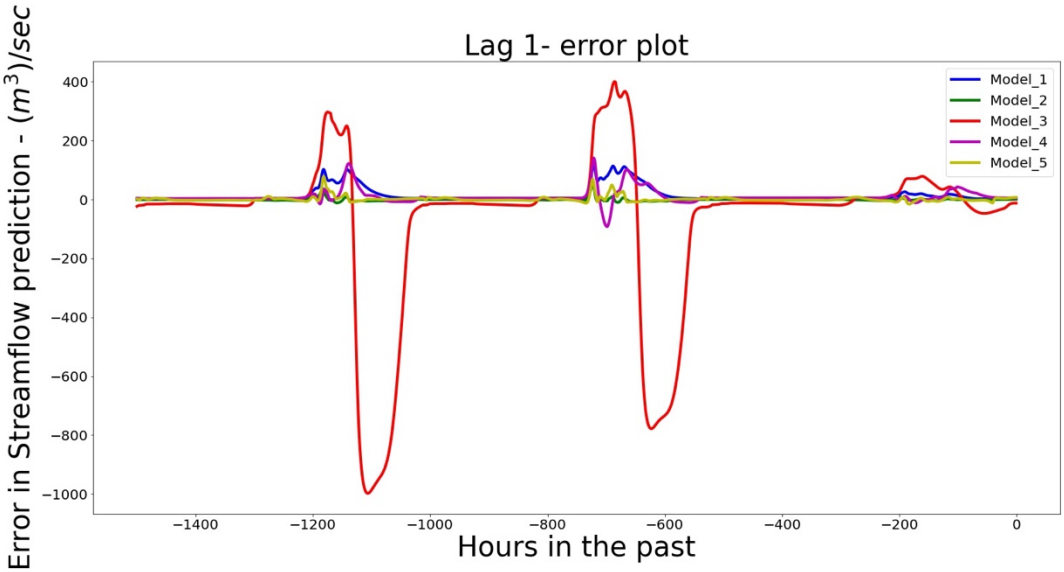

Lag 46- error plot
Model_1
Model_2
Model_3
Model_4
Model_5
Error in Streamflow prediction - $(m^3)/sec$
40
30
20
10
0
−10
−20
−30
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0
Hours in the past

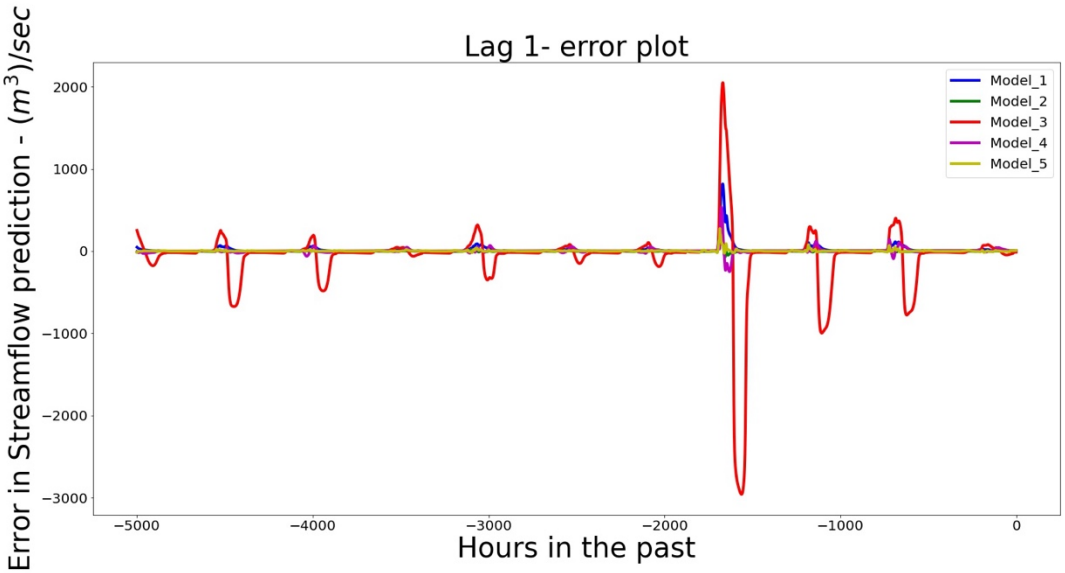

Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
200
150
100
50
0
−50
−100
−150
−5000
−4000
−3000
−2000
−1000
0

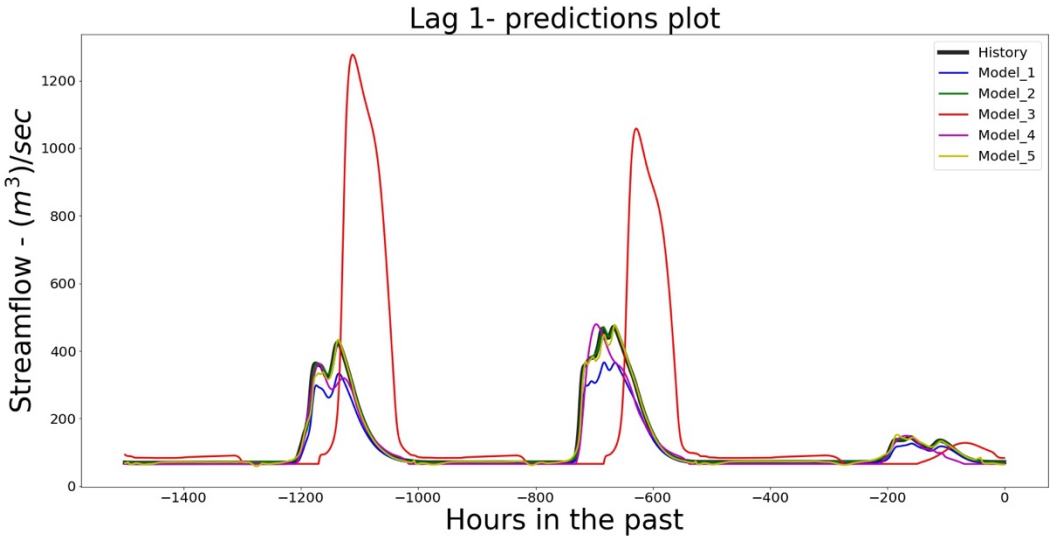

Lag 46- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
200
175
150
125
100
75
50
25
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

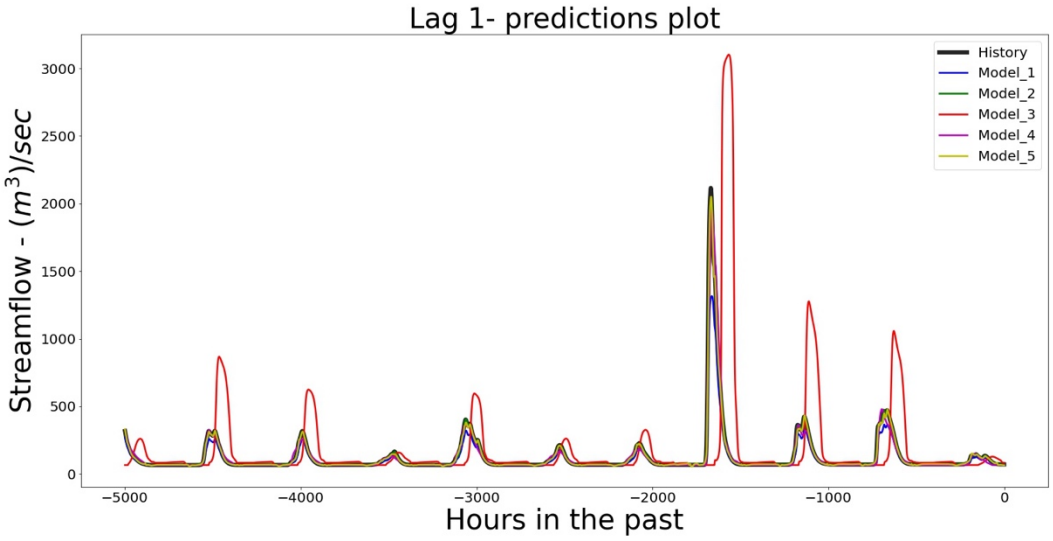


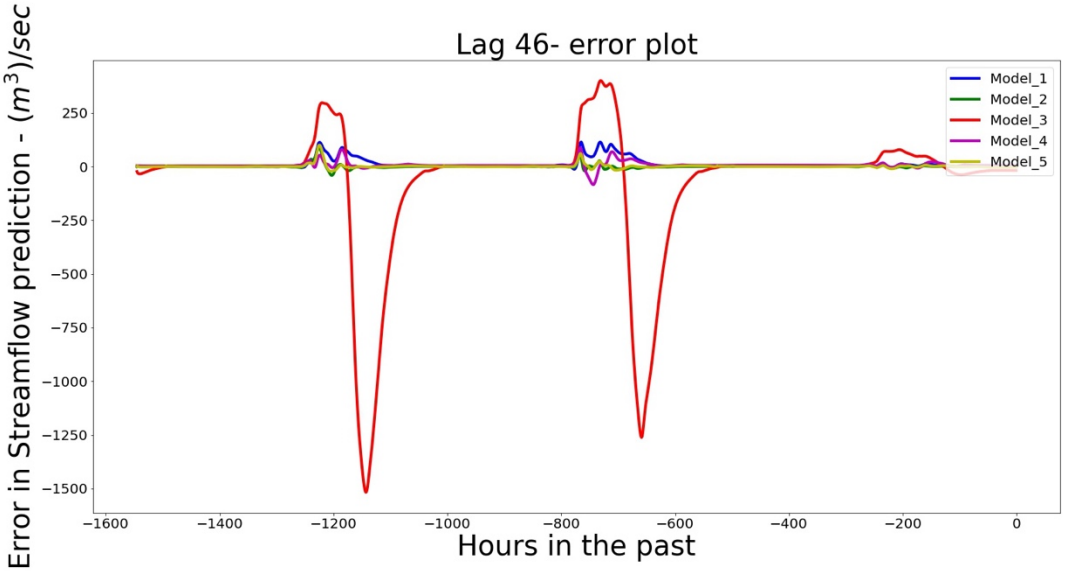
Similarly For location 434514: (3 events vs 10 events)

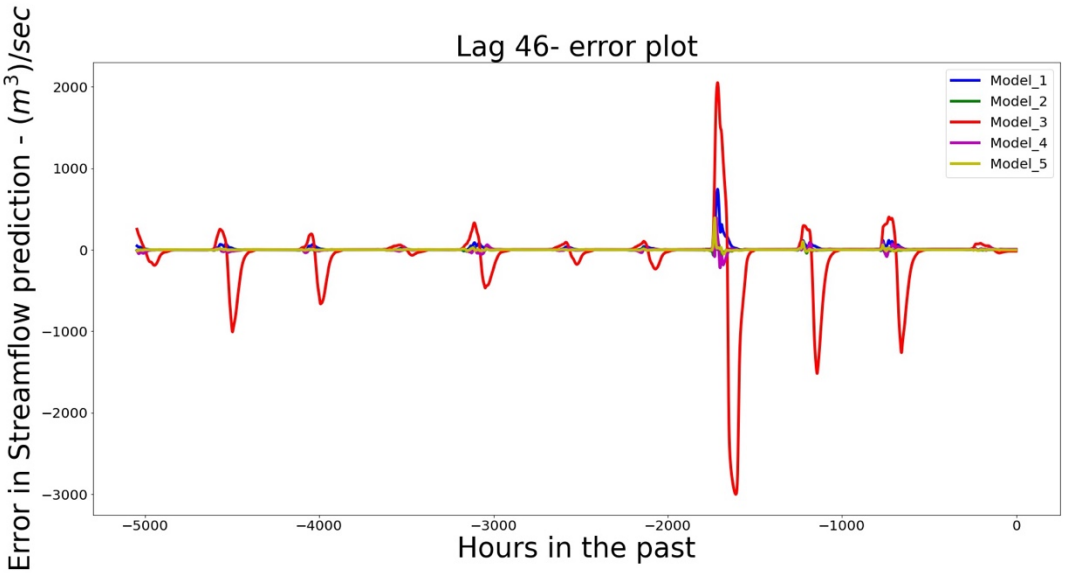


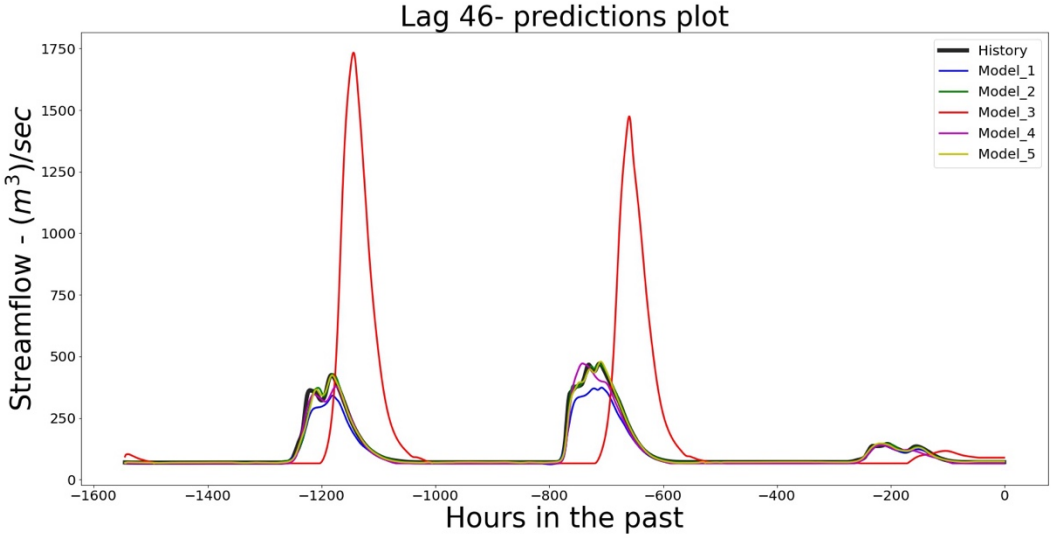

Lag 1- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
2000
1000
0
−1000
−2000
−3000
−5000
−4000
−3000
−2000
−1000
0

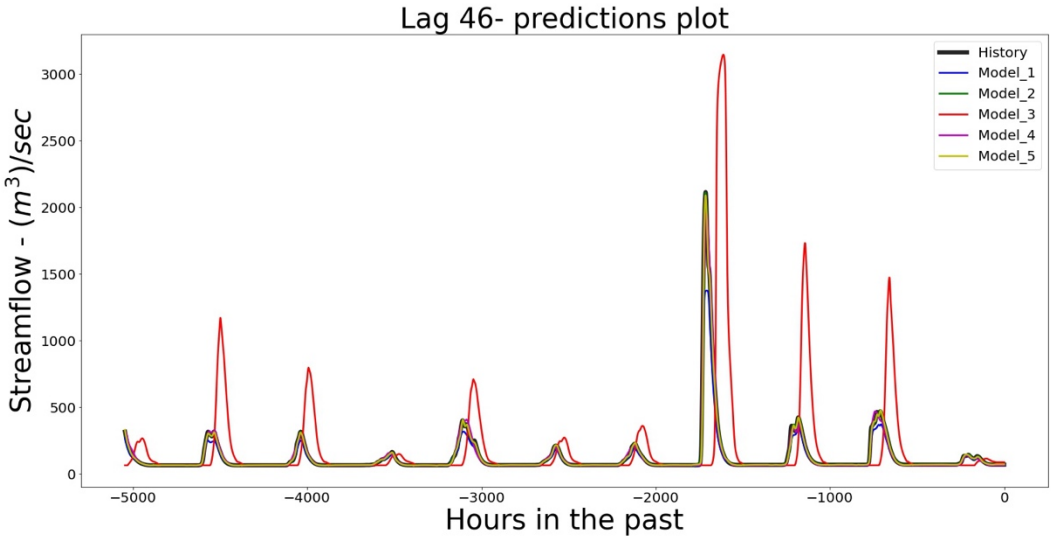

Lag 1- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
1200
1000
800
600
400
200
0
−1400
−1200
−1000
−800
−600
−400
−200
0


Lag 1- predictions plot
History
Model_1
Model_2
Model_3
Model_4
Model_5
Streamflow - $(m^3)/sec$
3000
2500
2000
1500
1000
500
0
−5000
−4000
−3000
−2000
−1000
0
Hours in the past


Lag 46- error plot
Model_1
Model_2
Model_3
Model_4
Model_5
Error in Streamflow prediction - $(m^3)/sec$
250
0
−250
−500
−750
−1000
−1250
−1500
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0
Hours in the past


Lag 46- error plot
Error in Streamflow prediction - $(m^3)/sec$
Hours in the past
Model_1
Model_2
Model_3
Model_4
Model_5
2000
1000
0
−1000
−2000
−3000
−5000
−4000
−3000
−2000
−1000
0


Lag 46- predictions plot
Streamflow - $(m^3)/sec$
Hours in the past
History
Model_1
Model_2
Model_3
Model_4
Model_5
1750
1500
1250
1000
750
500
250
0
−1600
−1400
−1200
−1000
−800
−600
−400
−200
0

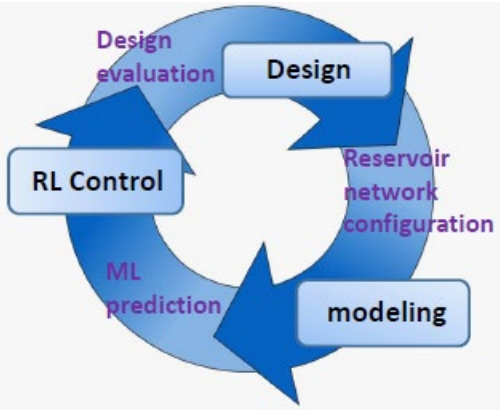


Few observations from above plots are that:

- Model 2 and Model 5 perform consistently better than the other models
- The errors sometimes corresponds to a shift of the prediction hydrograh by few hours right or left to the original hydrograph. However, Peak of the predicted hydrographs and original ones are very similar
- As we go downstream, Rainfall only model starts to decrease in performance. This may be due to the fact that we are taking an average rainfall of all the catchments that directly contribute to flow at that particular link.

(Note: we also want to investigate with new models that does not take future values as an input)

Ideas/aims for future extramural project:



In a future extramural project, the ultimate goal is to design an active reservoir system for flood mitigation and control. The iterative design process consists of three major components: reservoir network design, flood forecast prediction, and reinforcement learning (RL) control. The machine-learning-based flood forecast model will be utilized to approximate the interaction between the control agent and the environment (i.e., the river network).
The results from this IIAI pilot project can be extended in developing the above-mentioned modeling module. The following specific aims shall be addressed: 1) incorporating the reservoir gate operations as additional input features in the machine-learning-based predictive model; 2) predicting the state (i.e., the configuration) of the river network in addition to the peak flow only; 3) developing a generalized machine learning model for various river networks.