IMPLEMENTATION OF MACHINE LEARNING MODELS IN RAINFALL PREDICITON

By:

Nishant Singh Kushwaha
Roll No: 11710098
B. Tech. Civil Engineering
NIT Kurukshetra

Under the Guidance of:

Dr. S. K. Gupta
Associate Professor
IIT(BHU)

NATIONAL INSTITUTE OF TECHNOLOGY
KURUKSHETRA – 136119

MAY 2020

CERTIFICATE

This is to certify that **Mr. Nishant Singh Kushwaha,** student of **B. Tech. 3rd Year Civil Engineering** at **National Institute of Technology, Kurukshetra;** has completed his project work under my guidance for a duration of 19 weeks starting from January 8, 2020 up to May 19, 2020.

The project involved implementation of Machine Learning methods on Rainfall Prediction. This project has helped him to get acquainted with latest technology that is being used today. The experience he gained from here will help him in his future. The work was completed by him with complete dedication and enthusiasm.

Dr. S. K. Gupta
Associate Professor
Civil Engineering Department
IIT(BHU)

ACKNOWLEDGEMNT

I express my gratitude to **Dr. S. K. Gupta,** Associate Professor, Department of Civil Engineering, IIT(BHU), Varanasi for allowing me to take up a project work under his guidance. He was my guide throughout my project work. He aided me at times and helped me to successfully complete the work.

I also express my gratitude to **Dr. K. K. Singh**, Professor, Department of Civil Engineer, NIT Kurukshetra for his contribution to my knowledge.

The time I spent at IIT(BHU) helped me to gain a lot of knowledge. It helped me to learn new things that have the potential of changing the world.

-Nishant Singh Kushwaha

ABSTRACT

Accurate and reliable prediction of rainfall and runoff is required for flood control, water supply planning, hydro-power generation and water management and harvesting. Three different Machine Learning methods (BAT-SVM, SVM (Support Vector Machine), Artificial Deep Neural Network (ADNN)) were used in this modelling. The dataset used was a rainfall dataset of East Uttar Pradesh from the year 1901 to 2017 downloaded from data.gov site. The SVM model yielded the best result on this dataset. The ADNN model failed starkly on this dataset because the data available was less than that is required for training a neural network model. However, when the ANN model was used to predict whether the rainfall will be above a threshold value or not, it gave satisfactory results. All the models were created on python programming language.

Contents

Introduction 6
Support Vector Machine 6
Bat Algorithm 7
Theory 7
Generation of Initial Population 8
Procedure for SVM Parameters Optimization by Bat Algorithm 8
Deep Neural Network (DNN) 9
Study Area and Data Sets 10
Study Area 10
Model Structures 10
Measures of Accuracy 11
Discussion and Result 12
Fitting an ANN model to the Data 12
Fitting an SVM Model to the Data 12
Fitting a Bat Algorithm based SVM to the Data 13
Comparison of the Three Models 14
Conclusion 15
References 16
List of Figures 17
List of Tables 18
Results 19

Introduction

Medium-term and long-term runoff prediction is crucial for water resources management because it provides important references for flood control scheduling, hydro-power generation and water supply and management plans. The proposed model for the same can be classified into groups: physically-based models and data-based models. Although physically-based models help researchers to better understand the process, these approaches are limited because of multitude of, as well as complexity of, the process involved and also by the scarcity of the data (ASCE 2000; Behzat et al. 2009). On the other hand, data-based model can provide accurate forecast with fewer data, albeit they fail to provide physical interpretation of the process. Therefore, data-based models become popular in medium-term and long-term prediction.

During the past couple of decades, a lot of improvement has been made in different models for prediction, for example, regression models, ANN models, SVM models and hybrid models. As SVM is based on risk minimization and optimization principle, hence, it is the most widely used model for streamflow prediction, flood stage prediction, reservoir level prediction and precipitation prediction.

The efficiency of SVM largely depends on training parameters C and the kernel function K. There are many ways of optimizing an SVM. However, the most widely used method is quadratic optimization. BAT algorithm was also used to optimize the SVM (i.e. to find the penalty parameter C) (10.1061/(ASCE)HE.1943-5584.0001269). There are many other SVM optimization, gradient descent algorithm, firefly algorithm. But, for this project only the two above mentioned algorithms were chosen.

The outline for this project is as follows. First a detailed description of the SVM, Bat algorithm and Deep Neural Network (DNN) is given. Second the characteristics of dataset and the model used is introduced. Third the quadratic optimization and bat algorithm optimization is presented.

Support Vector Machine

SVM was first proposed by Vapnik (10.1061/(ASCE)HE.1943-5584.0001269), inspired by the statistical machine learning theory. It has good generalization capability and strong theoretical foundations. In recent years, SVM has gained more and more popularity for electrical, wind speed, economic and traffic accident forecasting. In this project basic SVM was used.

SVM constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class, since in general the larger the margin the lower the generation error of the classifier.

The model produced by support vector classification depends only on a subset of the training data, because the cost function for building the model does not care about training points that lie beyond the margin. Analogously, the model produced by Support Vector Regression depends only on a subset of the training data, because the cost function

for building the model ignores any training data close to the model prediction.

Given the training vectors $x_i \in \mathbb{R}^p, i = 1, \ldots, n$, in two classes and vector $y \in \{1, -1\}^n$, SVC solves the following primal problem:

$$\max_{w,b,\zeta,\zeta^*} \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^*)$$

$$subject\ to\ y_i - w^T \Phi(x_i) - b \leq \varepsilon + \zeta_i,$$

$$w^T \Phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*,$$

$$\zeta_i, \zeta_i^* \geq 0, i = 1, \ldots, n \quad (1)$$

Its dual is

$$\min_{\alpha,\alpha^*} \frac{1}{2} (\alpha - \alpha^*)^T Q (\alpha - \alpha^*) + \varepsilon e^T (\alpha + \alpha^*) - y^T (\alpha - \alpha^*)$$

$$subject\ to\ e^T (\alpha - \alpha^*) = 0$$

$$0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, \ldots, n \quad (2)$$

where e is the vector of all ones, C > 0 is the upper bound, Q is an n by n positive semidefinite matrix, $Q_{ij} \equiv K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j)$ is the kernel. Here training vectors are implicitly mapped into a higher (maybe infinite) dimensional space by the function Φ.

Hence, the decision function is:

$$\sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + \rho \quad (3)$$

(Scikit Learn)

Where, K is the kernel function. A kernel is a function that changes the dimension of the data, which makes it easier for the model to create the hyperplane. There are three main kernel function in SkLearn library: linear, polynomial and the radial bias function (rbf). However, a custom kernel function can also be used in the model.

Bat Algorithm

Theory

Bat algorithm is a new metaheuristic method based on the echolocation behavior of bats, which was first introduced by Yang (2010).

Bats usually adjust their loudness and pulse emission rate when approaching the target. In order to better understand the echolocation characteristics of microbats, the following idealized rules were supposed (Yang 2010):

1. All bats use echolocation to sense distance, and they also "know" the difference between food/prey and background barriers in some magical way.
2. Bats fly randomly with velocity V_i at position X_i with a fixed frequency f (or wavelength λ), varying wavelength λ (or frequency f) and loudness $A0$ to search for prey. As the sound travels in the air at a speed of 340 m/s at room temperature, and the speed is given by
   $$v = \lambda \times f \quad (4)$$
   They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission r∈[0,1], depending on the proximity of their target;
3. Assume that the loudness varies from a large (positive) $A0$ to a minimum constant value Amin. Researches show that the loudness decreases once a bat was found to have approached its target. For the minimization optimize problem, the smaller the fitness is, the better the solution will be. Therefore, in order to simplify the calculation, $A0$ of all bats was assumed to be equal to the initial

fitness, and Amin was assumed to be 0.

The new positions x_i^t, velocities v_i^t, and frequencies f_i of bats in a d-dimensional search space are defined as Eqs. (5)– (7)

$$f_i = f_{min} + (f_{max} - f_{min})\beta \quad (5)$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i \quad (6)$$

$$x_i^t = x_i^{t-1} + v_i^t \quad (7)$$

Where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution; $x*$ = current global best location among all the bats; f_{min} is usually set to 0 and $f_{max} = O(1)$; $I \in [1, n]$, which represents the number of bats; and n is the bat population, which always takes the value of 20.

For location search, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk

$$x_{new} = x_{old} + \theta A^t \quad (8)$$

where $\theta \in$ [-1, 1] = random number; and $A^t =$ average loudness of all the bats at time *t*.

in the process of bat predation, once the prey is found, the loudness usually decreases, while the rate of pulse emission increases. So, the loudness A^i and the rate of pulse emission r_i are updated based on Eqs. (9) and (10)

$$A_i^{t+1} = \alpha A_i^t \quad (9)$$

$$r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)) \quad (10)$$

where α = loudness decay factor; γ = pulse increase factor; and both of them are constants. Simply, we assume that α=γ, and they are restricted to (0.92, 0.98).

Generation of Initial Population

In general, the convergence of optimization largely depends on the distribution of bats in the search domain. For the purpose of increasing the diversity of the initial population, cubic mapping of chaos was introduced to generate the initial population

$$y(i+1) = 4y(i)^3 - 3y(i) \quad (11)$$

where i represents the number of bats; $-1 \leq y(i) \leq 1, i = 1, 2, \ldots\ldots$, n; and n represents the population of bats.

The procedure for generation of the initial population with cubic mapping list is as follows:

Step 1: Randomly generates a vector in the d-dimensions, domain, which means initialization of the first bat $Y_1^0 = (y_{1,1}^0, y_{1,2}^0, \ldots\ldots, y_{1,d}^o)$ where d represents the dimension of optimization;

Step 2: Generate the other $n-1$ bats by use of Eq. (11), where $Y_1^0 = \left(y_{1,1}^0, y_{1,2}^0, \ldots\ldots, y_{1,d}^o\right)$

Step 3: Use Eq, (12) to map the population Xgenerated by cubic mapping into the search domain of optimization

$$x_{i,d}^0 = x_{d,min} + \left(1 + y_{i,d}^0\right)\frac{x_{d,max} - x_{d,min}}{2} \quad (12)$$

Step 4: Start optimization if X_i^0 satisfies the constraint condition; otherwise return to step 1.

Procedure for SVM Parameters Optimization by Bat Algorithm

The process for optimizing SVM parameters is as follows:

Step 1: In preparation for optimization, choose the objective function $f(x) = (x_1, x_2, \ldots\ldots, x_d)^T$, where d represents the

dimension parameters of interest; in this study, cross validation based mean square error (MSE) was chosen as the objective function (fitness function), d was set to 1, and x_1 is the penalty parameter (C);

Step 2: Generate the initial population of n bats including frequency, velocity, position, loudness, and pulse emission rate, with the initial positions generated by cubic mapping as mentioned above;

Step 3: Determine the global optimal solution. Calculate the initial fitness, find the global optimal solution of all bats, and assign the initial fitness to the initial loudness;

Step 4: Global search. Update the position and velocity of all bats based on Eqs. (6) and (7); make sure the current position and velocity do not exceed the search domain, and if not, use the upper or lower bounds to replace the corresponding variable; recalculate the fitness and global optimal solution;

Step 5: Local search. Calculate the new position of the bat according to Eq. (8) while the bat meets the local search condition (rand > r_i), afterwards, calculate the new fitness;

Step 6: Determine whether the new position updated in Step 5 was accepted or not. Accept the new position if new fitness is smaller than the previous fitness calculated in Step 4 and rand < A_i; after that, calculate loudness and the rate of pulse emission according to Eqs. (9) and (10);

Step 7: Determine the current global optimal solution. For all bats, if the current fitness is smaller than the historic global optimal solution, then assign the value of current fitness to the global optimal solution history; in the end, assign the value of global optimal solution history to current global optimal; and

Step 8: Check whether the calculation can be stopped or not. Stop the calculation if the maximum iterations are reached or if the difference between the current global optimal and the previous one is less than 10−3; otherwise, repeat Steps 3 and 4 until the termination criterion is reached.

Deep Neural Network (DNN)

A Neural Network is similar to a neuron, where input is taken and after some processing the output is fired.

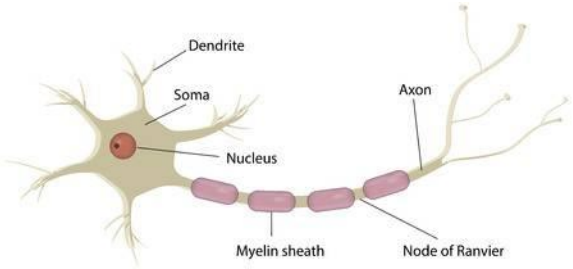

Figure 1 A Natural Neuron

In an ANN (artificial neural Network) each link has a weight (w) and bias (b). the output from a neuron is taken as

$$sum = \sum_{i=1}^{n} (w_i\, x_i + b_i)$$

where i represents the number of elements in a neuron, i = 1, 2, …. n. The sum is passed through an activation function called sigmoid function. The sigmoid function fires when a certain condition is met. So, an ANN actually fires or does not.

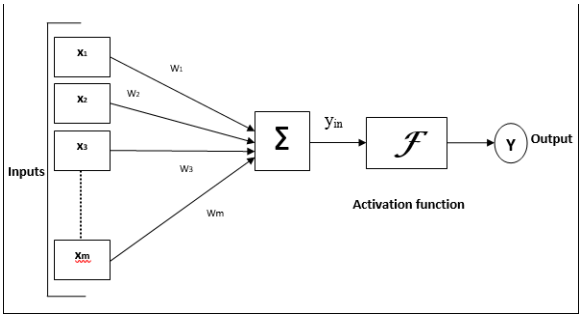

Figure 2 An Artificial Neuron

When in an ANN the number of hidden layers becomes more than one it becomes a Deep Neural Network (DNN). In a DNN each and every element of a layer is liked with each and every element of the next layer as shown. Each link has its own weight (w) and bias (b). The output from each layer is calculated as mentioned above and each layer has its own sigmoid function, which either fires or does not.

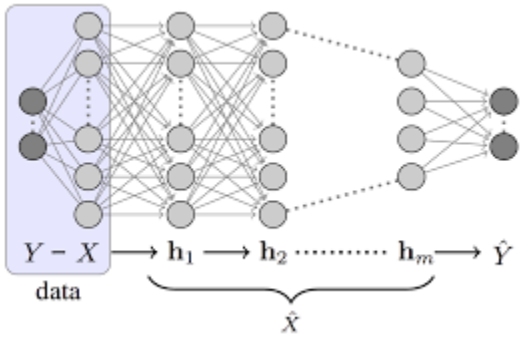

Figure 3 Deep Neural Network

The final layer in the DNN is the output layer. It represents the number of cases in which an output is desired. For example, if the ADNN is used to classify the digits then there are 10 possible cases: 1, 2, 3, 4, 5, 6, 7, 8 9, 0. If the model is used for predicting a certain value that is either above a threshold or not it outputs in yes(1) on no(0). The only caveat with an ADNN is that it requires large amounts of data for training purpose.

Study Area and Data Sets

Study Area

In this project precipitation data from data.gov.in site was used. The data available was from a duration of 1901 to 2017. However, specifically for training purpose the data from East UP was fed into the models. For training purpose data from the year 1901 to 2015 was used, the rest of the data was used for testing and result generation. The rainfall data available to us was in mm. The data was first processed to remove all the malicious data that could create problem to the model. The processed data was used to create the model. If for some reason error occurs while training the data the data should be normalized to [0, 1] through

$$R' = \frac{R_i - R_{min}}{R_{max} - R_{min}}$$

before feeding it into the model. Where $R' =$ scaled value, dimensionless; $R_i, R_{max}\ and\ R_{min}$ represent the original monthly precipitation values.

Model Structures

One of the most important steps in developing of a satisfactory forecasting model is the selection of input variables. The input variables should be selected such that there are enough parameters for the model. For this project six different model types were selected as listed in Table 1.

In Table 1, $y(t)$ represents the monthly precipitation that needs to be predicted; $x(t-1), x(t-2), \ldots., x(t-6)$ represent the precipitation in the same month for the previous years. For Example, if we are generating a dataset for the month of January of the year 2000 $y(t)$, to be predicted, then

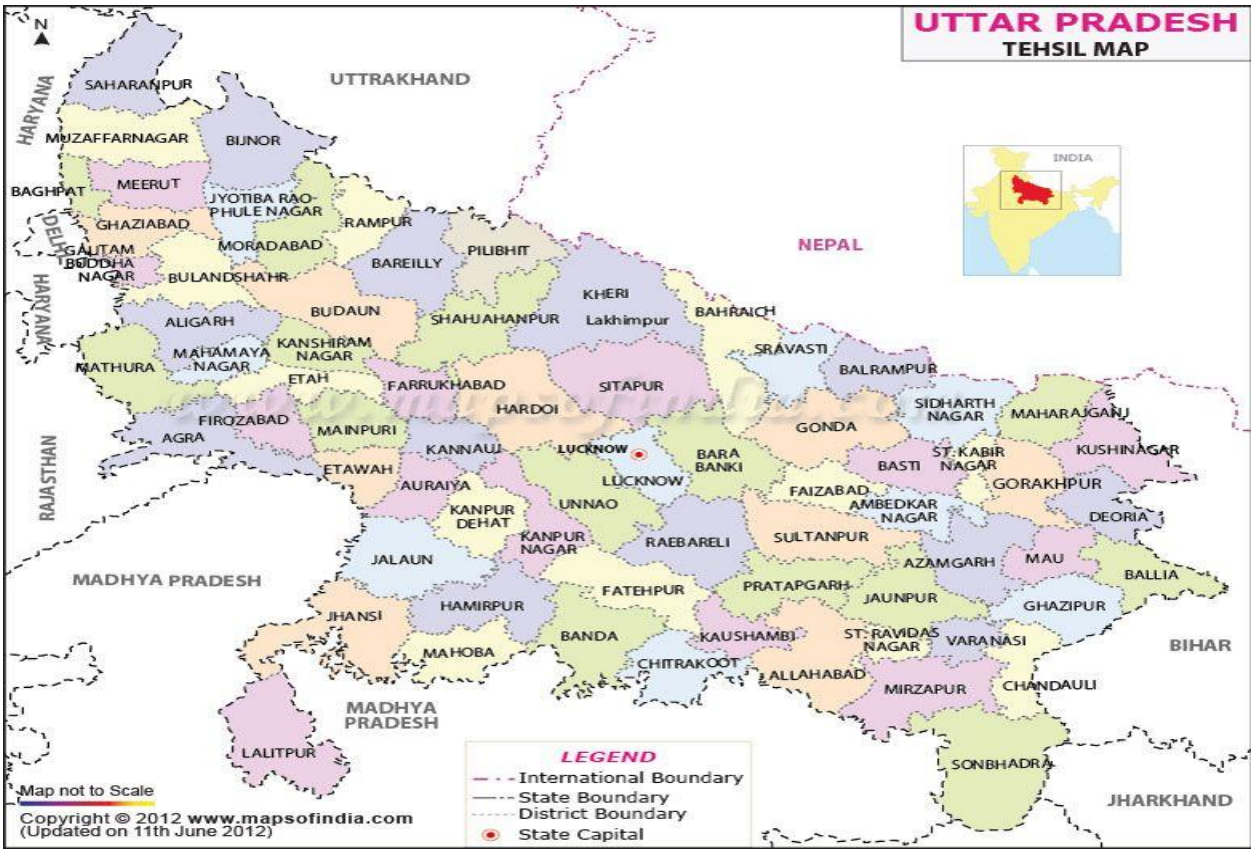

Figure 4: Map of Uttar Pradesh

$x(t-1)$ represents the rainfall in the month of January in the year 1999, and so on.

Table 1. Model Structures for Forecasting Precipitation

Model	Model Input
M1	$y(t) = f[x(t-1)]$
M2	$y(t) = f[x(t-1), x(t-2)]$
M3	$y(t) = f[x(t-1), (xt-2), x(t-3)]$
M4	$y(t) = f[x(t-1), x(t-2), x(t-3), x(t-4)]$
M5	$y(t) = f[x(t-1), x(t-2), x(t-3), x(t-4), x(t-5)]$
M6	$y(t) = f[x(t-1), x(t-2), x(t-3), x(t-4), x(t-5), x(t-6)]$

Measures of Accuracy

The performances of the models in monthly mean streamflow prediction training and testing are evaluated by using various standard statistical performance evaluation criteria—in this paper the root mean square error (RMSE), the mean absolute percentage error (WMAPE), the correlation coefficient (R), and the coefficient of efficiency (CE), which are defined respectively from Eq. (14)– (17).

The RMSE evaluates how closely the predictions match the observations; the WMAPE measures the absolute percentage error of the prediction; R measures how well the predicted runoff correlates with the observed runoff; and the CE reveals how well the predicted time series matches the observed time series. Smaller RMSE and WMAPE values imply a better prediction model. Likewise, bigger R and CE imply a better prediction model -

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Q_{o,t} - Q_{m,t})^2} \quad (14)$$

$$WMAPE = \frac{\sum_{t=1}^{n}\left|\frac{Q_{0,t} - Q_{m,t}}{}\right| Q_{o,t}}{\sum_{t=1}^{n} Q_{0,t}} \quad (15)$$

$$R = \frac{\sum_{t=1}^{n}(Q_{o,t} - \overline{Q_{0,t}})(Q_{m,t} - \overline{Q_{m,t}})}{\sqrt{\sum_{t=1}^{n}(Q_{o,t} - \overline{Q_{0,t}})^2}\sqrt{\sum_{t=1}^{n}(Q_{m,t} - \overline{Q_{m,t}})^2}} \quad (16)$$

$$CE = 1 - \frac{\sum_{t=1}^{n}(Q_{m,t} - \overline{Q_{m,t}})^2}{\sum_{t=1}^{n}(Q_{o,t} - \overline{Q_{o,t}})^2} \quad (17)$$

Where $Q_{o,t}$ and $Q_{m,t}$ represent the observed rainfall and model output respectively; $\overline{Q_{o,t}}$ and $\overline{Q_{m,t}}$ represent the mean observed and modeled precipitation series, respectively; and n denotes the number of input vector.

Discussion and Result

Fitting an ANN model to the Data

The DNN used in the project was a 3 hidden layer feed forward and back propagation with one input and output layer each. A sigmoid function was considered as the activation function for each neuron. The termination criteria were set to 5000 epochs and the complete dataset was fed through the network in each epoch. TensorFlow was used as a tool to create the neural network with the code being written in Python programming language. Table 2 (page 12) represents the efficiency of different models.

Four different DNN models were tested with the number of hidden neurons as follows: 7-3-6, 7-10-14, 7-14-15, 7-500-500. All the networks were trained on each of the training models M1-M6.

As Table 2 clearly shows that all the models yielded same efficiency, this means that the network has failed on our dataset. However, for such a small dataset custom activation function and accurate number of hidden layers with accurate number of neurons in each layer is required.

Fitting an SVM Model to the Data

For this project purpose the SkLearn module for SVM was used. Two parameters were set for the training purpose i.e., C = **set** and the kernel function as rbf (radial bias function). C is the penalty parameter; in layman's term it is the width of the support vector line that an SVM generates. Kernel Function is used to change the dimension of the data, which makes the construction of support vector easy. Each of the models were trained using the SVM. As it was intended as the number of input parameters increased the accuracy of the model started increasing. The run time of the code ranged from 3-5 minutes. For experimental purpose the model could also be trained by changing the parameters of the SVM.

Table 3 (page 13) shows that the minimum RMSE (35.0405) and WMAPE (0.34097) The maximum R (0.953917) and CE (0.880444) in the testing phase were observed from M6. The best performance in the testing phase (**RMSE =** 35.0405**,**

Table 2: Prediction Performance of ADNN structures

Model		Testing				
		RMSE	WMAPE	R	CE	Accuracy
M1	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333
M2	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333
M3	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333
M4	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333
M5	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333
M6	7-14-15	120.541	1	0	-0.41483	20.8333
	7-10-14	120.541	1	0	-0.41483	20.8333
	7-3-6	120.541	1	0	-0.41483	20.8333
	7-500-500	120.541	1	0	-0.41483	20.8333

Note 1:Best Performance in Bold

WMAPE = 0.34097**, R** = 0.953917**,** and **CE** = 0.880444**)** was recorded from M6. According to the criteria of optimal model selection, M6 should be selected for the rainfall prediction.

Fitting a Bat Algorithm based SVM to the Data

Bat algorithm was proposed based on the features of the bats' echolocation; they automatically adjust the wavelength of their emitted pulses and adjust the rate of pulse emission depending on the proximity of their target (Yang and He 2013). The results depend in the range of frequencies. The maximum frequency depends largely on the domain size of the optimization object. In this project maximum frequency was set to 2. The following parameters were set for the BAT algorithm: Dimension = 2, Population Size = 20, Generations = 1000, Loudness = 0.05, Pulse Rate = 0.005, Minimum Frequency = 0.o, Maximum Frequency = 2.0, Lower Bound = 0.98, Upper Bound = 1.38. The value of C was found to 1.3302. It takes about 3-4 minutes to run the code. The modelling parameters if all the models are shown in Table 4 (Page 14).

Table 3: Prediction Performance Indices of SVM

	Testing				
Model	RMSE	WMAPE	R	CE	Accuracy
M1	50.5887	0.42946	0.880727	0.750805	75.0805
M2	46.8567	0.41666	0.946566	0.786215	78.6215
M3	44.2425	0.41226	0.946759	0.809405	80.9405
M4	43.9742	0.41348	0.931972	0.811709	81.1709
M5	42.8812	0.37173	0.950680	0.820953	82.0953
M6	**35.0405**	**0.34097**	**0.953917**	**0.880444**	**88.0444**

Note 2: Best Performance indicated in Bold

Table 4: Prediction Performance Indices of BAT-SVM

	Testing				
Model	RMSE	WMAPE	R	CE	Accuracy
M1	51.0677	0.430087	0.880485	0.746064	74.6064
M2	46.0921	0.404893	0.944427	0.793136	79.3136
M3	45.8188	0.436303	0.947032	0.795582	79.5582
M4	44.6193	0.414139	0.934233	0.806144	80.6144
M5	43.2771	0.379074	0.947504	0.817632	81.7632
M6	**40.2331**	**0.370372**	**0.952324**	**0.842385**	**84.2385**

Note 3: Best performance Indicated in Bold

Table 5: Performance of ADNN, SVM, BAT-SVM

	Testing				
Model	RMSE	WMAPE	R	CE	Accuracy
ADNN	120.541	1	0	-0.41483	20.8333
SVM	**35.0405**	**0.34097**	**0.953917**	**0.880444**	**88.0444**
BAT-SVM	40.2331	0.370372	0.952324	0.842385	84.2385

Note 4: Best Performance indicated in Bold.

M6 shows the best outcome in testing phase. From the above, M6 was chosen to represent the best performance of BAT-SVM.

The complete BAT SVM was created on Python Programming Language with the SkLearn Module.

Comparison of the Three Models

In order to verify the effectiveness of the proposed model, the best performances of each method (ADNN, SVM, BAT-SVM) were compared in terms of the statistical error involving RMSE, WMAPE, R and CE. The results are shown in Table 5.

Table 5 represents the statistical performance criteria in SVM is superior than to those in BAT-SVM and ADNN. During training the ADNN model took the least time to run, but, failed starkly on the dataset. However, BAT-SVM and SVM took almost equal time to run, but, the best result was obtained from the SVM model.

The observed and calculated precipitation by the ADNN, SVM and BAT-SVM are shown in figure 5. The SVM and BAT-SVM model

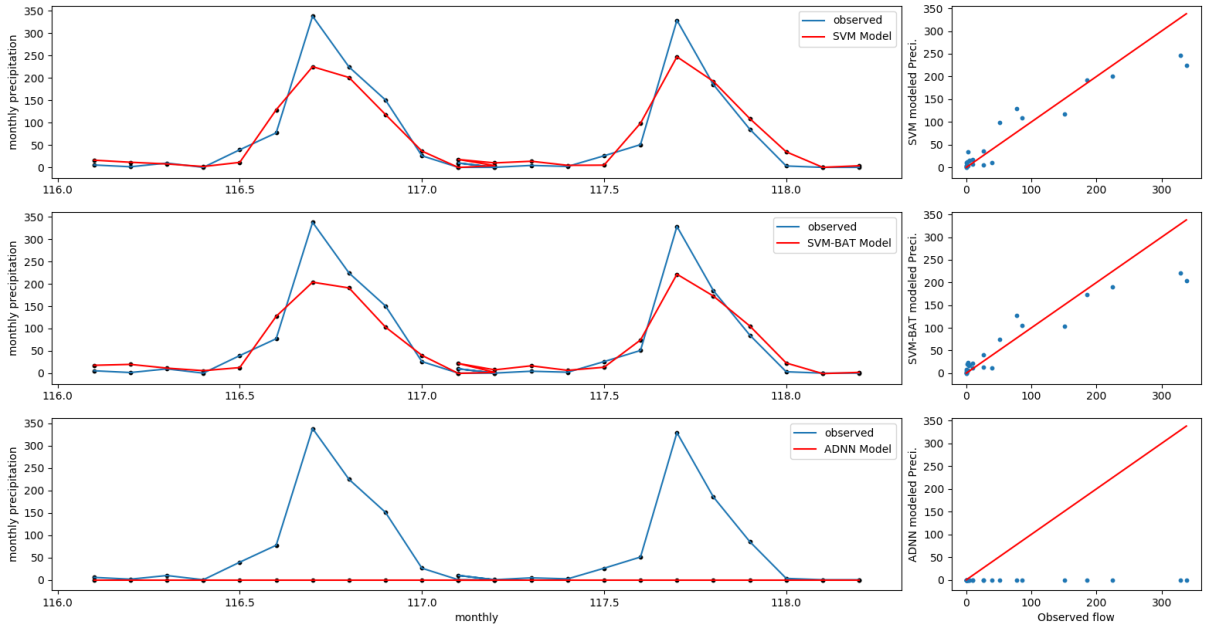

Figure 5: Observed and Predicted Precipitation by SVM, BAT-SVM, ADNN

give nearly close approximation of the precipitation. The SVM model tested line is closest to the observed line in comparison to the line from SVM model. Albeit, the differences between the predicted and observed extreme precipitation modelled from SVM and BAT-SVM are very small, the SVM shows better results than the BAT-SVM. The prediction accuracy for the continuous wet dry years was higher than that of abrupt years.

The fitted line equation and the coefficient of efficiency between calculated and observed precipitation obtained from SVM is superior to the other models. This means that SVM performance is better than ADNN and BAT-SVM in the monthly precipitation prediction.

On the whole, the results indicate that SVM model is powerful technique to model the monthly precipitation and can provide a better performance compared to ADNN and BAT-SVM.

Conclusion

Medium-term and long-term runoff prediction is an important part of engineering hydrology; it is also of great importance to water resources management. There are many different ways to predict precipitation. In this project SVM, BAT-SVM and ADNN were used to predict the monthly rainfall. For the purpose of evaluating the performance of the models, the monthly rainfall data from data.gov.in site was used. The data was of East Uttar Pradesh from the year 1901 to 2107. Six different model structures with various input variables were suggested to develop a satisfactory prediction model. The statistics RMSE, WMAPE, R and CE were evaluated in the testing phase. The results show that M6 performs the best.

After comparing the best fitted ANN, SVM, BAT-SVM models, it is clear that the SVM model shows the best efficiency.

The result from each model is provided at the back.

References

Scikit Learn
https://scikit-learn.org/stable/

SkLearn SVM
https://scikit-learn.org/stable/modules/svm.html

Matplotlib
https://matplotlib.org/

American Society of Civil Engineers
(ASCE)HE.1943-35584.0001269
ASCE 2000: Behzad et al. 2009

List of Figures

Figure 1 A Natural Neuron 9
Figure 2 An Artificial Neuron..... 10
Figure 3 Deep Neural Network..... 10
Figure 4: Map of Uttar Pradesh 11
Figure 5: Observed and Predicted Precipitation by SVM, BAT-SVM, ADNN 15
Figure 6: Result- SVM with M6. 20
Figure 7: Result- SVM with M5. 21
Figure 8: Result- SVM with M4. 22
Figure 9: Result- SVM with M3. 23
Figure 10: Result- SVM with M2. 24
Figure 11: Result- SVM with M1. 25
Figure 12: Result- BAT_SVM with M6. 26
Figure 13: Result- SVM_BAT with M5. 27
Figure 14: Result- SVM_BAT with M4. 28
Figure 15: Result- SVM_BAT with M3. 29
Figure 16: Result- SVM_BAT with M2. 30
Figure 17: Result- SVM_BAT with M1. 31
Figure 18: Result- for all the ADNN models. 32
Figure 19: Comparison of best from all the three models. 33

List of Tables

Table 1. Model Structures for Forecasting Precipitation 11
Table 2: Prediction Performance of ADNN structures 13
Table 3: Prediction Performance Indices of SVM 14
Table 4: Prediction Performance Indices of BAT-SVM 14
Table 5: Performance of ADNN, SVM, BAT-SVM 14

RESULTS

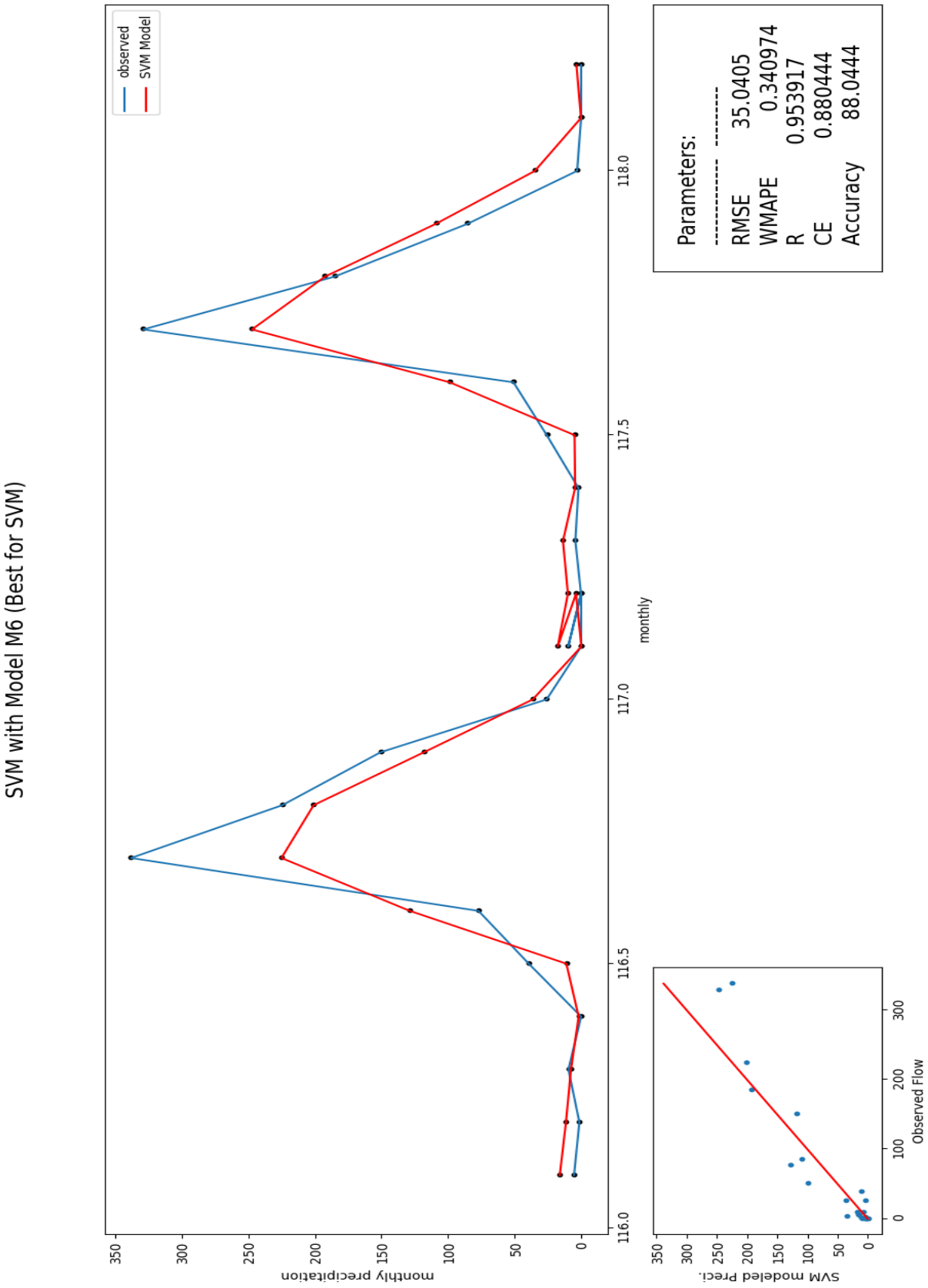

Figure 6: Result- SVM with M6.

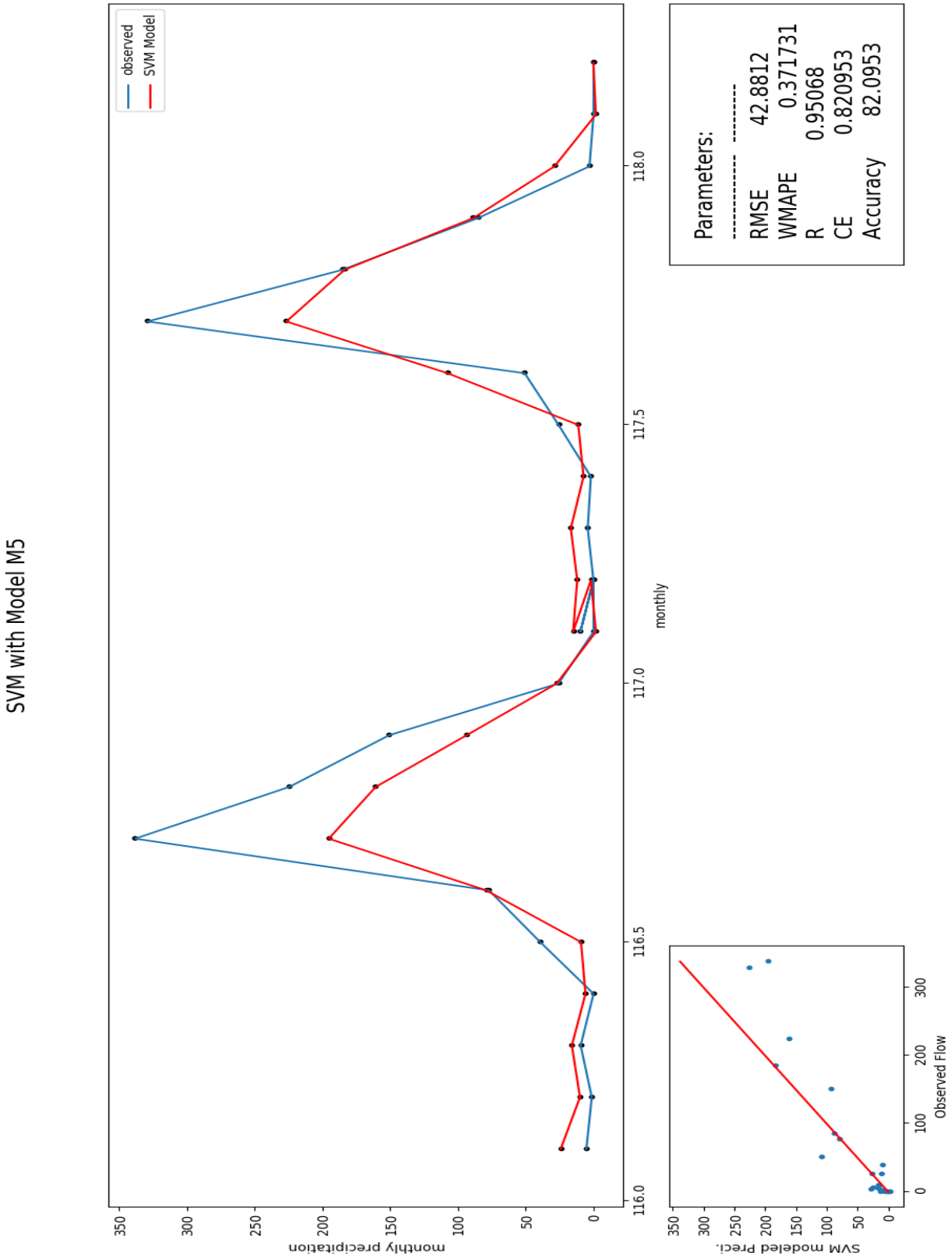

Figure 7: Result- SVM with M5.

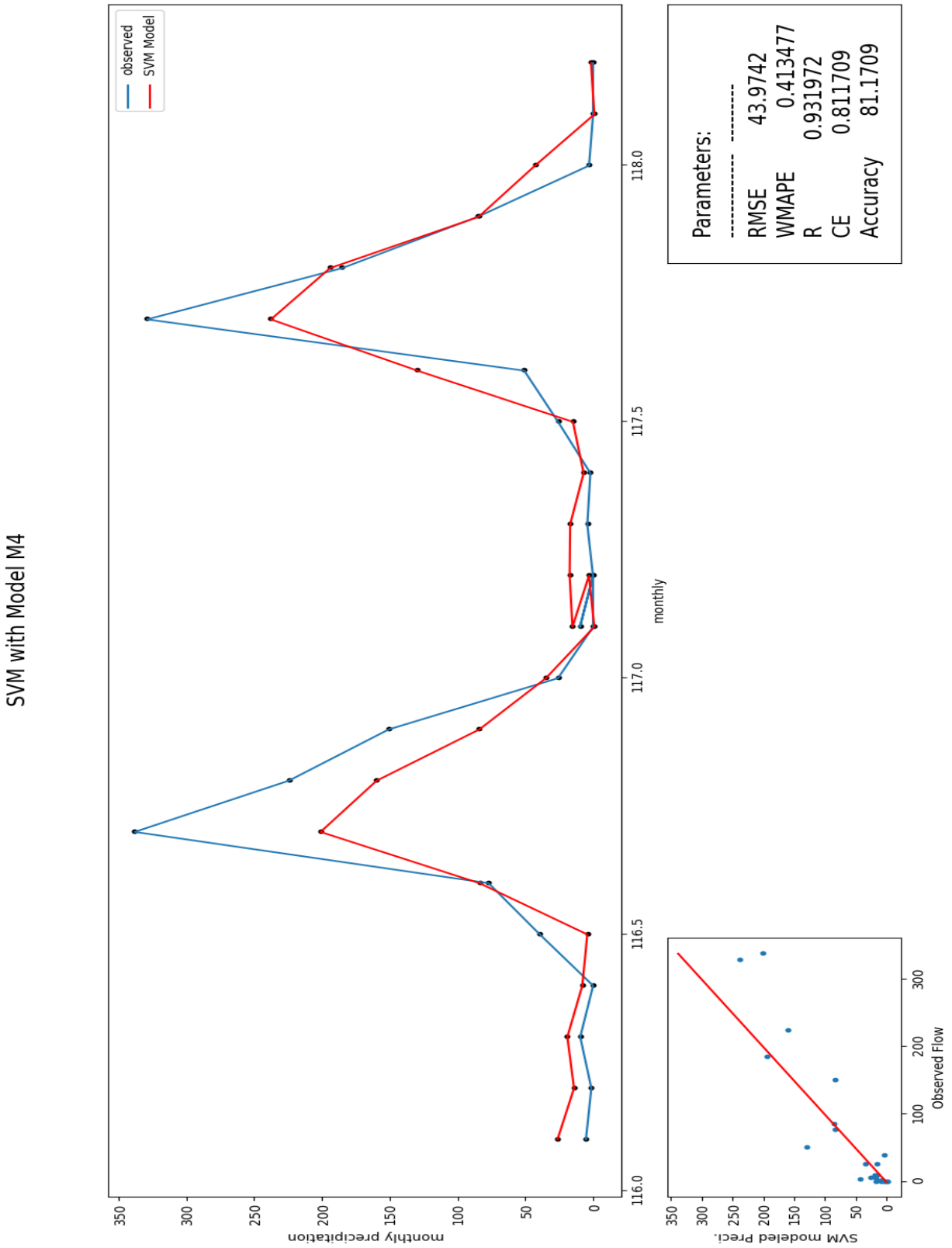

Figure 8: Result- SVM with M4.

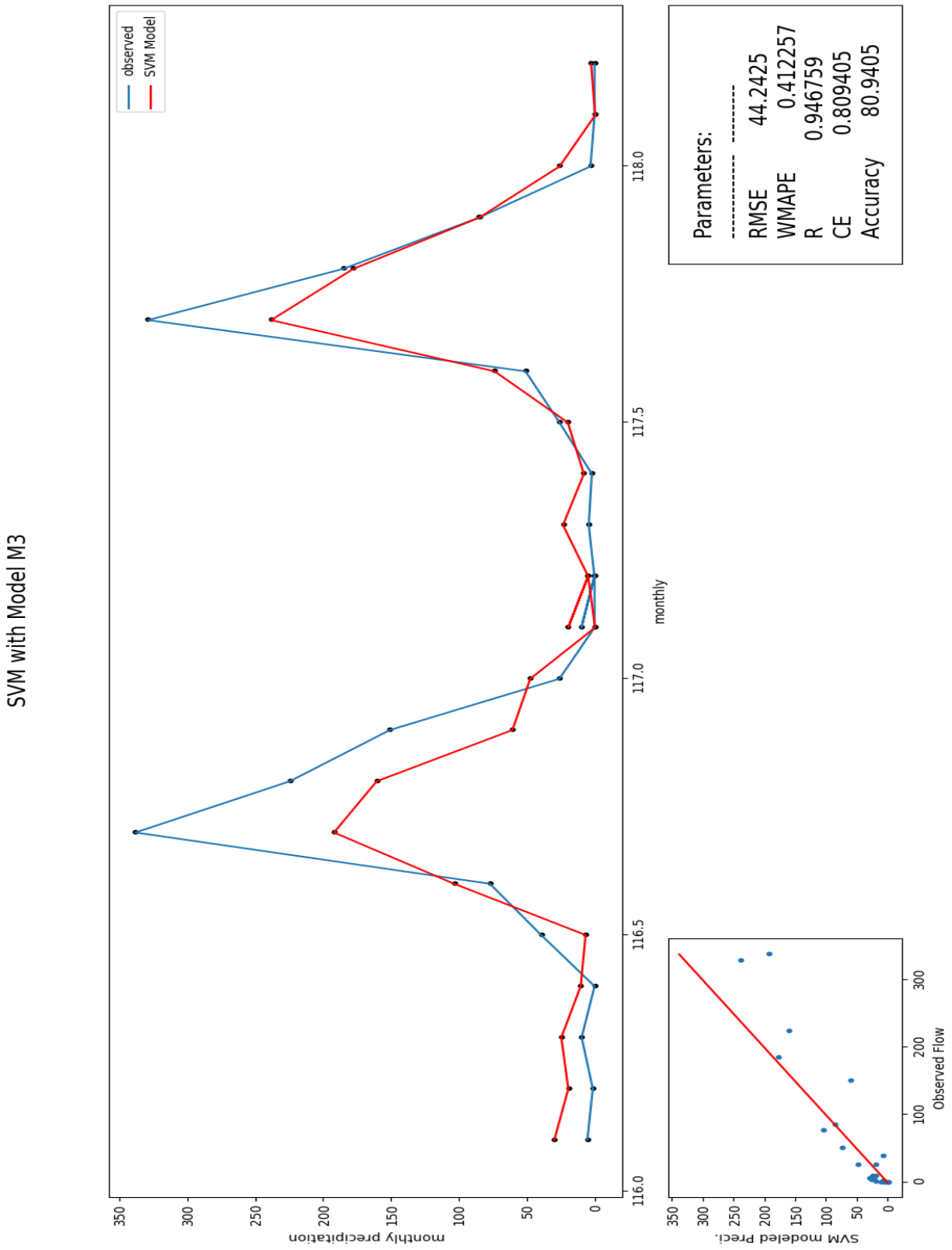

Figure 9: Result- SVM with M3.

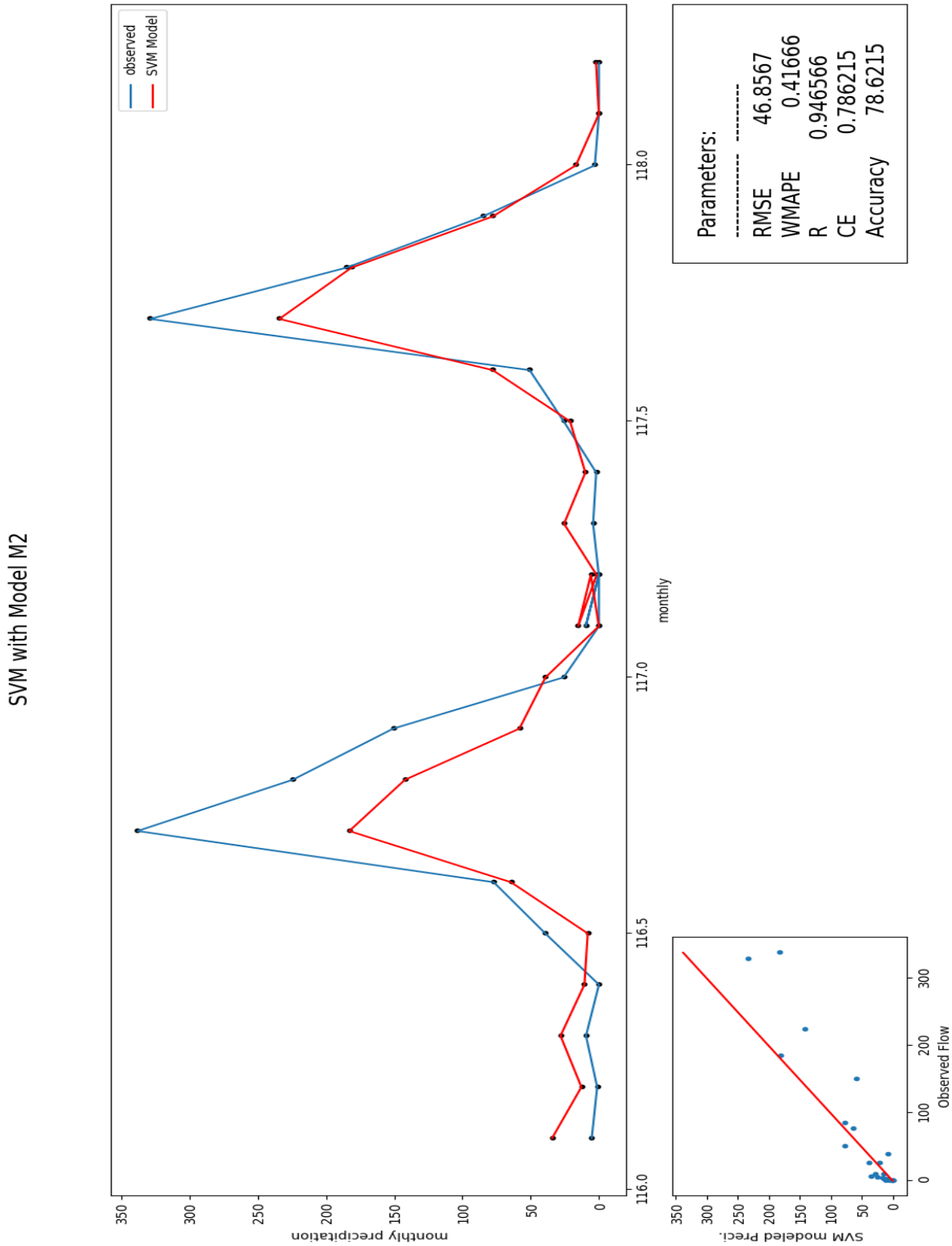

Figure 10: Result- SVM with M2.

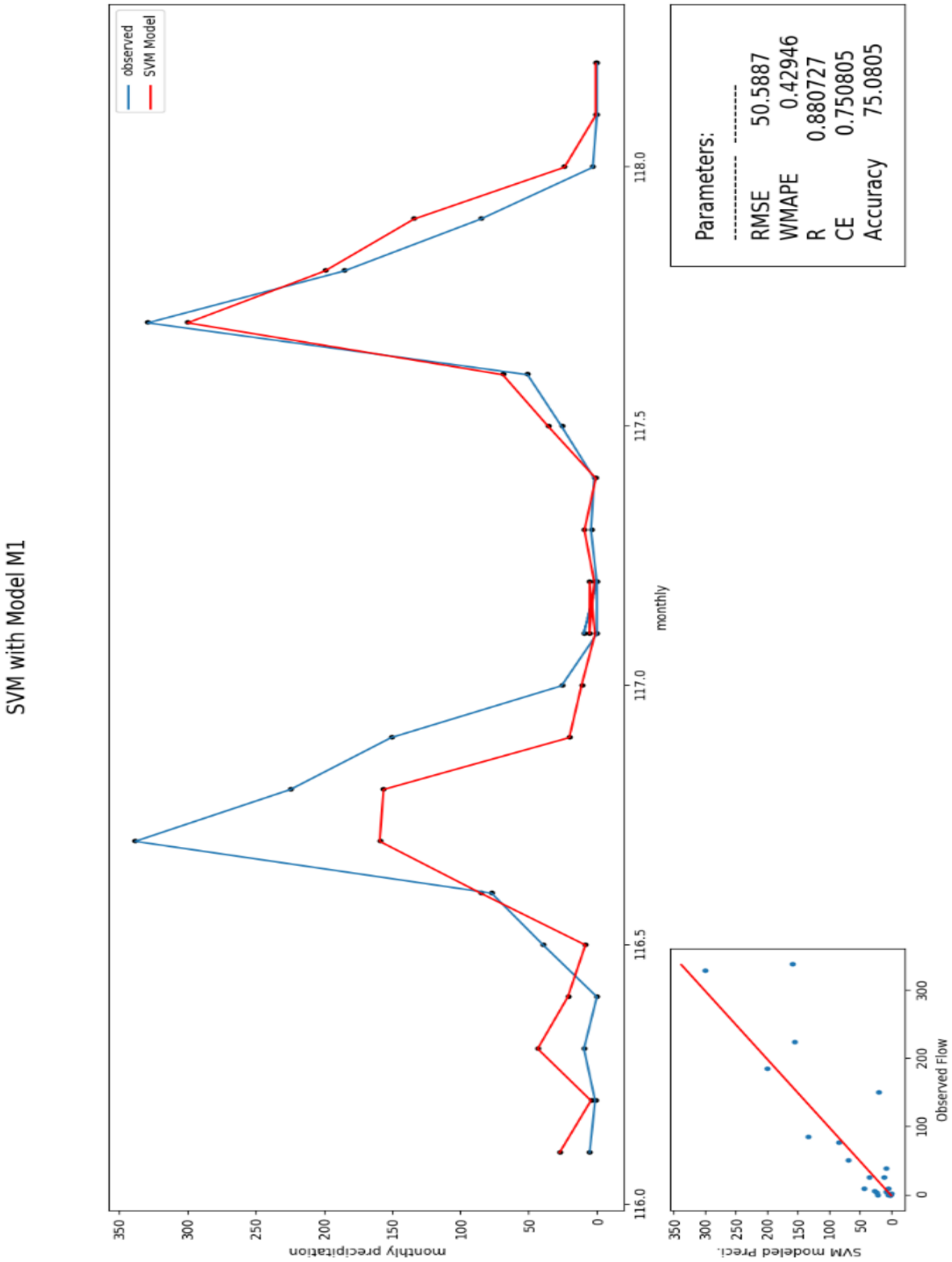

Figure 11: Result- SVM with M1.

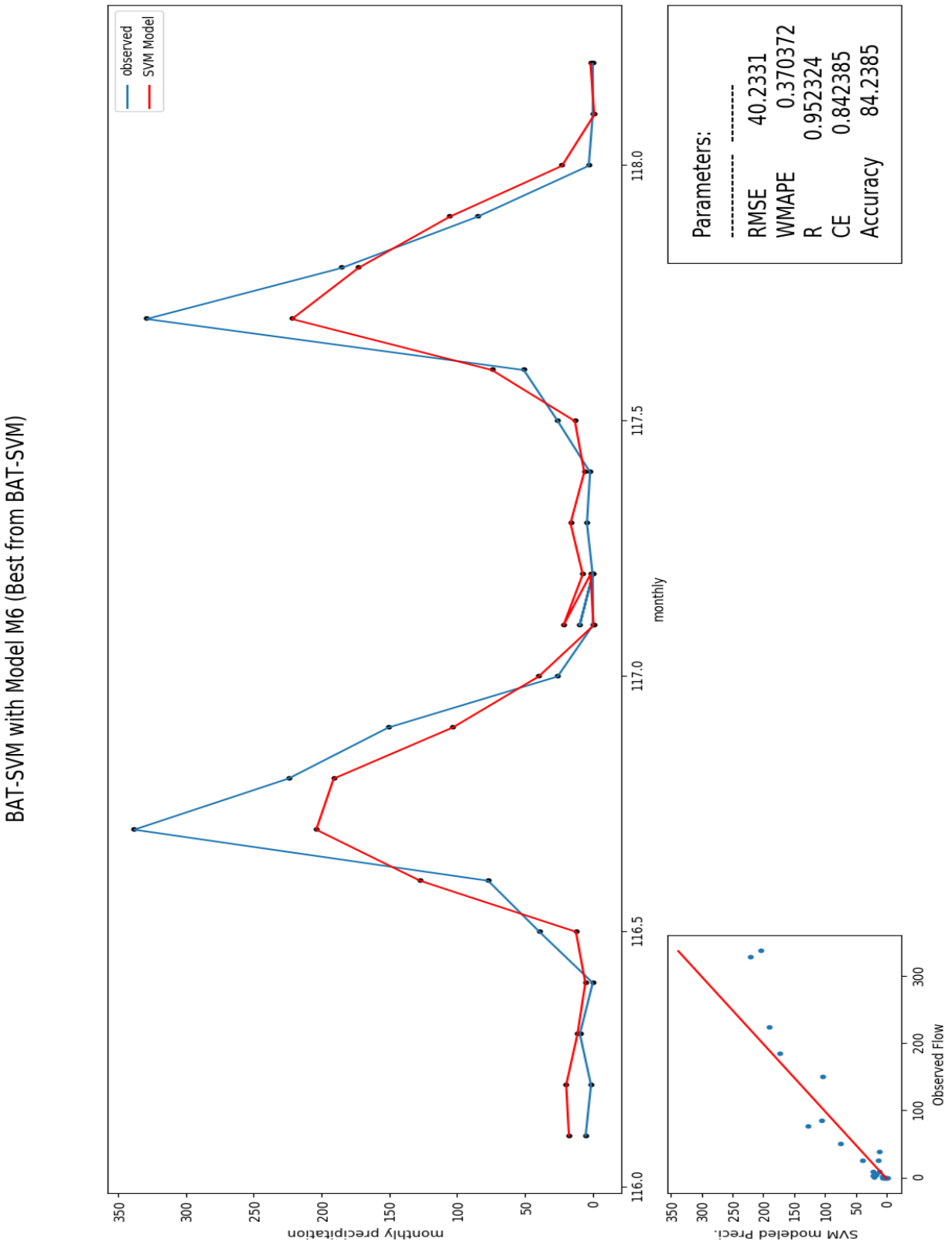

Figure 12: Result- BAT_SVM with M6.

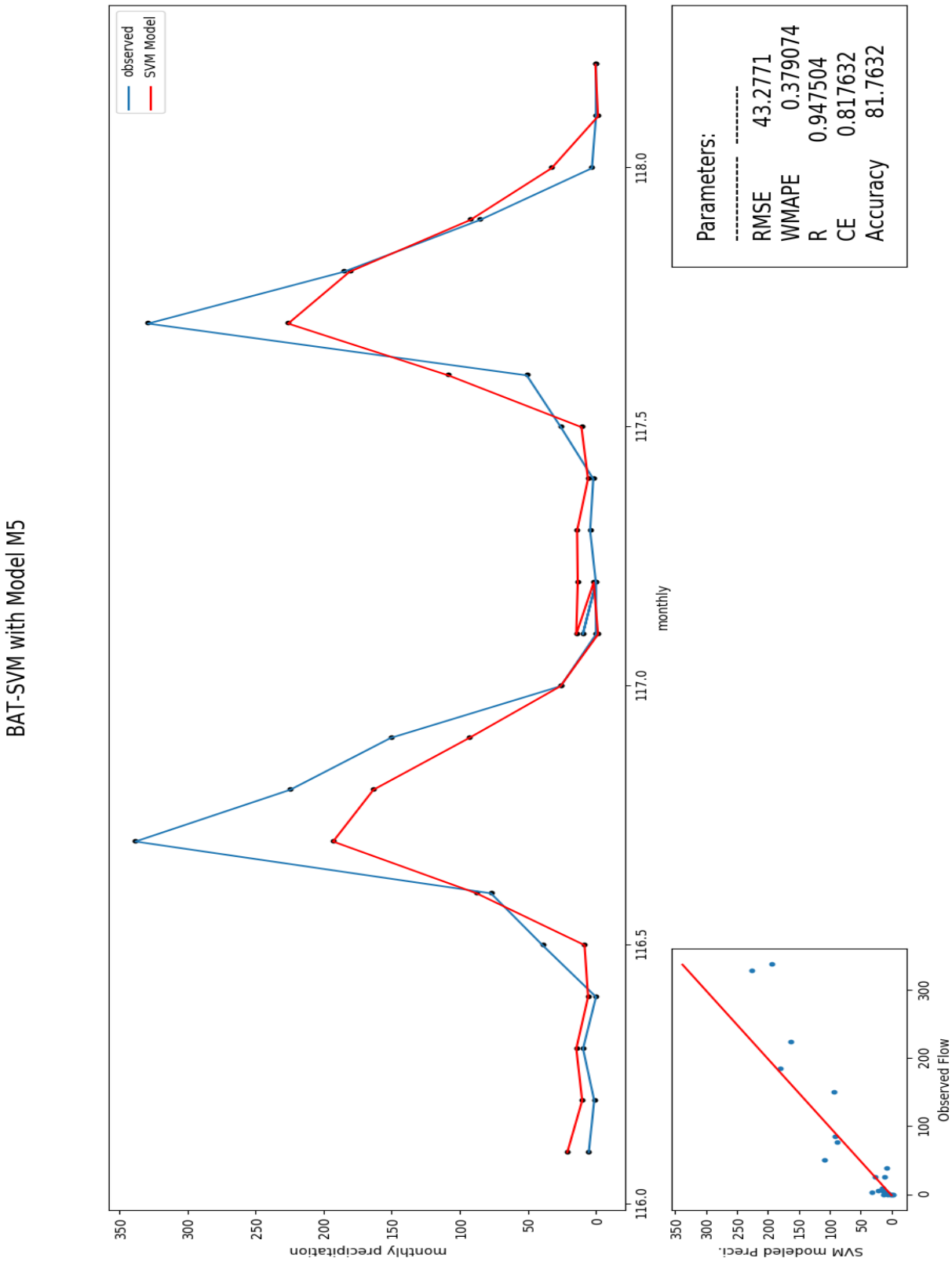

Figure 13: Result- SVM_BAT with M5.

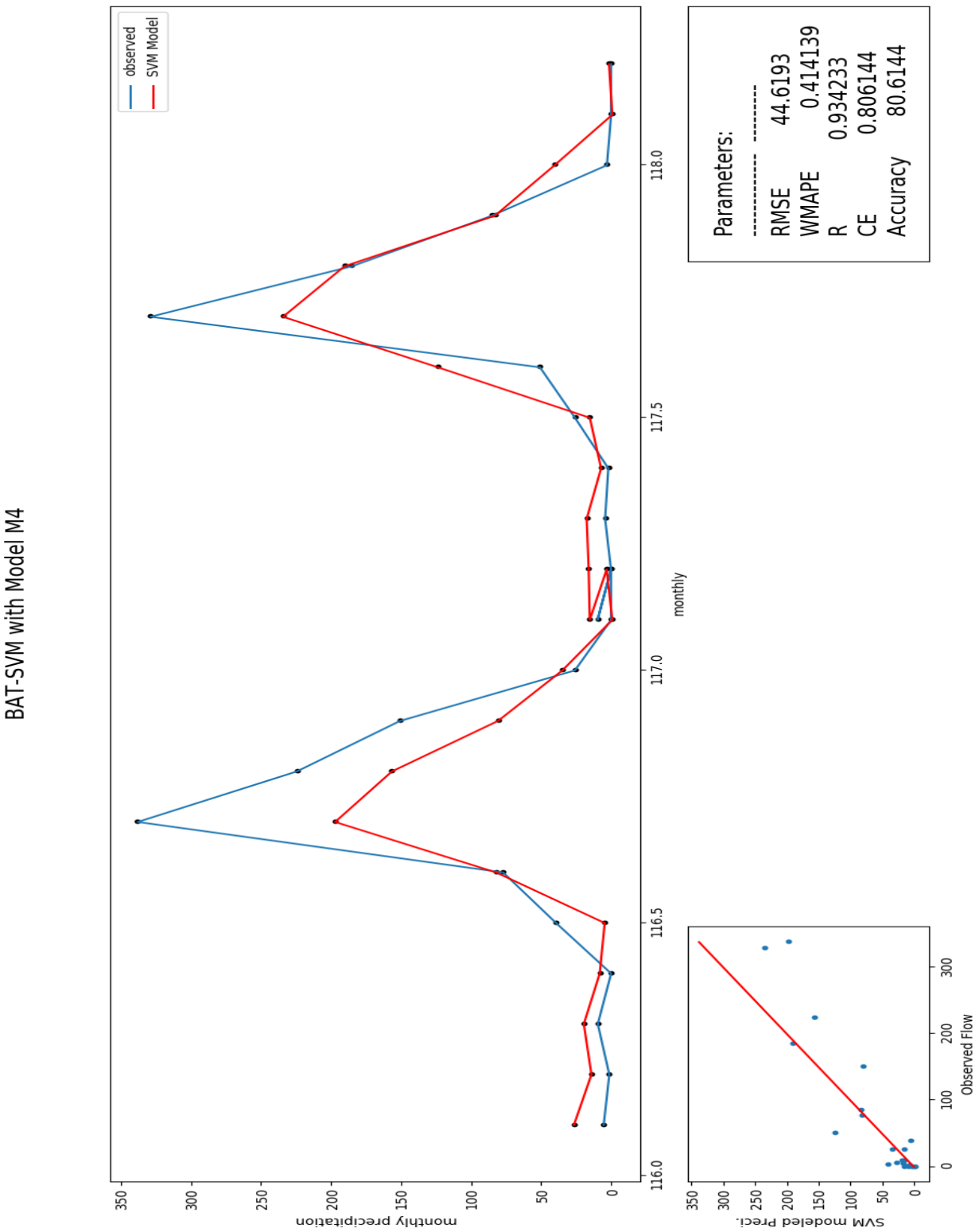

Figure 14: Result- SVM_BAT with M4.

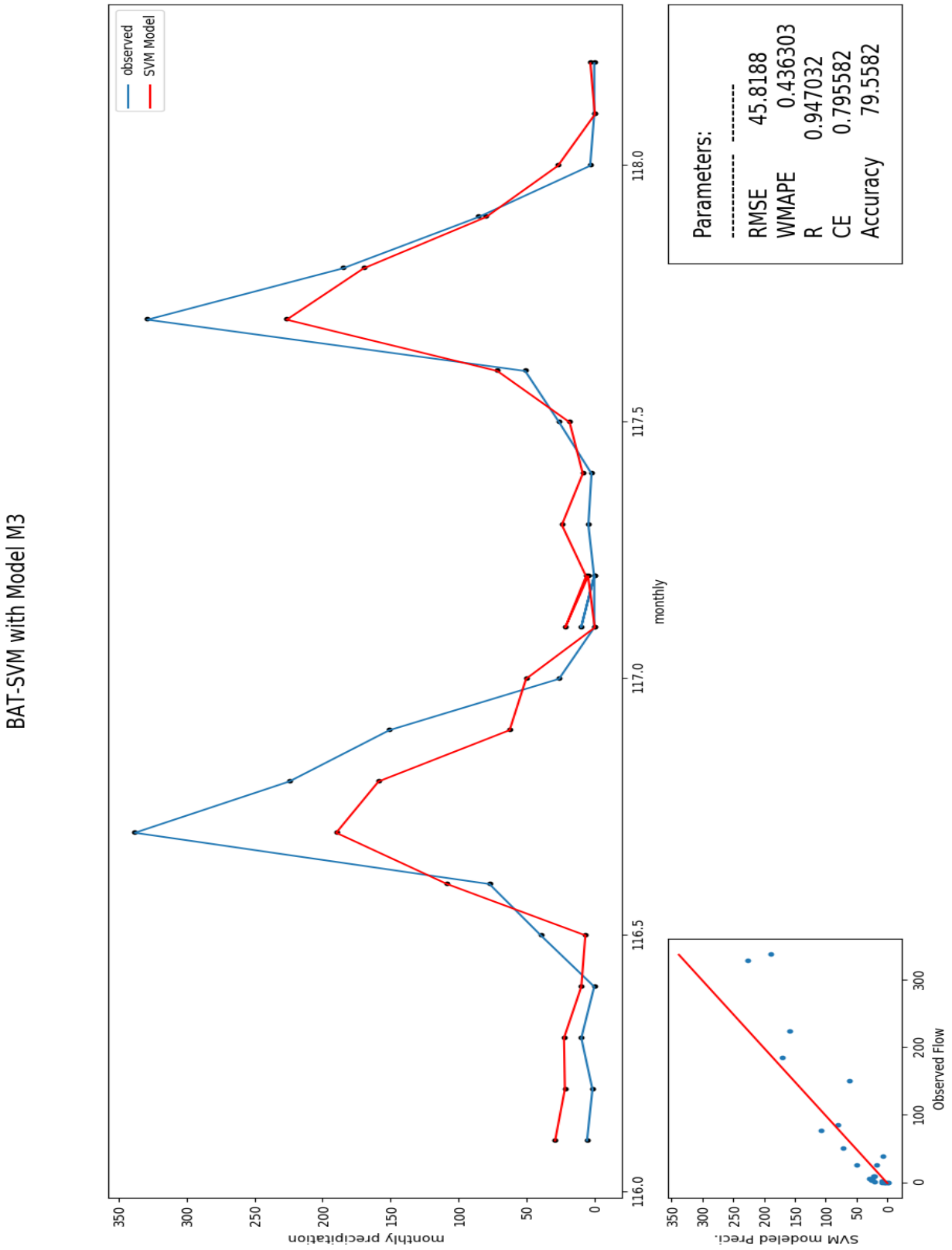

Figure 15: Result- SVM_BAT with M3.

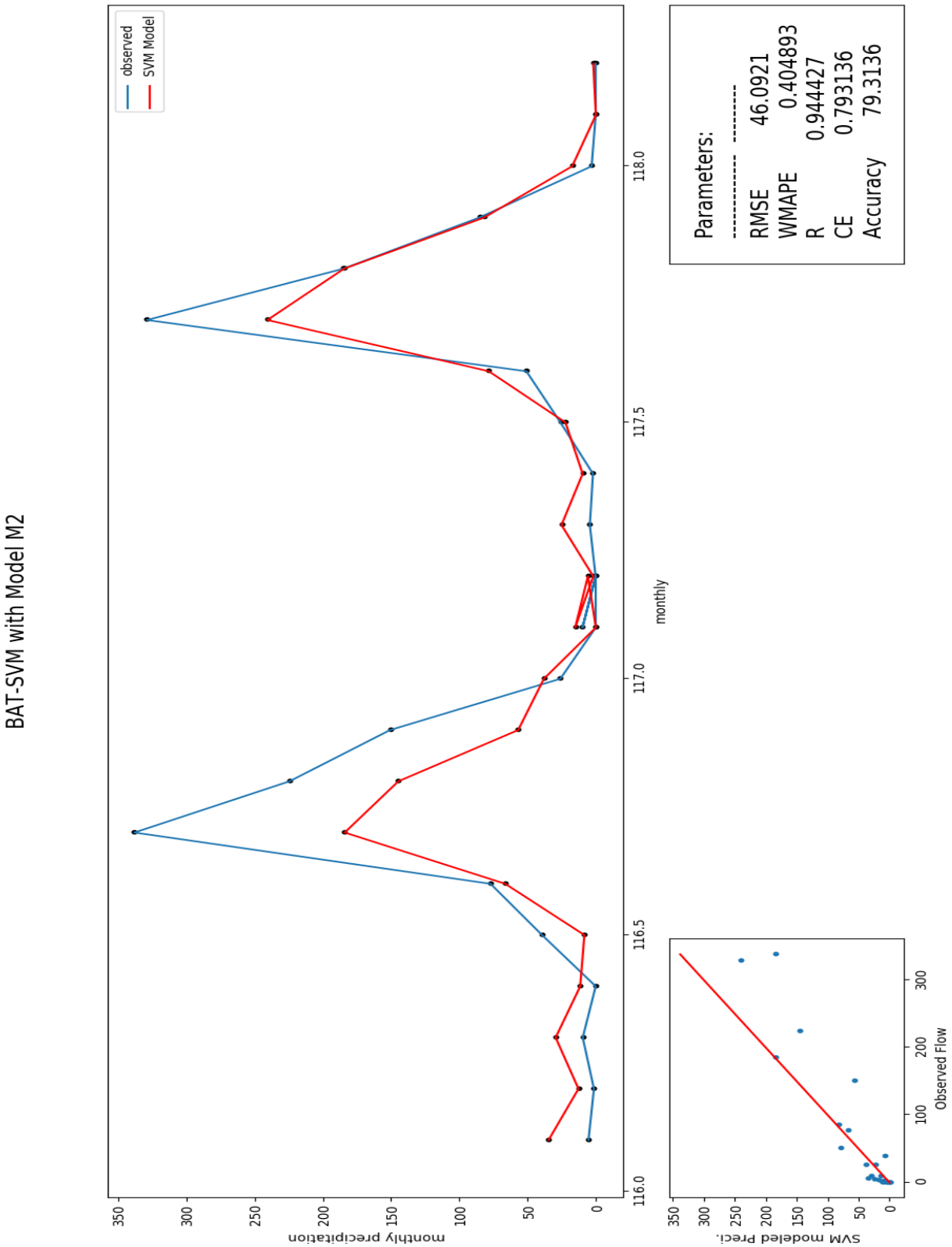

Figure 16: Result- SVM_BAT with M2.

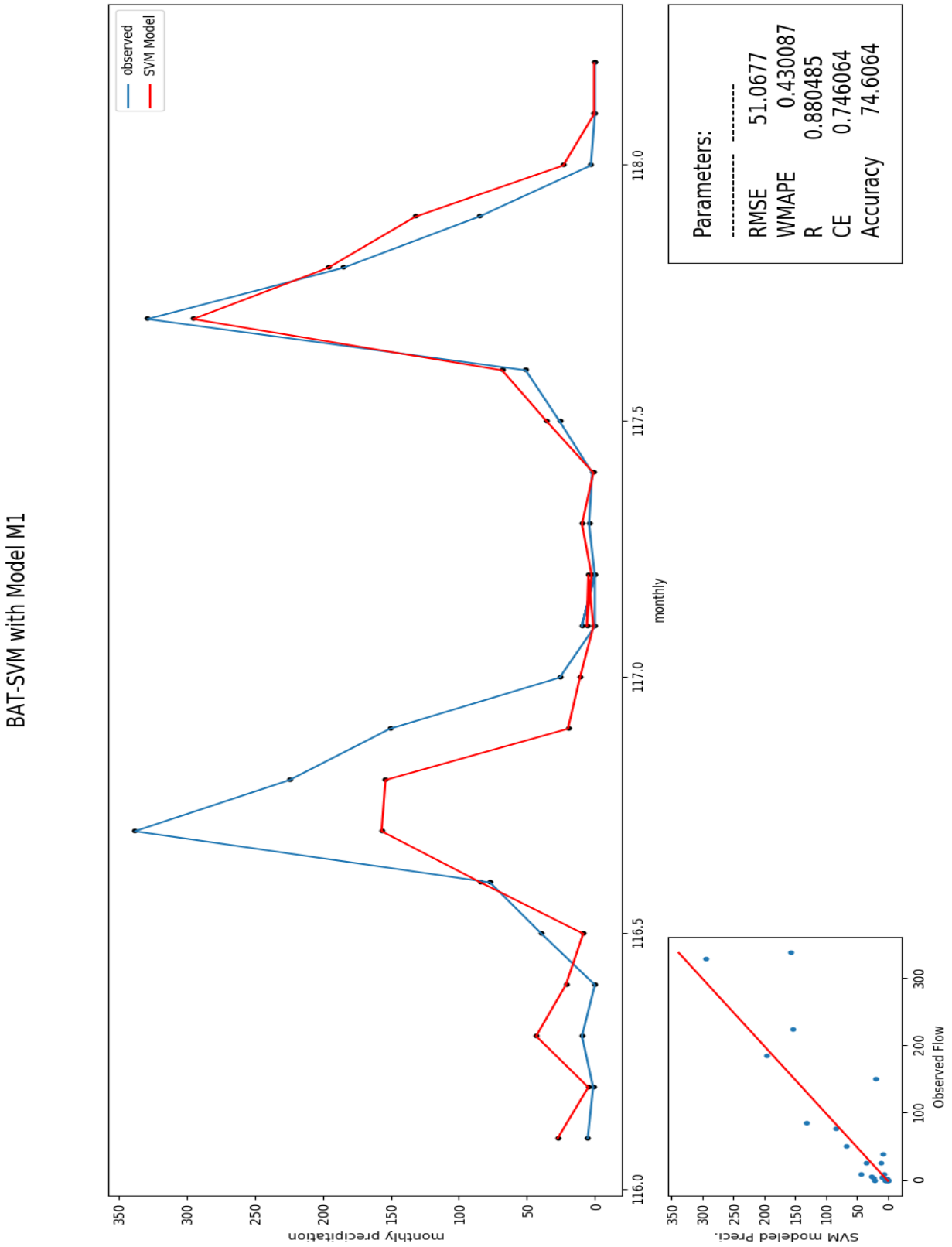

Figure 17: Result- SVM_BAT with M1.

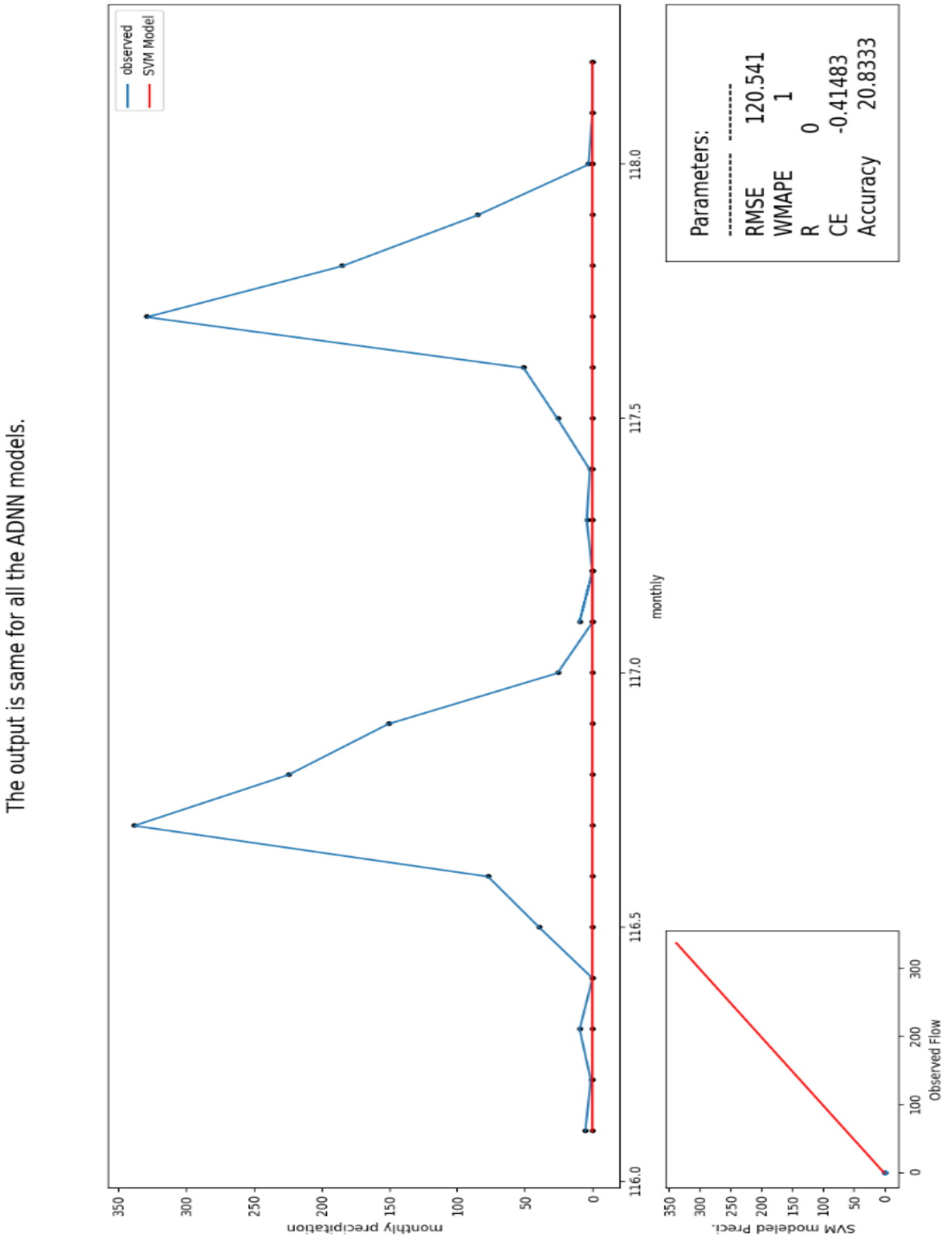

Figure 18: Result- for all the ADNN models.

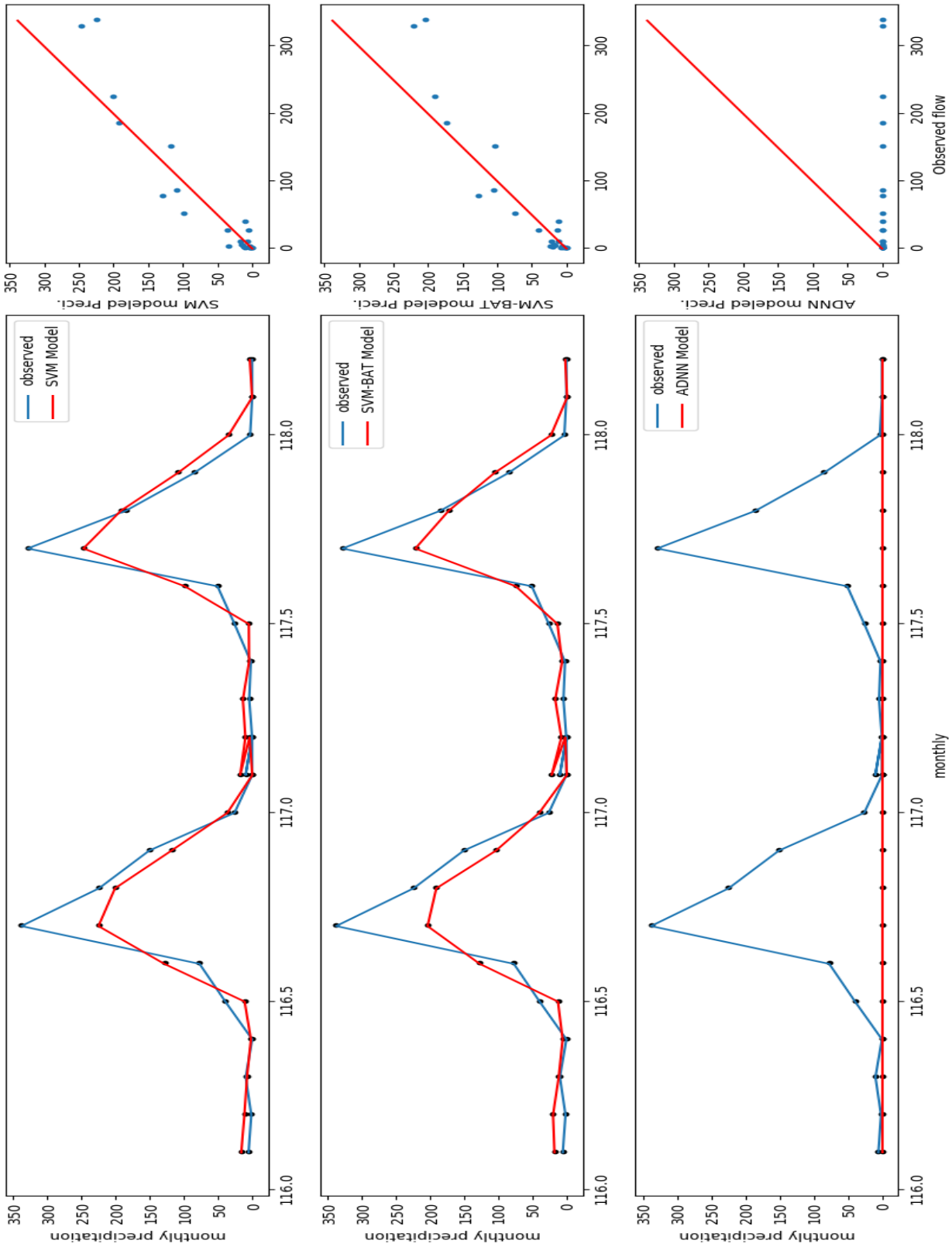

Figure 19: Comparison of best from all the three models.