

CHAPTER V

RESULTS AND DISCUSSIONS

5.0 Introduction

The results gathered in this research are rather extensive due to the varying circumstances that may occur. It is therefore appropriate to present the overall results for all the three approaches ANN, KNN and FTS in advance. They are organized based on possible cases of users in conjunction with the temporal demarcation that spans daily, weekly and monthly duration. This will assist the effort of performing the required comparative analysis.

5.1 Data Input and Output (Daily, Weekly and Monthly) for (User A, User B and User AB)

Overall, the total number of online users is approximately 90 people. In terms of the values, only a noticeable difference can be detected between the maximum and minimum number of users online. The average maximum seems to lie within 60 people while the average minimum revolves around 50 people. As such, the two only differs by roughly 10 people or 10% of the total sample.

Traffic data is gathered over the period of 180 days, divided equally by the range of 50 days. The traffic of user A is frequently higher than user B until day 150. It was observed that the widest difference can be witnessed approximately on day 140 of which user A

reaches its peak while user B plummets into a very low degree. Surprisingly, after that, they converge into a common trend of fluctuation.

Without the need for in-depth analysis, it can be seen from the traffic data that the highest spike is conventionally preceded by the lowest decline. Hypothetically, it could be a seasonal phenomenon that exceeds the scope of the research. This should not be a cause of worry however as the entire trend of the data is not significantly undermined by this pattern.

In this study, the capital city of Libya, Tripoli, was selected as source of traffic data required for ANN based implementation of WIMAX traffic forecasting analysis. Indicated in figures are the daily data input/output recorded data during 180 days which includes the maximum and minimum online of user-A, user-B and user AB along with their WiMAX traffic of MIMO-A, MIMO-B and MIMO-AB.

The following figures denote, it can be seen that the daily maximum number of online users were found to vary between approximately (49.64) and (81.80) and the daily minimum number of online users were found to vary between approximately (21.64) and (56.83), while the daily traffic of MIMO-A user were found to vary between approximately (2.04e8) and (6.954e8). During the same period of observation, the traffic of MIMO-B user was found to vary between approximately (3.632e7) and (3.837e8).

Figure 29: Daily Max/Min of User A

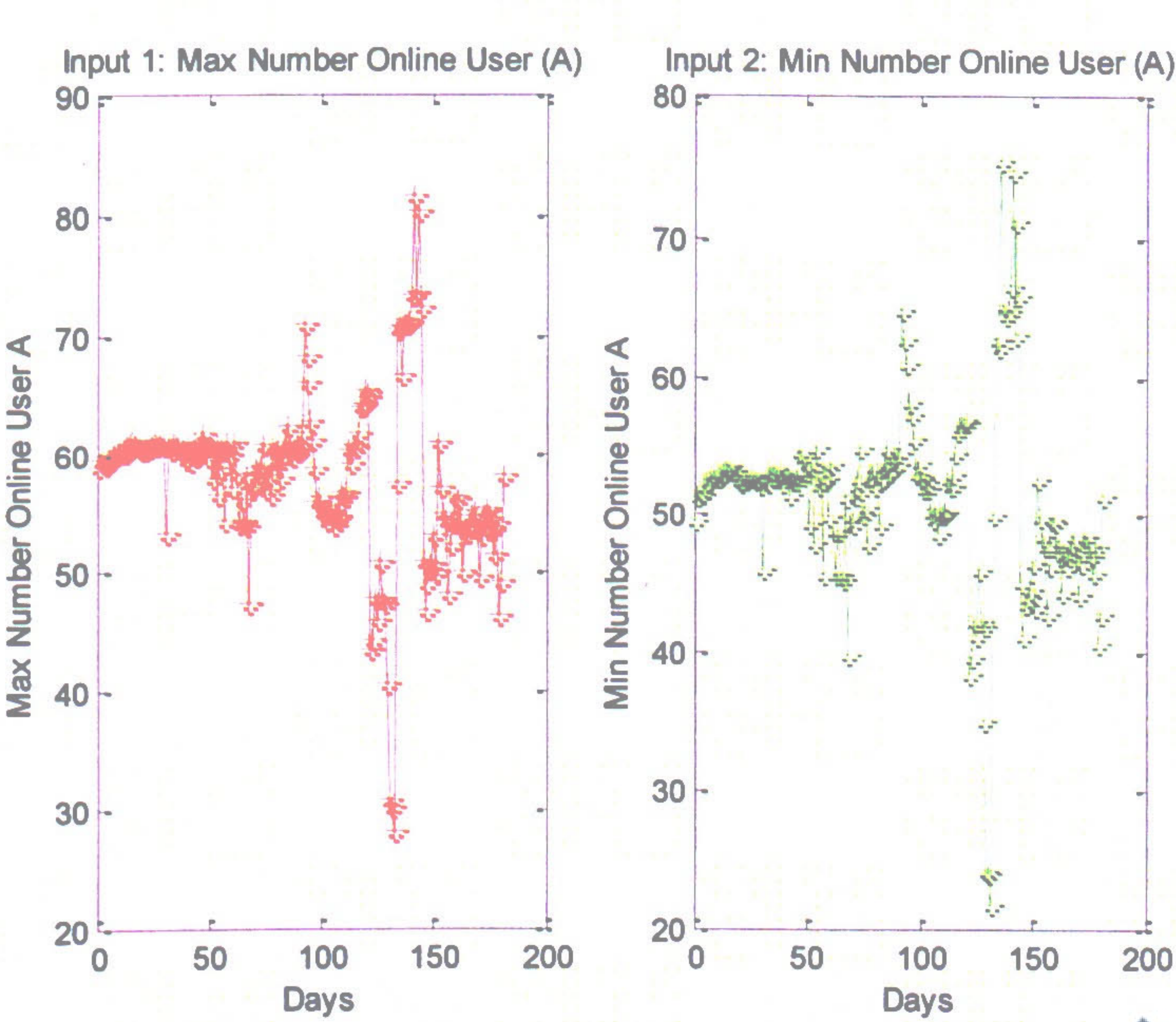


Figure 30: Daily Traffic of User A

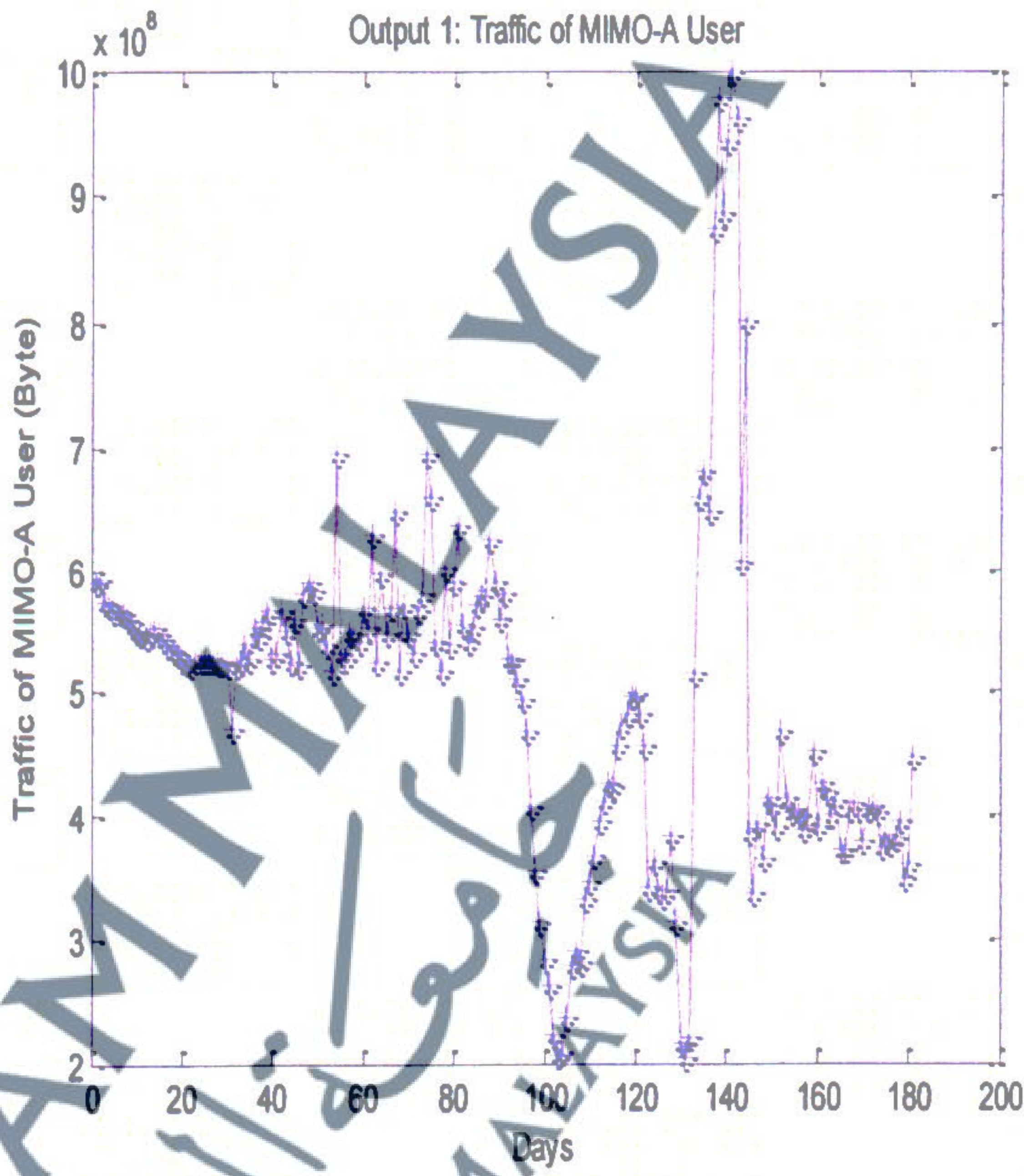


Figure 31: Daily Max/Min of User B

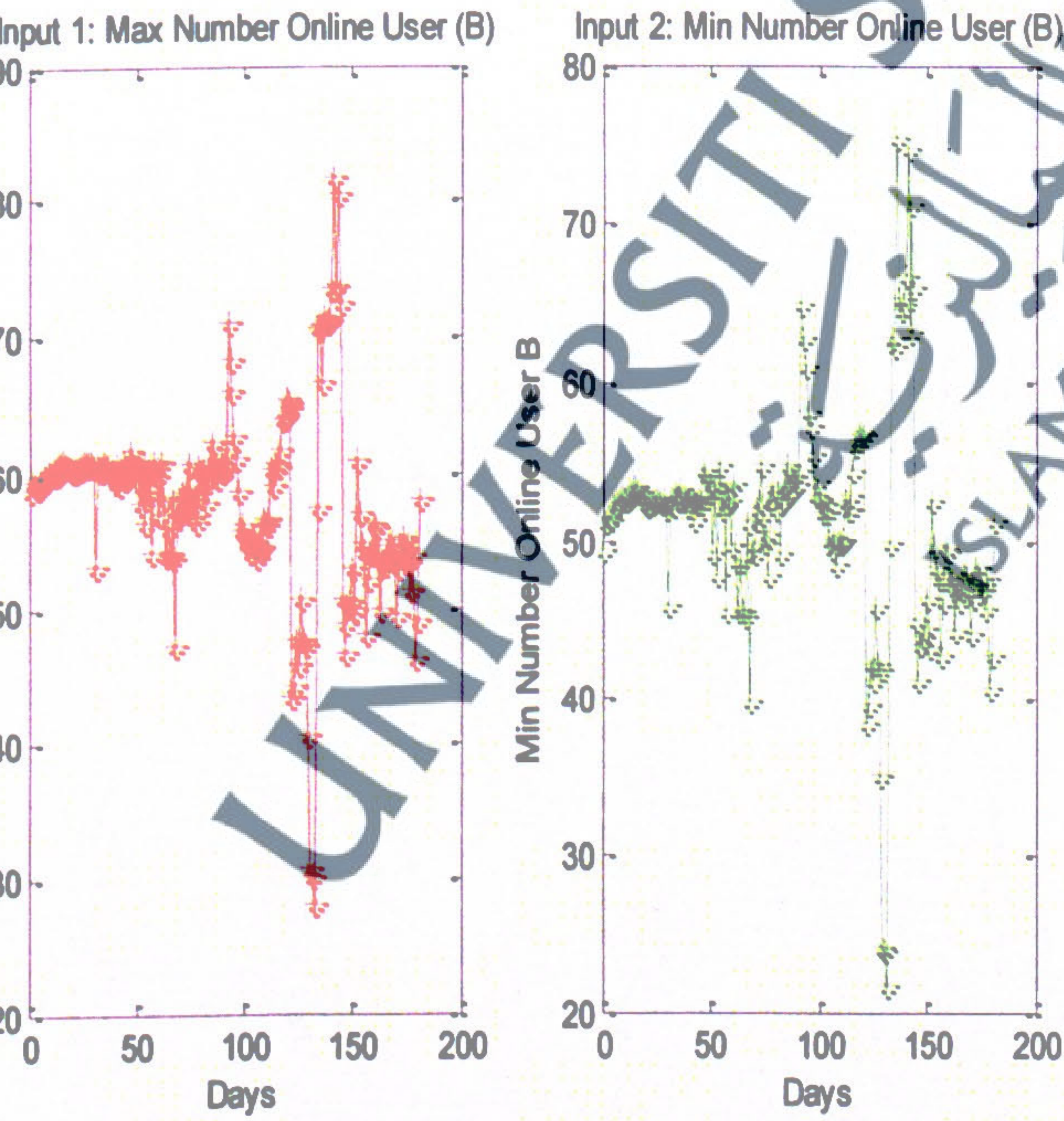


Figure 32: Daily Traffic of User B

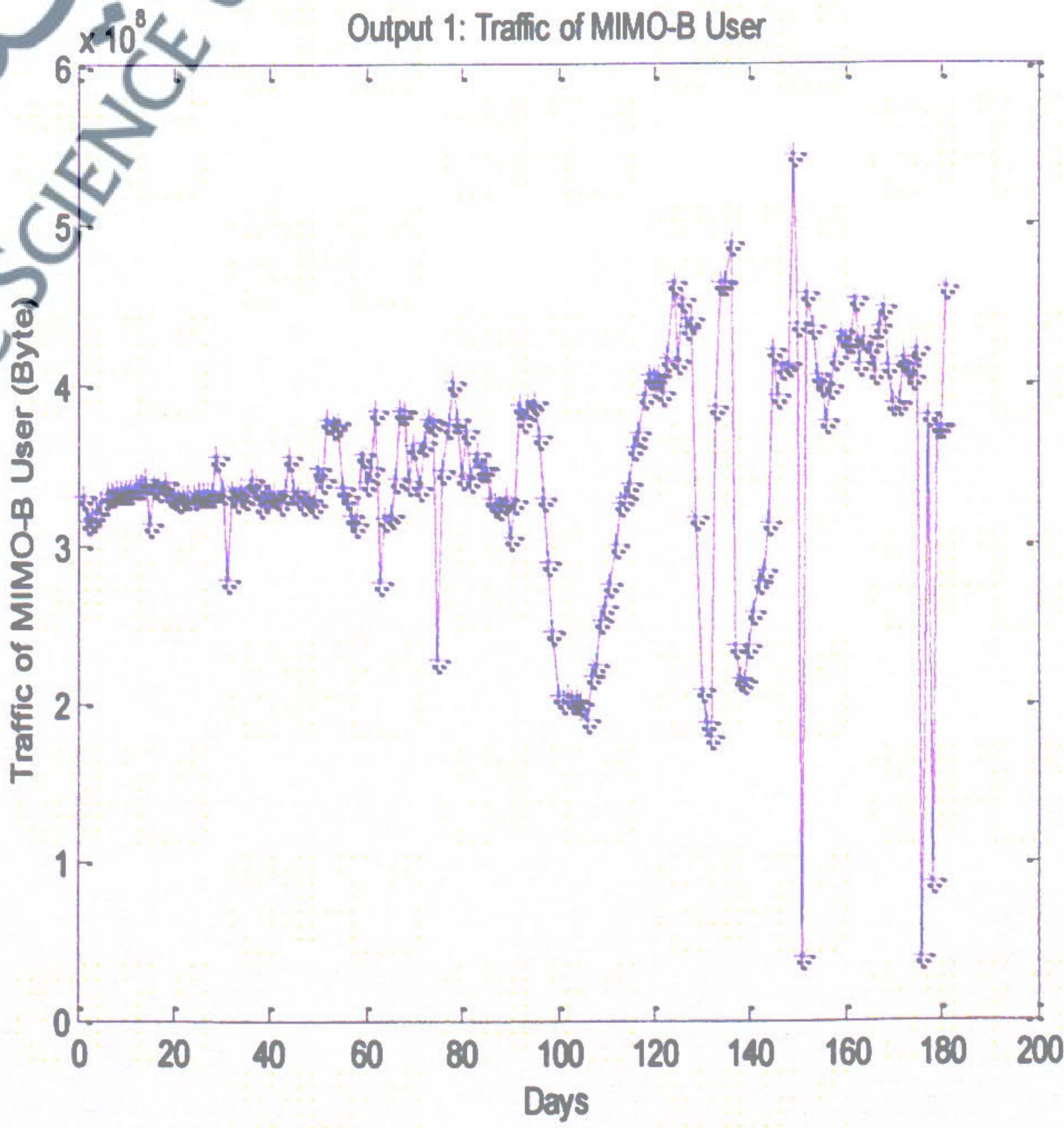


Figure 33: Daily Max/Min of User AB

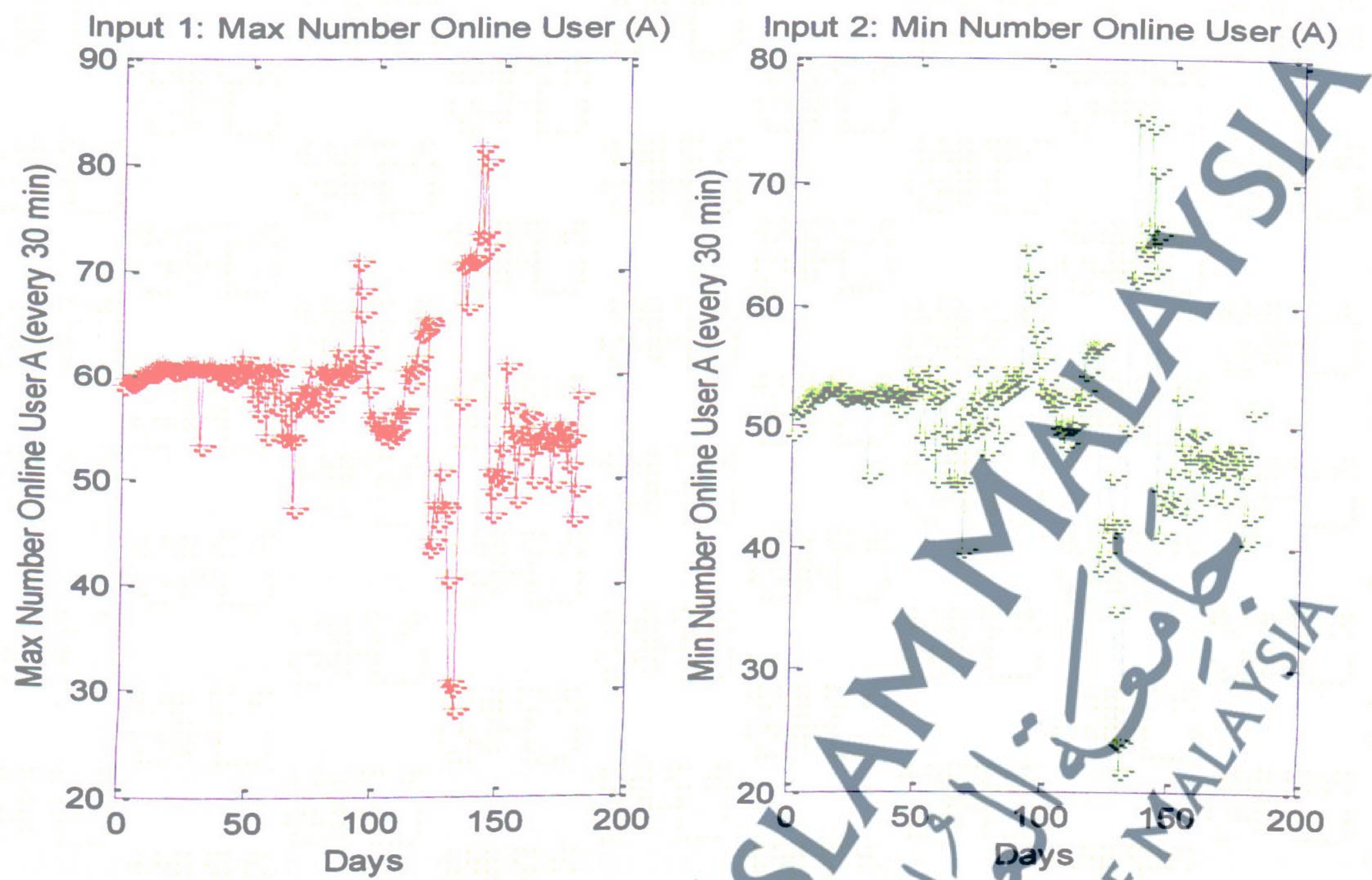
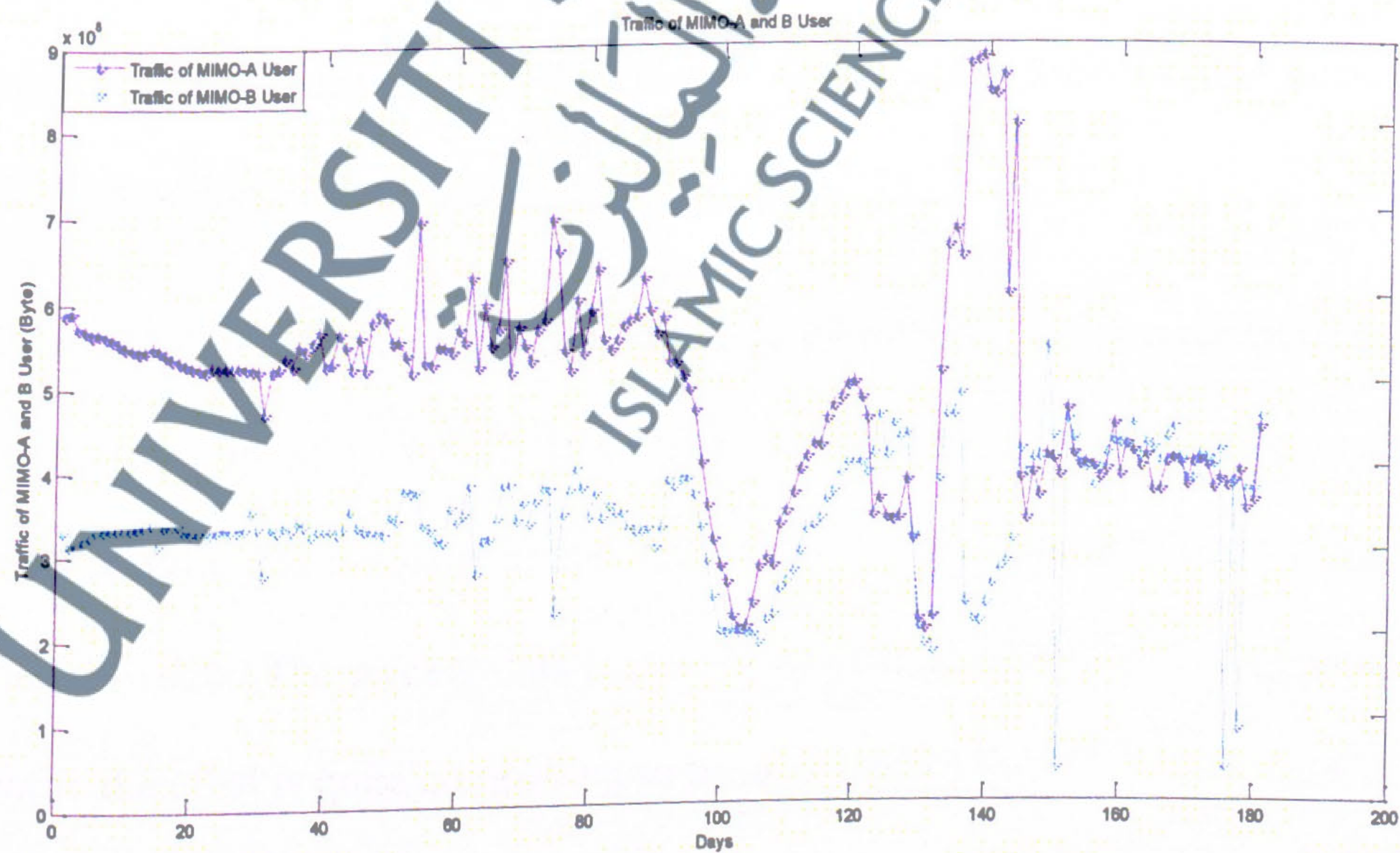


Figure 34: Daily Traffic of User AB



The weekly data can be derived from the daily data by grouping the days into one week. All of them are illustrated as weekly data and traffic of user A (Figure 35, Figure 36), weekly data and traffic of user B (Figure 37, Figure 38), and weekly data and traffic of user AB (Figure 39, Figure 40). The following figures denote, it can be seen that the weekly maximum number of online users were found to vary between approximately (54.20) and (60.85) and the weekly minimum number of online users were found to vary between approximately (51.90) and (53.89).

It is quite obvious that the trend for the weekly data fluctuates less than the daily one. When the behavior of a set of data is put together into a designated class, the mean or collective tendency (Bello et al., 2013) will emerge. More often than not, this affinity will generalize the manifestation of individual ones into an accumulated trend that is normally less striking.

The data indicates that the 20th week is the more interesting point (Stegle et al., 2009; Kalaitzis & Lawrence, 2011) of the diagram for both users. It can be observed that the traffic plunges into the lowest point just before the 20th week and then ascends to the paramount after the week as well. The abrupt decrease and increase that occurs greatly within a short period of time is puzzling.

Assembling data on a weekly basis provides a medium perception on the trend that lies in between daily and monthly data. In this sense, the depiction is usually rather satisfactory. It is not as specific and unsteady as the daily data. Furthermore, it is not too unchanging as the monthly data. The weekly data (Kaltenbrunner et al., 2010) offers a balanced view of the entire data that is quite compelling to the analyzer.

Figure 35: Weekly Min/Max of User A

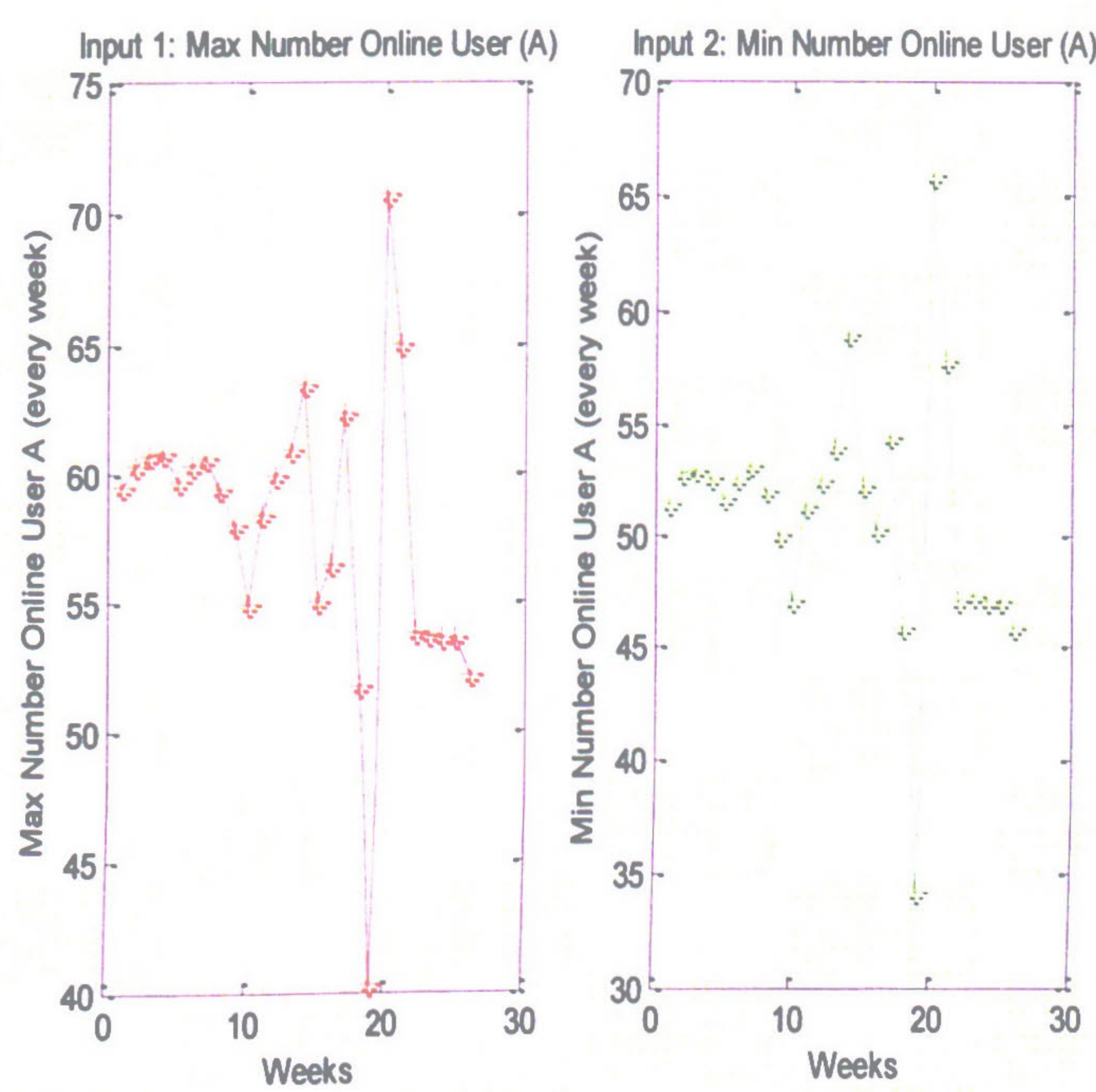


Figure 36: Weekly Traffic of User A

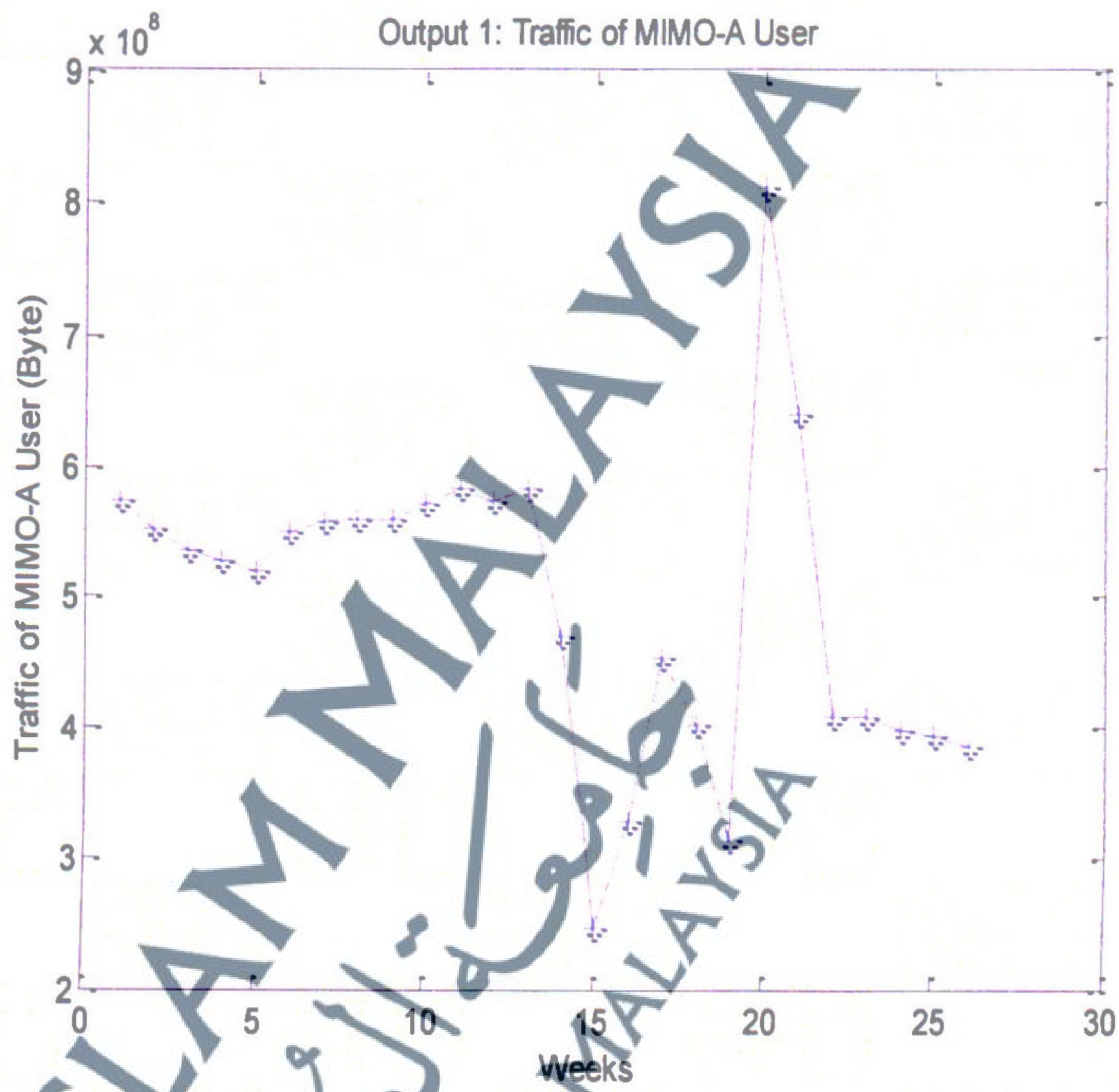


Figure 37: Weekly Min/Max of User B

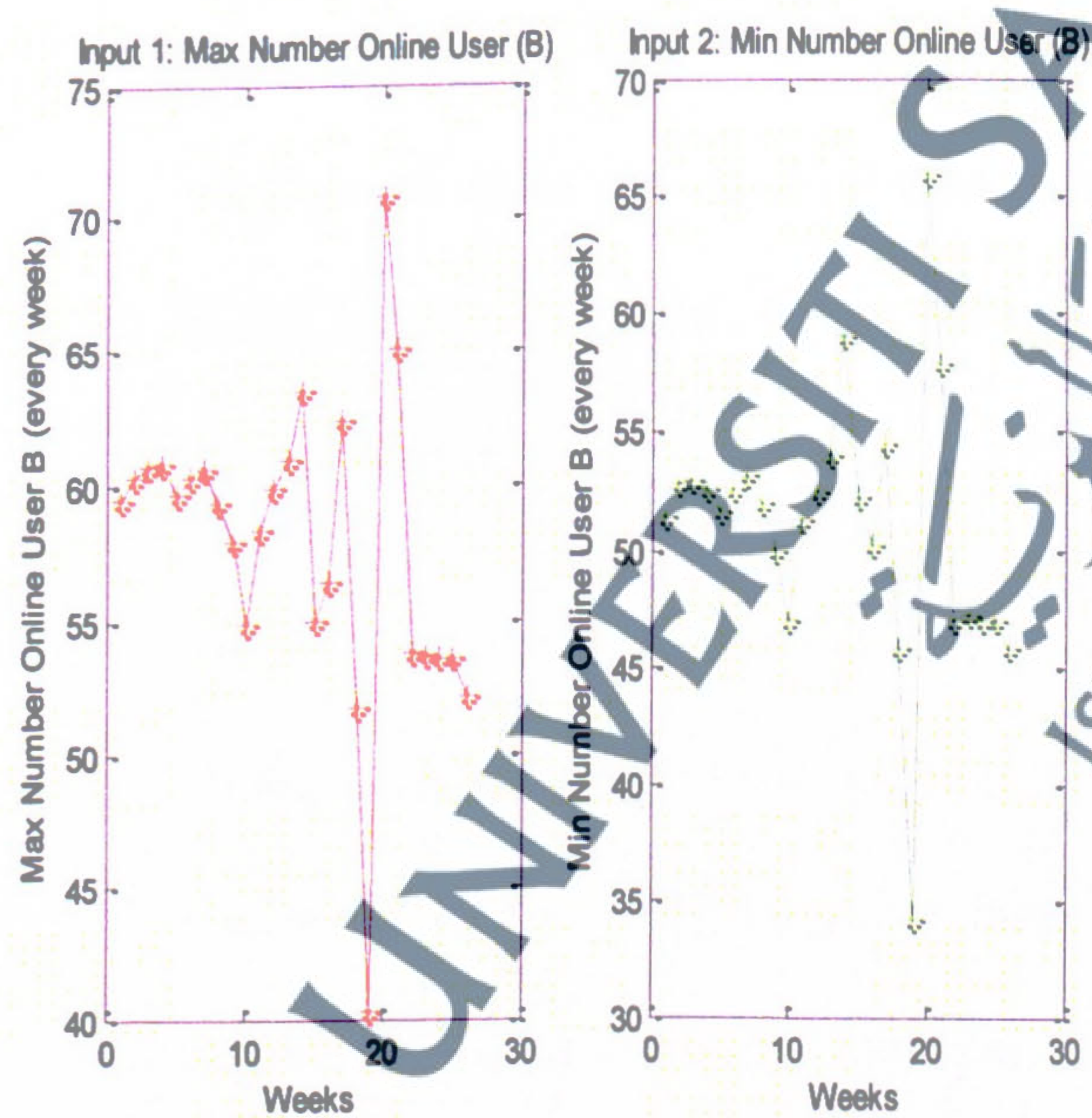


Figure 38: Weekly Data Input/output of User B

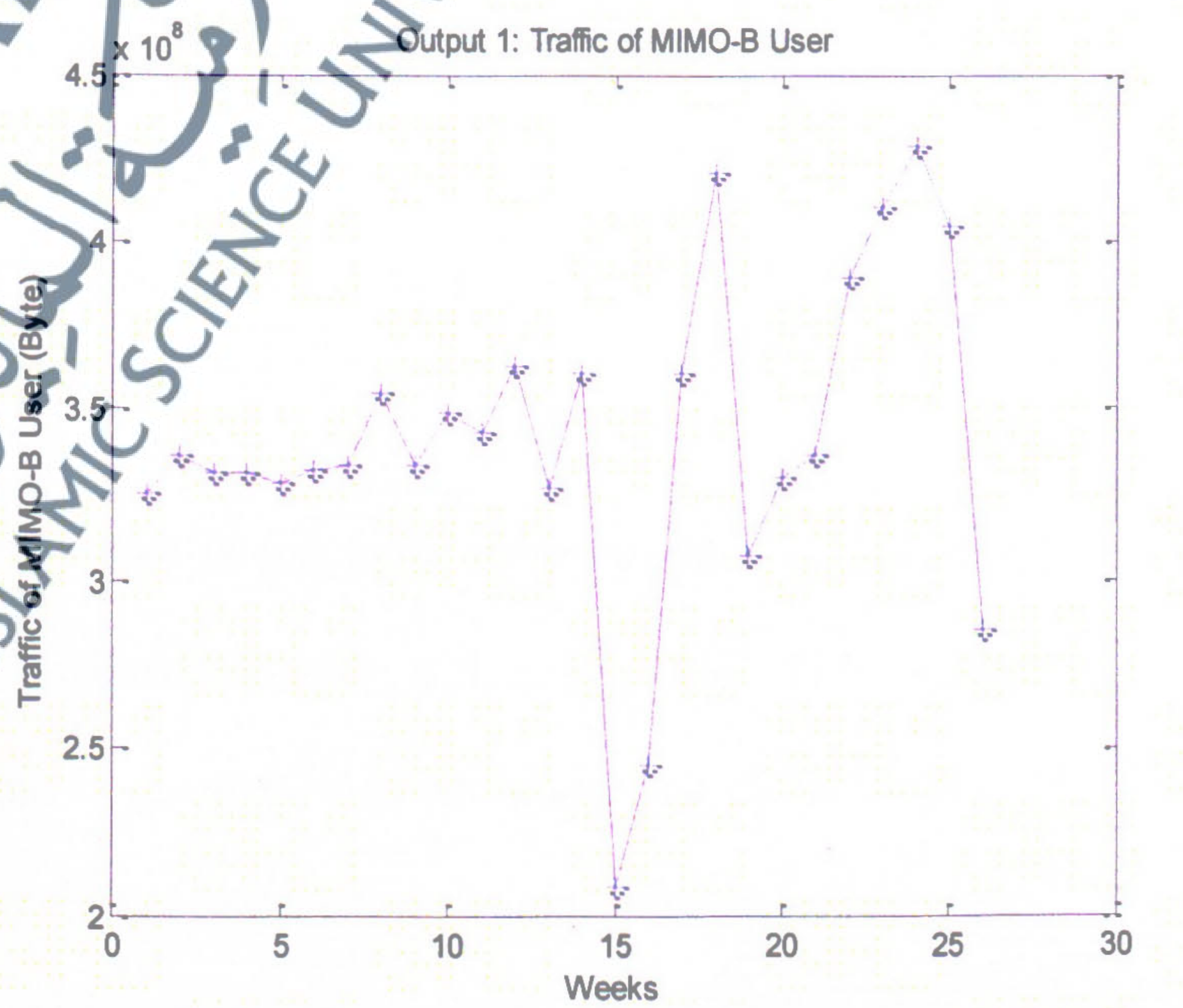


Figure 39: Weekly Min/Max of User AB

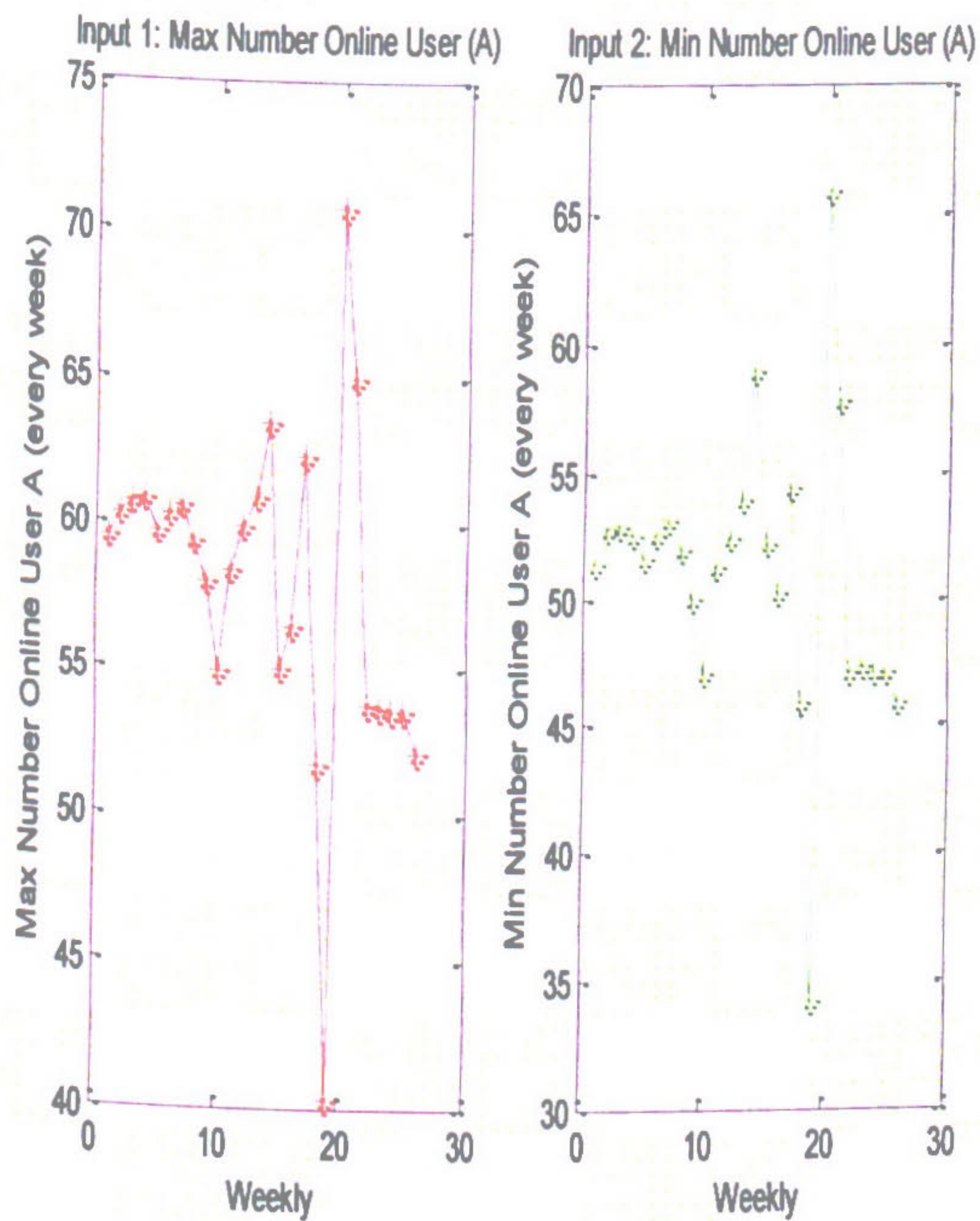
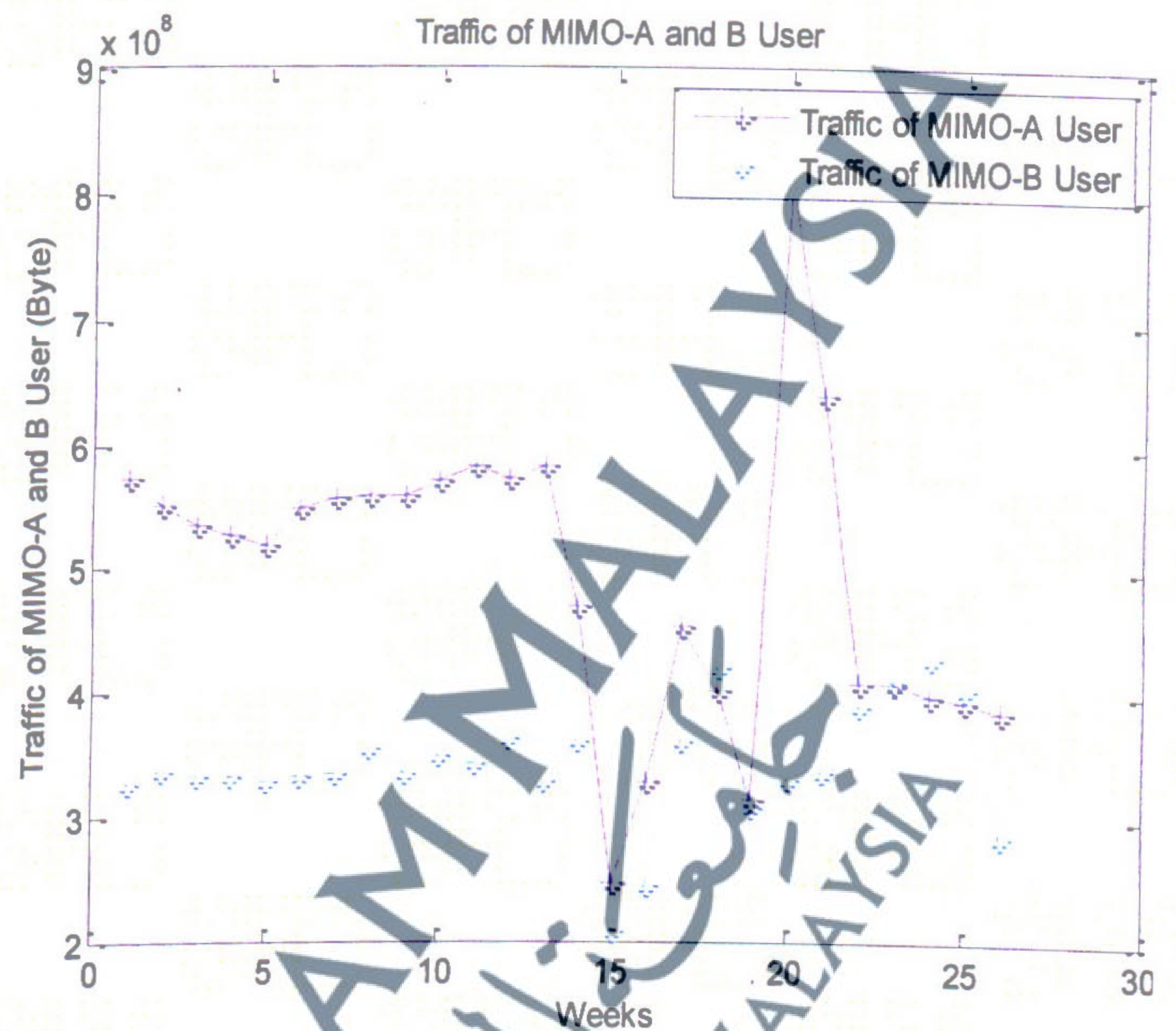


Figure 40: Weekly Traffic of User AB



The monthly data for the WiMAX traffic is illustrated in monthly user A (Figure 41, Figure 42), monthly user B (Figure 43, Figure 44), and monthly user AB (Figure 45, Figure 46). Similar to their predecessors, the data comprise of the maximum / minimum number of users and the traffic for each category. Monthly data (Waewsaka et al., 2014) is the longest duration of data for this research. There is no quarterly data (Lorek, 2014) included in the study.

From the aspect of expressivity, monthly data is the least sensitive class of temporal change when compared to its two counterparts, the daily and weekly data. This can be attributed to the wide range of data that is grouped together under the division. As a result, the data exhibits minimal fluctuation, emanating minute displacement over progress (Sabzpooshani & Mohammadi, 2014).

The traffic distribution begins with a slow increase at the initial stage and climbs the peak on the third month (Yang & Li, 2011). It suffers a sudden plunge into the lowest value on the fourth month of the overall duration. This is opposes the trend of the maximum / minimum number of online users which show low value on the third month and subsequently a rise on the following month.

Comparatively, the overall traffic of user A is persistently higher than user B. The difference between the two is truly excessive. It is apparent from the graph of monthly traffic of user AB that the summit of user B does not even match the statistical of user A. However, it is not within the scope of this research to study the causality of this phenomenon.

Figure 41: Monthly Min/Max of User A

Figure 42: Monthly Traffic of User A

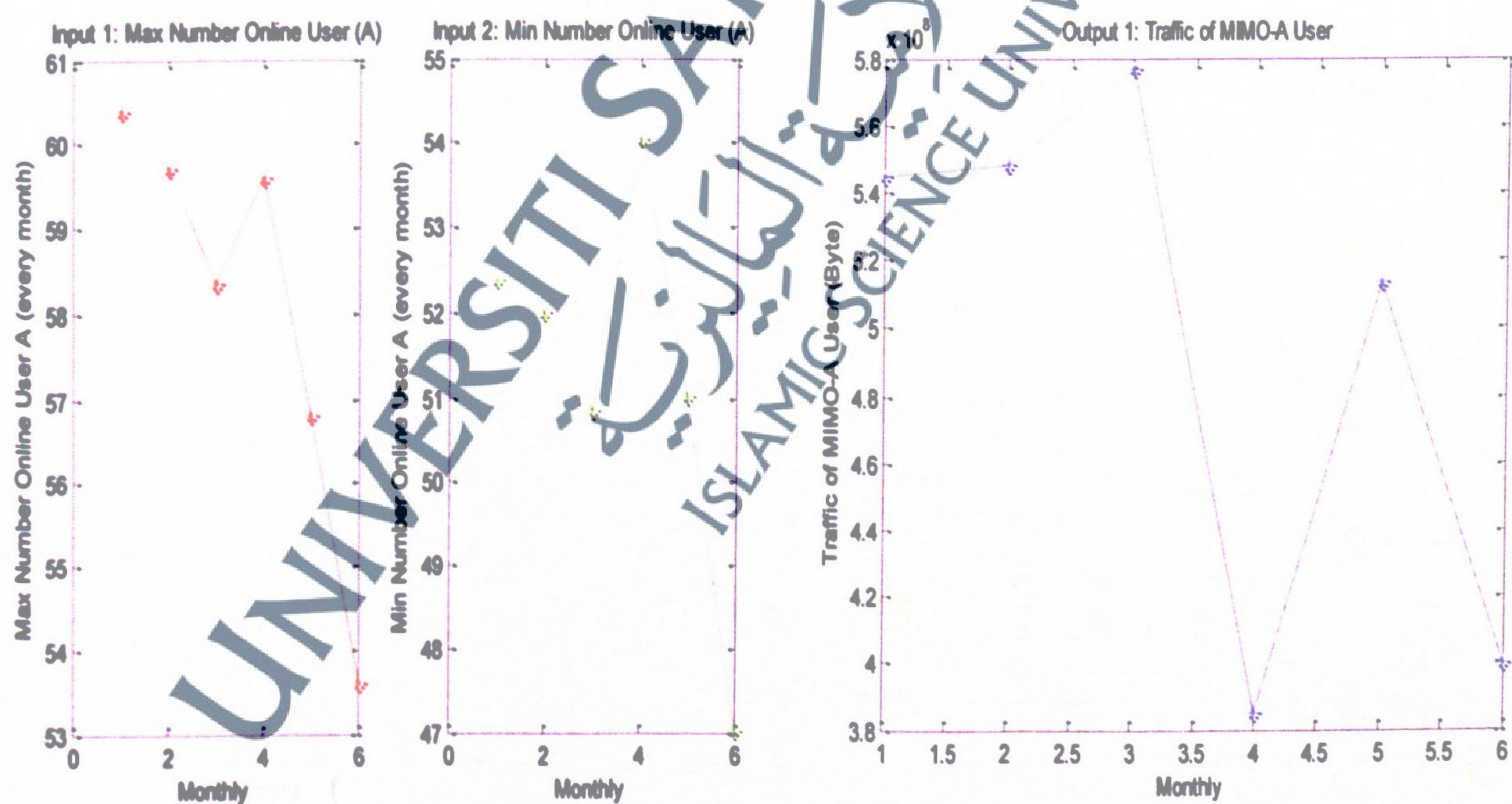


Figure 43: Monthly Min/Max of User B

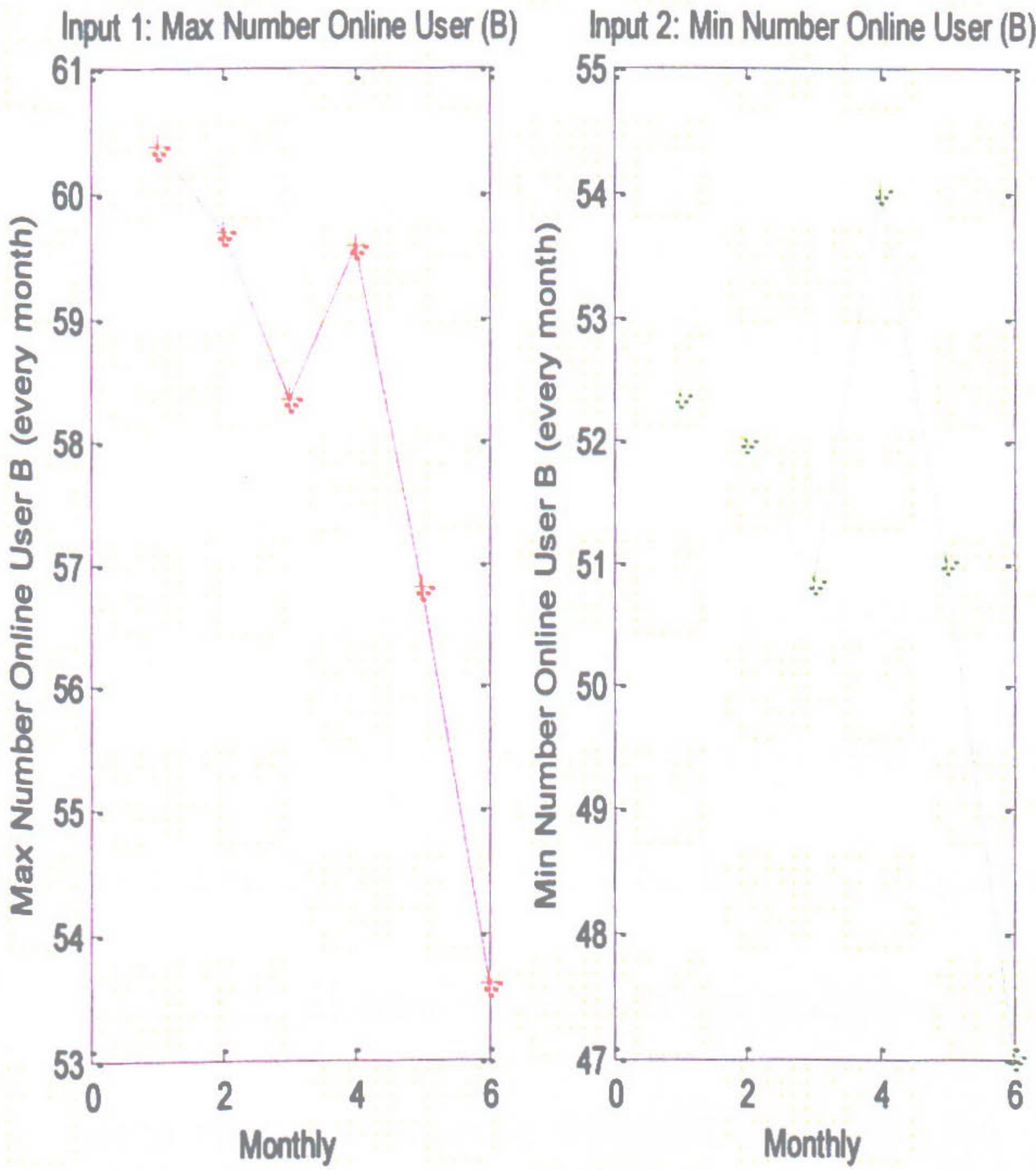


Figure 44: Monthly data input/output of user B

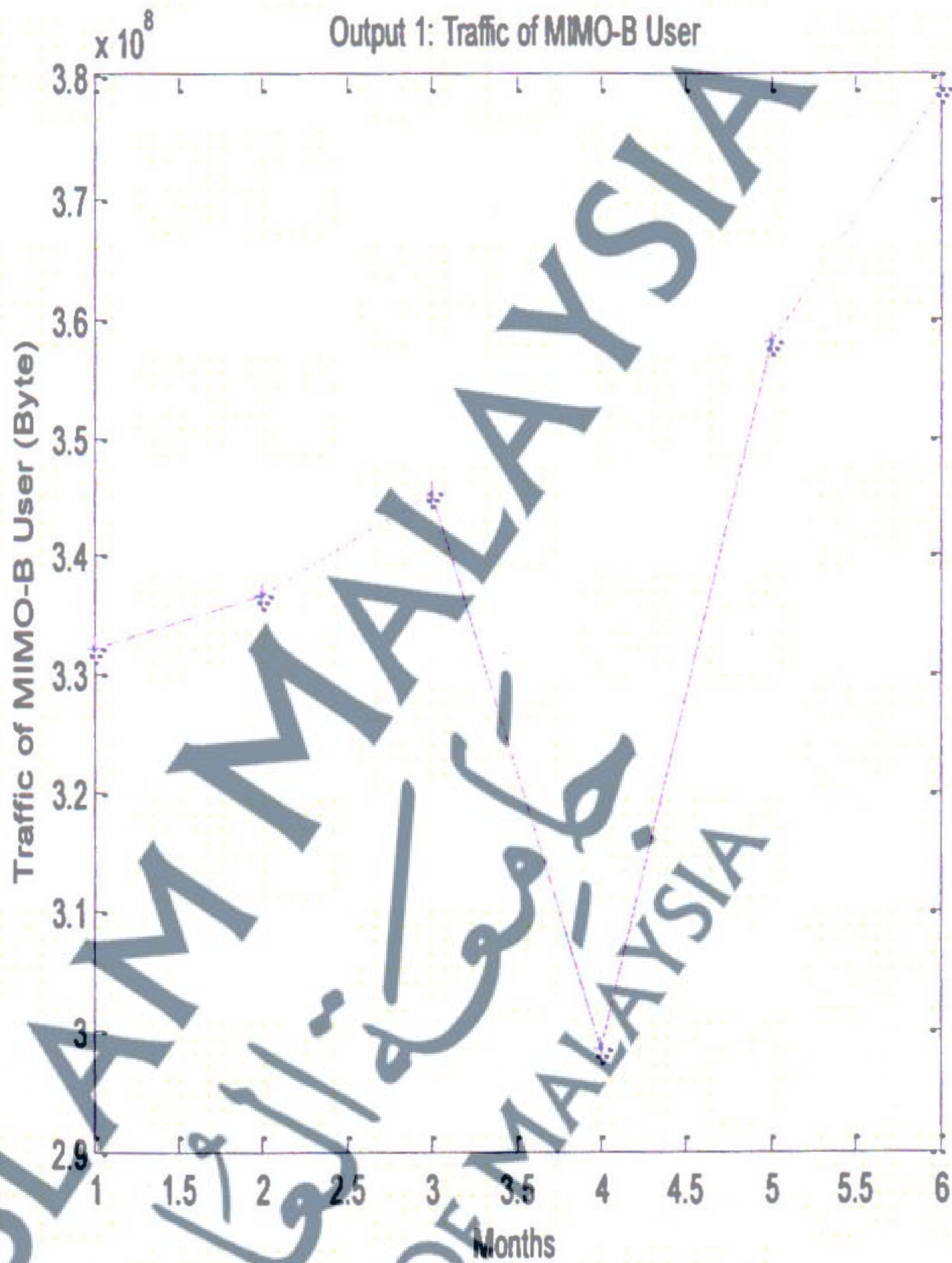


Figure 45: Monthly Min/Max of User AB

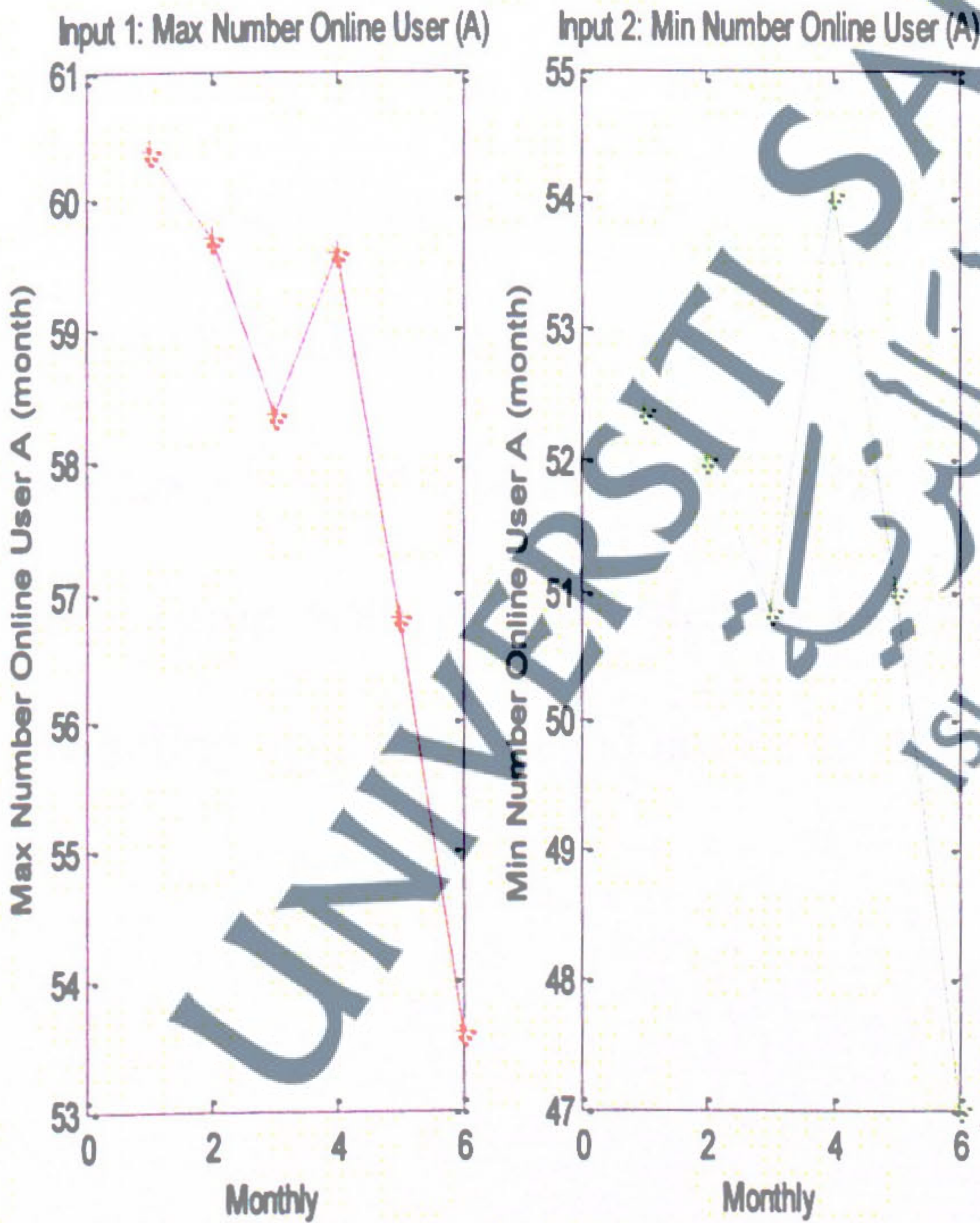
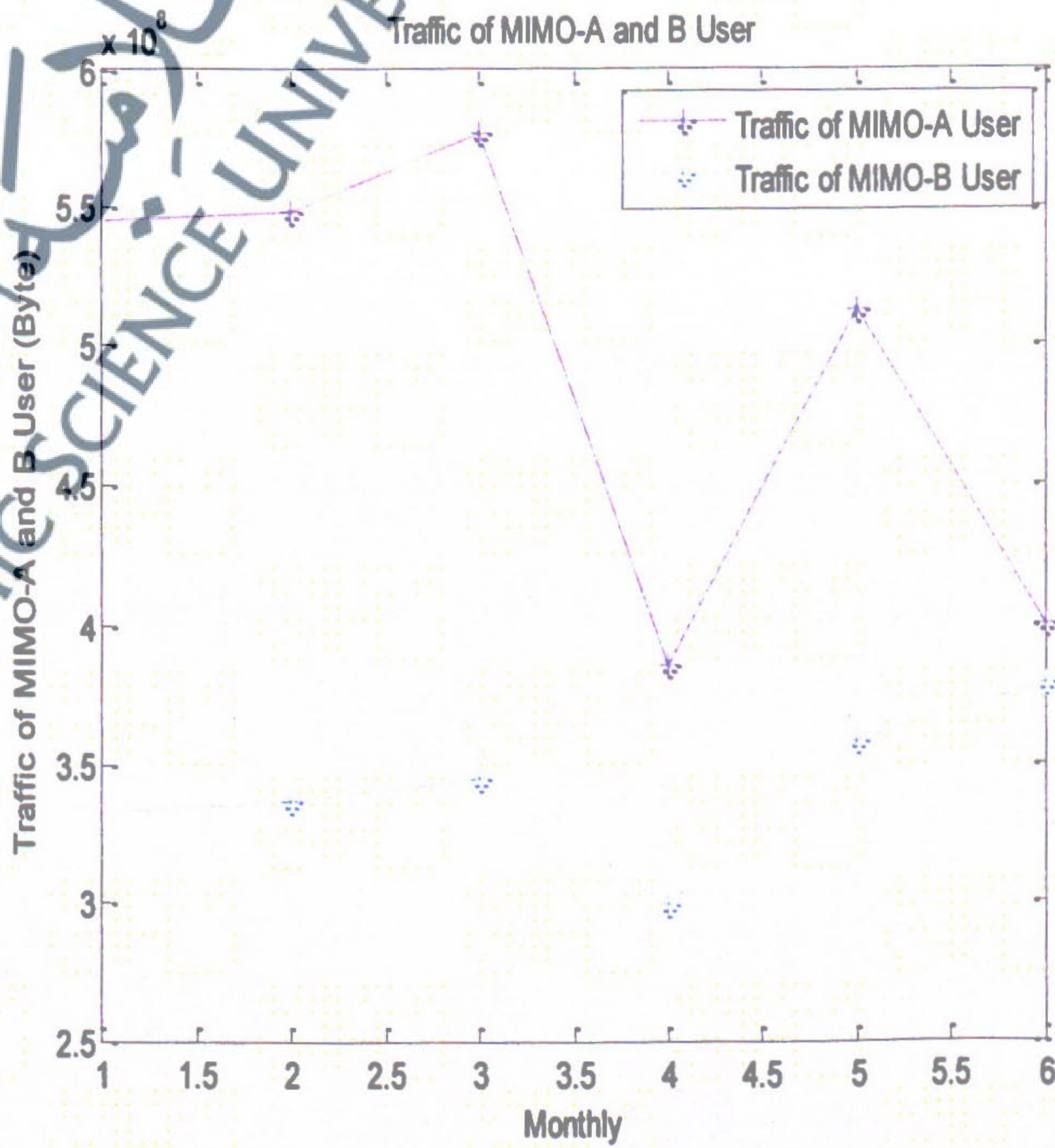


Figure 46: Monthly Traffic of User AB



5.2 Data Input and Output for KNN (Daily, Weekly and Monthly)

The result for k – nearest neighbor is organized based on the number of neighbor's K . There are five instances of k utilized in this research. They are $k = \{1, 3, 5, 7, \text{ and } 9\}$. Without the loss of generality, it is assumed that the usage of odd numbered k on the performance of the forecasting will not be that significantly different from even numbered $k = \{2, 4, 6, 8, \text{ and } 10\}$.

The general naive algorithm for KNN (Li et al., 2003) is illustrated below. It calculates the distance $D(X, Y)$ between the unknown data Y by comparing it with all the data in the instance $X = X_1 \dots X_N$. The Euclidean distance is the most common measure of distance. Once the distance between the unknown and all the instances are known, only the k set of data that are nearest would be chosen for classification. For instance, if $k = 3$, then only three instances with the nearest distance are considered.

For the daily traffic of user A (Figure 47), the result is summarized below (Table 3). The best number for K is 1 or 3 neighbors. Here, both selections of K offer the lowest error of 0.0007 and the highest modeling efficiency of 0.9673. On the other hand, the worst error is 0.0009 with the lowest efficiency at 0.9582. This occurs when the number of closest neighbor is set at 3 points for K . Other value of K such as 5 also results to a commendable forecasting with an error degree as marginal as the best choice of K . However, the modeling efficiency would not be as impressive.

Figure 47: KNN Daily Testing of User A

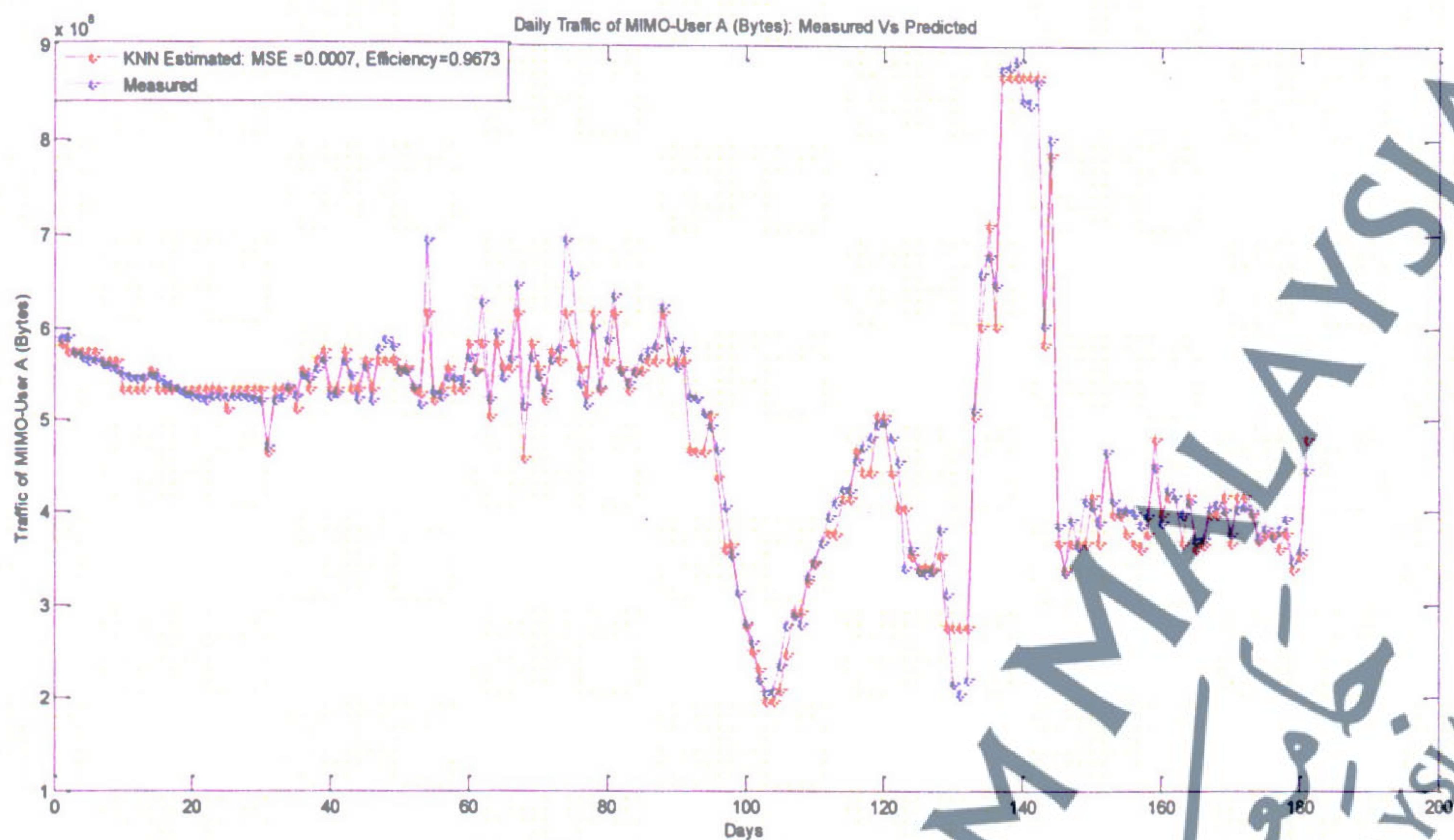


Table 3: KNN Daily Testing of User A

USER A DAILY		
K	MSE	M _{eff}
1	0.0007	0.9673
3	0.0009	0.9582
5	0.0007	0.9673
7	0.0008	0.9610
9	0.0007	0.9653

Contrary to the daily result of user A that shows preference for a particular value of K when it comes to minimizing the forecasting error, it seems that for user B (Figure 48), this predilection is nonexistent. All the variants of K (Table 4) portray similar lowest error of 0.0005. Nevertheless, the highest efficiency is achieved at 0.9758, which transpires when K is set at 7. The efficiency of other values of K does not digress

significantly from the maximum one. The worst efficiency is found at 0.9751 when K is defined at the value of 5. Note the difference between the lowest efficiency and the highest one. They differ by a meager value of 0.0007.

Figure 48: KNN Daily Testing of User B

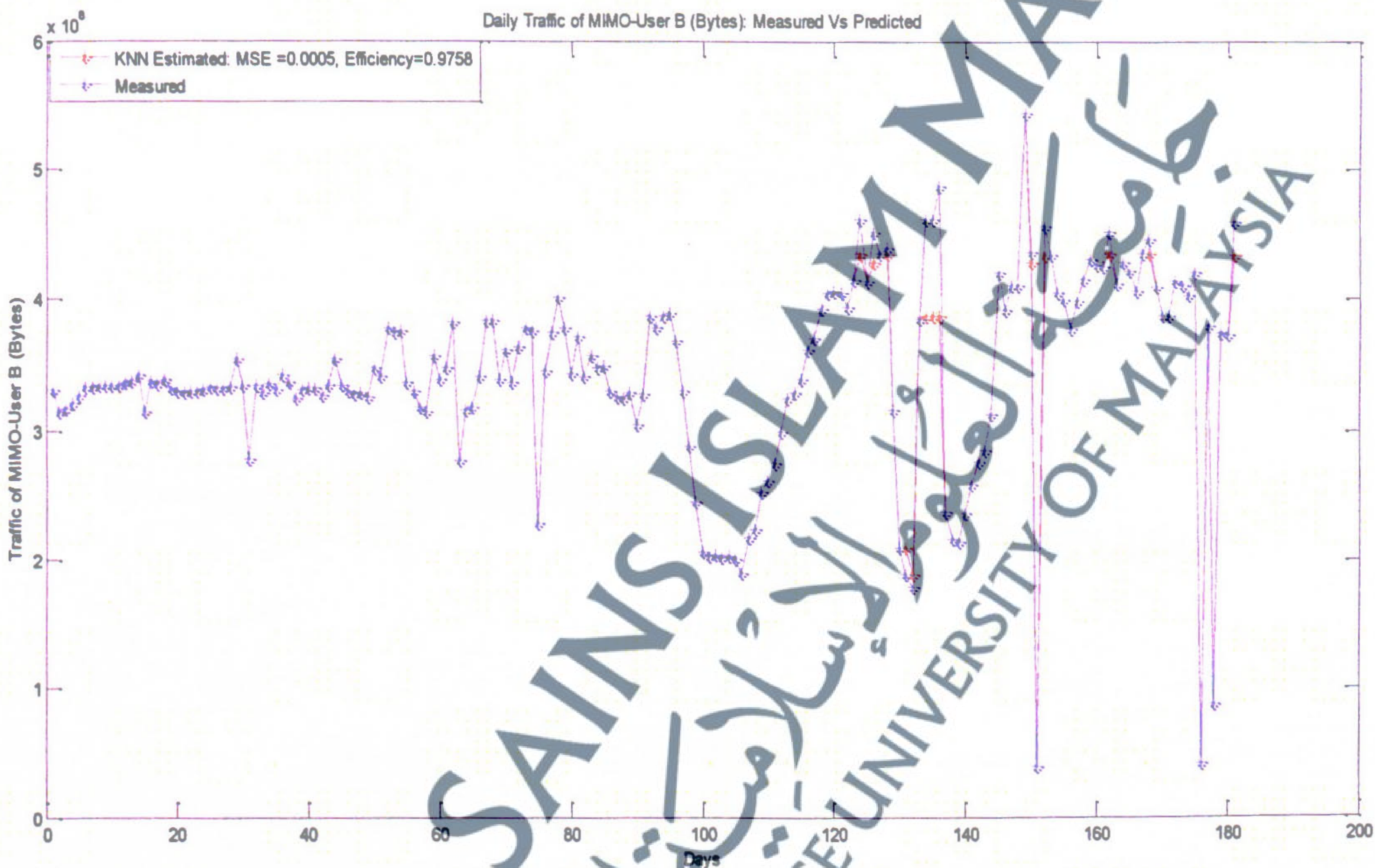


Table 4: KNN Daily Testing of User B

USER B DAILY		
K	MSE	M _{eff}
1	0.0005	0.9755
3	0.0005	0.9752
5	0.0005	0.9751
7	0.0005	0.9758
9	0.0005	0.9753

Similar to the preceding case of user B, the best choice of $K = 7$ (Table 5) is demonstrated again for user AB (Figure 49) whereby the least error is attained at 0.0004 and the highest efficiency projected at 0.9791. It is quite alarming to notice that the best value for K for user A is also the worst for user AB. When $K = 1$, the approach shows the highest error (0.0011) and lowest efficiency (0.9414). Average performance can be afforded when $K = 5$. Here, the error suffered is 0.0007 while the efficiency is 0.9473, which is slightly higher than the worst.

Figure 49: KNN Daily Testing of User AB

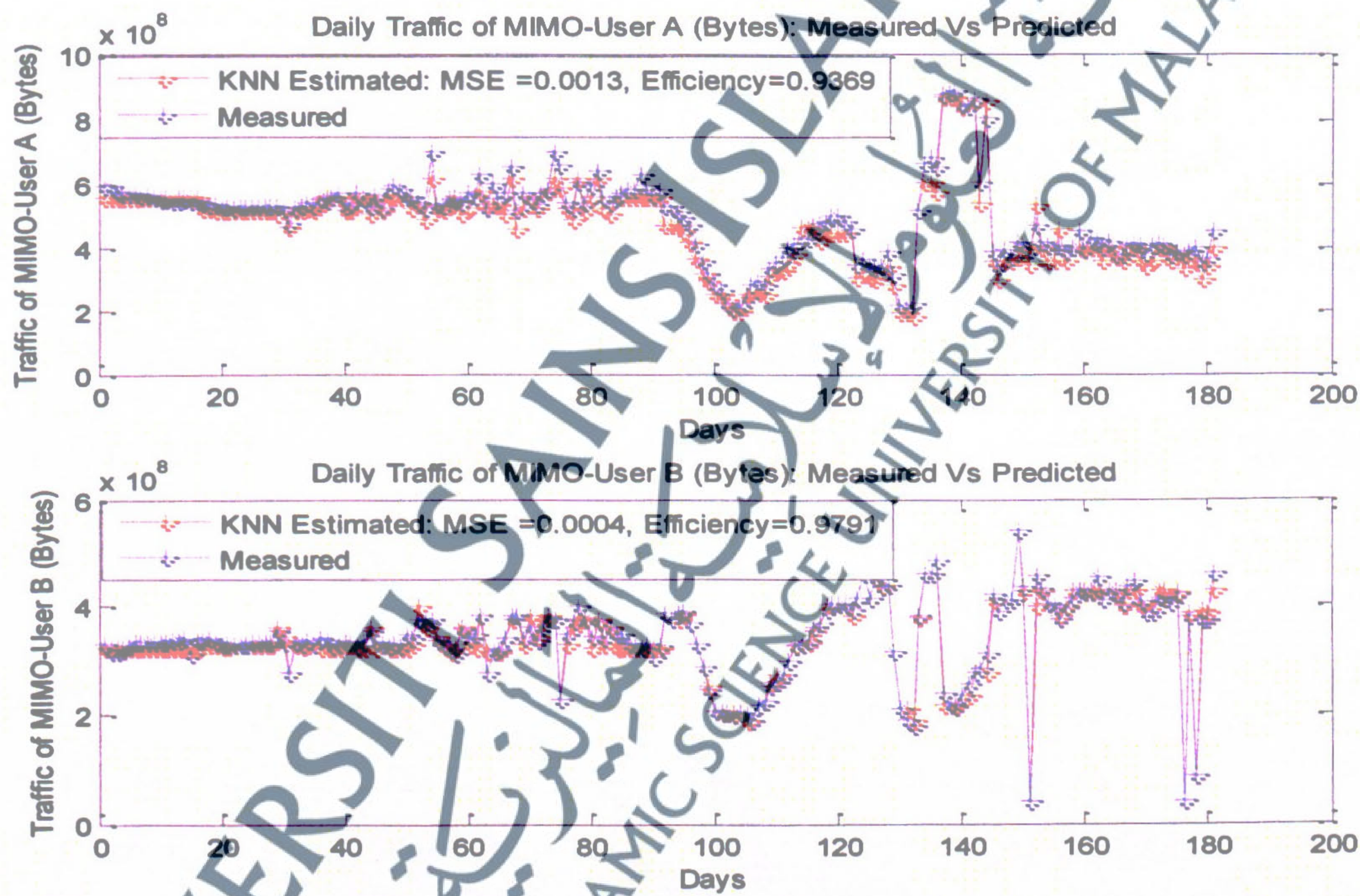


Table 5: KNN Daily Testing of User AB

USER AB DAILY		
K	MSE	M _{eff}
1	0.0011	0.9414
3	0.0011	0.9414
5	0.0007	0.9473
7	0.0004	0.9791
9	0.0004	0.9653

The best performance for the weekly forecasting of user A (Figure 50) is when K is set at 7 (Table 6). In this particular configuration, the lowest error is made possible at 0.0004 with the highest efficiency at 0.9803. Other choices of K would also portray comparable error performance. Both K=3 or K=5 would result to the lowest error (0.0004) as the best K=7. The efficiency for these near optimal alternatives of K is quite impressive too, at 0.9801. The worst occurs when K=9 where the highest error (0.0006) and lowest efficiency (0.9732) are recorded.

Figure 50: KNN Weekly Testing of User A

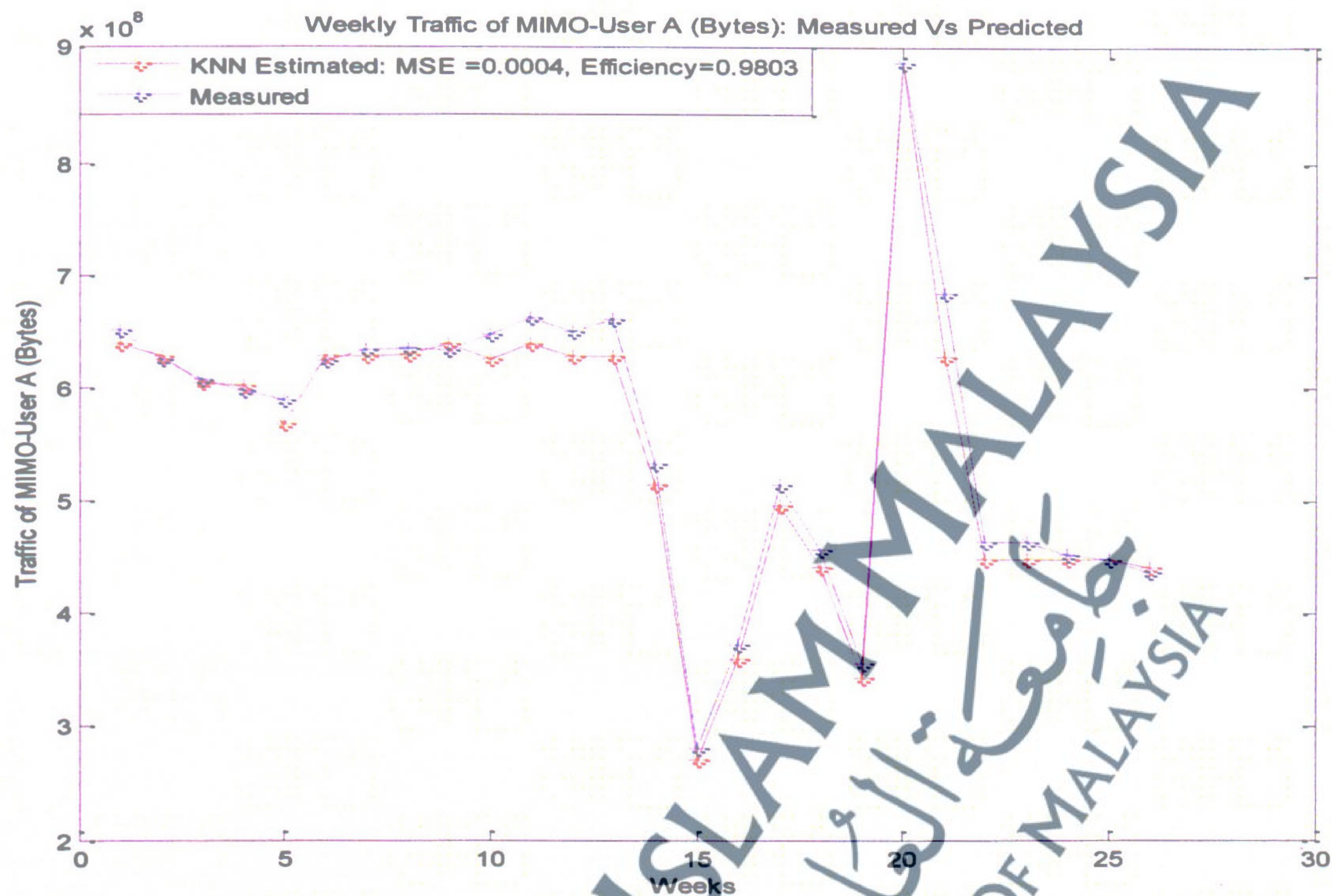


Table 6: KNN Weekly Testing of User A

USER A WEEKLY		
K	MSE	M _{eff}
1	0.0005	0.9788
3	0.0004	0.9801
5	0.0004	0.9801
7	0.0004	0.9803
9	0.0006	0.9732

Equally compelling as the weekly performance of user A, the value of $K = 7$ repeats its dominance over superiority for user B (Figure 51). At this value (Table 7), the error is lowest (0.0001) and the efficiency is highest (0.9873). The worst error is 0.0004 with

inferior efficiency of 0.9700, which is instigated when K is defined at 9. It seems that poor performance transpires when K is set to values other than 7. In addition, the efficiency for other values of K= (1 or 3 or 5 or 9), beside the optimal one, is significantly severe as well. They are approximately lower than the highest efficiency of 0.9893 by more than 0.0100 and revolve around 0.9700.

Figure 51: KNN Weekly Testing of User B

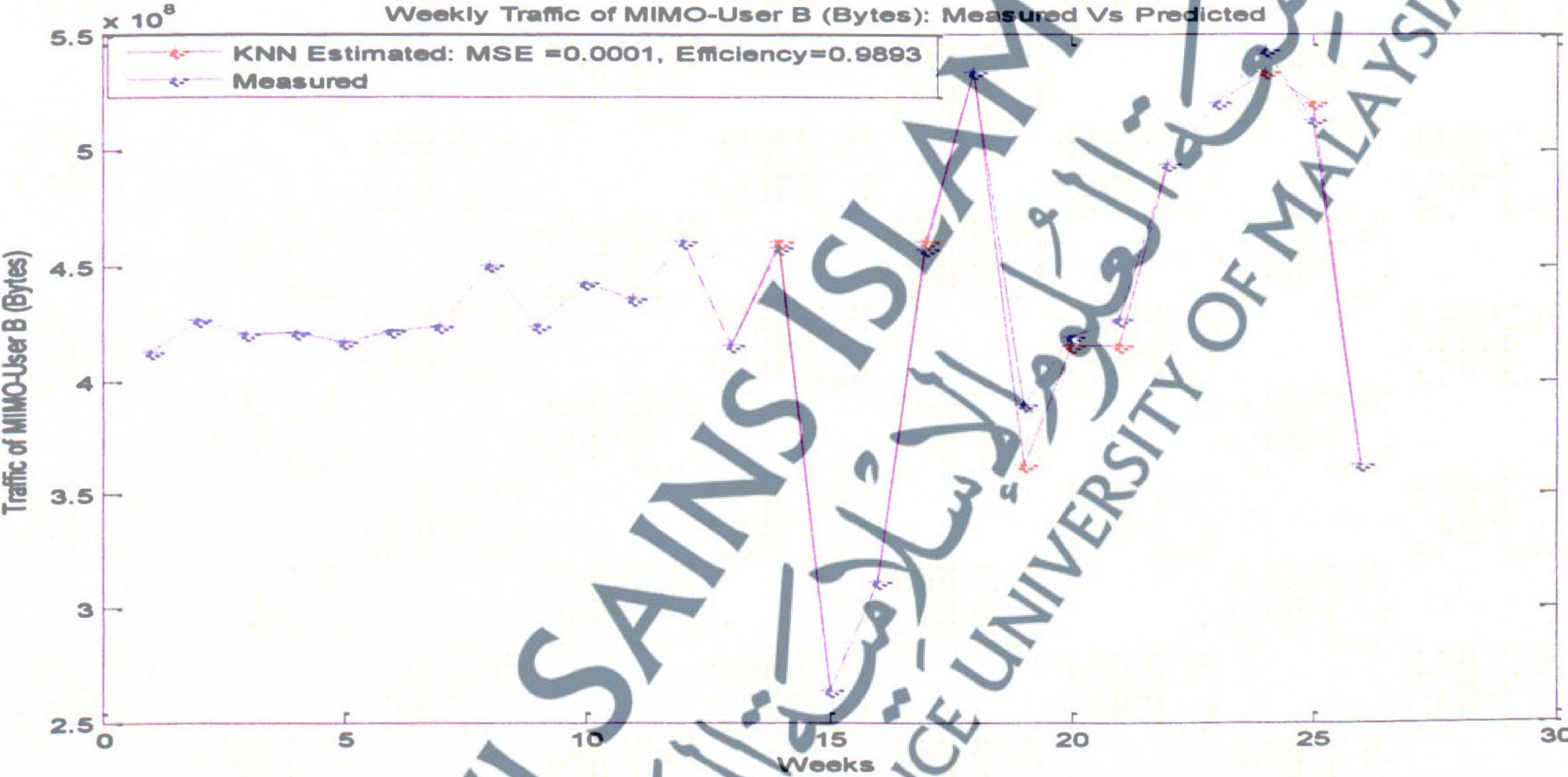


Table 7: KNN Weekly Testing of User B

USER B WEEKLY		
K	MSE	M _{eff}
1	0.0004	0.9706
3	0.0004	0.9706
5	0.0004	0.9708
7	0.0001	0.9893
9	0.0004	0.9700

Clearly, the most suitable value of K is 7 for the forecasting of weekly traffic. As shown in the weekly testing of user AB (Table 8) where this value shows unparalleled distinction when compared against its counterparts, the same form of advancement is shown here for the weekly traffic of user AB (Figure 52). When $K = 7$, the error is lowest at 0.0004 and the efficiency is highest at 0.9679. However, contrary to earlier cases, the optimality shown at this juncture is not as impressive as the worst one. Error is highest (0.0006) when $K = 7$ but efficiency is lowest (0.9670) when $K = 5$.

Figure 52: KNN Weekly Testing of User AB

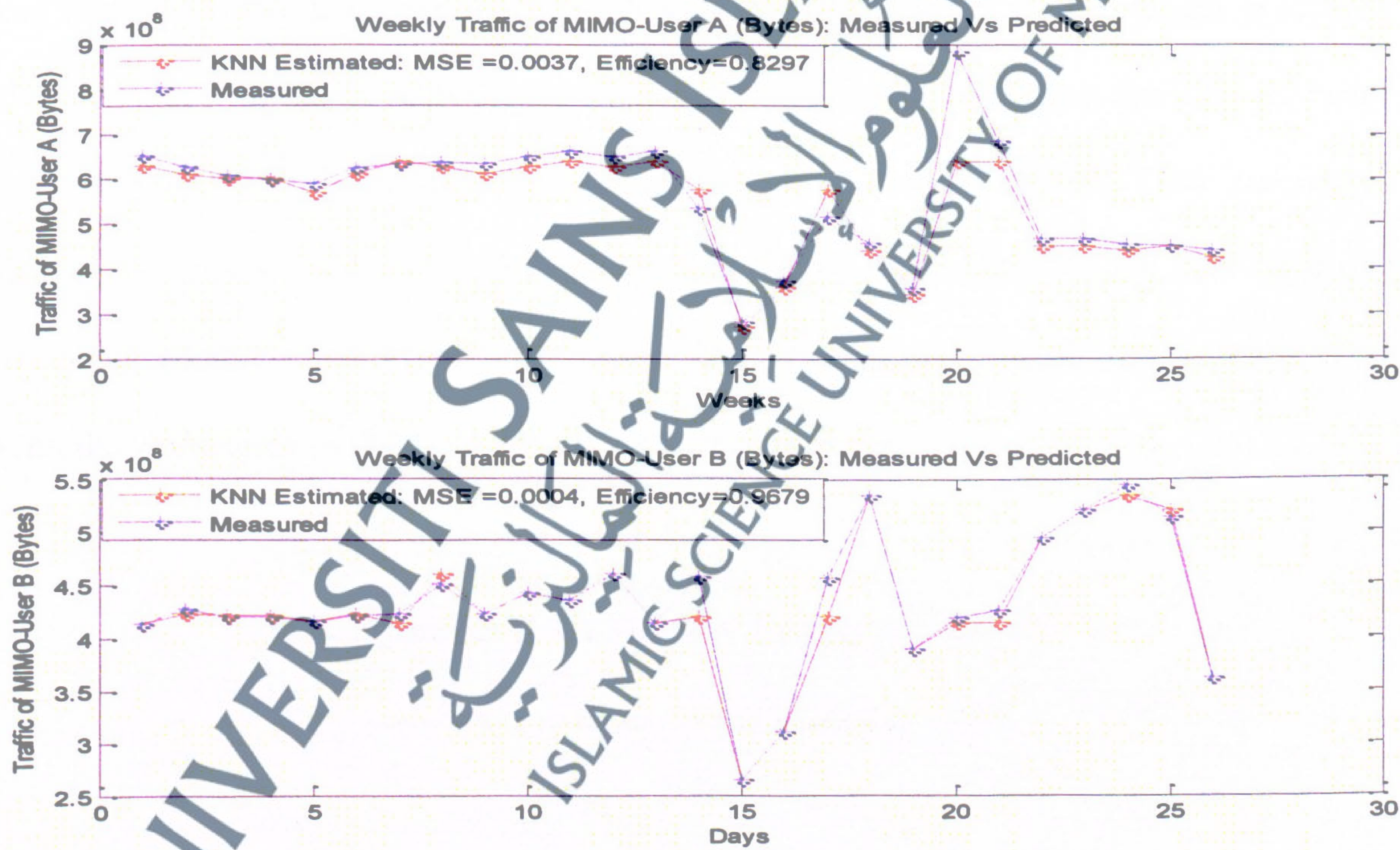


Table 8: KNN Weekly Testing of User AB

USER AB WEEKLY		
K	MSE	M _{eff}
1	0.0004	0.9678
3	0.0004	0.9678
5	0.0004	0.9670
7	0.0004	0.9679
9	0.0006	0.9678

In the weekly testing of the k-nearest neighbor, it can be gathered that certain value of K can minimize error and maximize efficiency. The value of $K = 7$ (Table 9) fails to repeat this advantage for the monthly testing of user A (Figure 53). Although it provides the best efficiency (0.9635), the setting also suffers the worst error at 0.0007. Thus, the most inviting alternative is $K = 1$, which offers the lowest error (0.0004) and near optimal efficiency (0.9633) that differs from the best one by a marginal 0.0002. The worst K is 9 with the worst error (0.0007) and worst performance (0.9621).

Figure 53: KNN Monthly Testing of User A

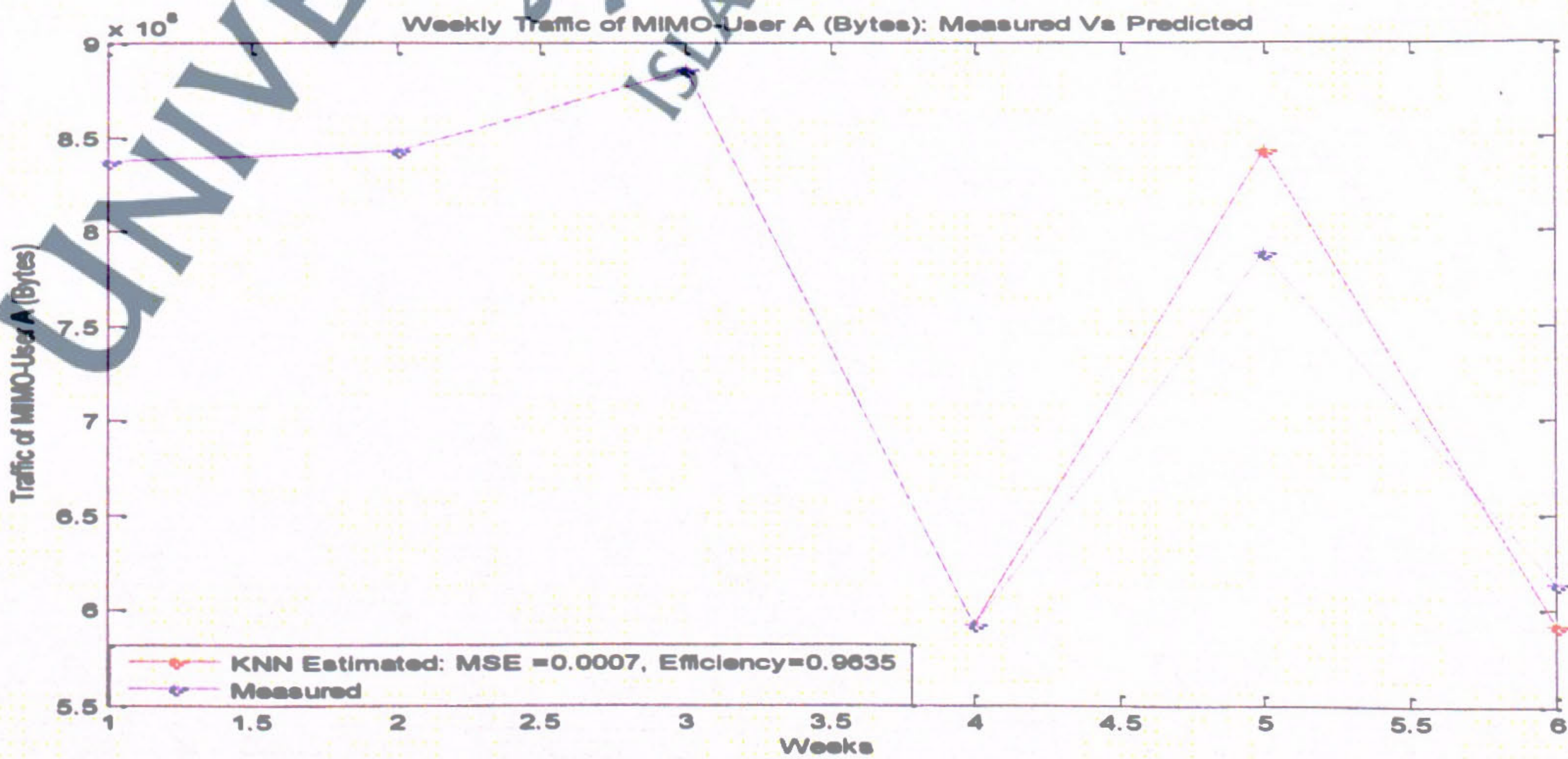


Table 9: KNN Monthly Testing of User A

USER A MONTHLY		
K	MSE	M _{eff}
1	0.0004	0.9633
3	0.0005	0.9621
5	0.0005	0.9688
7	0.0007	0.9635
9	0.0007	0.9621

For the case of monthly testing of user B (Figure 54), all the values of K (Table 10) result to the near least error (0.0001). However, the best K is set at 7 again of which both performance measures is optimized. Here, the error is non detectable at 0.000 and the efficiency is maximum at 0.9963. The lowest efficiency (0.9921) can be found at K = 3 and K = 9, although the value is comparatively similar to the best one. Consistently high efficiency values for all the values of K for the monthly testing indicates the suitability of the approach in forecasting, especially when the duration is long. Finally, the medium value of efficiency (0.9941) can be found when K = 5.

Evidence derived from the result of the monthly testing for user AB (Figure 55) confirms the affinity of $K = 7$ as the most favorable number of referential clusters for the KNN approach (Table 11). When the parameter is defined in this manner, the error is computationally negligible (0.000) and the efficiency is undeniably at its maximum with the value of 0.9963. The most inferior performance occurs when $K = 3$ and $K = 5$ where the error is worst at 0.0001 and the efficiency is least at 0.9921. Even at its worst, the monthly performance of the forecasting for user AB is still commendable.

Figure 55: KNN Monthly Testing of User AB

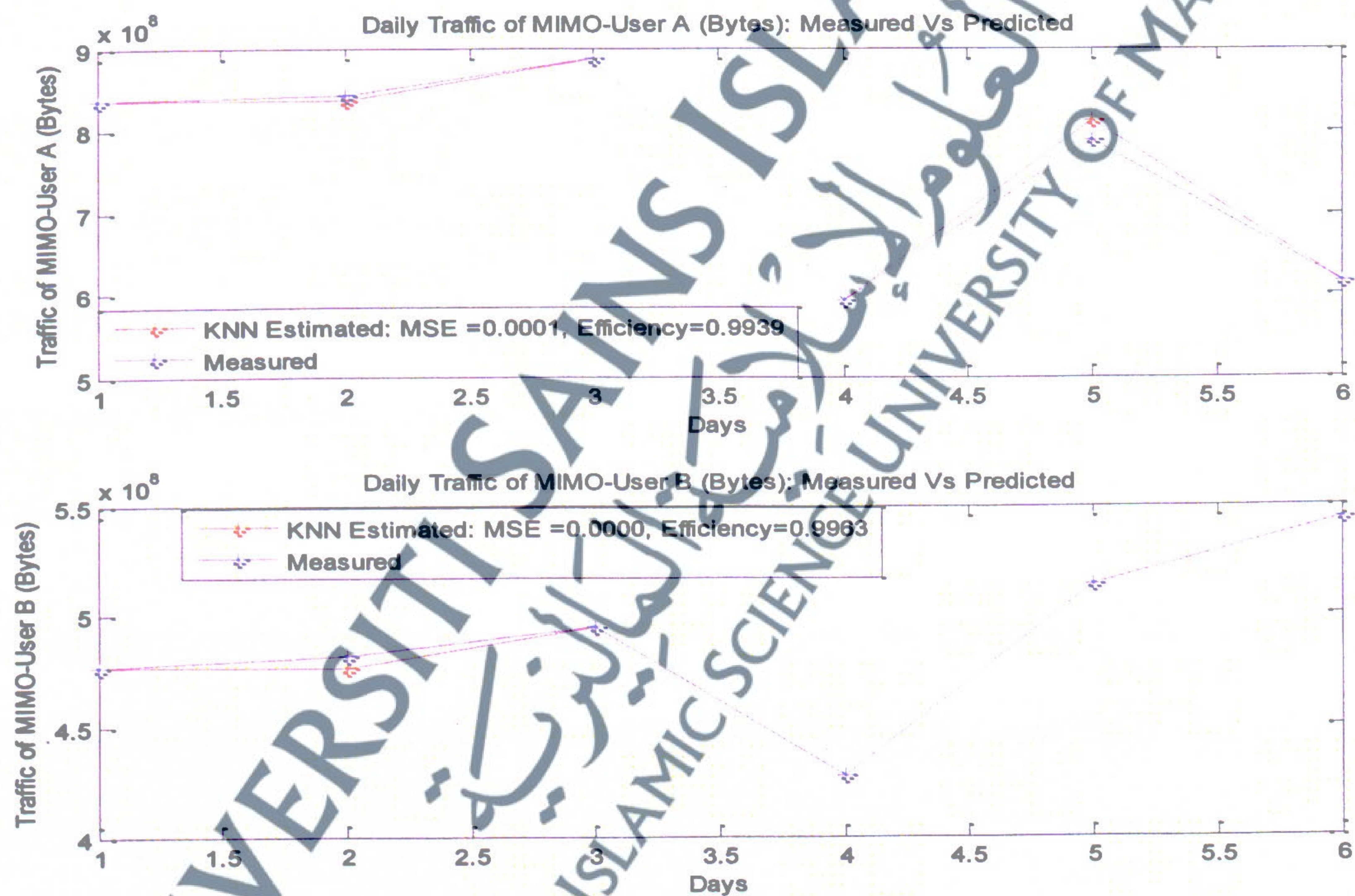


Table 11: KNN Monthly Testing of User AB

USER AB MONTHLY		
K	MSE	M _{eff}
1	0.0001	0.9933
3	0.0001	0.9921
5	0.0001	0.9988
7	0.0000	0.9963
9	0.0001	0.9921

5.3 Data Input and Output for ANN (Daily, Weekly and Monthly)

Similar to the k-nearest neighbor (KNN), the result for artificial neural network (ANN) is also measured based on the mean square error (MSE) and model efficiency (M_{eff}). It also includes the regression indicator (Ismail et al., 2009; Guerard, 2013), denoted by R, that measures how well a particular model can predict new values when given a collection of past data. The difference is quite subtle, but it must be stressed upon that it is not the same as finding the difference between the predicted and actual value.

For instance, suppose that a set of data between (0, 80) is given to the artificial neural network for training. This data is used to learn the classification between integer values into alphabetic ones, thus (0, 80) → [A ... F]. If it is required to predict the impact of (VALUE = 62), then this does not involve any new data since it falls under the range of (0, 80). However, if it is (VALUE = 95), then it is considered a case of new data since it is beyond the range of (0, 80). The regression (R) calculates the capacity of the model to deal with new situations such as the latter.

The artificial neural network (ANN) contains two sets of result. The first result exhibits the training part of the approach while the second result narrates the testing portion. All the other approaches, which are k-nearest neighbor (KNN) and fuzzy time series (FTS) only include their testing results. The training phase of ANN is incorporated into the result because it is quite extensive. In other words, considerable computational effort is invested into it.

Between the two algorithms for forecasting the daily traffic of user A (Figure 56), TrainLM (Figure 57, Figure 58) seems superior to TrainSCG (Figure 59, Figure 60) in terms of error and efficiency (Table 12). The performance of TrainLM from the aspect of testing is best comparatively. Here, the error reaches its minimum at 0.0010 and efficiency reaches its maximum at 0.9523. On the other hand, TrainSCG exhibits the worst performance with the most error (0.0017) and least efficiency (0.9212).

Table 12: ANN Cross Validation of Daily Traffic for User A

[20 10 1]	$T_{Daily(A)} = f_A(X_{user(Max)}, X_{user(Min)})$				
	Training			Testing	
	Algorithms	MSE	M _{eff}	R	MSE
<i>TrainLM</i>	0.000099	0.9464	0.97596	0.0010	0.9523
<i>TrainSCG</i>	0.001033	0.9447	0.97166	0.0017	0.9212

Figure 56 Daily traffic of MIMO User A: Measured vs Prediction

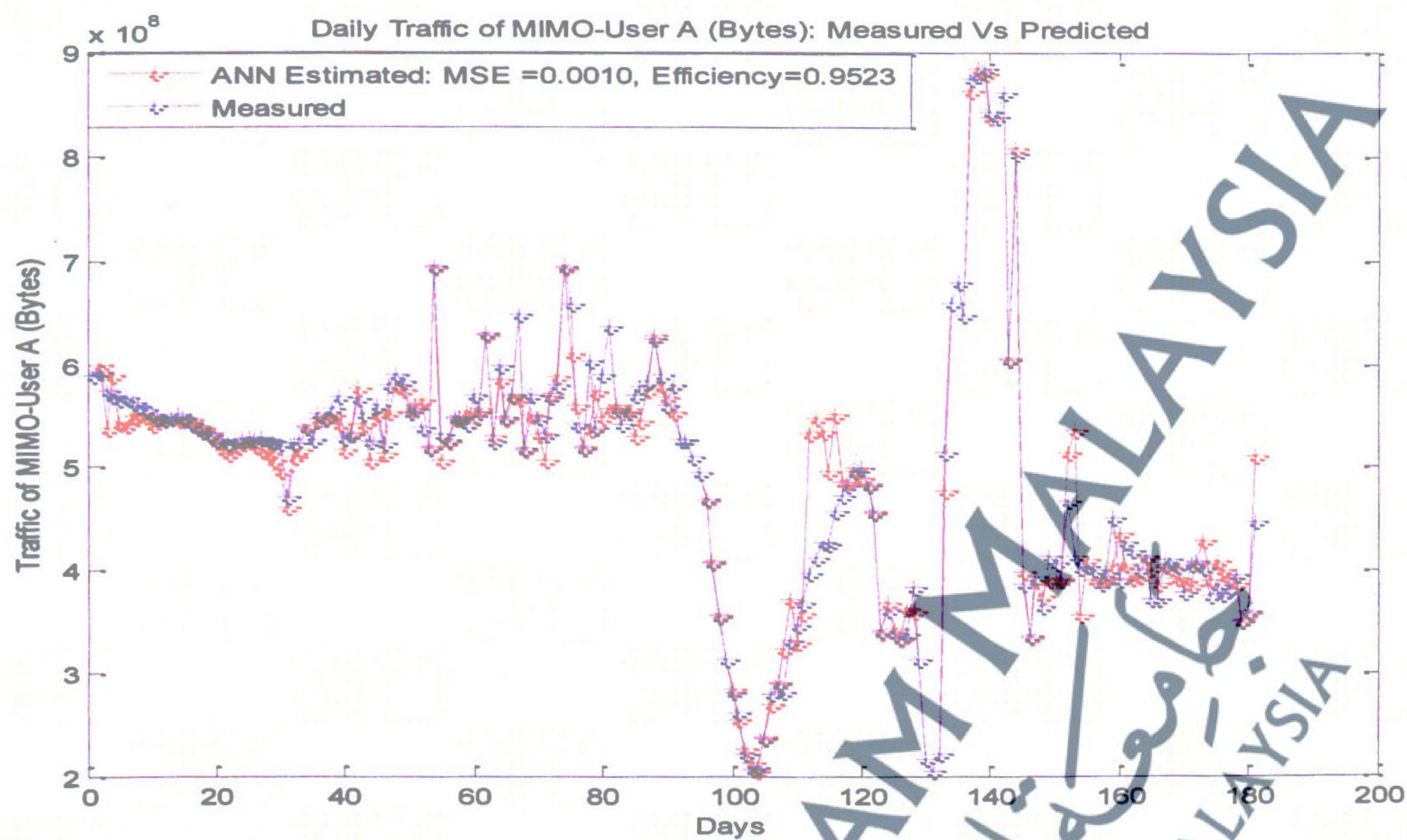


Figure 57: ANN Daily Training for User A - TrainLM



Figure 58: ANN Daily Testing for User A – TrainLM

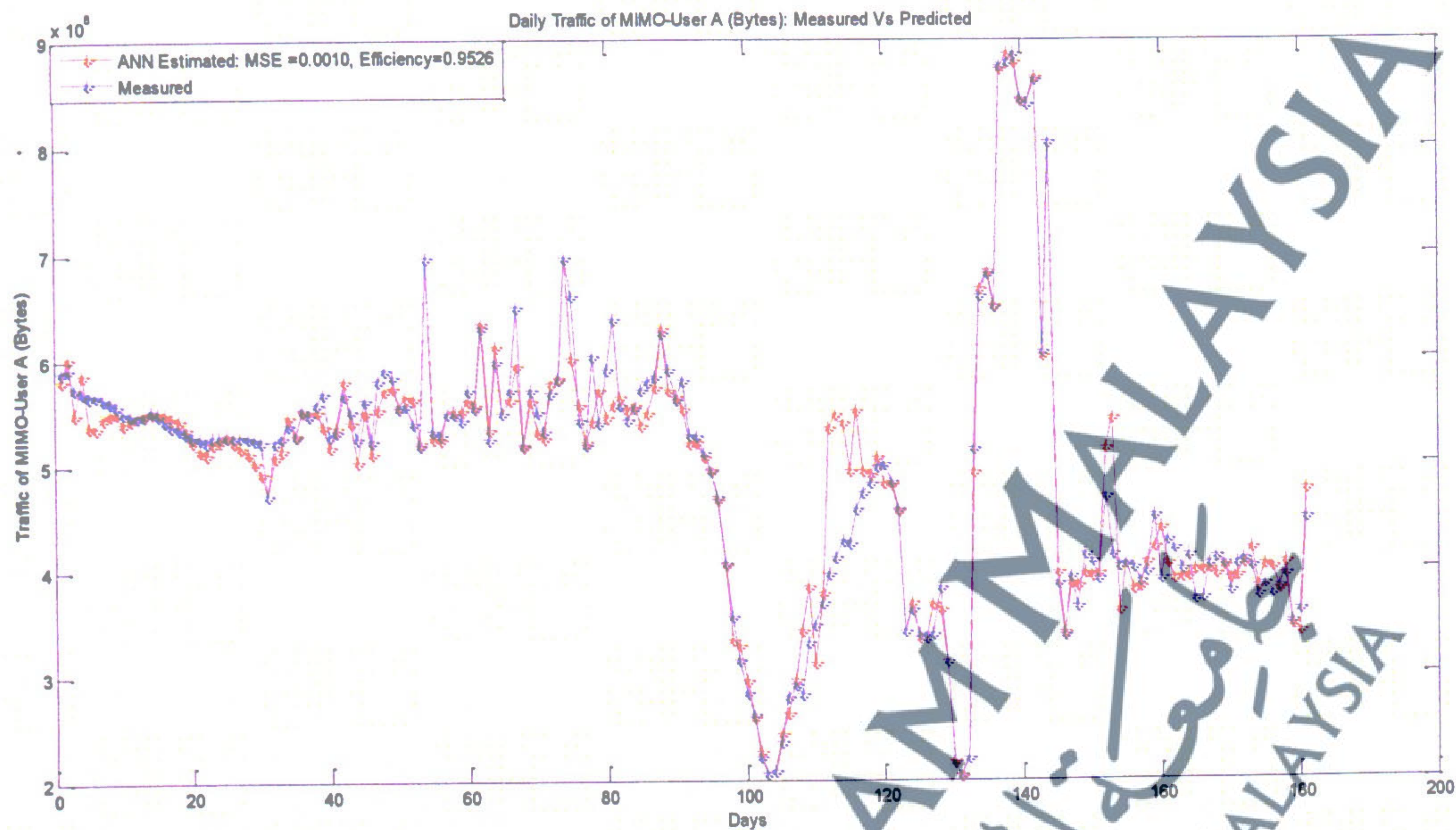


Figure 59: ANN Daily Training for User A – TrainSCG

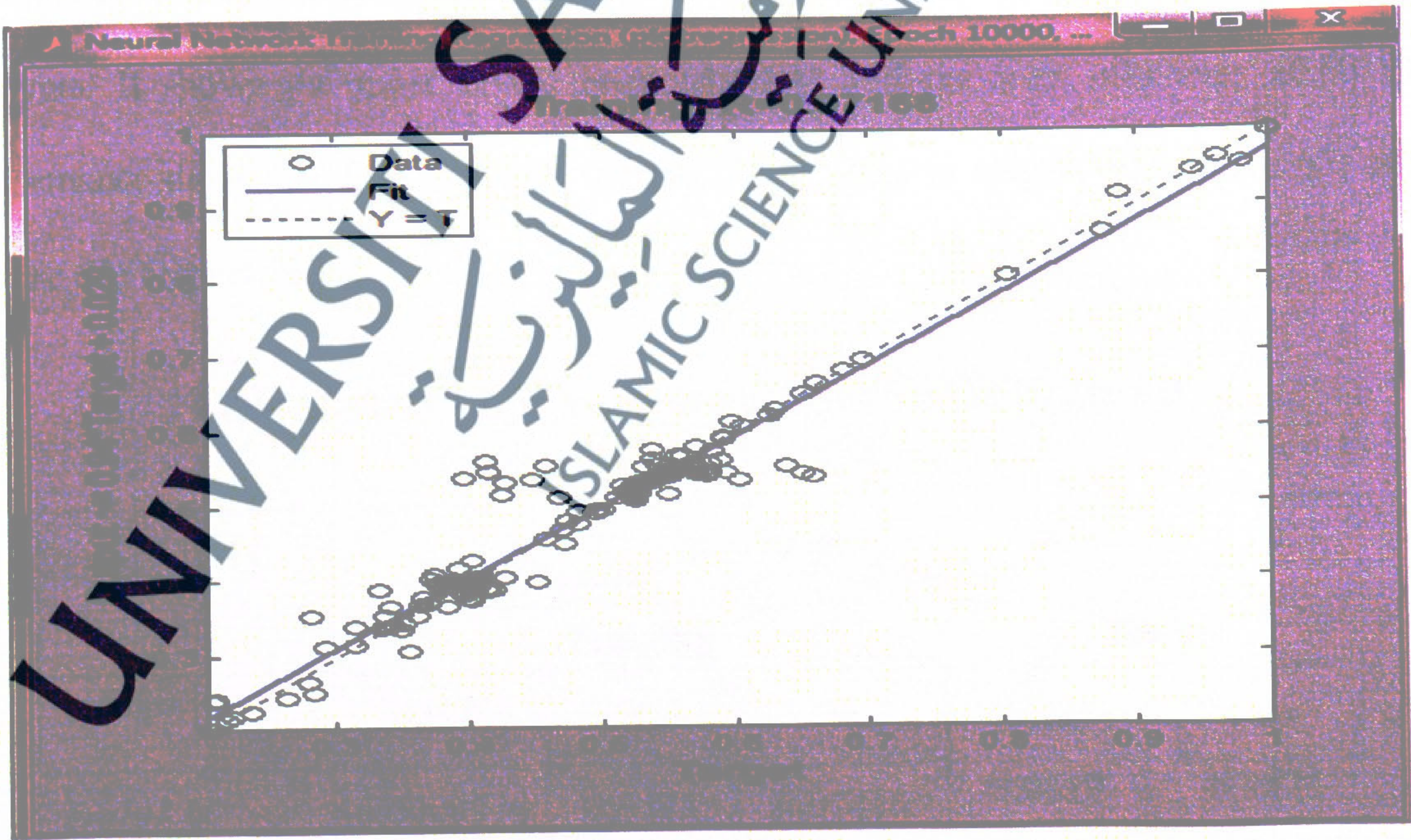
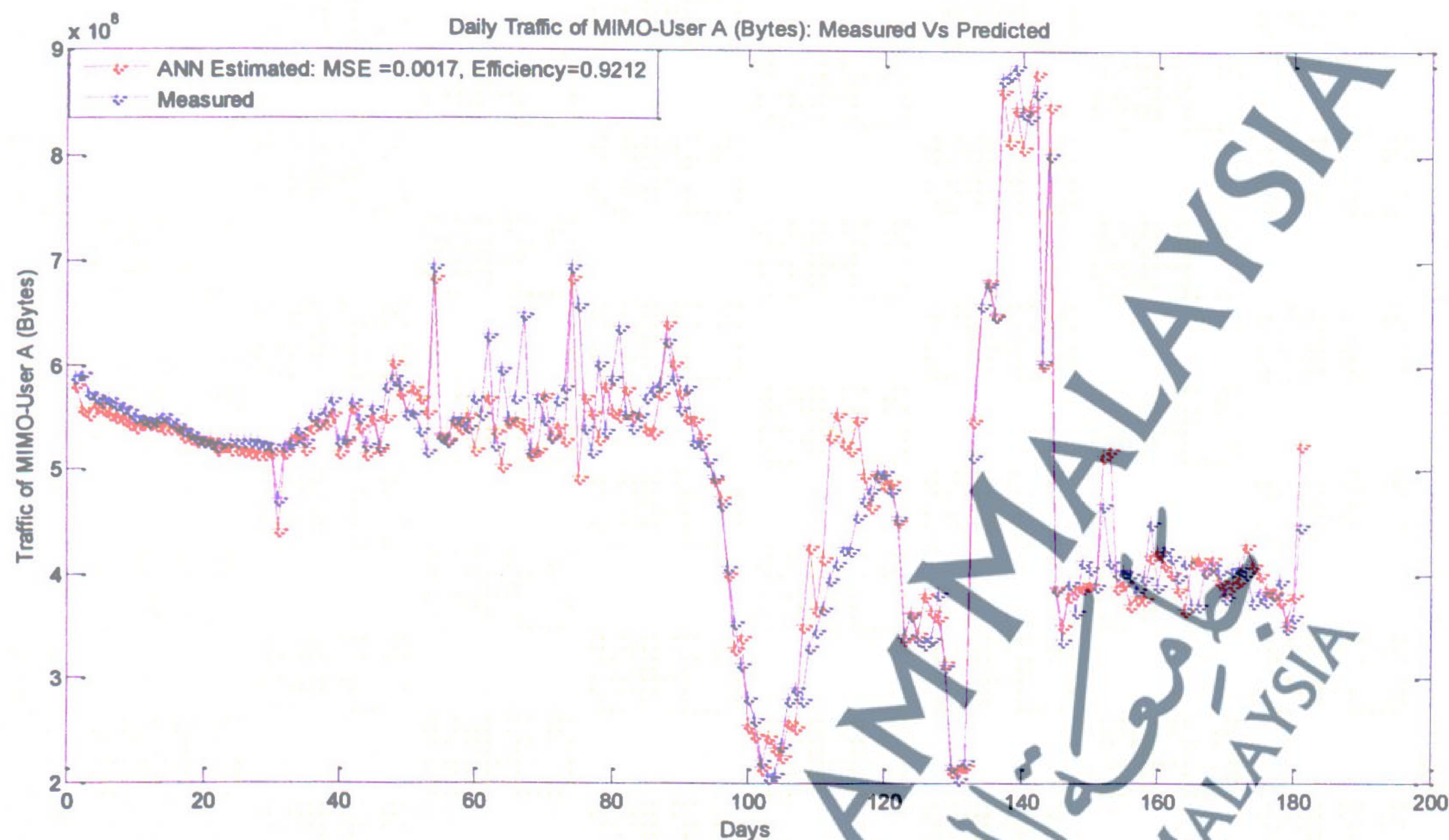


Figure 60: ANN Daily Training for User B - TrainSCG



Similar performance of TrainLM is witnessed for the forecasting of daily traffic of user B (Table 13, Figure 61). Here, it projects optimality by attaining the least error (0.0010) and the most efficiency (0.9472). TrainSCG on the other hand, performs rather poorly on both accounts. It shows the most severe error (0.0038) and the least efficiency (0.8010). Performance details of TrainLM and TrainSCG are given in (Figure 62, Figure 63) and (Figure 64) respectively.

Table 13: ANN Cross Validation of Daily Traffic for User B

[20 10 1]	$T_{Daily(B)} = f_B(X_{user(Max)}, X_{user(Min)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.0009974	0.9473	0.97316	0.0010	0.9472
<i>TrainSCG</i>	0.001027	0.9407	0.89443	0.0038	0.8010

Figure 61: Daily traffic of MIMO User B: Measured VS Prediction

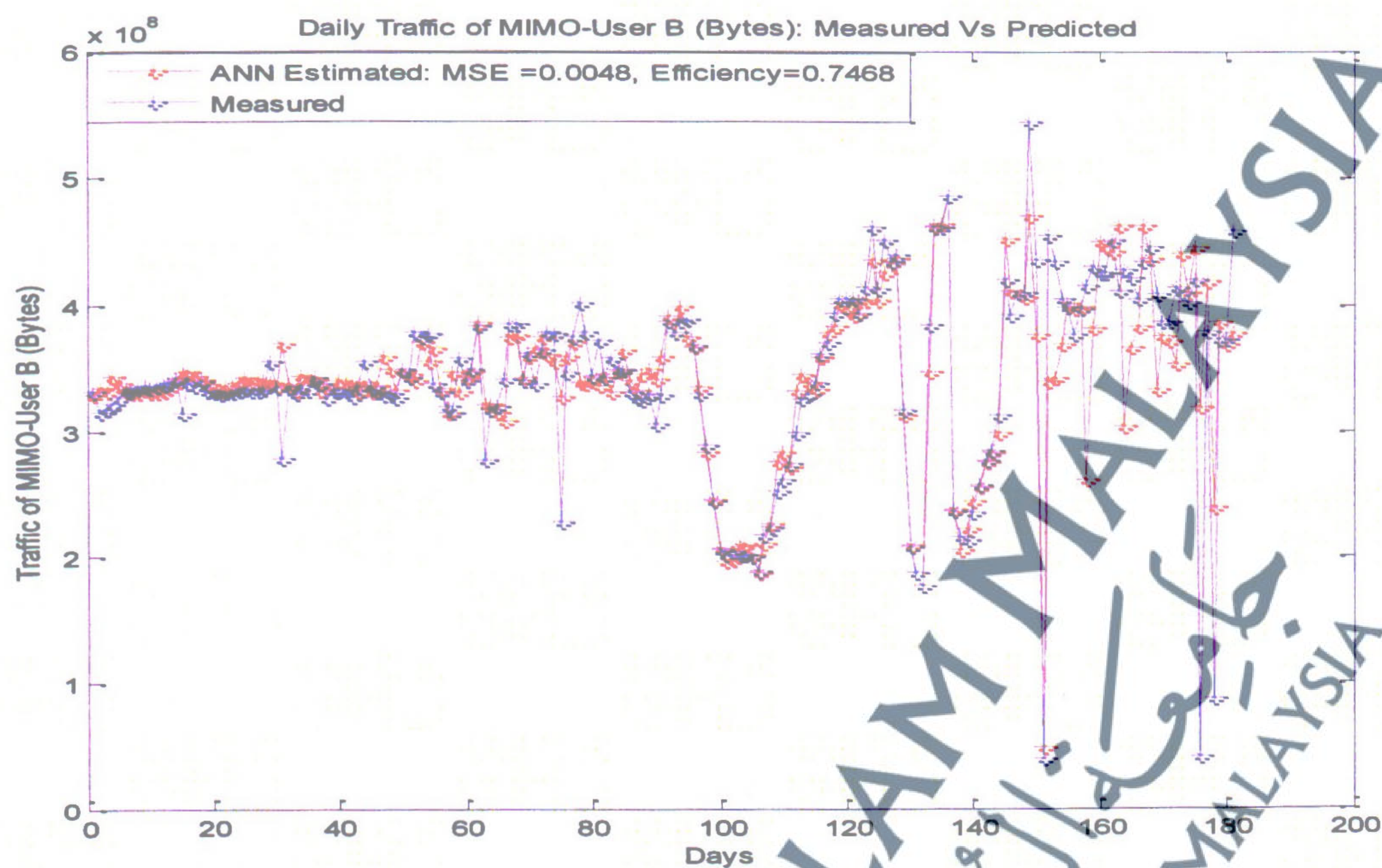


Figure 62: ANN Daily Result for User B – TrainLM

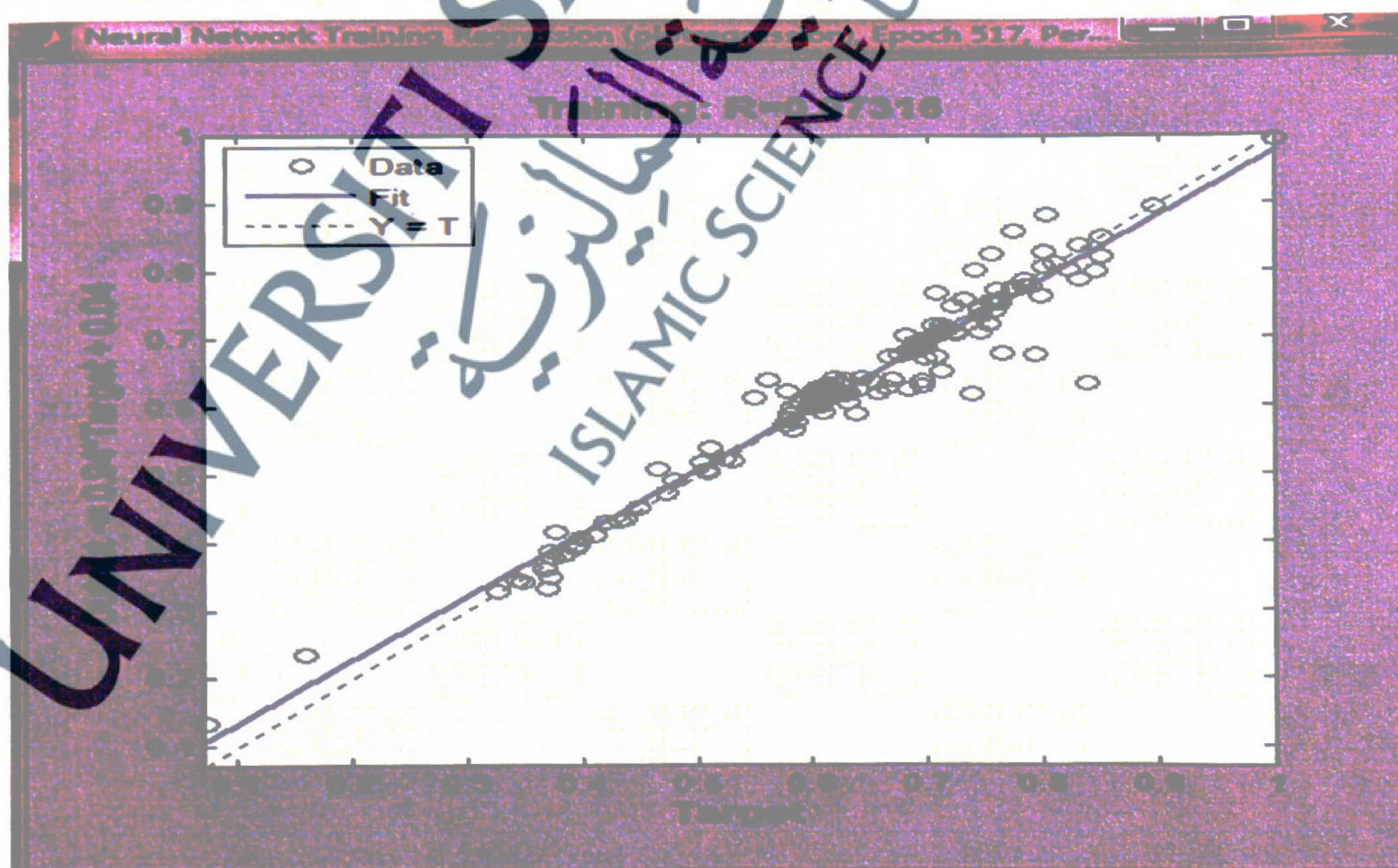


Figure 63: ANN Daily Result for User B – TrainLM

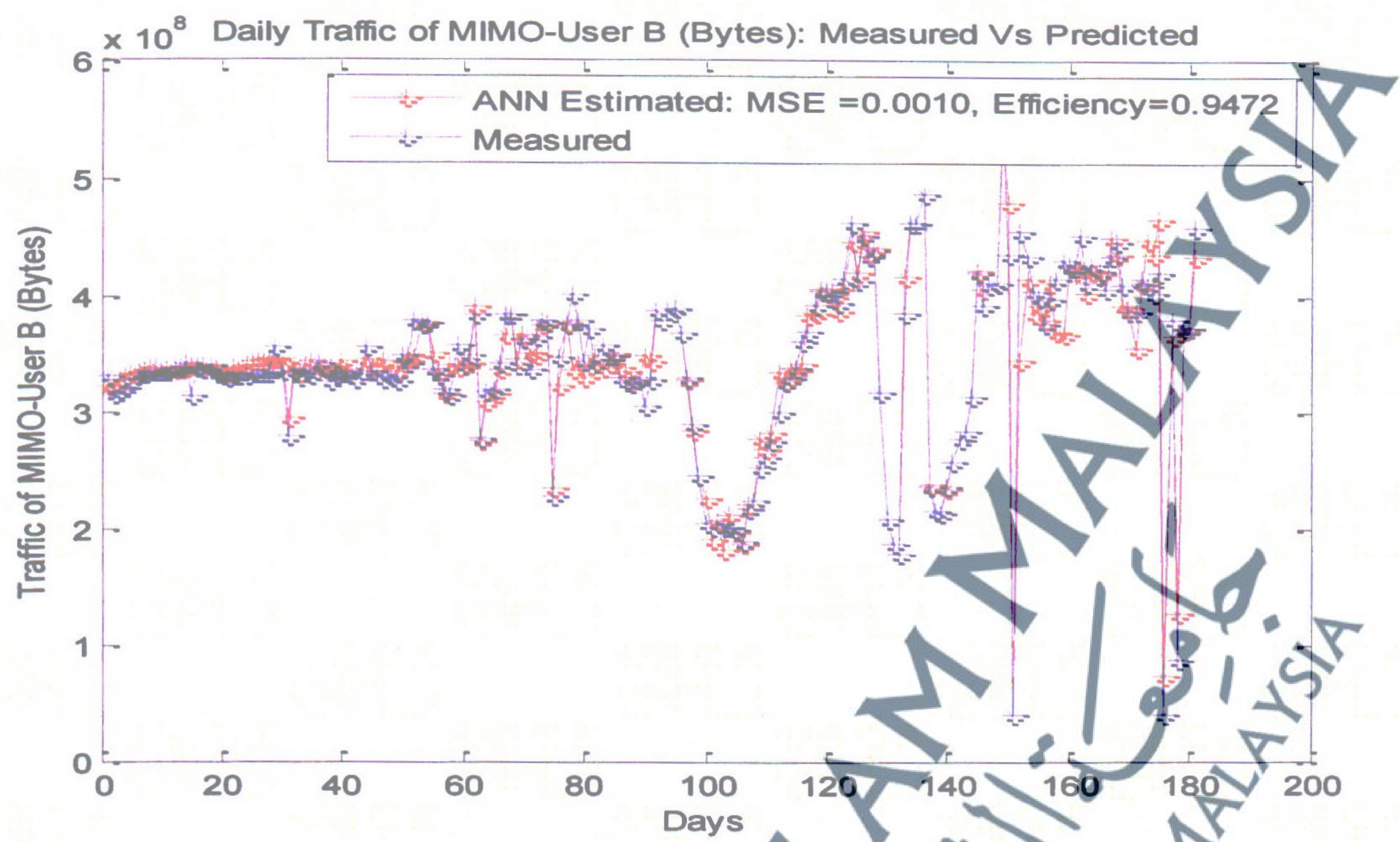
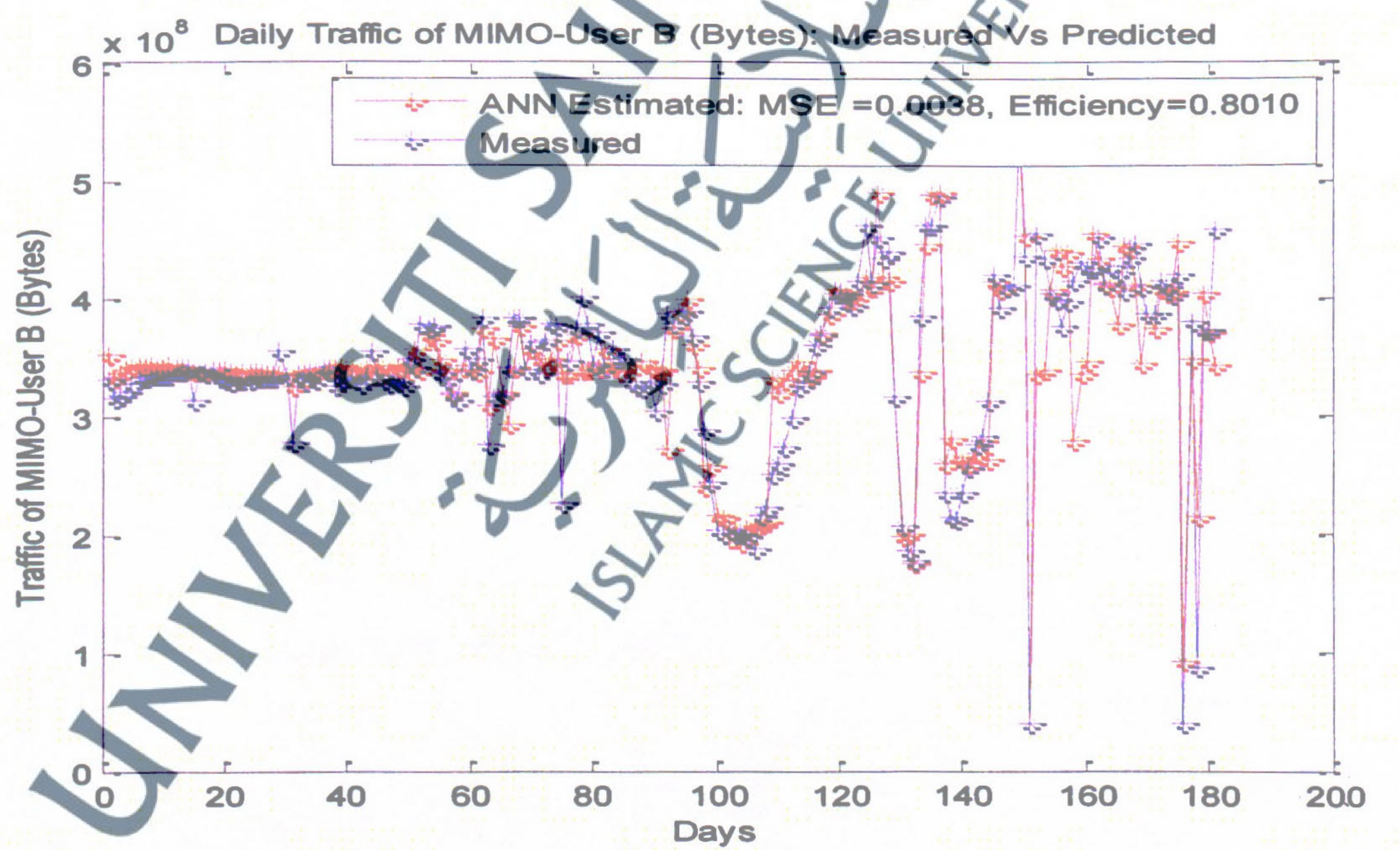


Figure 64: ANN Daily Training Result of User B – TrainSCG



Least error (0.0014) and highest efficiency (0.9277), TrainLM (Figure 65, Figure 66) outperforms TrainSCG (Figure 67, Figure 68) yet again in the daily traffic of user AB (Table14, Figure 69). The performance of TrainSCG is rather severe when contrasted against TrainSCG. Its error (0.0056) is four times worst than the former algorithm. The efficiency (0.9159), although lower than TrainLM, is not as previous in comparison. Evidently, TrainLM is consistently outstanding to TrainSCG in all the three cases - user A, user B, and user AB for the daily duration.

Table 14: ANN Cross Validation of Daily Traffic for User AB

[20 10 2]	$T_{Daily}(AB) = f_{AB}(X_{user(Max)}, X_{user})$				
	Training			Testing	
Algorithms	MSE	Meff	R	MSE	Meff
<i>TrainLM</i>	0.000998	0.9998	0.96057	0.0014	0.9277
<i>TrainSCG</i>	0.001538	0.9638	0.91877	0.0056	0.9159

Figure 65: Daily traffic of MIMO User AB: Measured VS Prediction

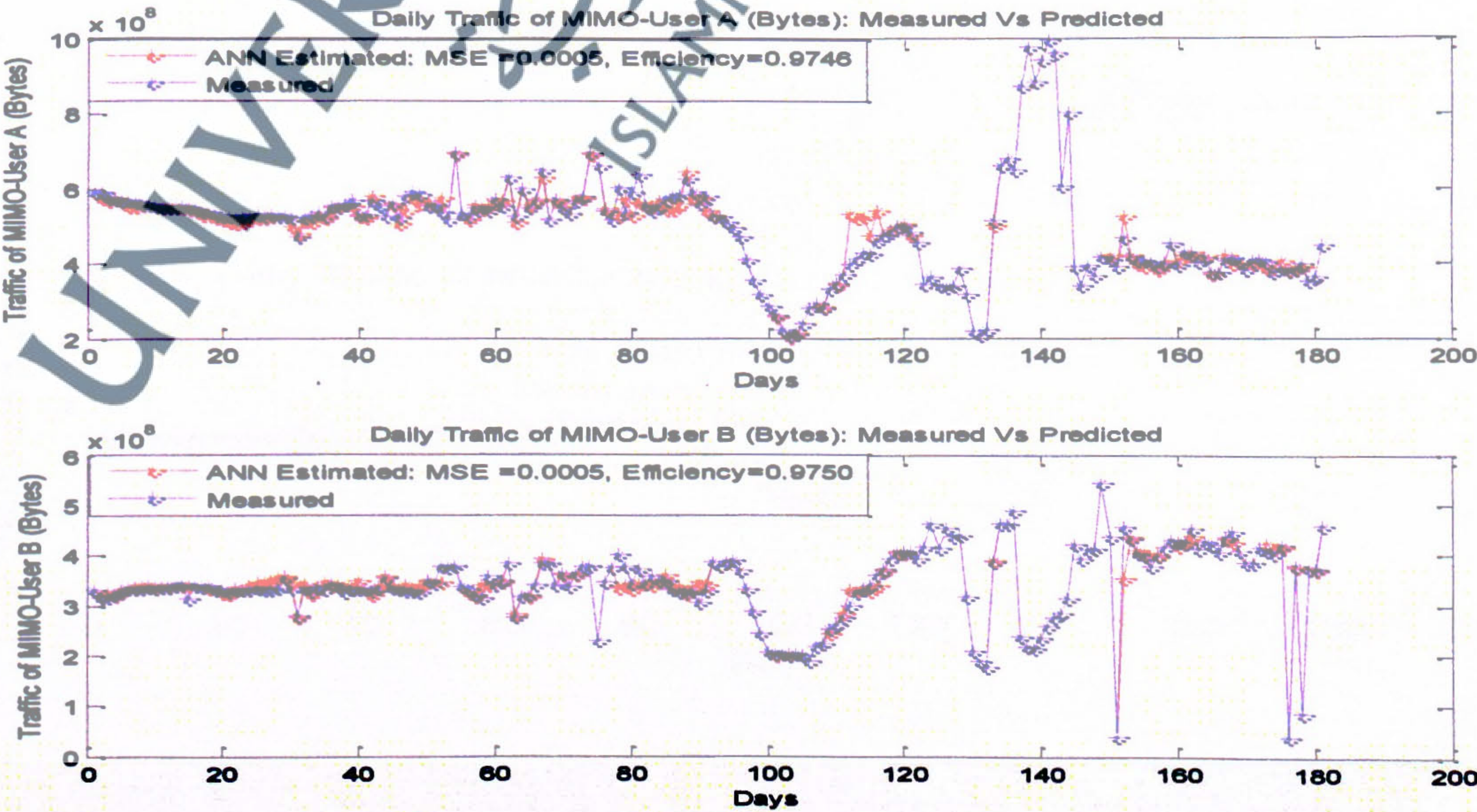


Figure 66: ANN Daily Training Result of User AB – TrainLM

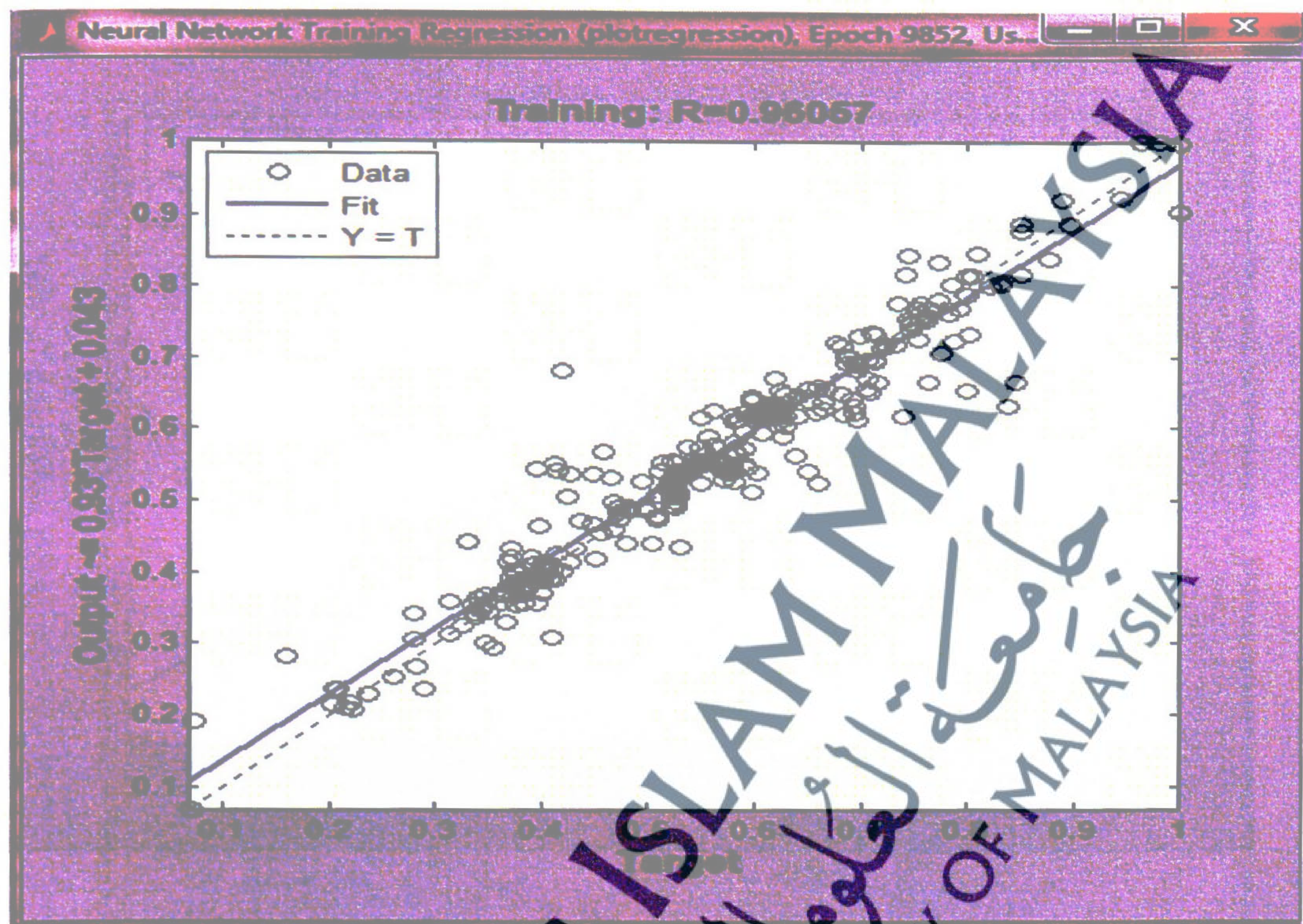


Figure 67: ANN Daily Traffic of User AB – TrainLM

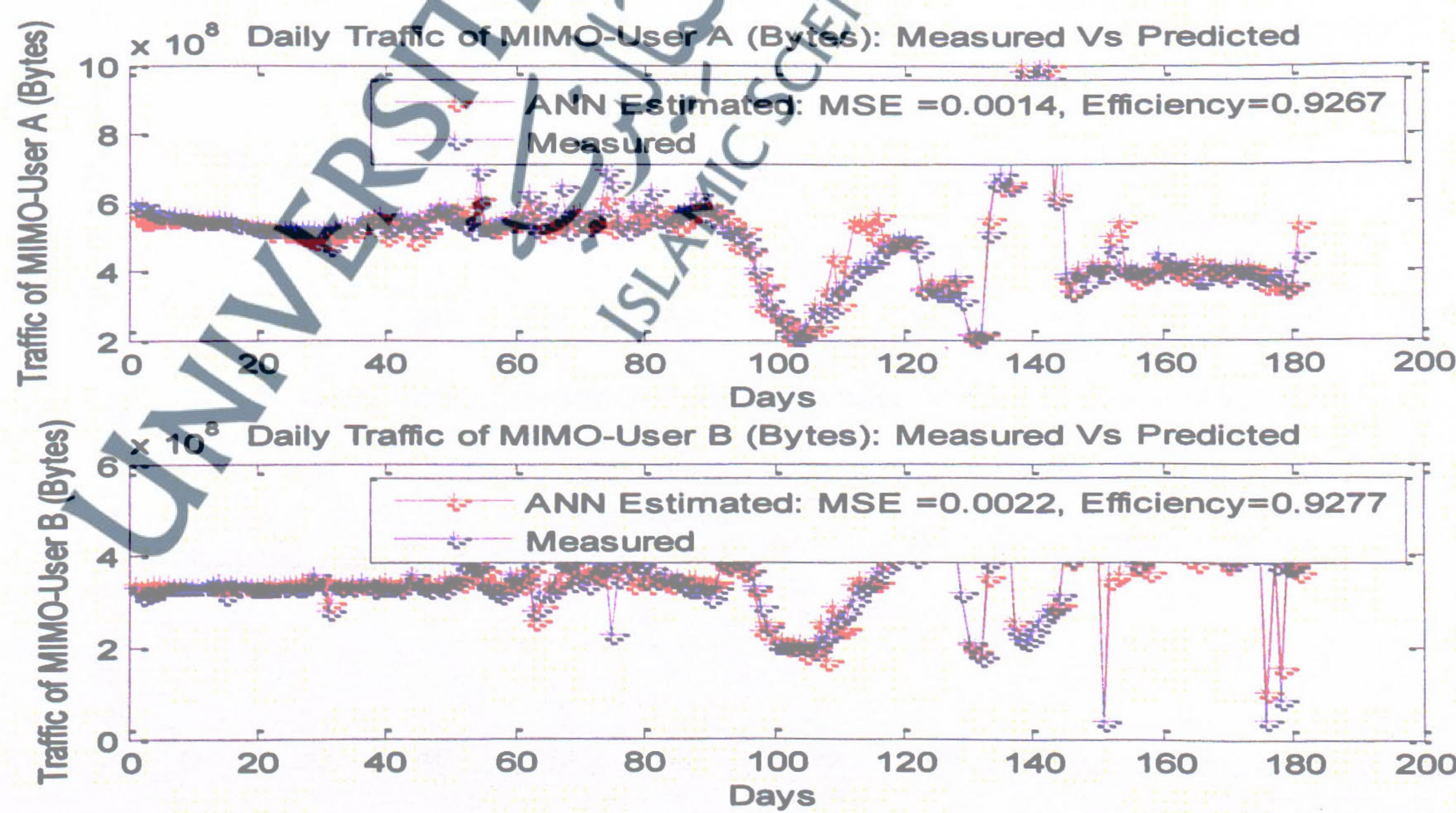


Figure 68: ANN Daily Training Result of User AB – TrainSCG

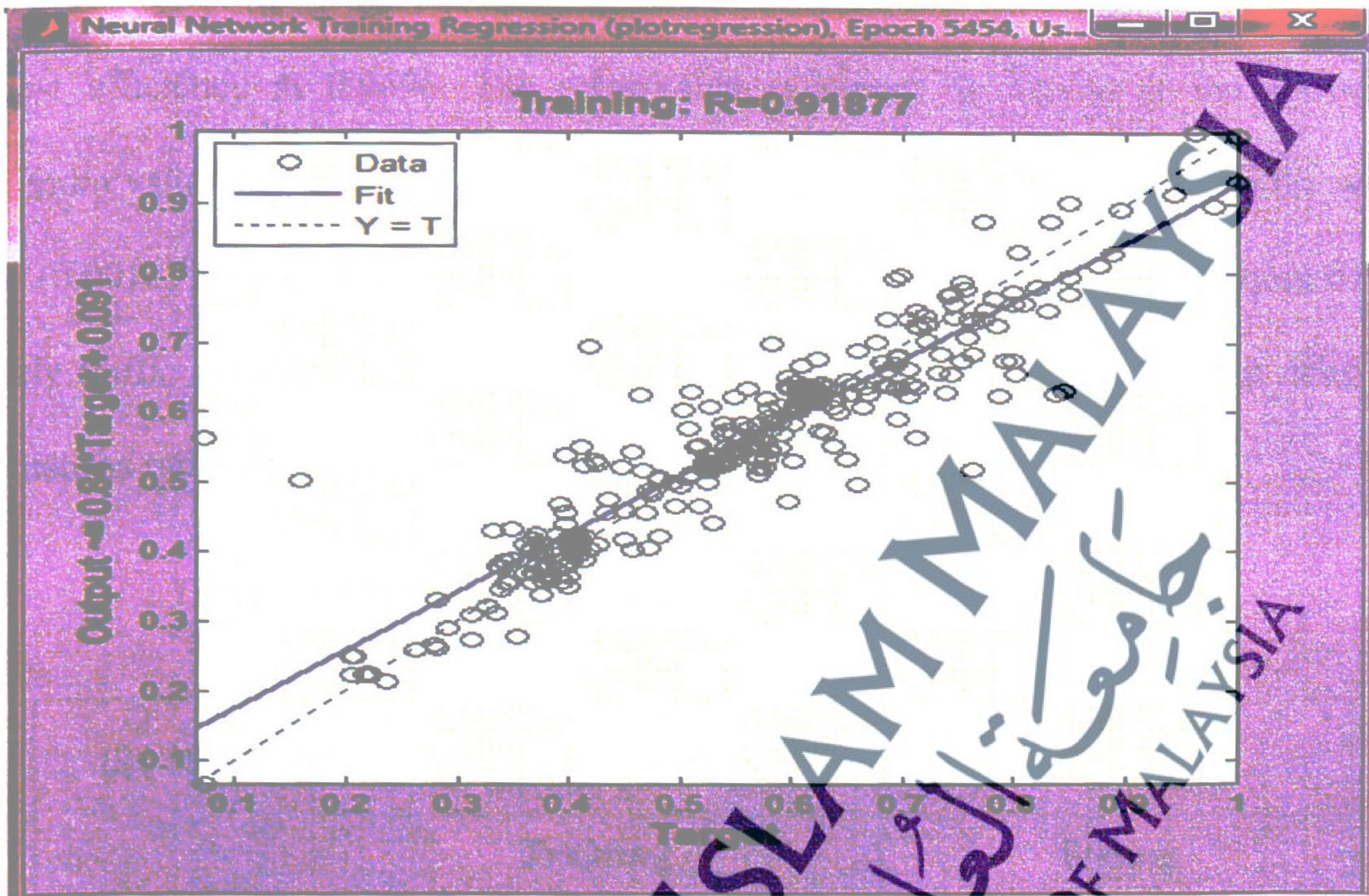
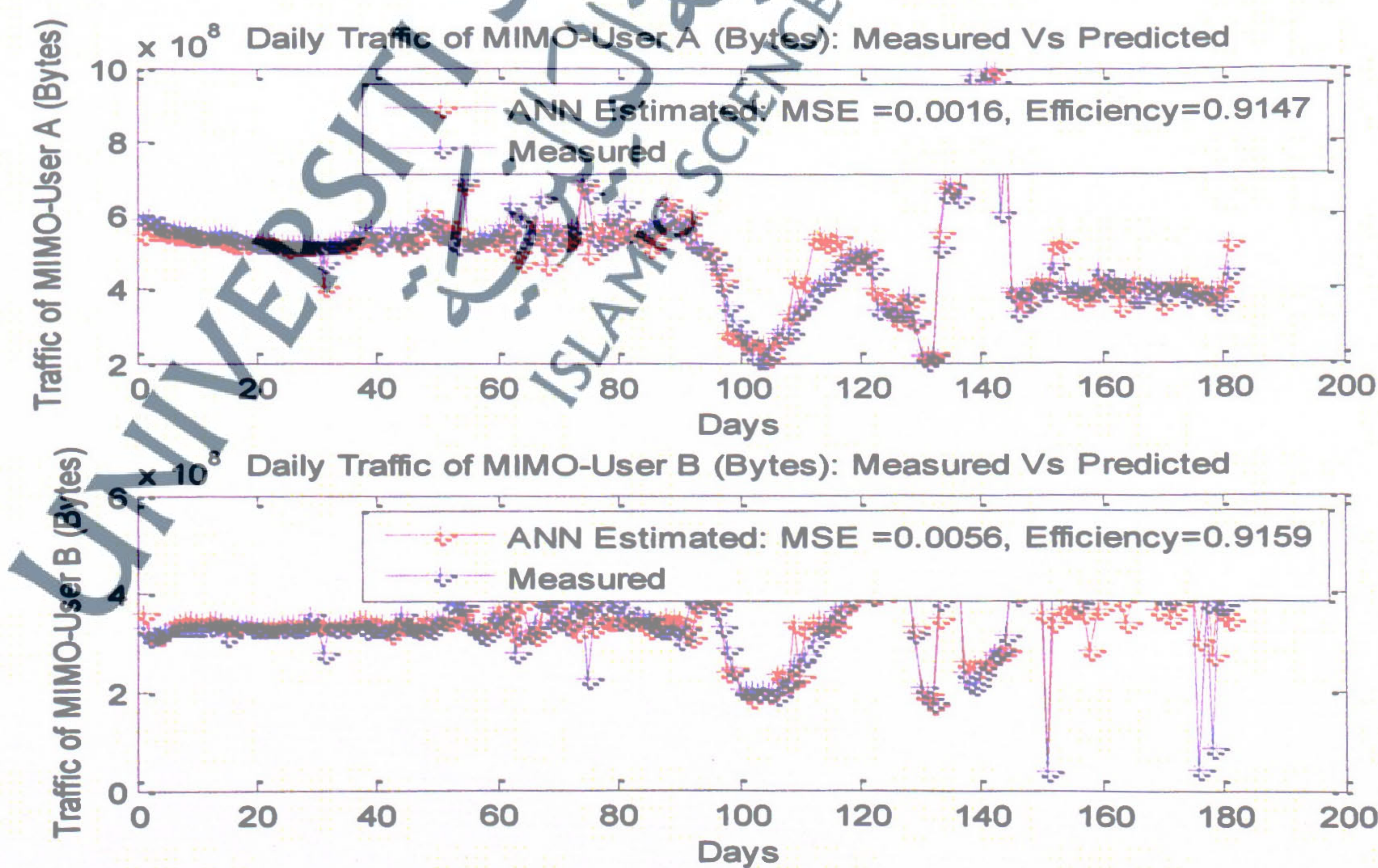


Figure 69: ANN Daily Traffic Result of User AB – TrainSCG



For the weekly result of User A (Table 15, Figure 70), the algorithm TrainLM (Figure 71, Figure 72) shows the most promising performance with the lowest error at 0.0007 and highest efficiency at 0.9696. The remarkable efficiency is almost at the allowable maximum value. On the other hand, TrainSCG (Figure 73, Figure 74) exhibits the worst error (0.0010) and efficiency (0.9540). It can be seen quite clearly that the pattern of the weekly traffic is more stable when compared to the daily one. In this sense, it does not fluctuate as rapidly.

Table 15: ANN Cross Validation of Weekly Traffic for User A

[20 10 2]	$T_{Weekly(A)} = f_A(X_{Max(A)}, X_{Min(A)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.00097138	0.95529	0.97112	0.0007	0.9696
<i>TrainSCG</i>	0.0099844	0.95404	0.97589	0.0010	0.9540

Figure 70: ANN Weekly Traffic Result of User A – TrainLM

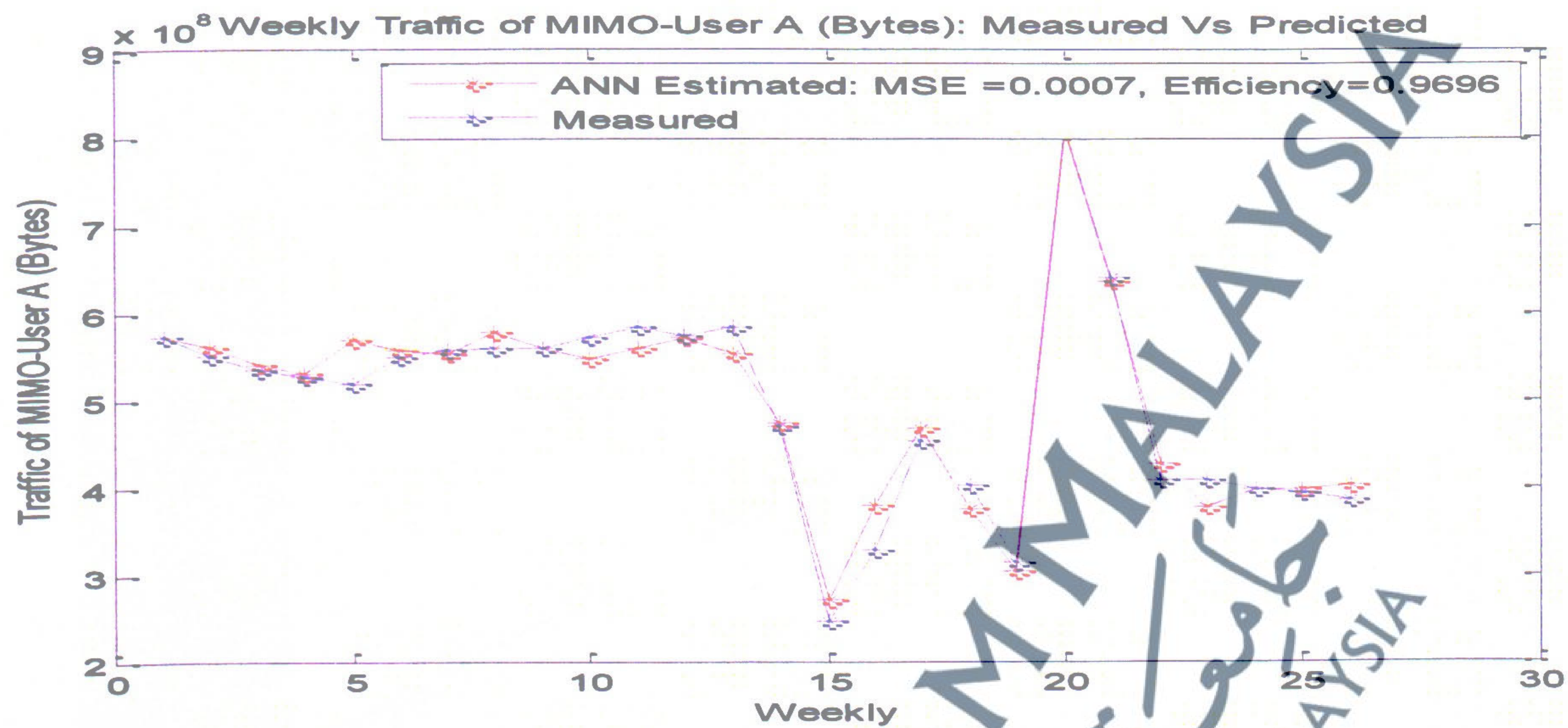


Figure 71: ANN Weekly Training Result of User A – TrainLM



Figure 72: ANN Weekly Training Result of User A – TrainSCG

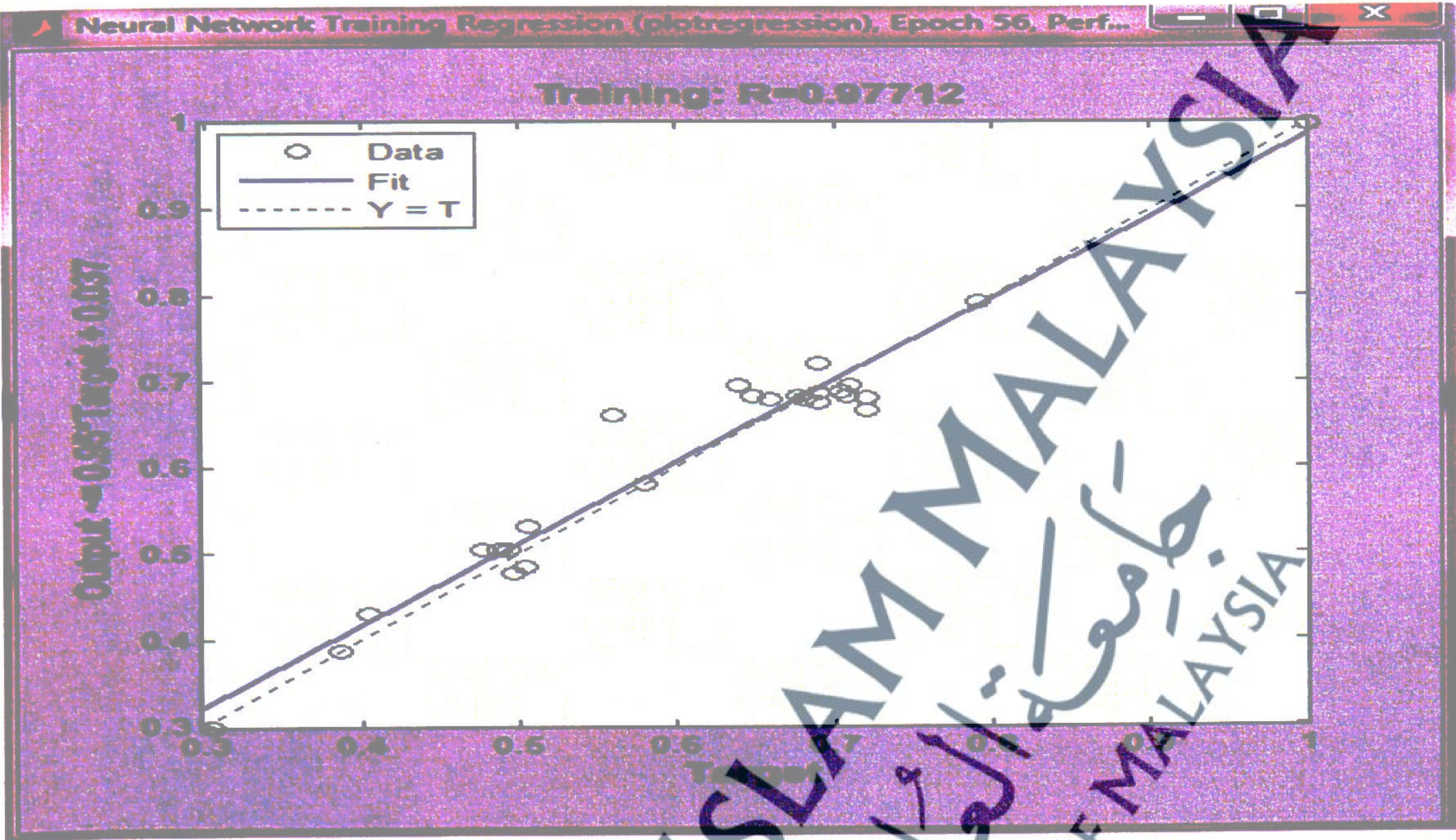


Figure 73: ANN Weekly Traffic Result of User A – Train SCG

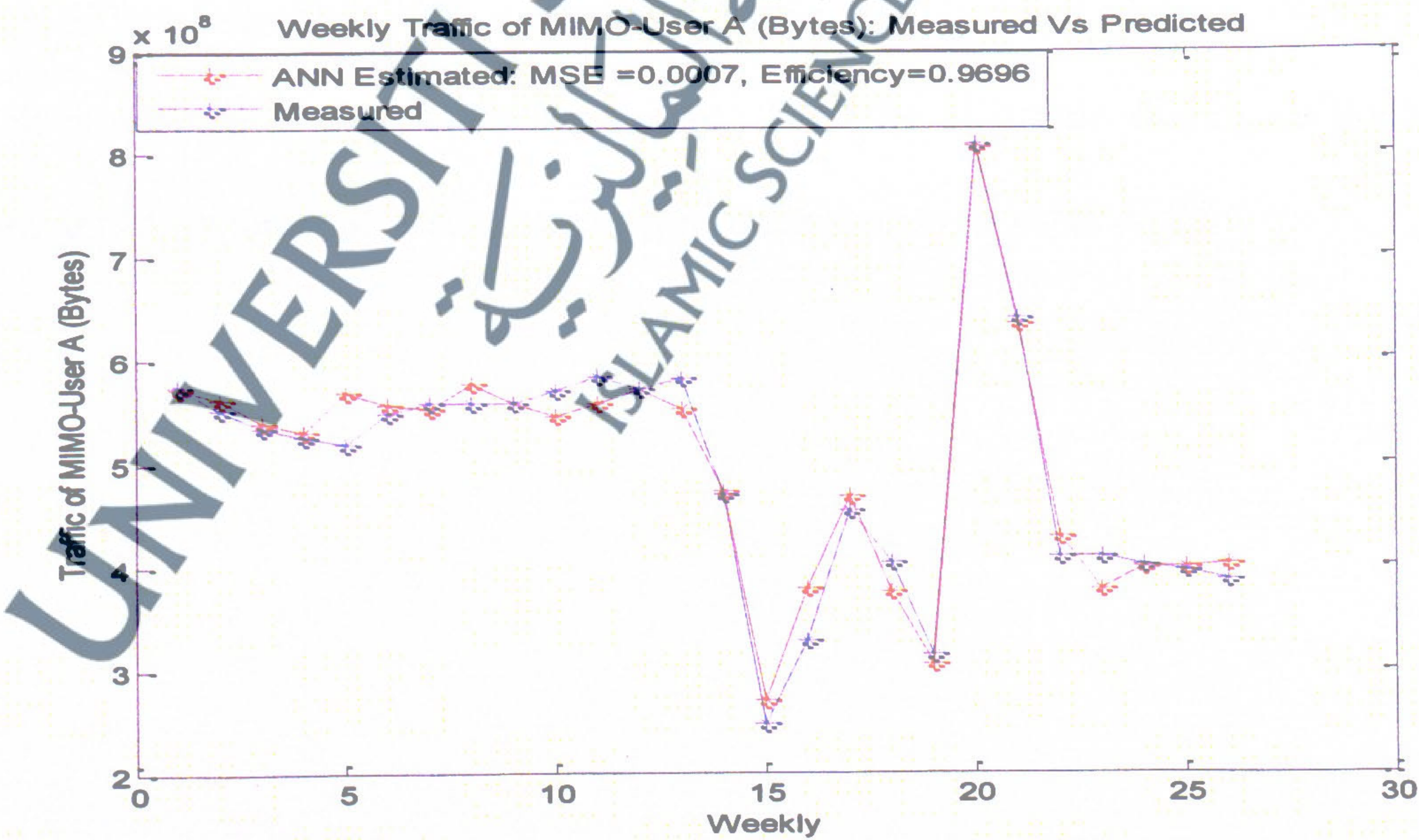
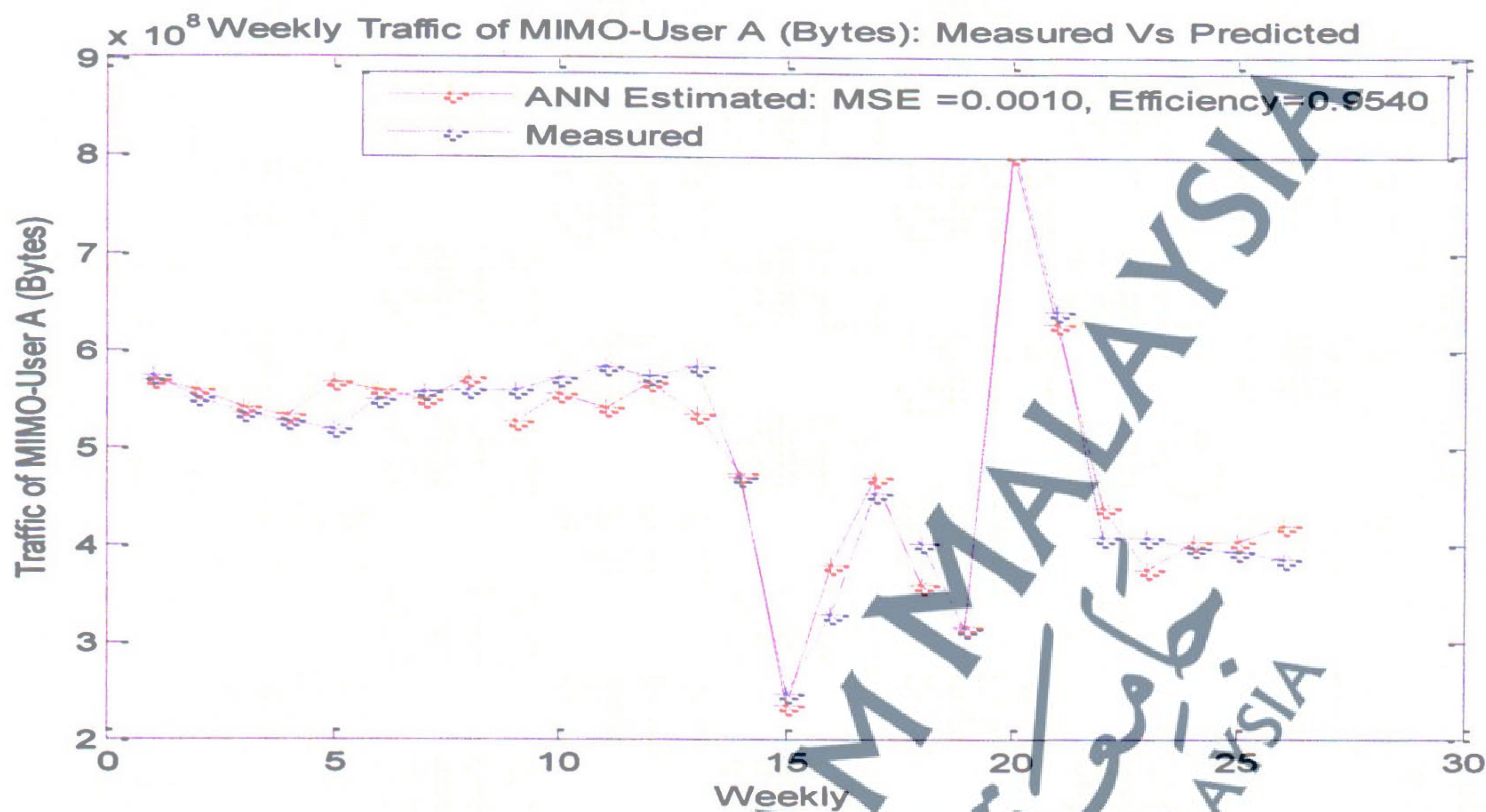


Figure 74: ANN Weekly Traffic Result of User A – TrainLM



A perceptible equality of performance can be seen in the performance of TrainLM and TrainSCG for the weekly traffic of user B (Table 16, Figure 75). Both of them portrays a similar degree of error at 0.0010. In addition, their efficiency in this particular setting is quite approximate as well. TrainLM (Figure 76, Figure 77) shows an efficiency of 0.9223 which differs only by 0.0002 with the efficiency of TrainSCG (Figure 78, Figure 79), which is deemed at 0.9221. In this regard, the latter algorithm displays a noteworthy improvement in terms of performance when compared to its other cases.

Table 16: ANN Cross Validation of Weekly Traffic for User B

[20 10 2]	$T_{Weekly(B)} = f_A(X_{Max(B)}, X_{Min(B)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.00099665	0.95422	0.96067	0.0010	0.9223
<i>TrainSCG</i>	0.009993	0.92209	0.96617	0.0010	0.9221

Figure 75: Weekly traffic of MIMO User B: Measured VS Prediction

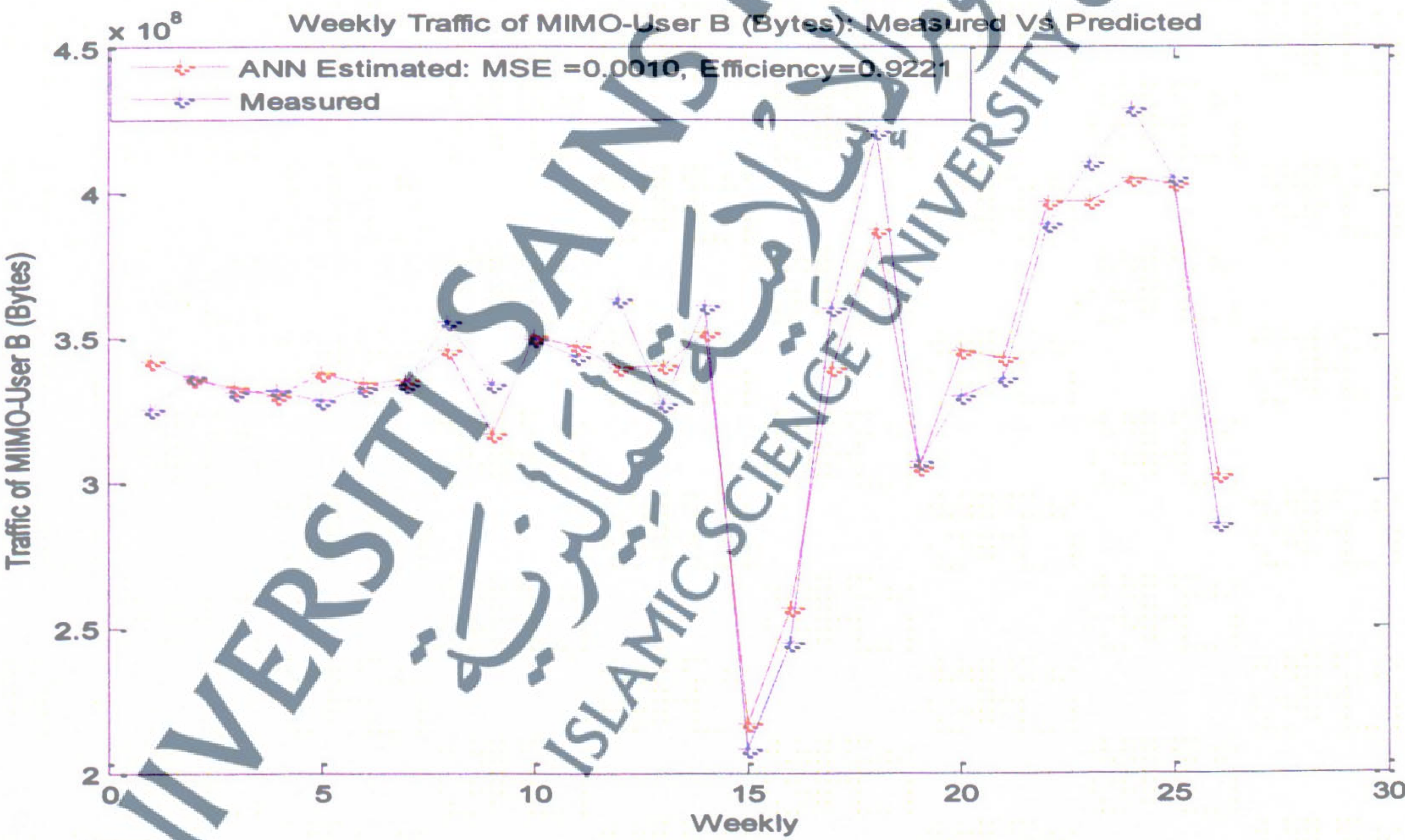


Figure 78: ANN Weekly Training Result of User B – TrainSCG

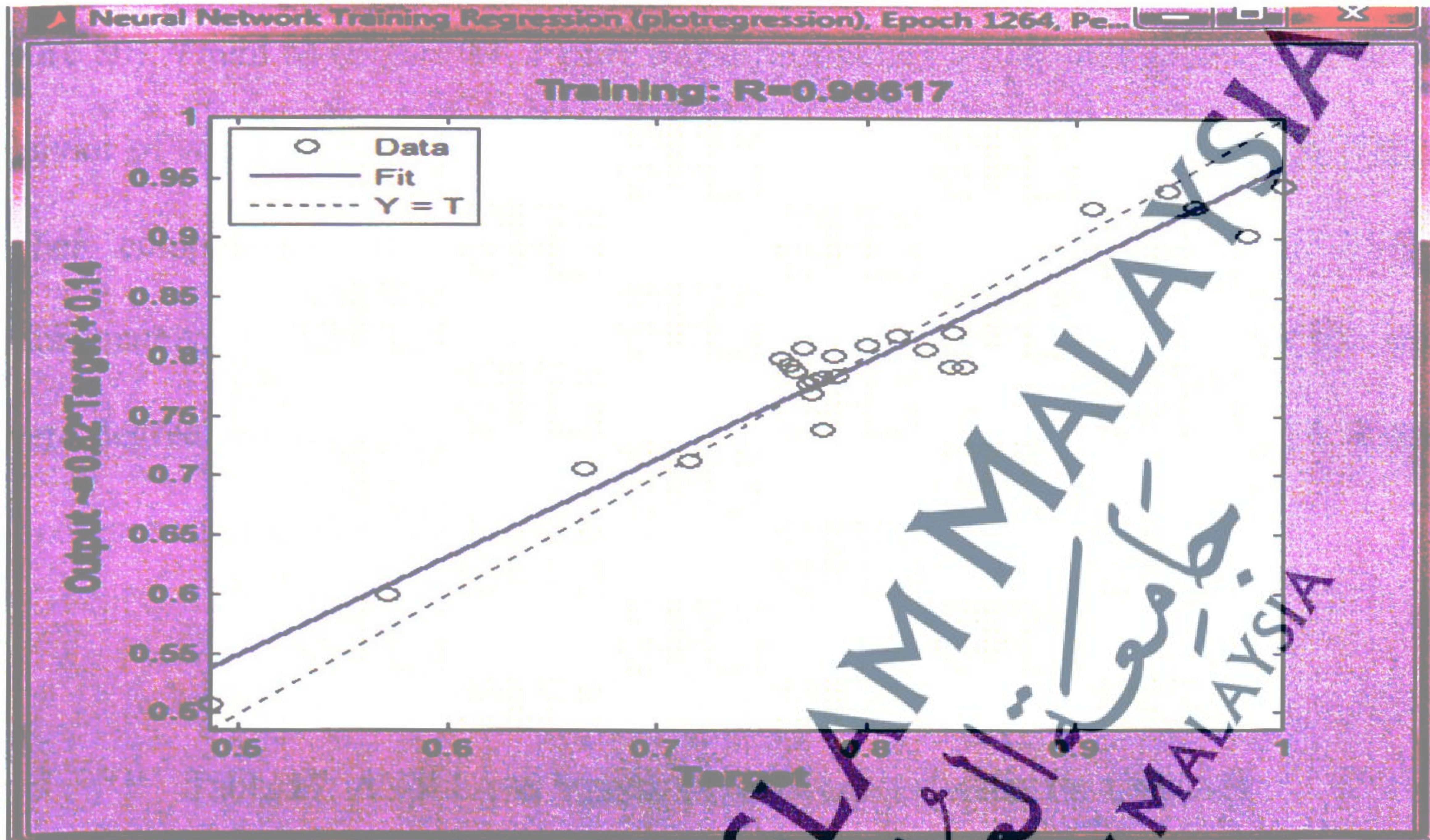
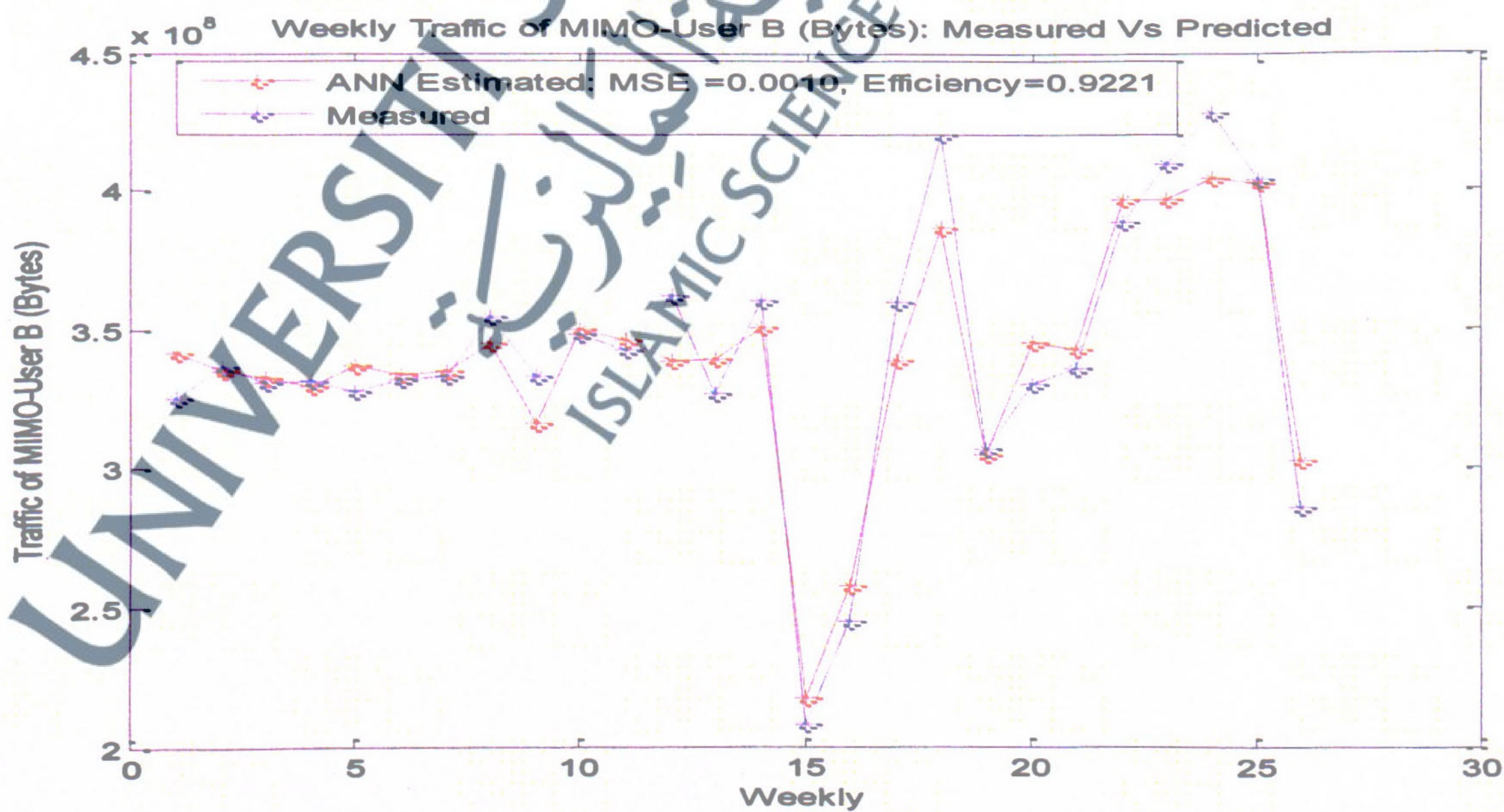


Figure 79: ANN Weekly Traffic Result of User B – TrainSCG



It is not often that the forecasting of the ANN algorithm can make a prediction that is really close to the actual data. In the case of the weekly traffic for user AB (Table 17, Figure 80), TrainLM (Figure 81, Figure 82) demonstrates a truly exemplary performance behavior of which the margin of error (0.0000) is too marginal for manifestation. This is further compounded by the extremely high portrayal of efficiency at 0.9983. Unfortunately, TrainSCG (Figure 83, Figure 84) achieves the opposite with the most severe degree of error at 0.0020. Although the efficiency is quite high at 0.9924, it does not compensate for the tendency for error.

Table 17: ANN Cross Validation of Weekly Traffic for User AB

[20 10 2]	$T_{Weekly (AB)} = f_A (X_{Max (AB)}, X_{Min (AB)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.00099665	0.99321	0.99889	0.0000	0.9983
<i>TrainSCG</i>	0.009993	0.098776	0.96615	0.0020	0.9924

Figure 80: Weekly traffic of MIMO User AB: Measured VS Prediction

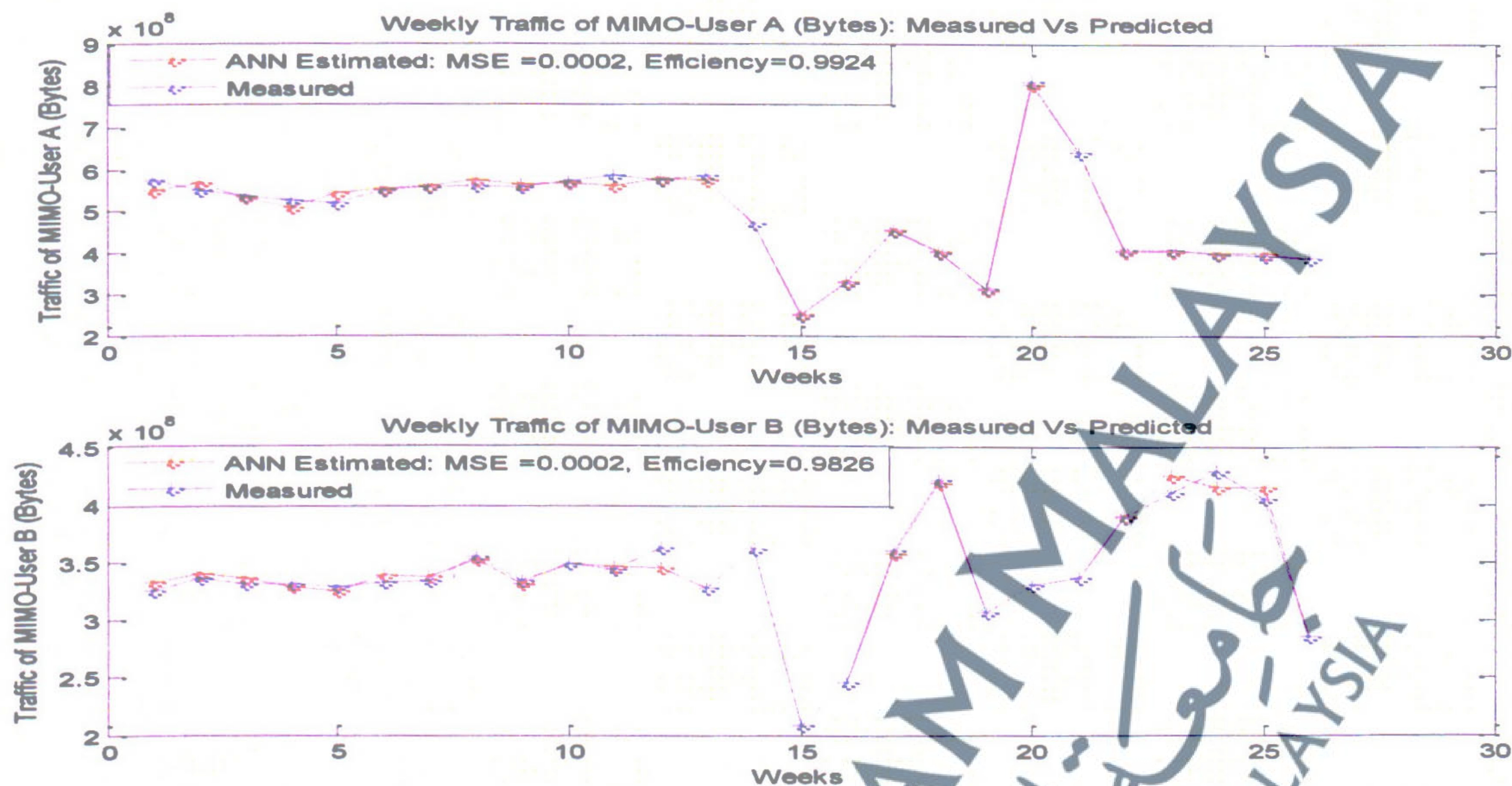


Figure 81: ANN Weekly Training Result of User AB – TrainLM

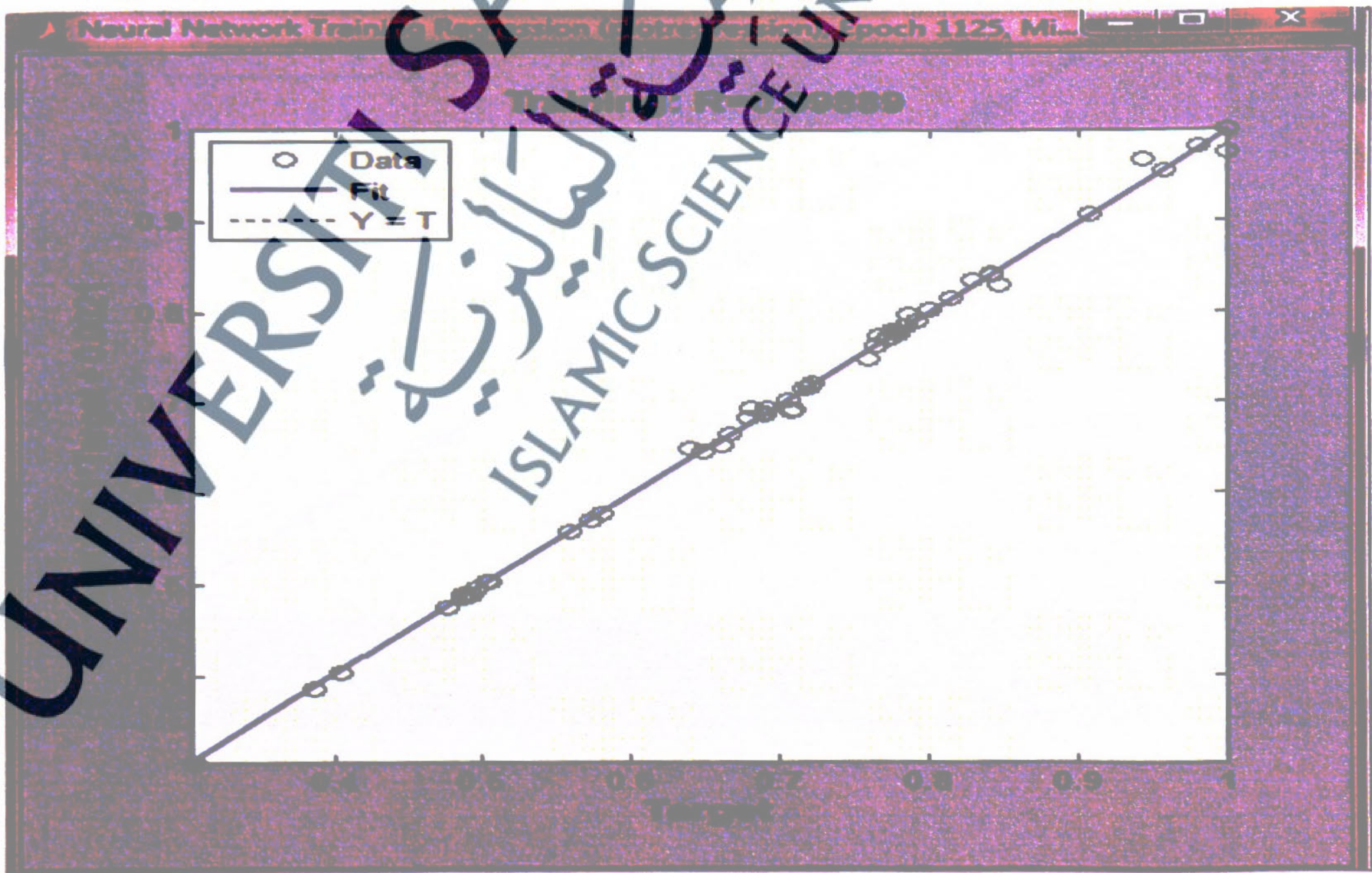


Figure 82: ANN Weekly Traffic Result of User AB – TrainLM

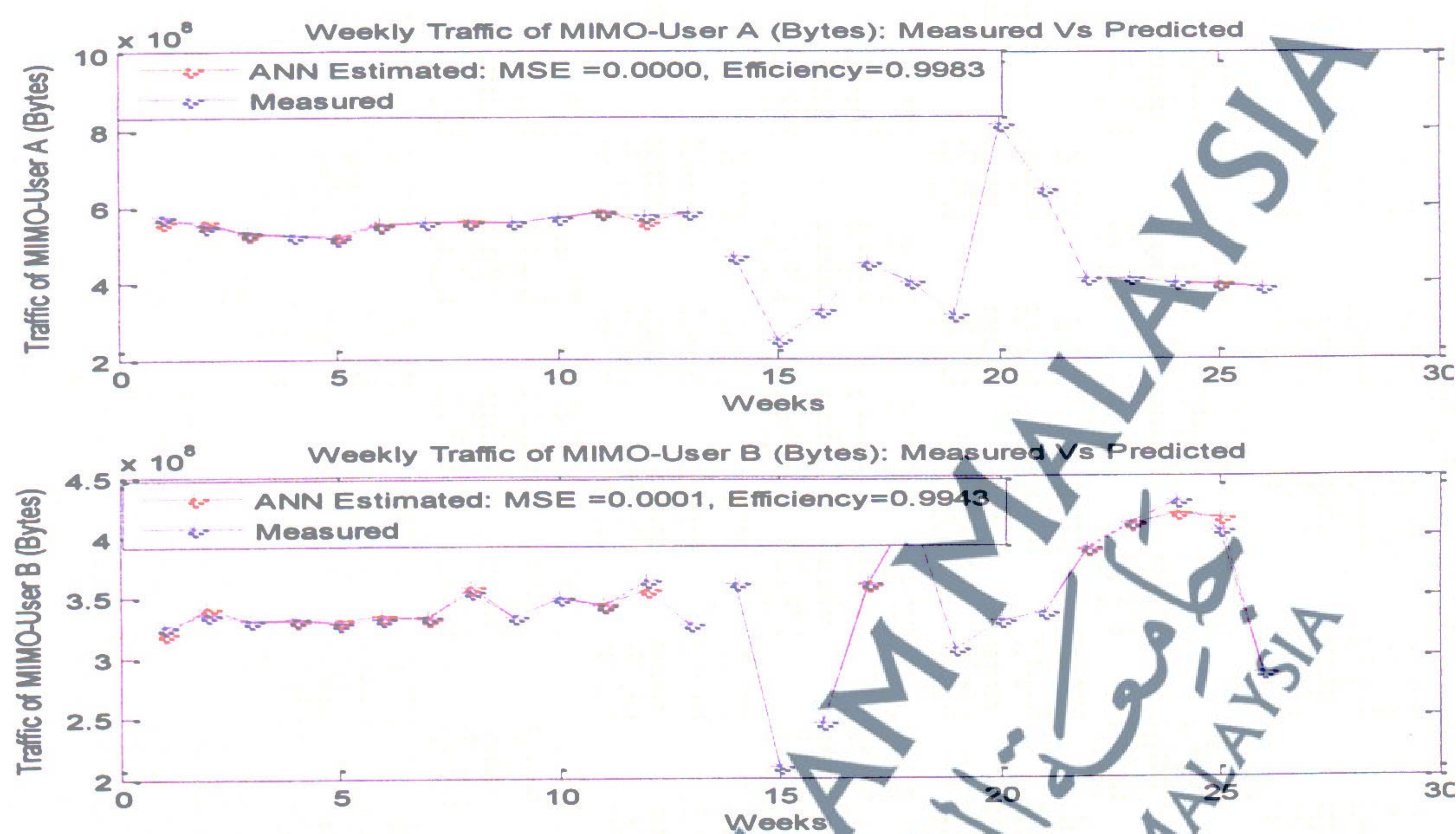
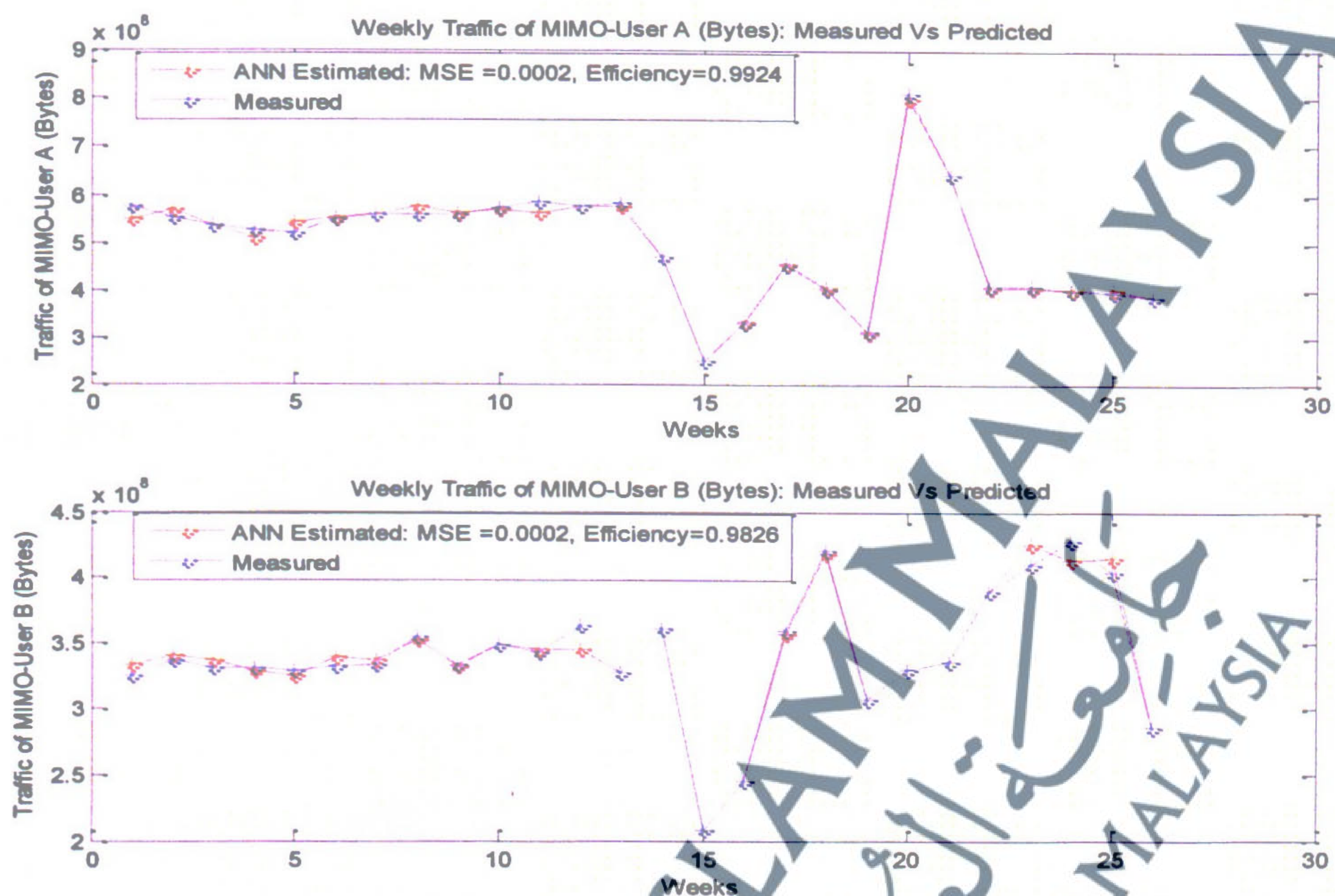


Figure 83: ANN Weekly Training Result of User AB – TrainSCG



Figure 84: ANN Weekly Traffic Result of User AB – TrainSCG



With regard to the monthly traffic forecasting of user A (Table 18, Figure 85), the performance of both algorithms, TrainLM (Figure 86, Figure 87), and TrainSCG (Figure 88, Figure 89), in terms of error are equally severe. Each exhibits an error of 0.0010. However, the efficiency of TrainLM is higher than TrainSCG. They are signified at values 0.9464 and 0.9444 respectively. This indicates that in certain situation, they can perform almost interchangeably.

Table 18: ANN Cross Validation of Monthly Traffic for User A

[20 10 1]	$T_{Monthly(A)} = f_A(X_{Max(A)}, X_{Min(A)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.0004	0.9464	0.97271	0.0010	0.9464
<i>TrainSCG</i>	0.0010	0.94465	0.97166	0.0010	0.9444

Figure 85: ANN Monthly Training of User A – TrainLM

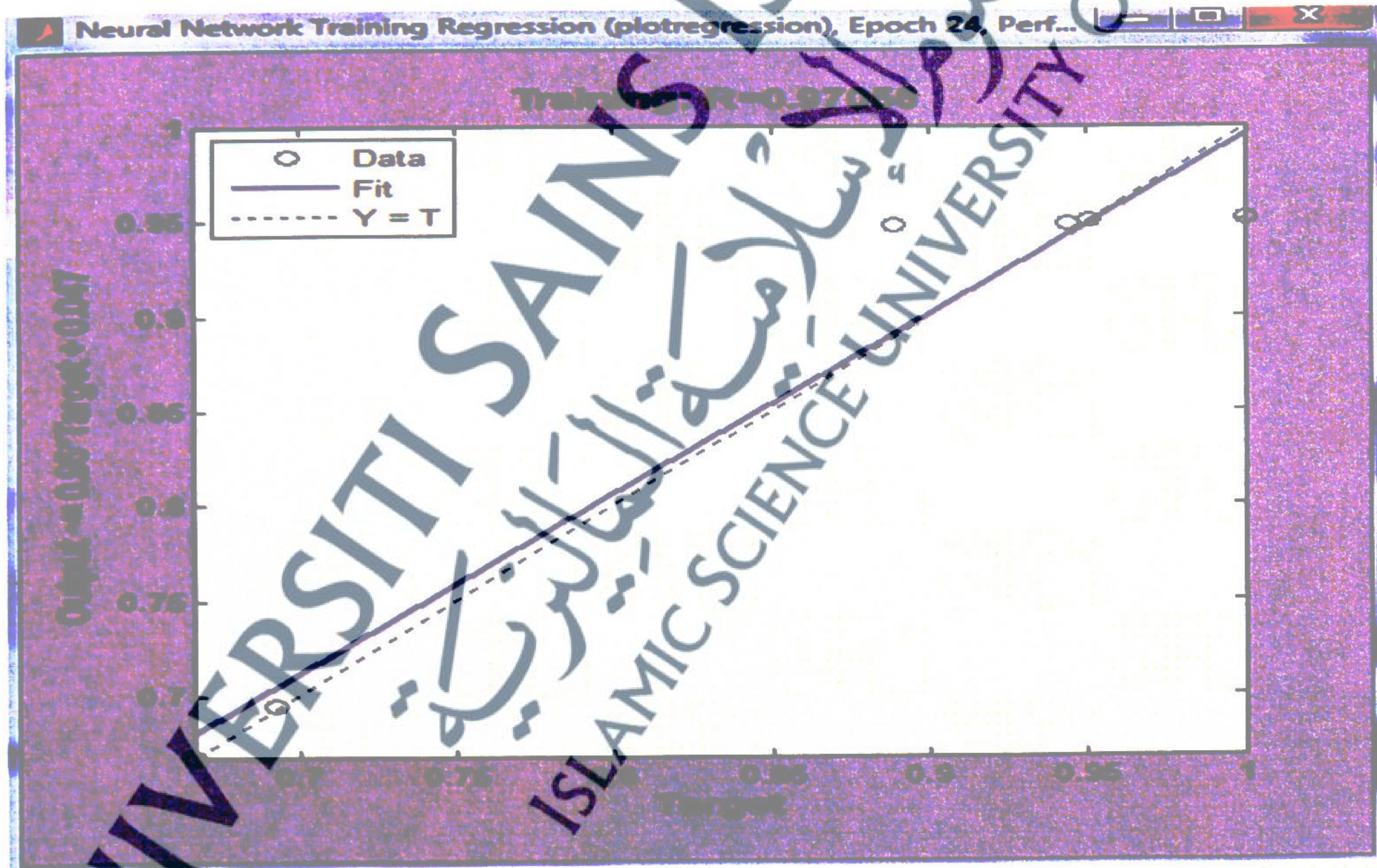


Figure 86: Monthly traffic of MIMO User A : Measured VS Prediction

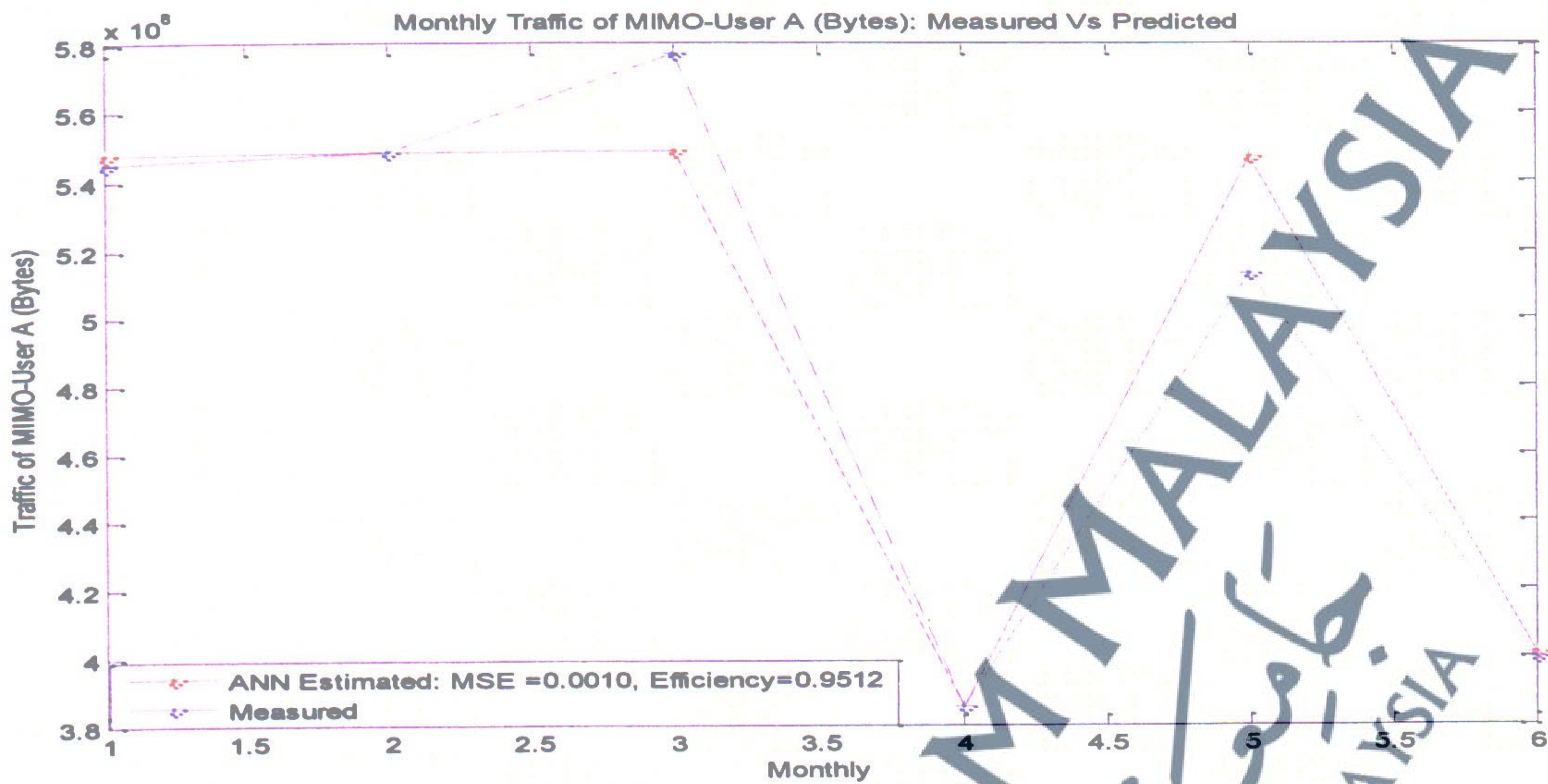


Figure 87: ANN Monthly Testing of User A – TrainLM

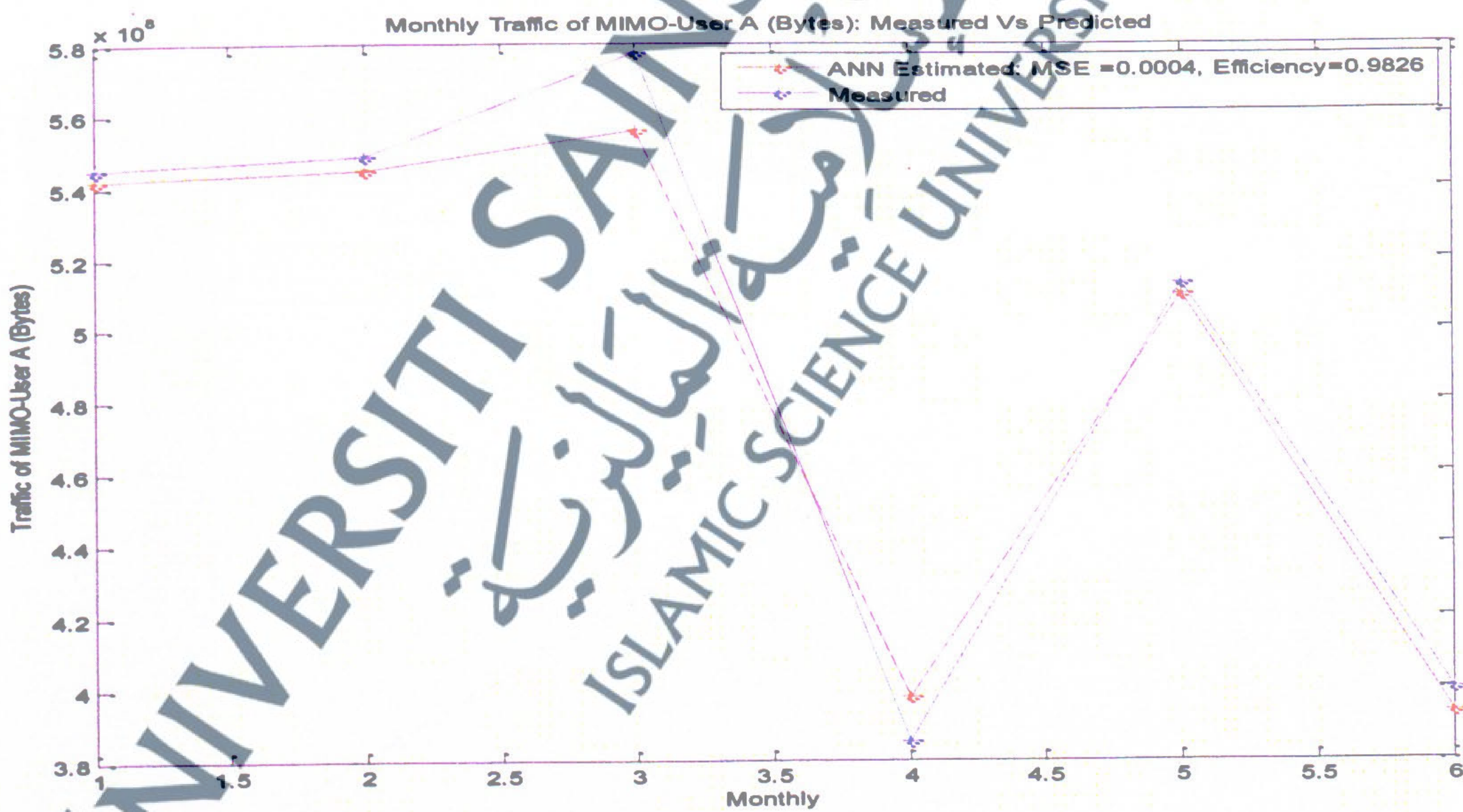


Figure 88: ANN Monthly Training of User A – TrainSCG

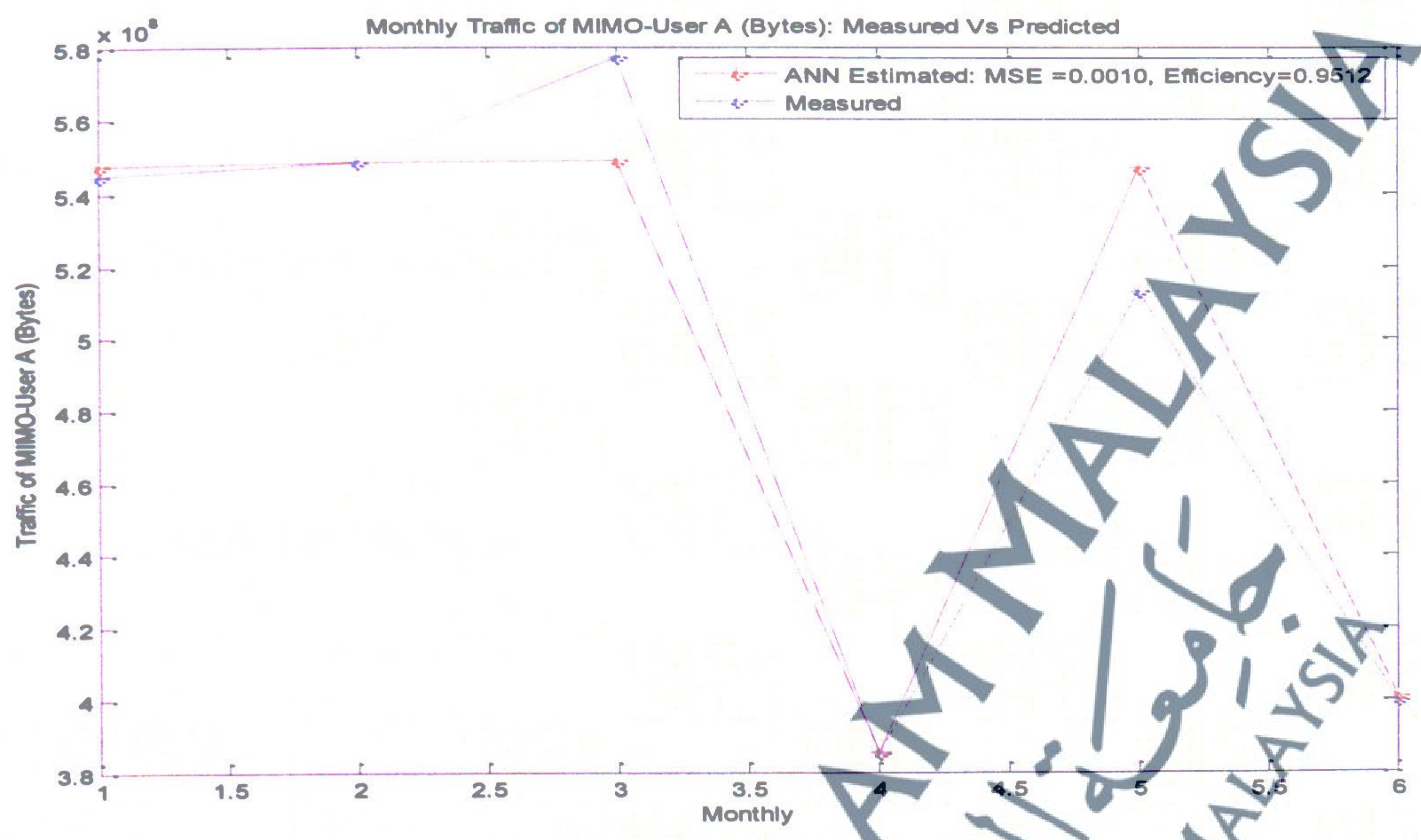
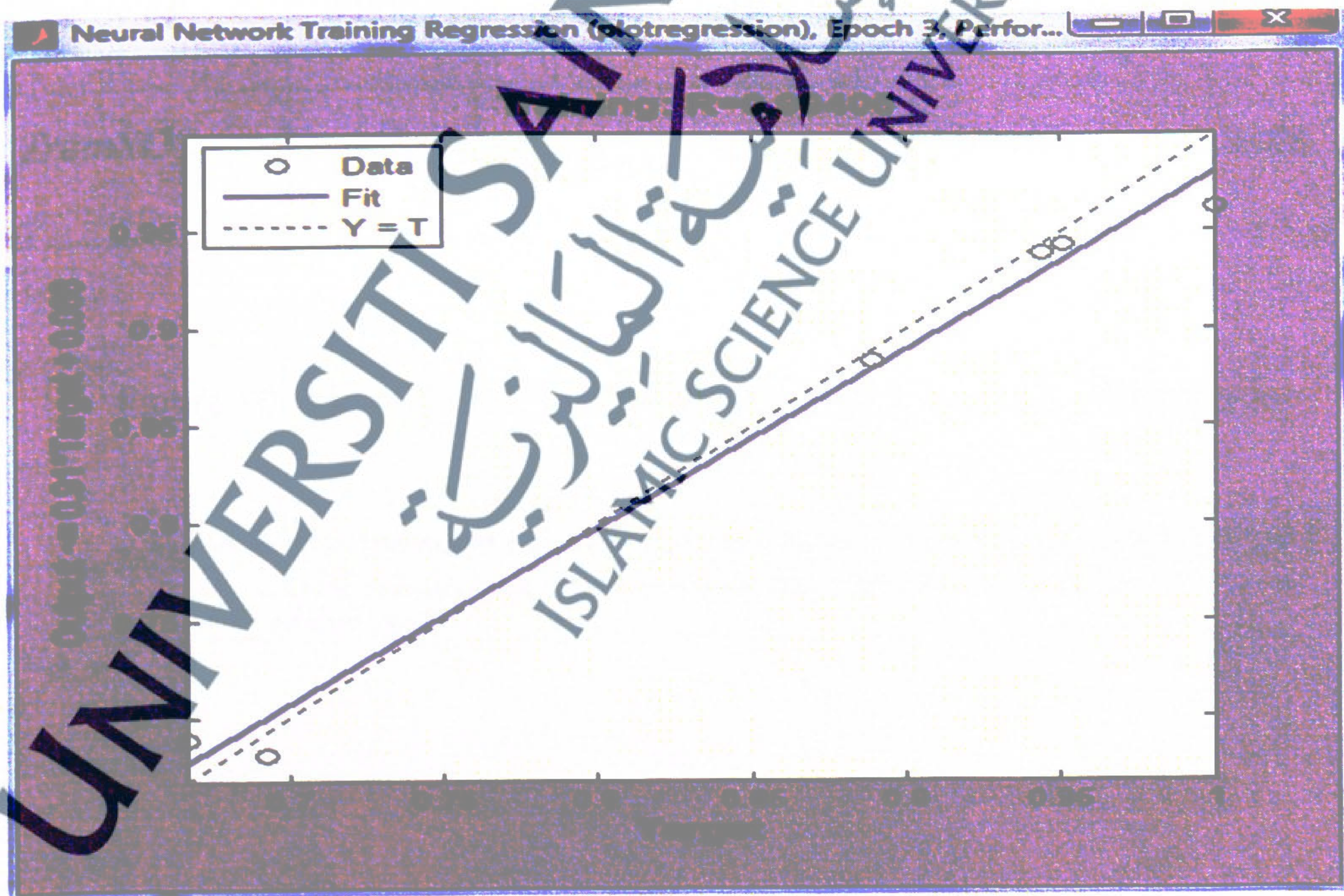


Figure 89: ANN Monthly Training of User A-Training SCG



TrainLM (Figure 90, Figure 91) displays a rather remarkable performance for the monthly traffic of user B (Table 19) with a near optimal error rate of 0.0001 and efficiency of 0.9891 that outperforms both measures of TrainSCG (Figure 92, Figure 93). On the other hand, the error rate of TrainSCG is comparatively poor at 0.0008 with efficiency 0.8348 which falls far behind the ones demonstrated by its counterpart.

Table 19: ANN Cross Validation of Monthly Traffic for User B

[20 10 1]	$T_{Monthly(B)} = f_B(X_{user(Max)}, X_{user(Min)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.0004	0.9464	0.97251 _u	0.0001	0.9891
<i>TrainSCG</i>	0.0010	0.94465	0.97166	0.0008	0.8348

Figure 90: Monthly traffic of MIMO User B: Measured VS Prediction

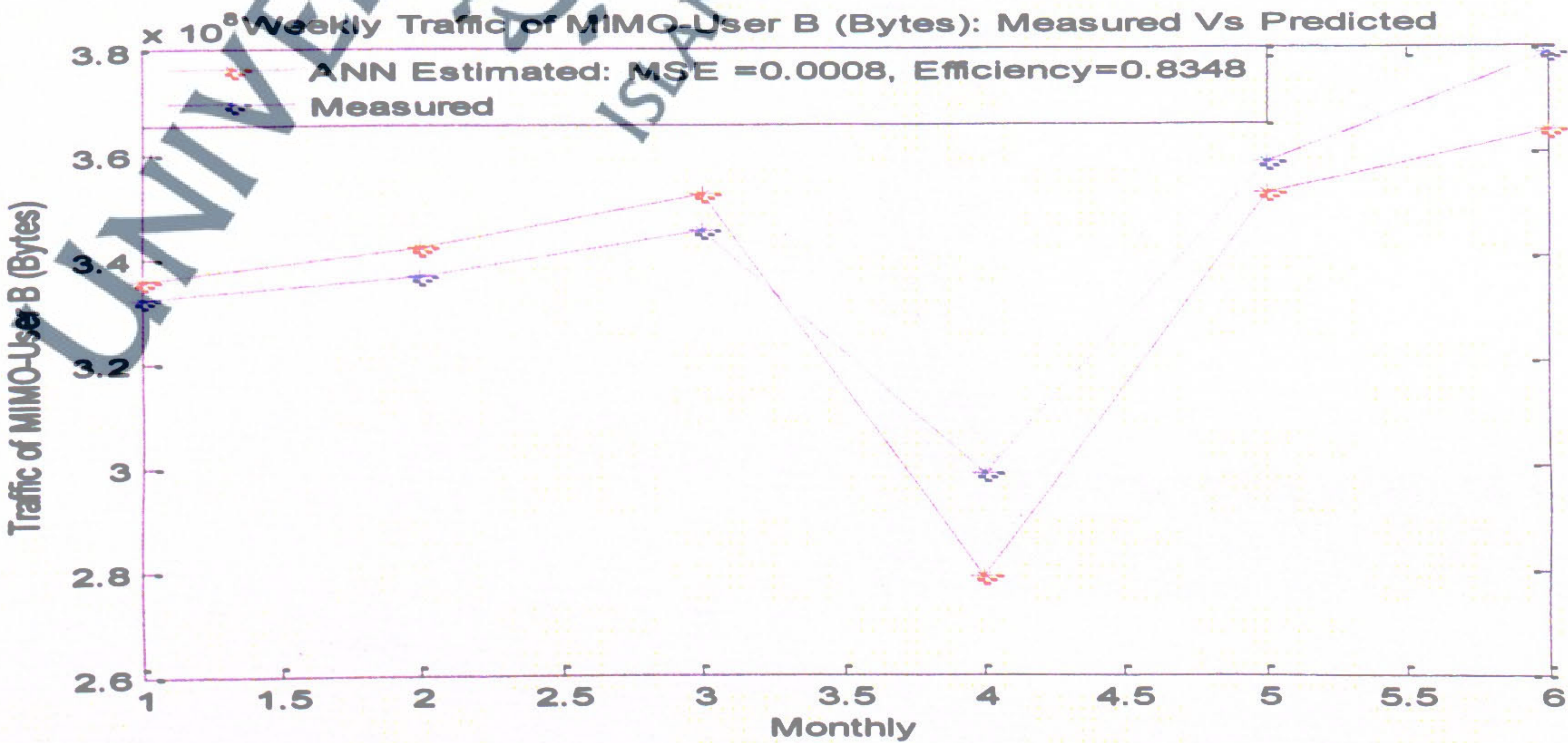


Figure 91: ANN Monthly Training of User B – TrainLM

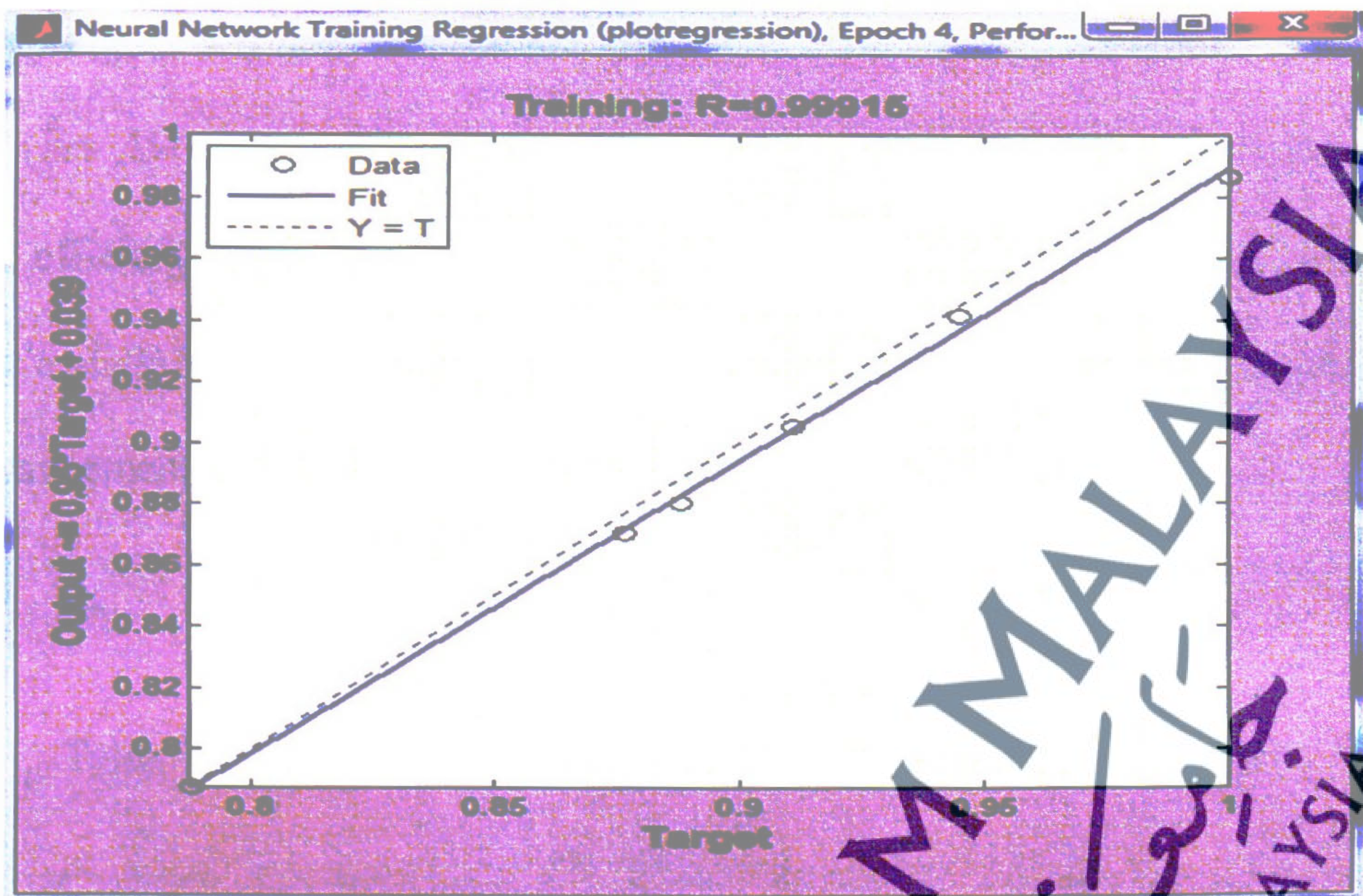
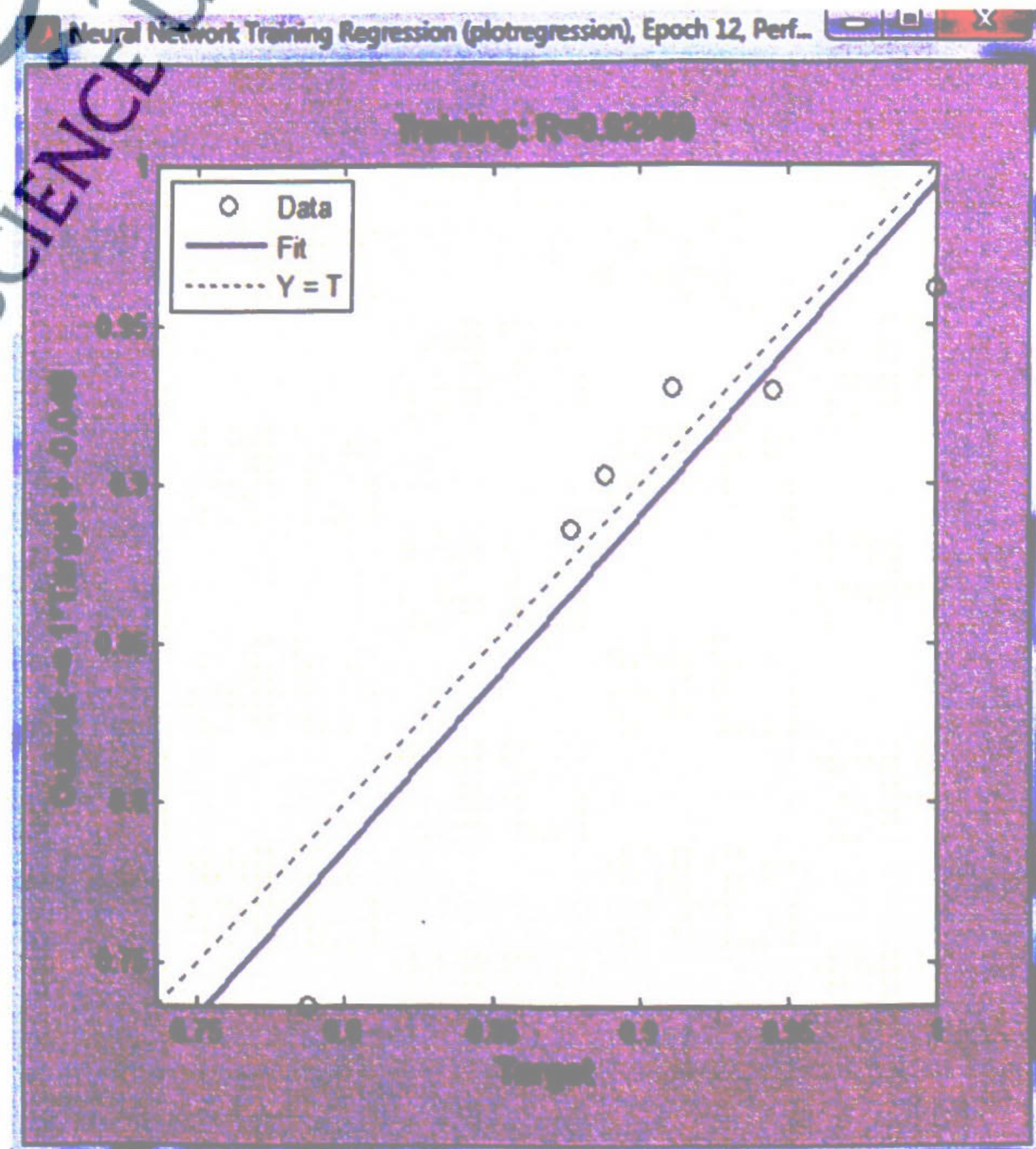
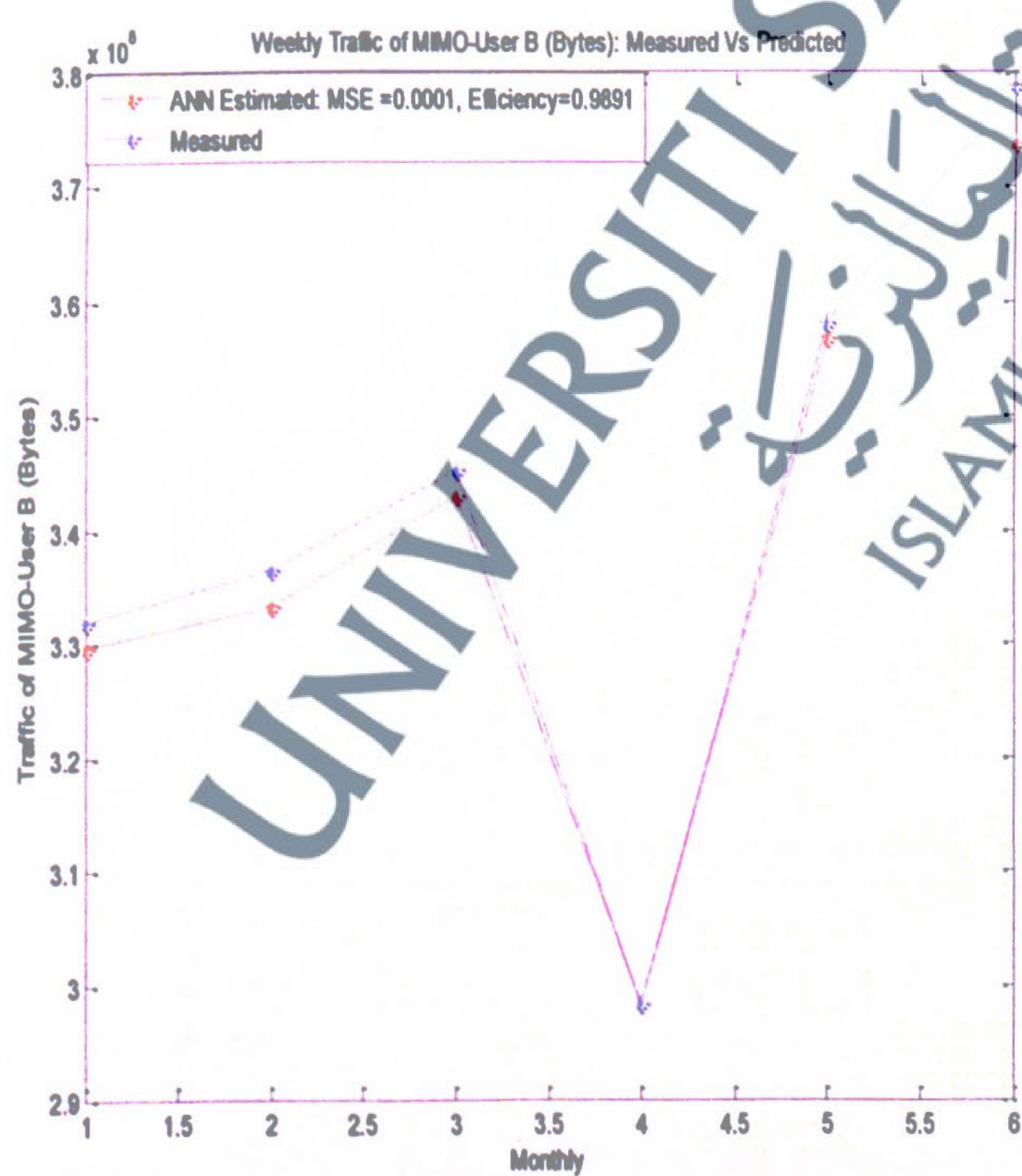


Figure 92: ANN Monthly Testing of User B – TrainLM Figure 93: ANN Monthly Training of User B – TrainSCG



Given its predilection to exhibit superior performance in forecasting (Table 20), TrainLM (Figure 94, Figure 95) does not deviate from this established behavior for the monthly traffic of user AB (Figure 96). It shows the lowest negligible error at 0.0000 and near maximum efficiency at 0.9983. TrainSCG (Figure 97, Figure 98) attempts to compete by projecting a slightly inferior error at 0.0003 and efficiency at 0.9842. This is quite a feat as the advancement of TrainLM is rather difficult to surpass.

Table 20: ANN Cross Validation of Monthly Traffic for User AB

[20 10 2]	$T_{Daily(AB)} = f_{AB}(X_{user(Max)}, X_{user(Min)})$				
	Training			Testing	
Algorithms	MSE	M _{eff}	R	MSE	M _{eff}
<i>TrainLM</i>	0.0009982	0.9998	0.96057	0.0000	0.9983
<i>TrainSCG</i>	0.001538	0.9638	0.91877	0.0003	0.9842

Figure 94: Monthly traffic of MIMO User AB: Measured VS Prediction

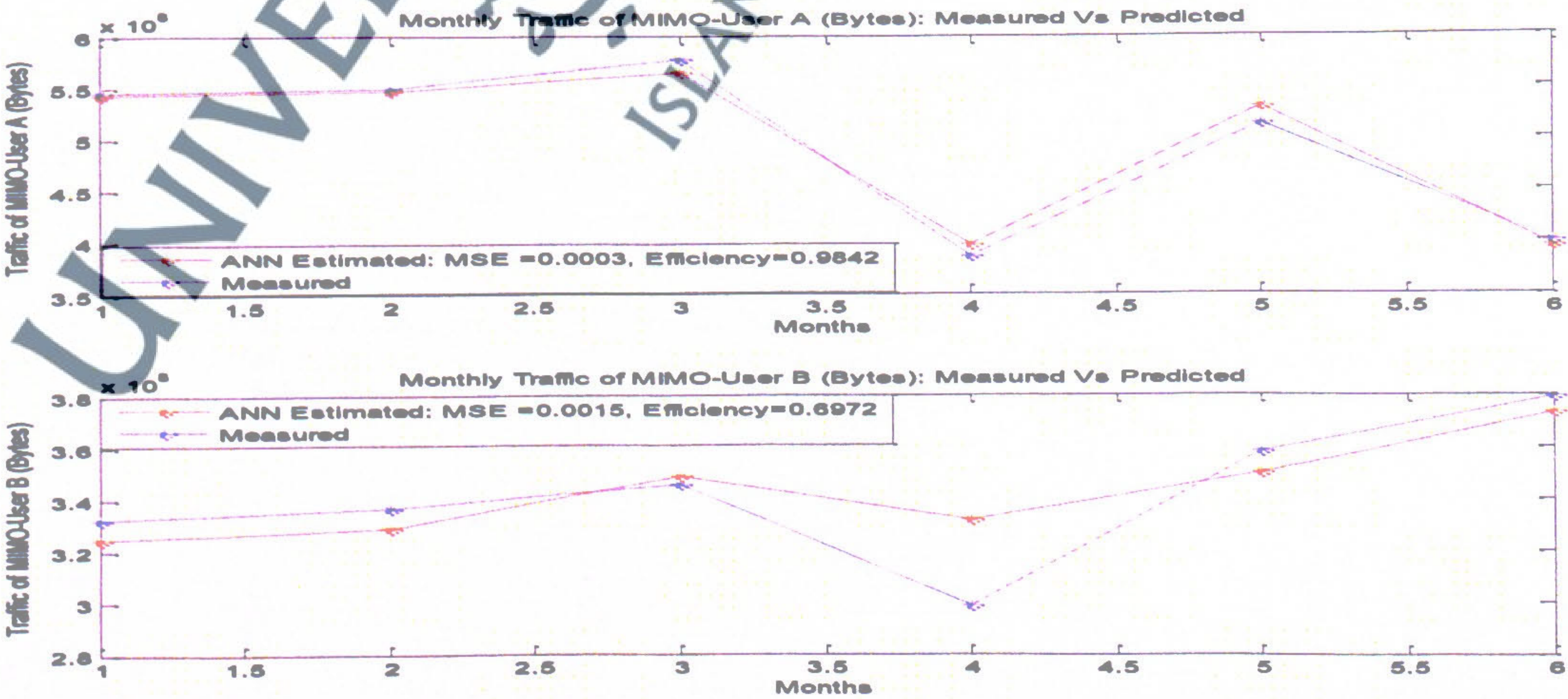


Figure 95: ANN Monthly Training of User AB – TrainLM

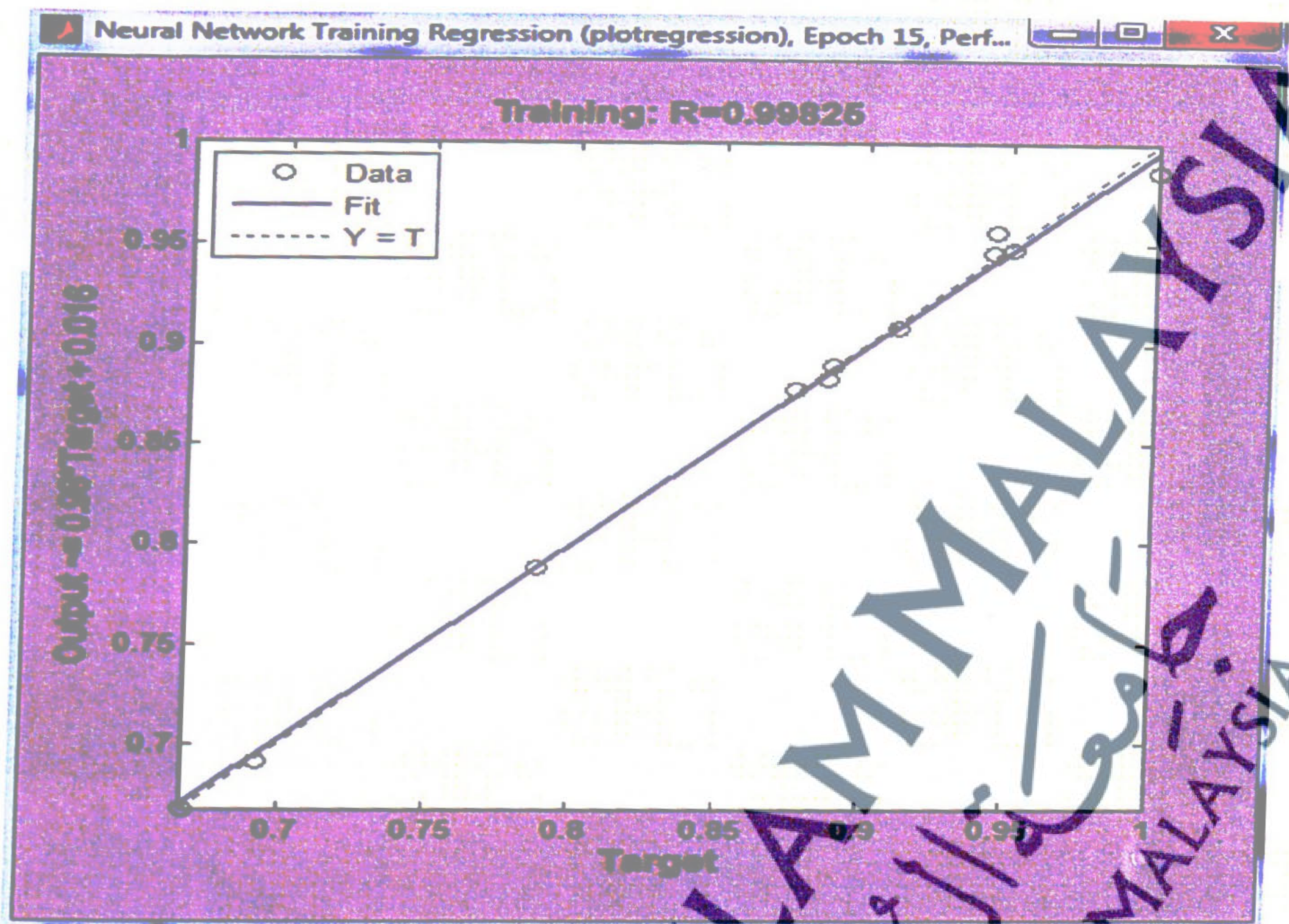


Figure 96: ANN Monthly Testing of User AB – TrainLM

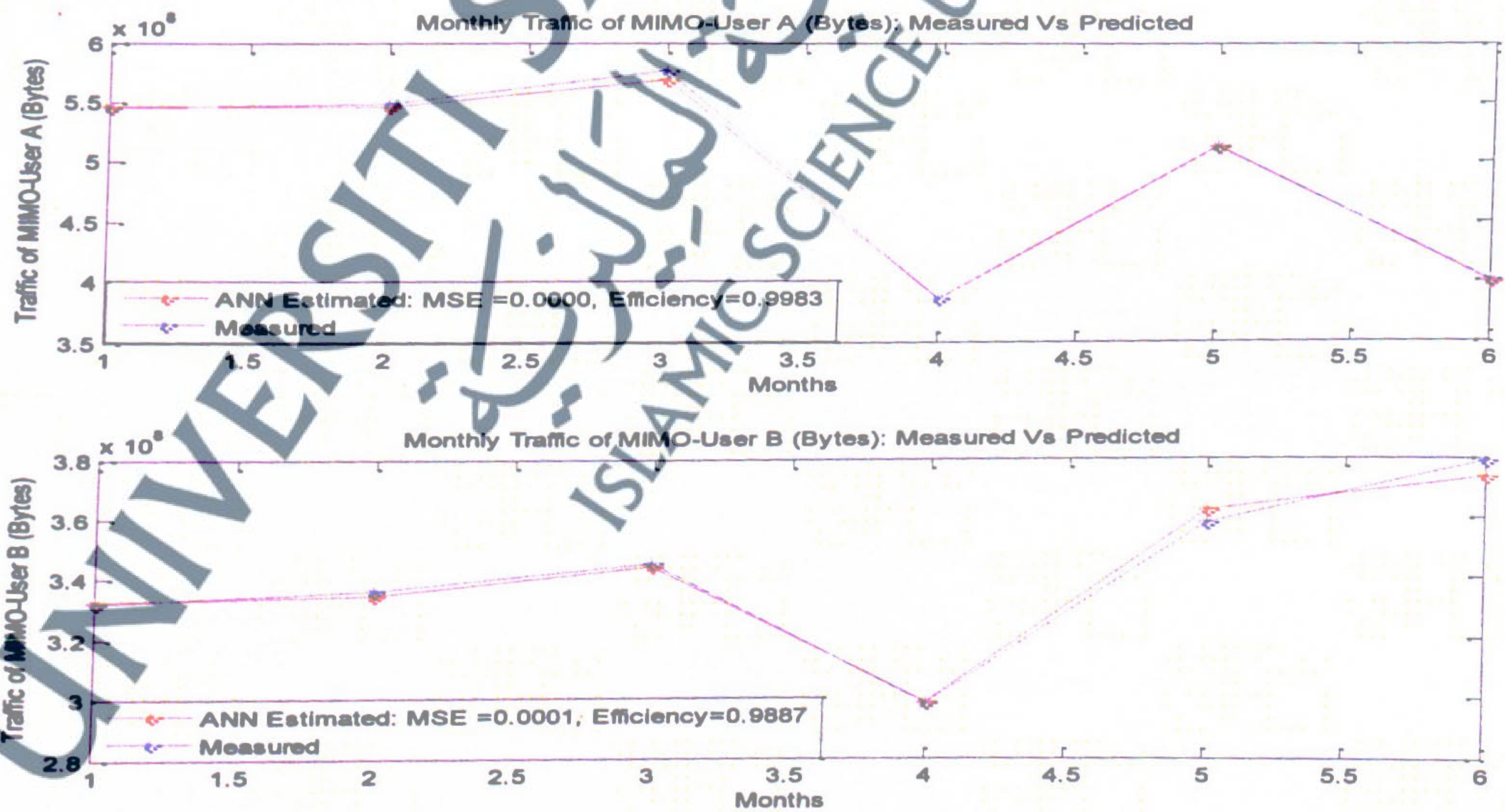


Figure 97: ANN Monthly Training of User AB – TrainSCG

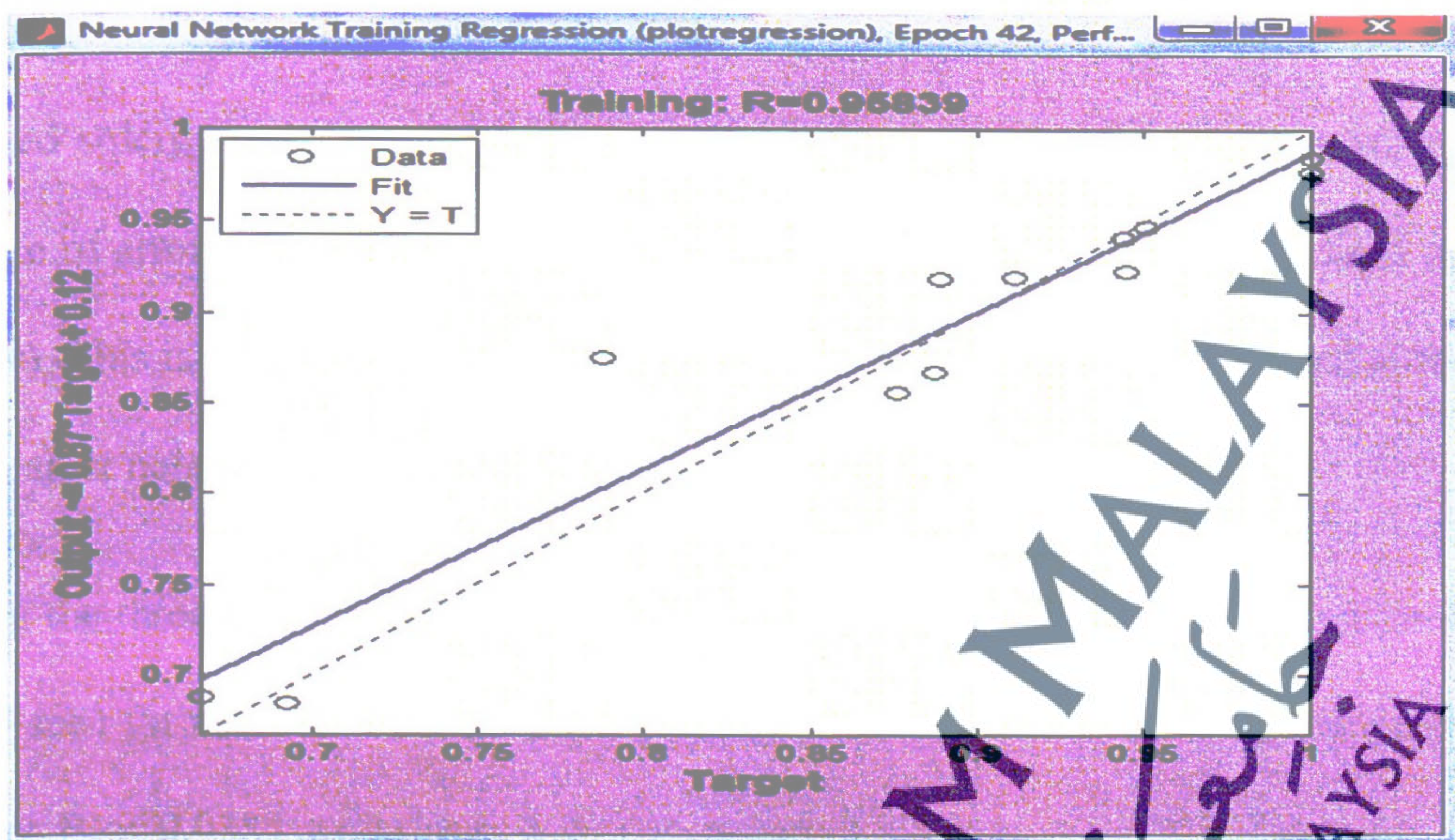
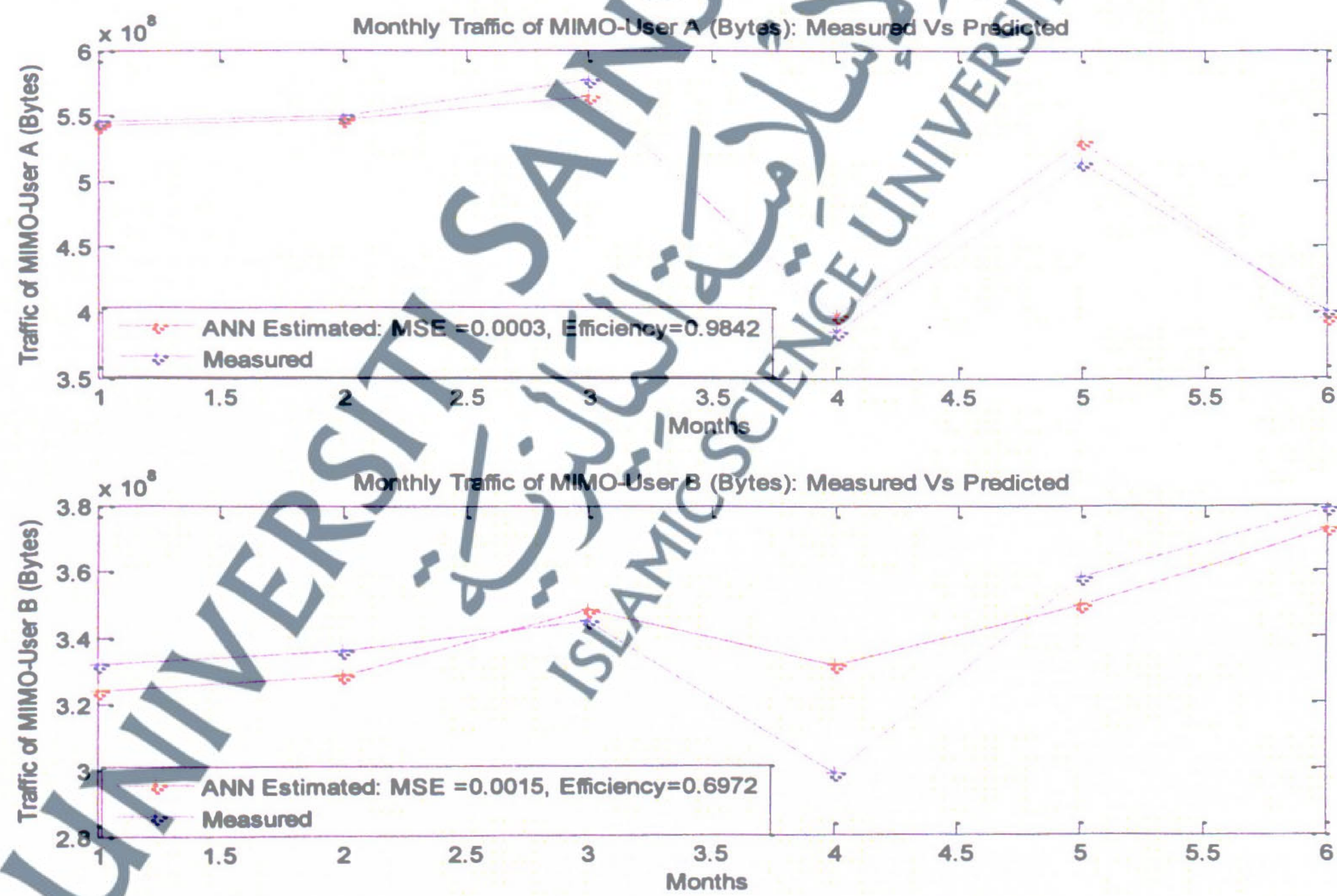


Figure 98: ANN Monthly Testing of User AB – TrainSCG



5.4 Data Input and Output for FTS (Daily, Weekly and Monthly)

Unlike previous intelligent approaches, the performance of fuzzy time series (FTS) is observed solely on the measure of mean square error (MSE). Emphasis is given on the measure of error as compared to the predicted value, which falls under the label of (MSE fitted v). This is equal to the measure of mean square error (MSE) as reported in the result of k-nearest neighbor (KNN) and artificial neural network (ANN).

Among the three cases of users (Table 21), the best daily forecasting for fuzzy time series can be seen for user AB with the least error calculated at 0.6575. The highest error occurs for user B at 0.9384 while user A lies in between the two at 0.8423. Thus, the average error for all the three cases is 0.8128. The graphs are given for user A (Figure 99), user B (Figure 100), and user AB (Figure 101).

Table 21: Exponential FTS Daily

	USER A	USER B	USER AB
MSE all	0.5070	0.4691	0.2870
MSE-in	0.6220	0.4834	0.5079
MSE-o	0.6958	0.6440	0.3950
l. intr	5760000.0 1.9322222E7	4190000.0 1.5641667E7	3720000.0 1.4936111E7
MSE fitted v	0.8423	0.9384	0.6575

Figure 99: FTS Daily Testing of User A

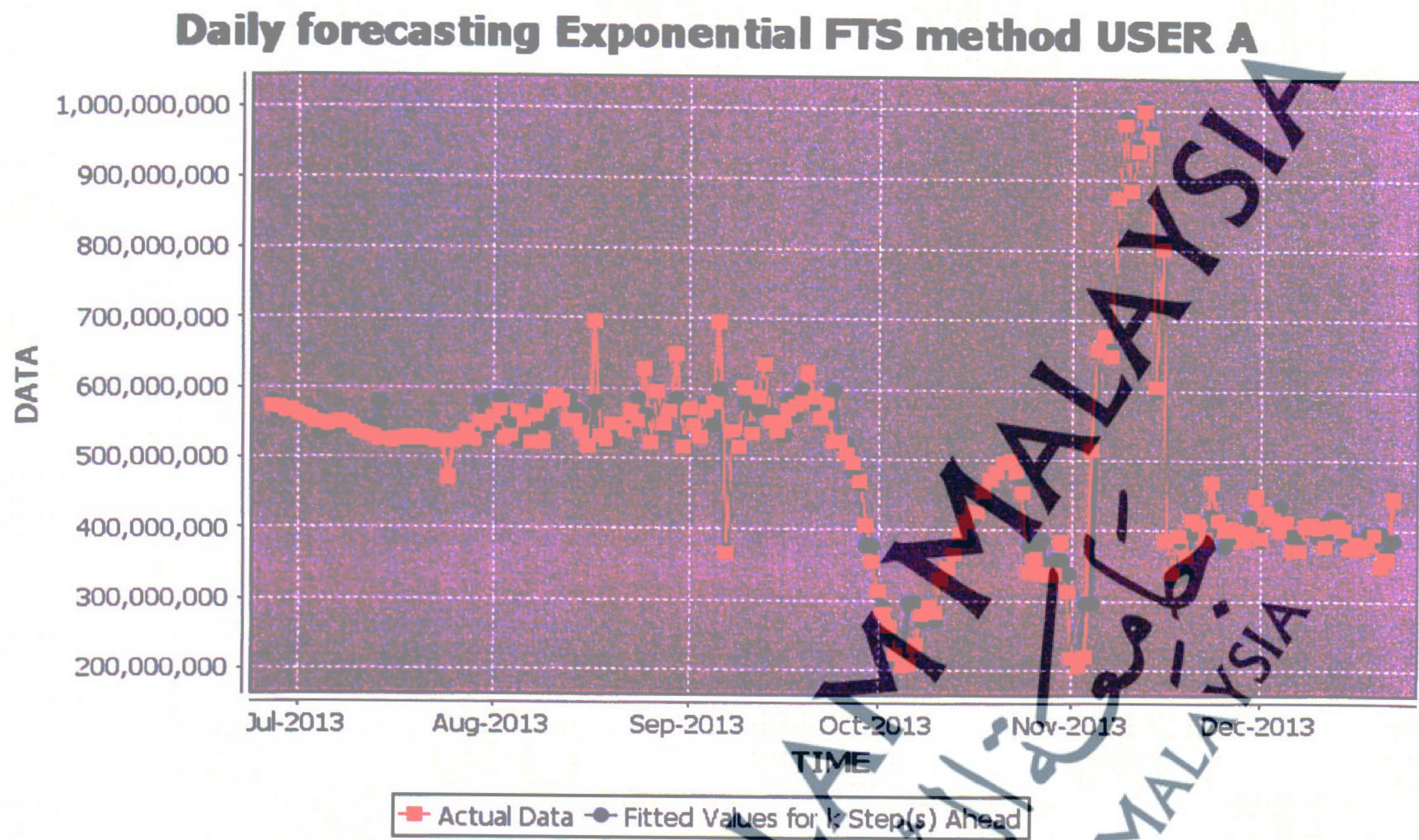


Figure 100: FTS Daily Testing of User B

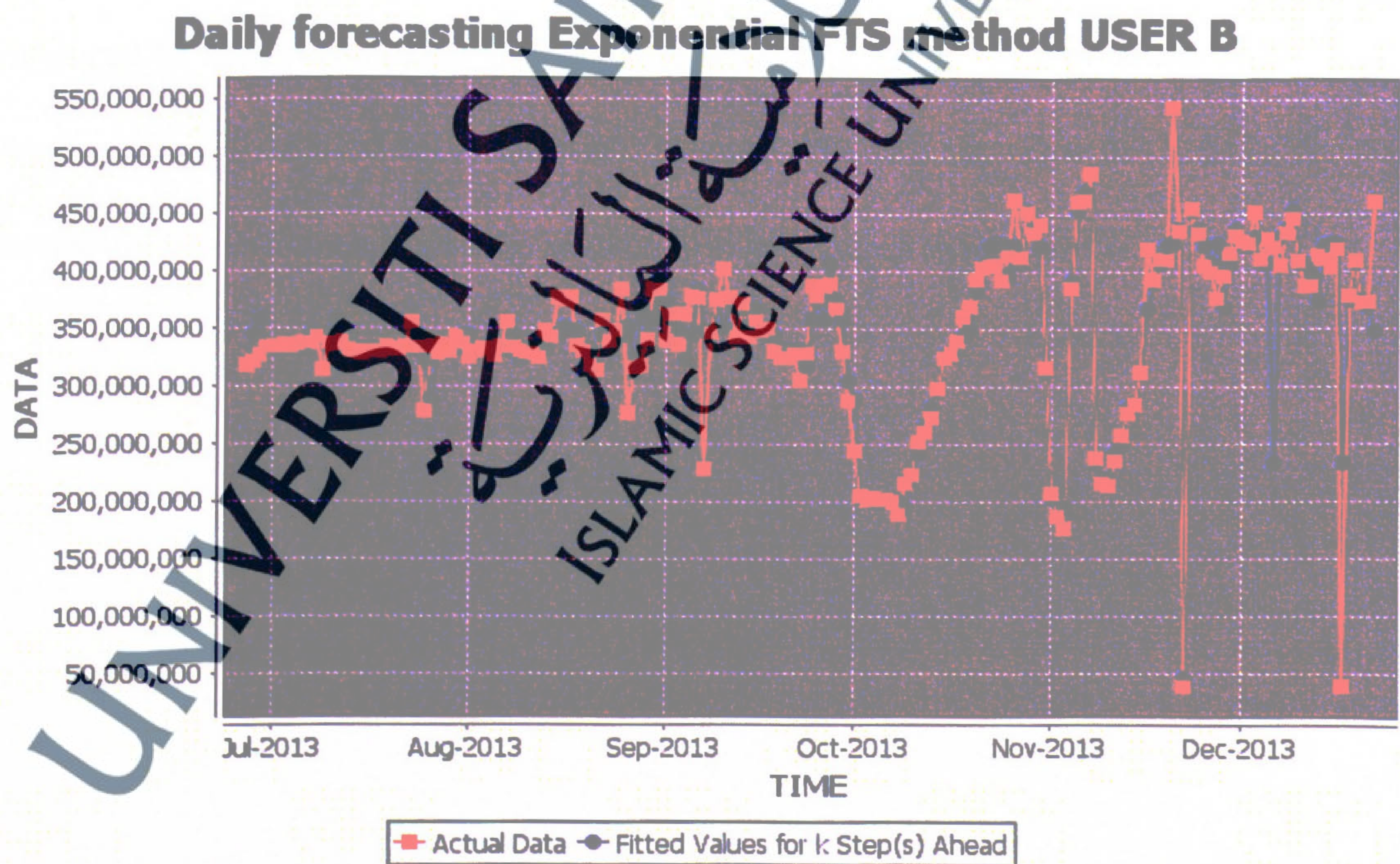
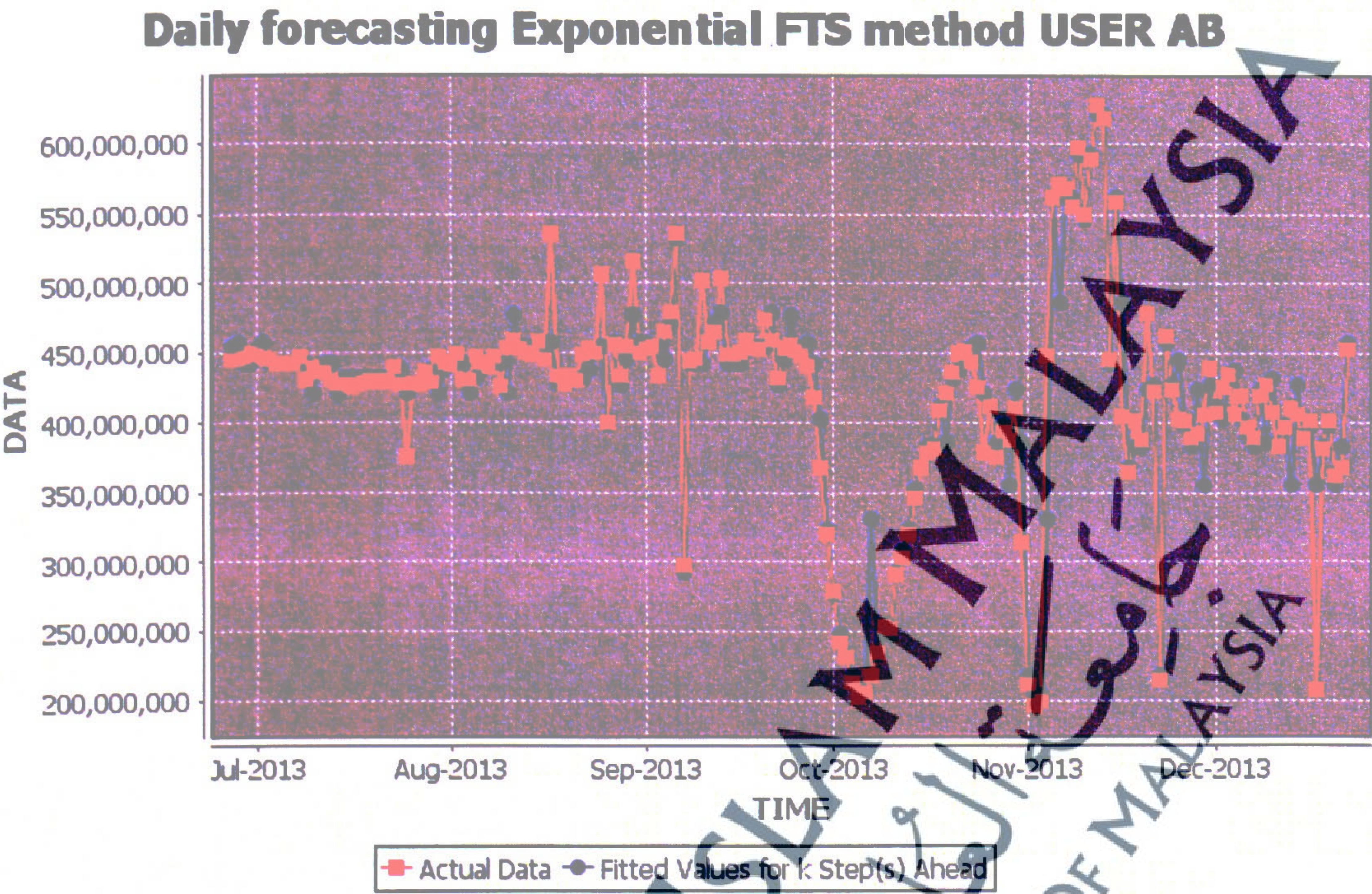


Figure 101: FTS Daily Testing of User AB



The entire result for the weekly forecasting of fuzzy time series is given below (Table 22) and depicted for user A (Figure 102), user B (Figure 103), and user AB (Figure 104). Compared to the daily result, the most optimal forecasting is exhibited by user B at the lowest error of 0.1650. This is followed by user AB with the error of 0.1743. The worst forecasting performance is shown by user A with error as severe as 0.3242.

Table 22: Exponential FTS Weekly

	USER A	USER B	USER AB
MSE all	0.8117	0.2924	0.1995
MSE-in	0.3125	0.1400	0.3900
MSE-o	0.9741	0.3509	0.2394
l. intr	5500000.0	1200000.0	2400000.0
	2.31E7	1.406E7	1.398E7
MSE fitted v	0.3242	0.1650	0.1743

Figure 102: FTS Weekly Testing of User A

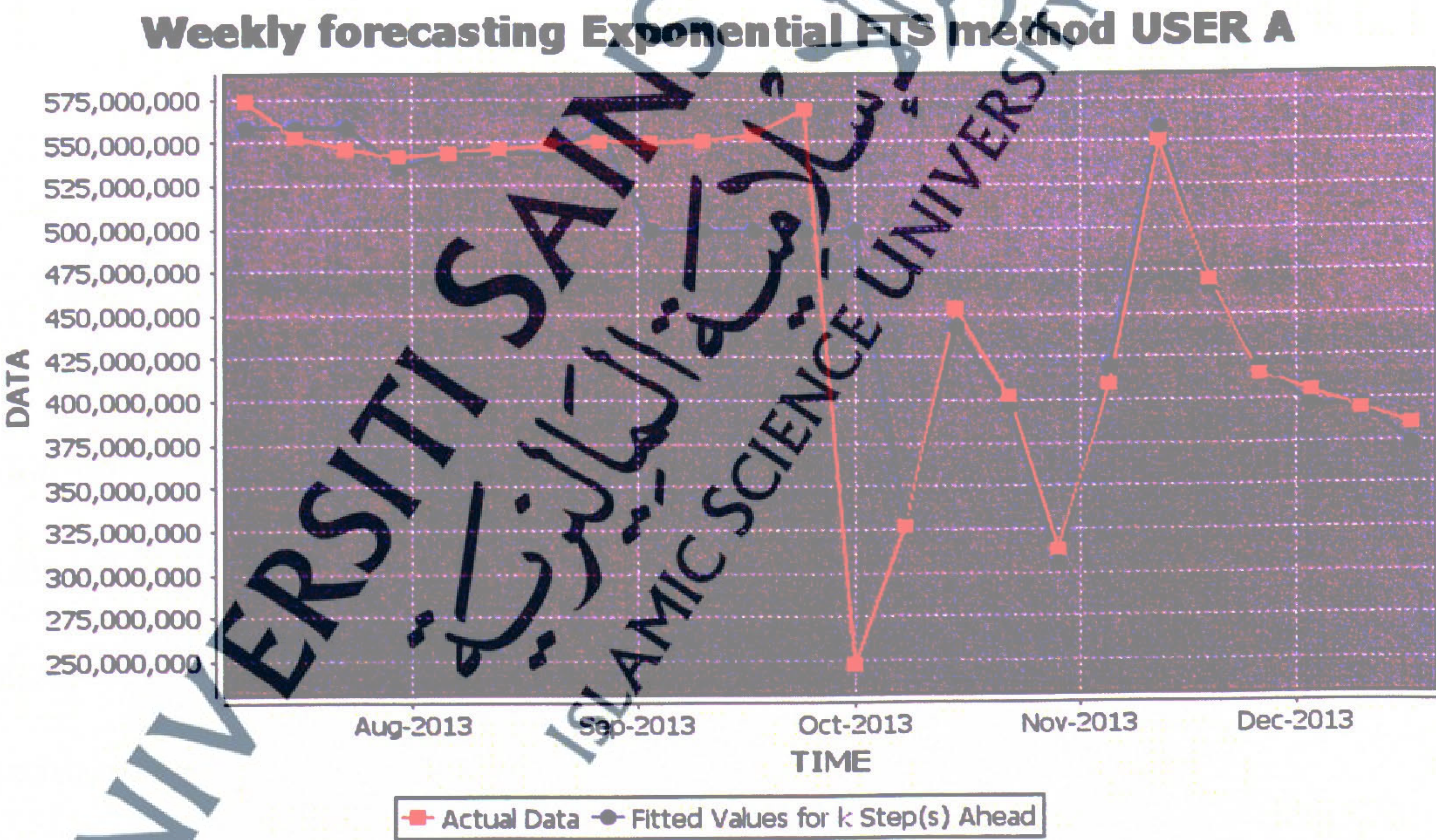


Figure 103: FTS Weekly Testing of User B

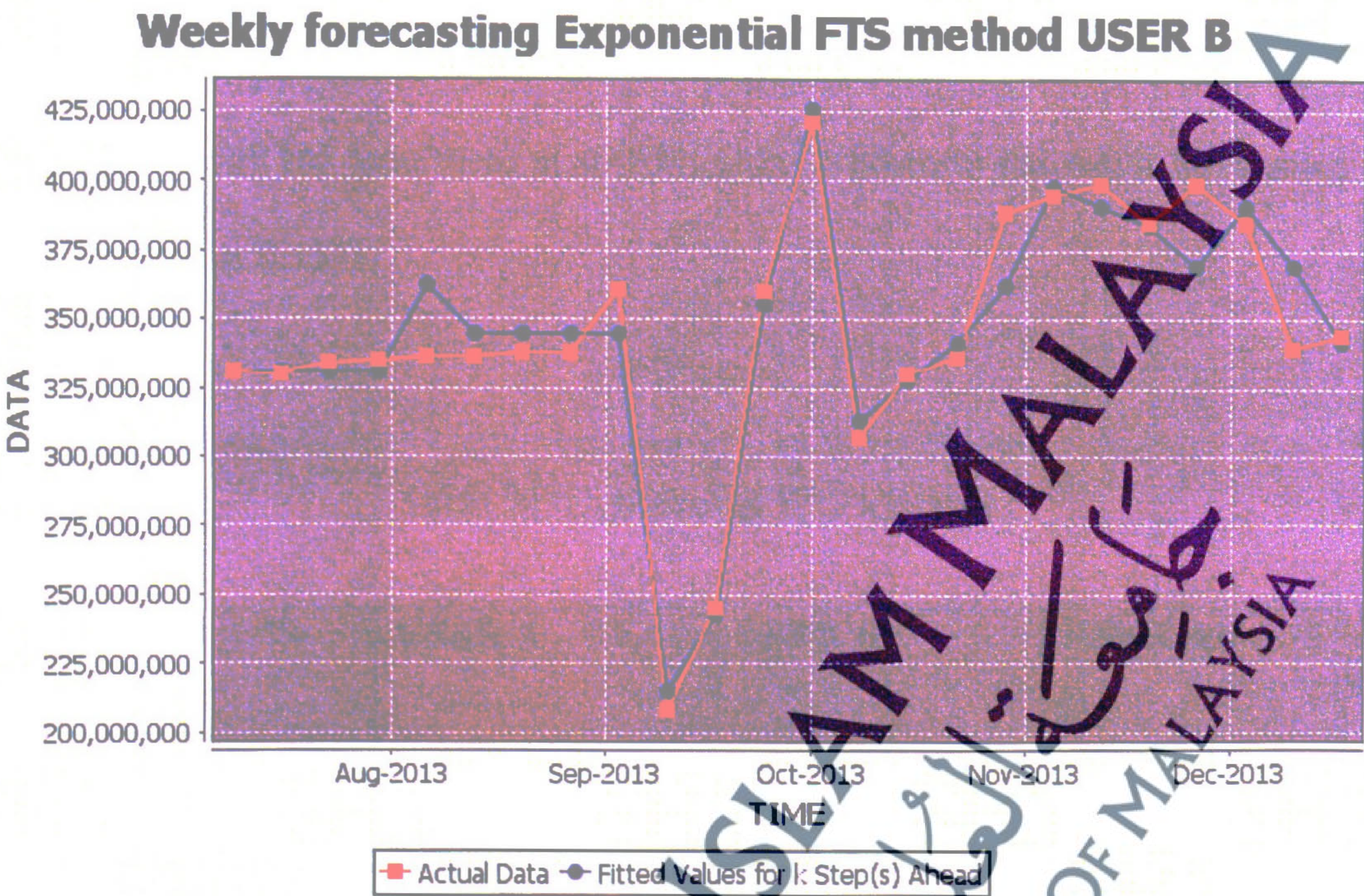
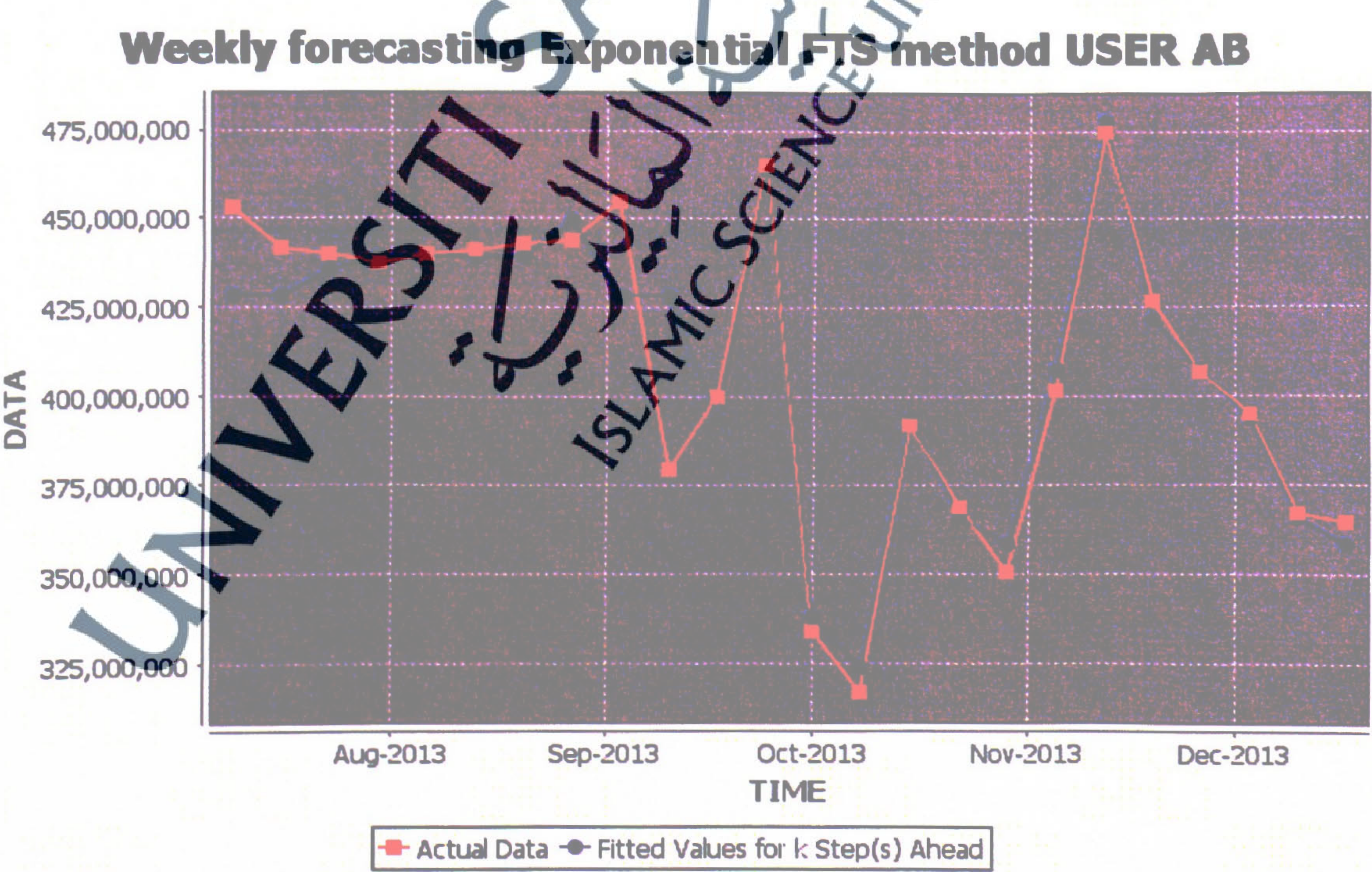


Figure 104: FTS Weekly Testing of User AB



The overall result of the monthly forecasting for fuzzy time series is tabulated below (Table 23) with visualization of user A (Figure 105), user B (Figure 106), and user AB (Figure 107). In this case, user AB shows superior performance in comparison with user A and user B with the least error at 0.1372. User B portrays the worst forecasting with error computed at 0.3531.

Table 23: Exponential FTS Monthly

	USER A	USER B	USER AB
MSE all	0.8123	0.2368	0.3226
MSE-in	0.2500	0.4000	0.6250
MSE-o	0.1083	0.3157	0.4301
l. intr	3000000.0	2000000.0	2500000.0
	4.23E7	1.55E7	2.24E7
MSE fitted v	0.2700	0.3531	0.1372

Figure 105: FTS Monthly Testing of User A

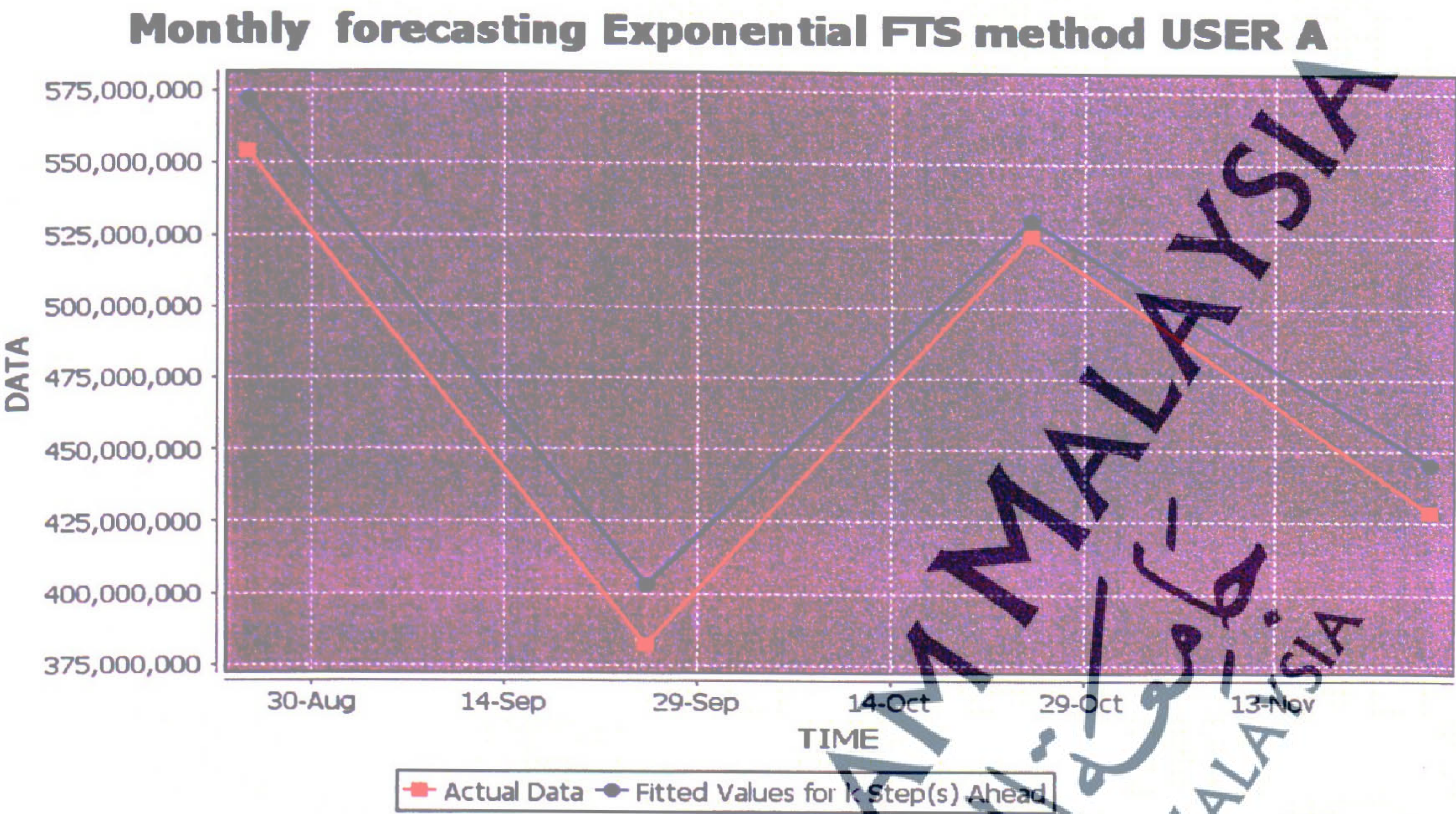


Figure 106: FTS Monthly Testing of User B

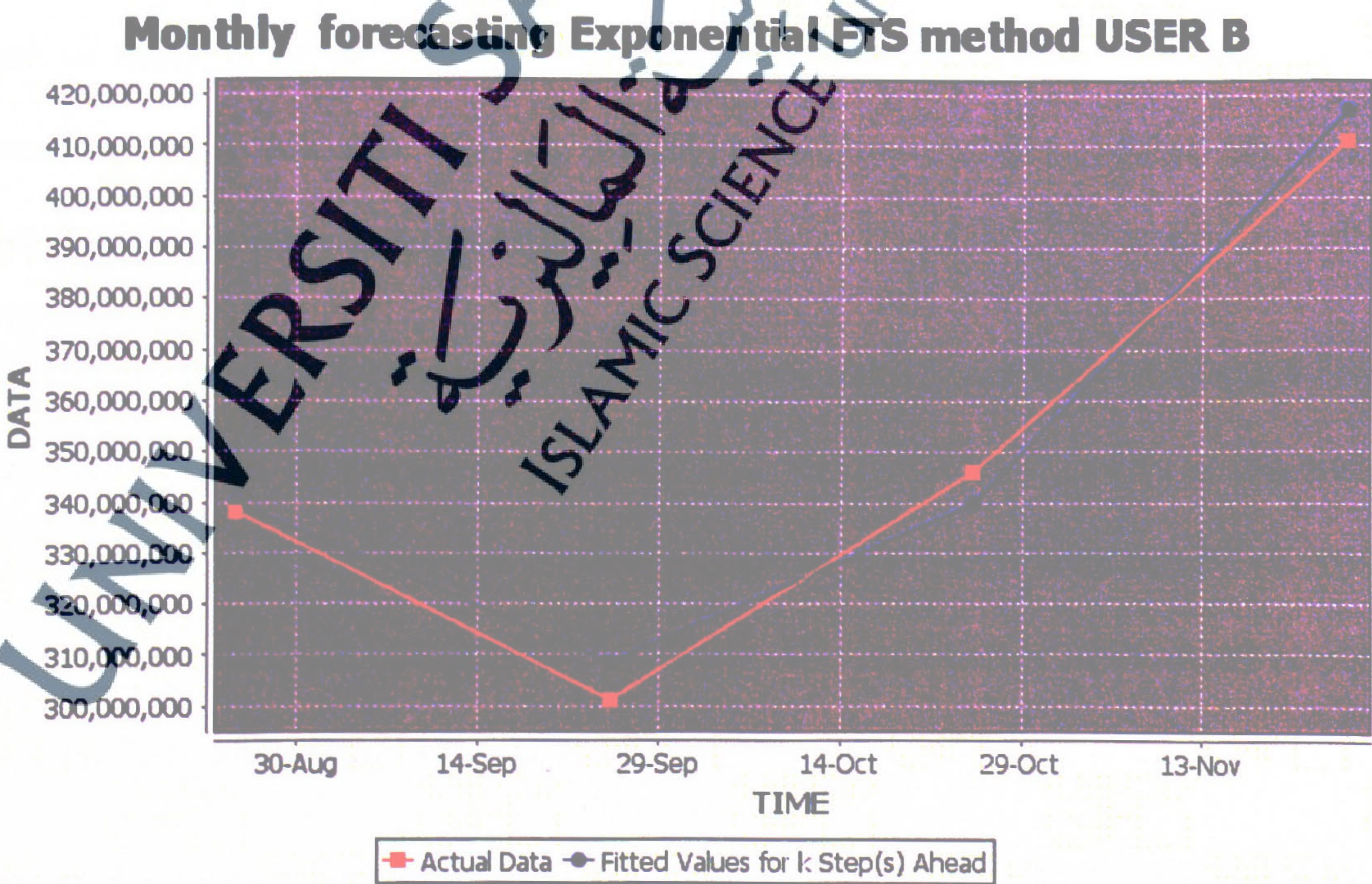
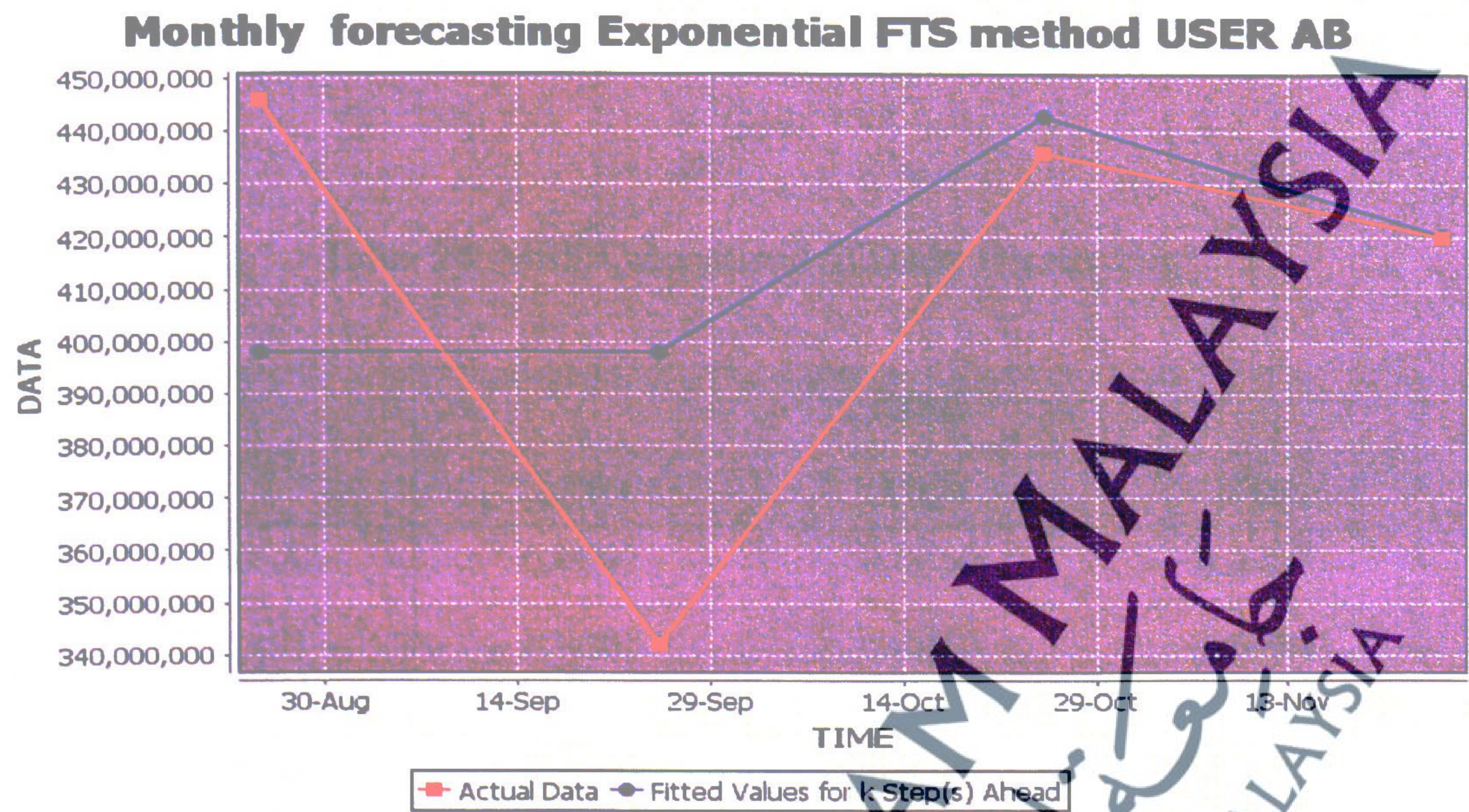


Figure 107: FTS Monthly Testing of User AB



5.5 Discussion

To perform the comparative analysis on the forecasting performance of k-nearest neighbor (KNN), artificial neural network (ANN) and fuzzy time series (FTS), only the extent of error in terms of mean square error (MSE) is examined. Following the convention of this research, they are presented based on the daily (Table 24), weekly (Table 25), and monthly duration (Table 26).

Obviously for the daily forecasting (Table 24), KNN outperforms both ANN and FTS for all the cases of user A, user B, and user AB. The error of ANN is nearly as diminished when compared to KNN. The former differs from the latter at best by 0.0003 (0.0010 – 0.0007) and at worst by 0.0010 (0.0014 – 0.0004). This implies that ANN works nearly as well as KNN on all the cases of daily forecasting.

However, the same cannot be said for the performance of FTS which displays the most serious severity when contrasted against KNN. The error exhibited by FTS is at best 10^3

times worst than KNN. It is not certain as to why the capacity of FTS to perform the daily forecasting is this inferior.

Table 24: MSE Comparison of Daily Forecasting

Users	MSE		
	KNN	ANN	FTS
User A	0.0007	0.0010	0.8423
User B	0.0005	0.0010	0.9384
User AB	0.0004	0.0014	0.6575

The weekly forecasting however (Table 25), reports a mixed impression for the performance of KNN as measured against its counterparts. Although KNN shows the best result for user A (0.0004) and user B (0.0001), it fails to do so for user AB (0.0004). Here, KNN is inferior to the error demonstrated by ANN, which seems to be optimal at the negligible rate of 0.0000.

For the sake of argument, it must be reminded that the error of 0.0000 does not imply perfection. In other words, it does not mean that the predicted value is exactly the same as the actual value gathered from the data. Instead, the statistical measure imparts the implicit notion that the error culminated by the prediction is too small to be perceptible by the experimentation.

For the rest of the cases, ANN attempts to follow the performance of KNN albeit at a less impressive rate. Regrettably, FTS repeats its poor performance in all the forecasting cases for the weekly forecasting. This time however, the deficiency in prediction is not as harsh as the one showed earlier in the daily forecasting whereby a smaller difference can be seen when compared with KNN. Observe the difference for user B, which is only 0.1649 (0.1650 – 0.0001).

Table 25: MSE Comparison of Weekly Forecasting

Users	MSE		
	KNN	ANN	FTS
User A	0.0004	0.0007	0.3242
User B	0.0001	0.0010	0.1650
User AB	0.0004	0.0000	0.1743

KNN regains its dominance in the monthly forecasting (Table 26). It narrates an unparalleled display of prediction for all the cases with the lowest error for user A (0.0007), user B (0.0000), and user AB (0.0000). In fact, it provides the most optimal forecasts with the error of 0.000 for two out of three cases in the research. This adamantly suggests the power of KNN in network traffic forecasting.

ANN tends to show a fluctuation of performance between cases. The error for user A (0.0010) is comparatively high when compared to the other cases of user B (0.0001) and user AB (0.0000). This is quite confusing, given the fact that the data of user A and user

B do not differ too dramatically. It is again unfortunate to mention the failure of FTS to perform favorably in forecasting as opposed to KNN and ANN.

Table 26: MSE Comparison of Monthly Forecasting

Users	MSE		
	KNN	ANN	FTS
User A	0.0007	0.0010	0.2700
User B	0.0000	0.0001	0.3531
User AB	0.0000	0.0000	0.1372

With regard to the perplexity that arises with the aforementioned fluctuation, it is a well known disadvantage that any form of intelligent decision making handled by ANN suffers opacity of reasoning (Dumitru & Maria, 2013; Wang et al., 2001; Vu, 1996). In other words, the mechanism employed by ANN to alter its weights in an iterative and simultaneous fashion, is not exactly transferable to the rational ground that can be accepted by humanistic reasoning (Mercier & Sperber, 2011).

Looking at the impressive result of k-nearest neighbor, it is intriguing to see the recurring phenomenon of $k = 7$ as the number of referential points in the algorithm for reaching peak performance. Now, arguing over the optimal value of k in this research can be rather counterproductive. There may not be an explanation, except for the fact that it is proven as such from the result of experimentation (Arlot, 2010).

Among the three approaches, KNN offers the highest accuracy in WiMAX traffic forecasting. This can be seen consistently in most if not all the cases in question. The only drawback in using KNN is the elusiveness of deciding the value of K or the number of nearest neighbors. As such, the unparalleled performance of KNN is achieved only when the value of K is assigned manually. In reality, finding the right value of K is a feat by itself.

The new model of ANN that is proposed in this research performs slightly less than KNN. It must be reminded however, that unlike KNN, the working of ANN is fully autonomous without the need of manual supervision. ANN can adapt to the changing scenarios in WiMAX traffic forecasting by learning the essential patterns within the data itself. In this regard, it is superior to KNN in terms of its versatility to accommodate change.

The worst performance of accuracy is exhibited by FTS, which repeatedly commits serious errors in prediction when compared to the other two approaches. Given the relationship of FTS with ANFIS, it is not entirely premature to assume that fuzzy related approaches in AI may not be entirely suitable in dealing with WiMAX traffic forecasting. A concise overview of the results is given below (Table 27).

Table 27: Overview of Results

		MSE (error)		
		ANN	KNN	FTS
User A	Daily	0.0010	0.0007	0.8423
	Weekly	0.0007	0.0004	0.3242
	Monthly	0.0010	0.0007	0.2700
User B	Daily	0.0010	0.0005	0.9384
	Weekly	0.0010	0.0001	0.1650
	Monthly	0.0001	0.0000	0.3531
User AB	Daily	0.0014	0.0004	0.6575
	Weekly	0.0000	0.0004	0.1743
	Monthly	0.0000	0.0000	0.1372

The most popular approach of dealing with this issue is the one derived from statistics. Here, the best value of k is approximated from the square root of total data (Duda, Hart & Stork, 2001), whereby (N) or more specifically $k = N^{1/2}$. In this research however, the total data is well over a million, suggesting that the value of $k \geq 1000$. Indeed, this is unreasonable given its excessive demand on computational resources. As for the poor predictive capacity of fuzzy time series (FTS), the cause is not clearly known. Although it can be quite advantageous when dealing with incomplete data (Hosseini, Fard & Baboli, 2011; Li et al., 2010), the accuracy of the approach for network traffic is yet to be fully explored. Attempting to explain the failure is therefore a hypothetical endeavor at this juncture.

Fuzzy time series requires the segregation of time series (Li et al., 2011) into a collection of fuzzy sets. This is not practiced for k -nearest neighbor and artificial neural network. Both accept the time series as it is without further decomposition. Rationalizing from this stance, perhaps the temporal division itself disrupts the forecasting ability of fuzzy time series that consequently deters effectiveness.

Some may argue that if segmenting the time series is the factor of failure, then why does fuzzy time series succeed in other areas of application (Wang, 2011; Aliev et al., 2006) beside network traffic? It must be reminded that it is considered a failure when compared to the other two intelligent approaches here. Despite its low rate of error, the deficiency is still considered extremely high in comparison.

5.6 Summary

Comparing the results of k-nearest neighbor (KNN), artificial neural network (ANN), and fuzzy time series (FTS), it is quite apparent that the best intelligent method for network forecasting is k-nearest neighbor. Its error of prediction is consistently the lowest for nearly all the temporal duration (daily, weekly, and monthly traffic) in all the cases (user A, user B, and user AB). On the other hand, the performance of ANN is slightly less than KNN on nearly all the varying situations. Without a doubt, fuzzy time series (FTS) shows the worst performance and lapses far behind KNN and ANN.