Forecasting malware propagations' dynamics with Machine Learning

Yassine Mehmouden

July 2024

Abstract

In this document, we consider the forecasting of malware epidemic metrics: the epidemic peak and the time to extinction (TTE).
Based on data obtained through simulations, we use machine learning (ML) tools (neural networks (NN) and support vector regression (SVR)) to forecast the dynamics of a malware epidemic in an interconnected network. At the beginning of a simulation, there is only one infected node, and the malware can propagate following different strategies, which will be discussed below. The network administrator can use a defense strategy based on intrusion detection systems (IDS). The simulation follows a basic SIR model (Susceptible - Infected - Recovered). An infected node can become susceptible with a certain probability, and a susceptible node can become recovered with a certain probability. We are interested in this work by forecasting the maximal number of infected nodes at a certain time (the epidemic peak) and the number of time steps after which there are no longer any infected nodes in the network (time to extinction), given some features of the propagation and the network, such as the number of nodes, the number of IDS used, etc.

Contents

List of Figures

1 Introduction

With the multiplication of interconnected networks, malware diffusion has become a massive issue because of how fast some malware can propagate in a network. We can see for instance the multiplication in recent years of computer worms that can replicate themselves without any human action throughout a network, generally an internet network. This kind of malware propagation can be worrying for network administrators and more generally for every person using internet-connected devices. This is why the question of forecasting propagation's metrics is interesting for prevention and to know which defense strategy can be adopted or which network architecture can be used to minimize for instance the epidemic peak and the time to extinction.
We use basic machine learning methods to forecast these metrics: neural networks with a feedforward architecture and support vector machines.
Unlike many forecasting works in the literature in which the data used is real data collected from real epidemics and malware infections, the data used to train the models and to predict is obtained by simulations. The simulator used is given some settings and then simulates the propagation of a malware in a network, the settings that can be used will be discussed below in section 2. The propagations looked at in this work start from a unique source in the network and, in the beginning, no vertex is resistant nor recovered but at each time step, susceptible vertices have the opportunity to become resistant, in our context, this can happen when someone upgrade his password to a more secure one for example.
We will use different features to forecast the metrics and make different predictions with various sets of features to compare and notice the impact of each feature. Moreover, we will gradually make the task more complex by adding more variables like the number of IDS used and limiting the number of features selected for the predictions. Section 2 will focus on the simulations and the settings used, in section 3, we present the models used for predictions and the results obtained and in section 4, we conclude and discuss what can be done in the future to complete this work.

2 Simulations

In this part, we will explain how the simulations are made to understand what will be predicted and which features can be used for this goal.
Firstly, as said in the introduction, the simulations aim to reproduce a malware's propagation within a network. The propagations start from a randomly chosen unique infected vertex of the graph that can diffuse the malware to its neighbors (the vertices it is connected to) according to different propagation dynamics policies.
To model the epidemics, we use a SIR model. The nodes of the networks can be in three states :

- "Infectious": the node is infected by the malware and can transmit it to its neighbors.
- "Susceptible": the node is susceptible and can be infected by the malware.
- "Resistant": the node is resistant, it cannot be infected by the malware anymore.

The simulation ends when there are no infected nodes left in the network. The simulation is based on time steps. During each step, all infected nodes have the opportunity to infect a neighbor and to become susceptible, and all susceptible nodes can become resistant. It is possible to use a defense strategy to detect any intrusion by watching some edges of the graph during each time step.

2.1 The network

To model the network, we use a connected graph. The graphs used in this work are randomly generated. We take the Delaunay triangulation of randomly generated points to obtain a random connected graph.

2.2 SIR model

To complete the SIR model used for our simulations, we use two probabilities :

- S $\rightarrow$ R: the probability that a susceptible node becomes resistant, when the device's user upgrades his password, for example.
- I $\rightarrow$ S: the probability that an infected node becomes susceptible.

2.3 Propagation dynamics

As written at the beginning of this section, an infected node can diffuse the malware according to different propagation policies :

- Unicast propagation: every infected node infects one randomly chosen susceptible neighbor at each time step. If it has no susceptible neighbors, it does not diffuse anything.
- Proportion propagation: given a proportion of neighbors to infect, every infected node tries to infect this proportion of randomly chosen susceptible neighbors at each step.
- Broadcast propagation: every infected node infects all its susceptible neighbors at each time step.
- High-degree unicast propagation: every infected node infects its susceptible neighbor with the highest degree.
- High-degree multicast propagation: every infected node infects all its susceptible neighbors with the highest degree.
- Reinforced high-degree multicast propagation: every infected node infects all its susceptible neighbors with the highest degree and a random number of its other susceptible neighbors.

2.4 Defense strategies

As written above, there is the possibility to use a defense strategy during the simulations. The strategies are based on intrusion detection systems. At each time step, we can watch some of the graph's edges and if during this step, one of the watched edges is used to propagate the virus, both nodes, the transmitter and the receiver, become susceptible. The first strategy is based on proximity, the second one is more random and aims to cover a bigger space.
Say at time step t, an intrusion is detected, we constitute a set of edges with all the edges linked to the node that was transmitting the malware, that are not resistant (not linked to a resistant node) and different from the watched edge in which we discovered the intrusion. If the proximity strategy is used, the edges watched at the time step $(t + 1)$ will be in priority selected in this set of edges. But if the other strategy is used, the edges watched at step $(t + 1)$ will be chosen in another set of edges composed of all the susceptible edges that are not in the first set of edges and different from the watched edges in which we discovered an intrusion at time step t. We can select the number of edges to watch from both strategies, they can be used together.

3 Models and Predictions

We need data to predict the epidemic peak and the TTE of a malware propagation with machine learning methods. Hence, we generate data with the simulator seen in the precedent section. This data needs preprocessing, so we remove outliers (simulations where the epidemic peak is too low, probably because the malware has been luckily detected during the very first steps of the simulation), we normalize all the features so no feature will have a bigger impact than another just because its values are in a bigger scale. Finally, we do one-hot encoding for categorical variables. It will only concern propagation dynamics here.

The networks used for predictions are feedforward networks, which seem the most efficient in epidemics forecasting tasks[1]. Even though the infections studied in this work are not epidemics, this kind of neural network seems to provide satisfactory results.

3.1 Forecasting the epidemic peak

Firstly, we will focus the study on the epidemic peak forecasting of a malware propagation, i.e. the maximal number of infected nodes during a time step. We also will do all the simulations on the same graph for the beginning.

3.1.1 Simple prediction

First, we fix the number of IDS for each strategy and focus only on one type of propagation dynamics. The dataset is divided into 3 (Training, Validation, and Test) with the rule 70/15/15. We use this neural network to begin:

	Layers	Features
1	Dense	160 neurons
2	Dropout	0.3
3	Dense	96 neurons
4	Dropout	0.3
5	Dense	96 neurons
6	Dropout	0.3
7	Dense	1

Table 1: Description of the first NN used

And we use these settings for the training:

Setting	Value
Optimizer	Adam
Learning rate	3×10^{-4}
Loss function	Mean absolute percentage error
Metrics	Root mean squared error
Early Stopping	Yes
Batch size	32

Table 2: First hyperparameters used for epidemic peak forecasting

To predict, we use the following features of the simulation:

- probability S $\rightarrow$ R
- probability I $\rightarrow$ S
- source node's degree
- Number of infected nodes after 5 time steps
- Number of resistant nodes after 5 time steps

[1] P. M. Datilo, Z. Ismail, J. Dare. A Review of Epidemic Forecasting Using Artificial Neural Networks. *International journal of epidemiologic research*, 6(3):132-143, 2019.

The parameters for the simulations are:

- S $\rightarrow$ R probability: randomly chosen according to a uniform distribution between 0 and 0.2
- I $\rightarrow$ S probability: randomly chosen according to a uniform distribution between 0.01 and 0.3
- source node: randomly chosen
- number of proximity IDS: 0
- number of "random" IDS: 8% of the number of edges, so 12

Finally, here is the graph used:

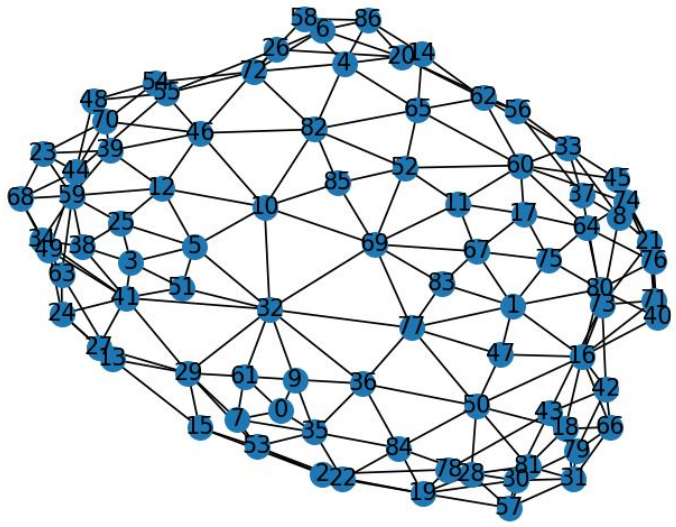
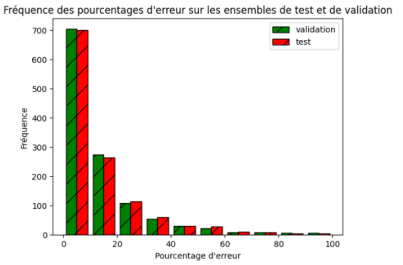


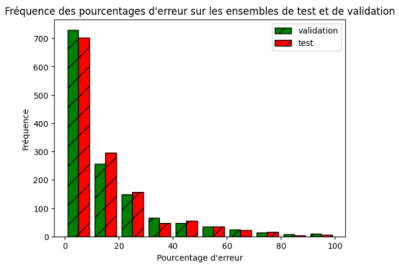
Figure 1: Graph used for simulations

With all the settings above, we obtain these results: [2]

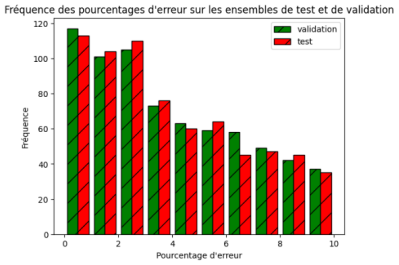
[2]Every time in bar graphs like this, the x-axis will represent the percentage of error of the prediction and the y-axis the number of predictions having this percentage of error. Hence, the values on the y-axis will depend on the dataset used for the prediction, but what matters here is the proportion of presence of each error percentage.



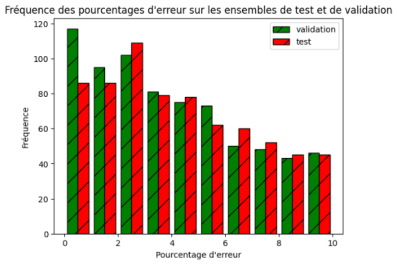
(a) Unicast propagation results



(b) Broadcast propagation results



(c) Unicast propagation results close-up



(d) Broadcast propagation results close-up

Figure 2: Results for the epidemic peak forecasting with everything fixed

The metrics used to evaluate the results of the models are the root mean squared error (RMSE), the mean absolute error (MAE), and the R2-score as defined below:

$$\text{RMSE} = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y_i} - y_i)^2}{n}}$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{yi}|$$

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{y_i} - y_i)^2}{\sum_{i=1}^{n} (\bar{y} - y_i)^2}$$

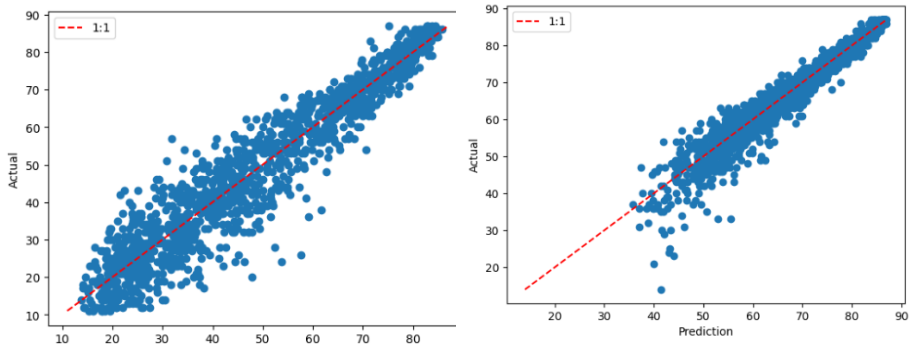
Where n is the number of predictions, $\hat{y_i}$ is the prediction of the real value of the epidemic peak y_i and $\bar{y}$ is the average epidemic peak.

Metric	Unicast propagation	Broadcast propagation
Train RMSE	6.28	7.22
Train MAE	4.41	5.12
Validation RMSE	6.93	7.57
Validation MAE	5.18	5.60
Test RMSE	6.83	7.55
R2-score	0.85	0.83

Table 3: Results for the epidemic peak forecasting with everything fixed

The results obtained are interesting, as we can see with the R2-scores that both are near 0.85 which is correct and is the sign of pretty accurate predictions. We consider an R2-score as good when it is above 0.8.

An SVR can also be used to predict the epidemic peak, with the same preprocessing, here are the results obtained:



(a) Unicast propagation results (b) Broadcast propagation results

Figure 3: Results for the epidemic peak forecasting by SVR with everything fixed

Metric	Unicast propagation	Broadcast propagation
Test RMSE	6.49	3.42
R2-score	0.9	0.93

Table 4: Results for the epidemic peak forecasting by SVR with everything fixed

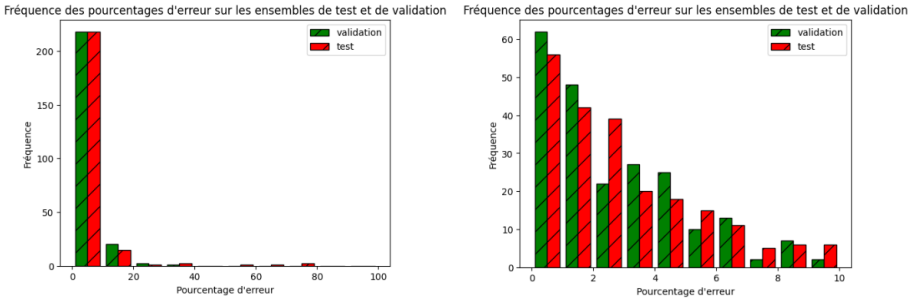
SVR provides better results and predictions, especially for broadcast propagation, for which the scores obtained are fully satisfactory. When forecasting the epidemic peak, SVR will almost always give better results than the neural networks tested here.

3.1.2 Variable defense strategy

Now, we keep everything like before but the defense strategy is variable: the number of random IDS n_r is chosen randomly according to a uniform distribution between 0 and 8% of the number of edges, and the number of proximity IDS is chosen randomly according to a uniform distribution between 0 and $\frac{n_r}{2}$.

Everything else is kept the same: network, settings of the simulations, etc.

We obtain these results for the broadcast propagation, which is the kind of propagation that always provides the best results:



(a) Broadcast propagation results (b) Broadcast propagation results close-up

Figure 4: Results for the epidemic peak forecasting with variable defense and broadcast propagation strategy

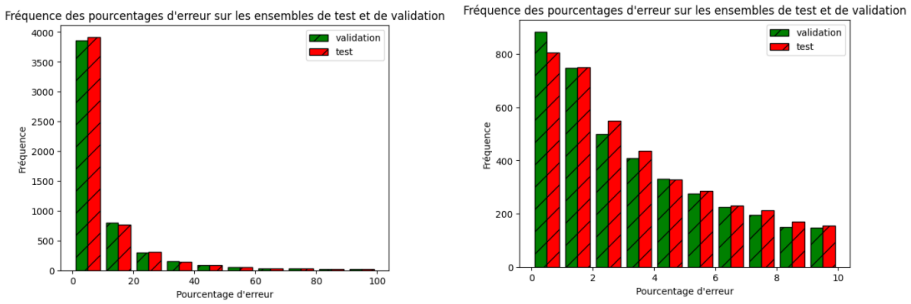
Metric	Broadcast propagation
Train RMSE	3.23
Train MAE	1.98
Validation RMSE	3.13
Validation MAE	2.29
Test RMSE	3.78
R2-score	0.90

Table 5: Results for the epidemic peak forecasting with variable defense and broadcast propagation strategy

The predictions are more accurate when defense strategies are variable as shown in the table above. The R2-score is better and the RMSE is lower than in the precedent section.
Even when trying on other graphs, the results are the same in this section, and for all the upcoming results.

3.1.3 Variable defense strategy and dynamics

In this part, the defense strategy and the propagation dynamics are variable. Everything else is kept the same:



(a) Results (b) Results close-up

Figure 5: Results for the epidemic peak forecasting with variable defense and dynamics

Metric	Value
Train RMSE	5.50
Train MAE	3.71
Validation RMSE	5.93
Validation MAE	4.02
Test RMSE	5.72

Table 6: Results for the epidemic peak forecasting with variable defense and dynamics

The results are less accurate than before but predicting the epidemic peak with this accuracy remains a good result.

3.1.4 Variable defense strategy and graph

In this part, the defense strategy and the graphs are variable, meaning the graphs will differ in every simulation. We fix the propagation dynamics to be a broadcast diffusion because this kind of diffusion provided the best results in previous parts.

We slightly change the network and the features of the prediction:

	Layers	Features
1	Dense	160 neurons
2	Dropout	0.3
3	Dense	128 neurons
4	Dropout	0.3
5	Dense	128 neurons
6	Dropout	0.3
7	Dense	96 neurons
8	Dropout	0.2
9	Dense	1

Table 7: Description of the second NN used

And we use these settings for the training :

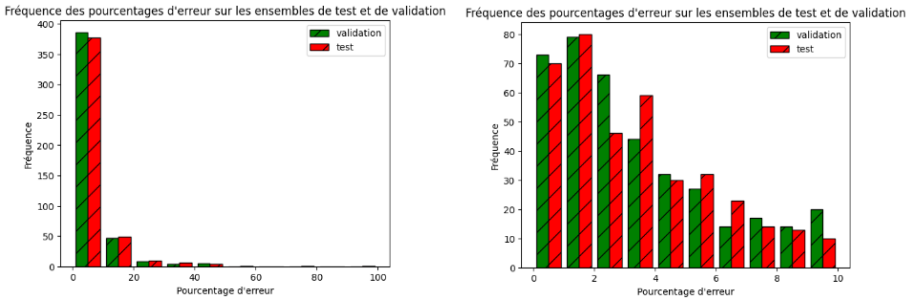
Setting	Value
Optimizer	Adam
Learning rate	3×10^{-4}
Loss function	Mean absolute percentage error
Metrics	Root mean squared error
Early Stopping	Yes
Batch size	32

Table 8: Hyperparameters used for following predictions

For the learning, we use the following features of the simulation:

- probabilities S $\rightarrow$ R and I $\rightarrow$ S

- source node's degree
- Number of infected nodes and resistant nodes after 5 time steps
- Number of "proximity" IDS, and "random" IDS
- Average distance in the graph
- Maximal, minimal, and average degrees in the graph
- Number of edges and nodes of the graph



(a) Results (b) Results close-up

Figure 6: Results for the epidemic peak forecasting with variable defense and graph, and broadcast propagation

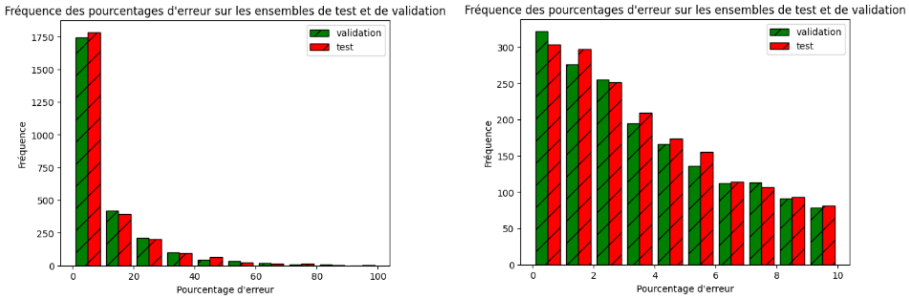
Metric	Value
Train RMSE	4.87
Train MAE	2.64
Validation RMSE	5.64
Validation MAE	3.94
Test RMSE	6.08
R2-score	0.92

Table 9: Results for the epidemic peak forecasting with variable defense and graph, and broadcast propagation

We still get a good accuracy in the predictions with a high R2-score of 0.92.

3.1.5 Everything variable

Finally, everything is variable, so in each simulation, the graph and its characteristics, the defense strategy, the propagation dynamics, and all the other settings are random.
We add the propagation dynamics in the features to predict the epidemic peak.



(a) Results (b) Results close-up

Figure 7: Results for the epidemic peak forecasting with everything variable

Metric	Value
Train RMSE	7.02
Train MAE	4.75
Validation RMSE	7.88
Validation MAE	5.67
Test RMSE	7.85
R2-score	0.91

Table 10: Results for the epidemic peak forecasting with everything variable

We also did an SVR to compare the results:

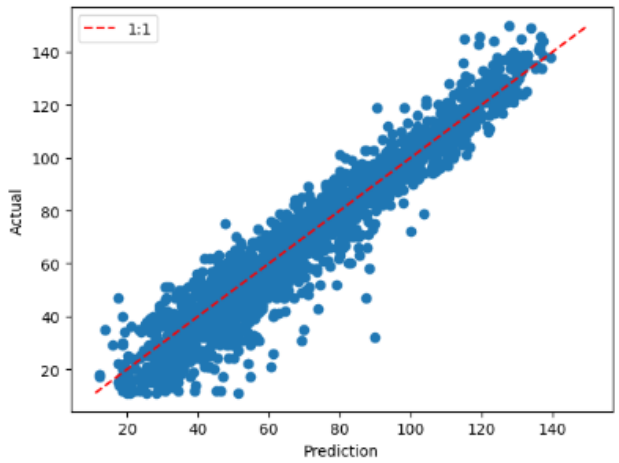


Figure 8: Results for epidemic peak prediction by SVR with everything variable

Metric	Value
Test RMSE	7.99
R2-score	0.92

Table 11: Results for SVR prediction with everything variable

The neural network and the SVR methods permit the prediction of the epidemic peak of a malware infection in a network with good accuracy, even when all features are different in every simulation (graph, strategies...). The R2-score obtained is high, above 0.9, and can be considered satisfactory. We get the best results when every parameter is fixed in the simulations and the propagation dynamics is broadcast type consisting in an R2-score of 0.93 and an RMSE on the test set of 3.42.

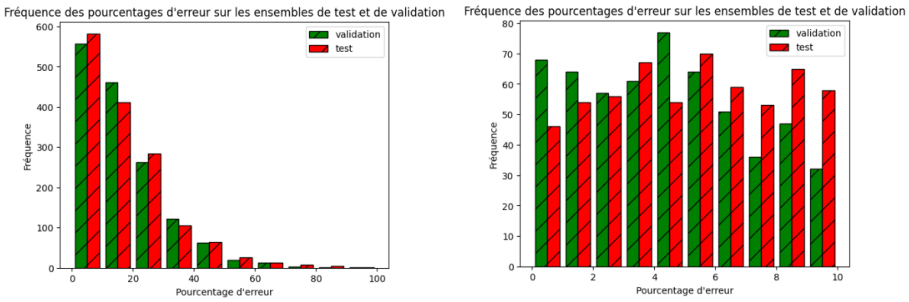
Even when we do not know all the parameters of the infection or the graph, we can predict with high accuracy the epidemic peak, the R2-score obtained is (almost) always above 0.9 and the RMSE is always near 6, meaning that the real epidemic peak predicted will in average be different from the real peak by only 6 units.

3.2 Forecasting the time to extinction

Now, we will focus on the time to extinction of a malware propagation, i.e. the time to have every node of the graph susceptible or resistant. To the best of my knowledge, this is not a topic that has been studied before with machine learning.

3.2.1 Simple prediction

For the beginning, we will fix everything like in the simple prediction of the epidemic peak, and we will use the network described in part 3.1.4.
We obtain these results:



(a) Results (b) Results close-up

Figure 9: Results for the TTE forecasting with everything fixed

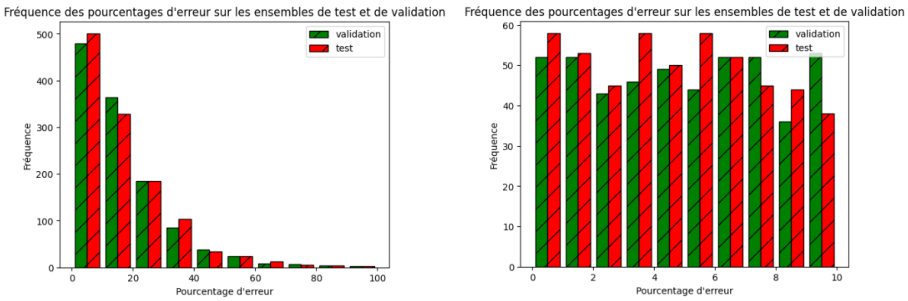
Metric	Value
Train RMSE	419
Train MAE	47
Validation RMSE	448
Validation MAE	85
Test RMSE	313
R2-score	0.92

Table 12: Results for the TTE forecasting with everything fixed

The values of the metrics are higher because of the differences between the rough estimate of the epidemic peak and the rough estimate of the TTE but the R2 score indicates that the prediction accuracy stays good. Moreover, we can see the proportion of time when a prediction has a certain amount of accuracy on the bar graph.

3.2.2 Variable defense strategy

Here, we keep everything like in the previous part, except the number of IDS that will now be variable like in part 3.1.2:



(a) Results (b) Results close-up

Figure 10: Results for the TTE forecasting with variable defense strategy

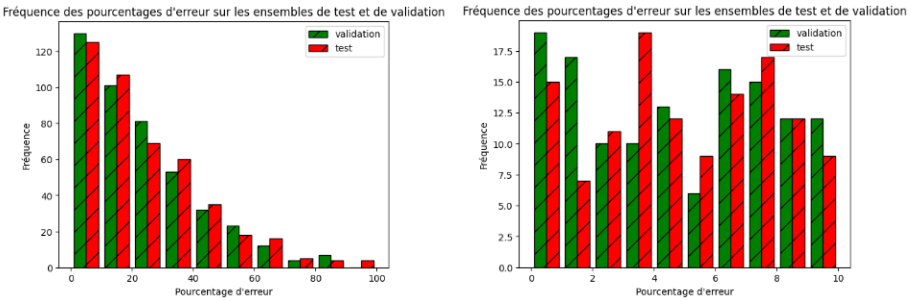
Metric	Value
Train RMSE	307
Train MAE	43
Validation RMSE	289
Validation MAE	81
Test RMSE	998
R2-score	0.26

Table 13: Results for the TTE forecasting with variable defense strategy

Now, the results are not good anymore, the R2-score is low, but as seen in the bar graph, many predictions are still close to the real values.

3.2.3 Variable defense strategy and graph

We now have variable graphs too. For the training, we use the same features as in part 3.1.4.



(a) Results (b) Results close-up

Figure 11: Results for the TTE forecasting with variable defense strategy and graph

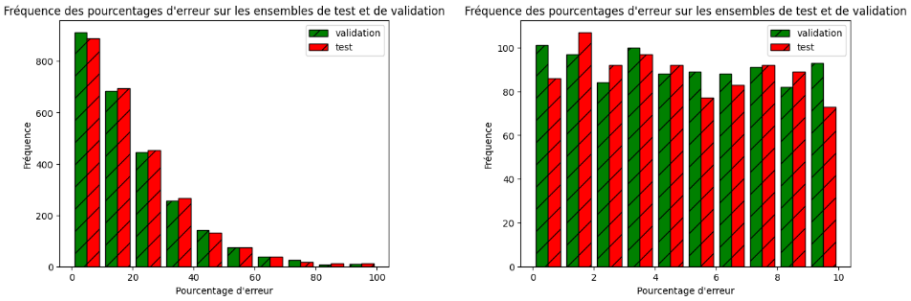
Metric	Value
Train RMSE	1314
Train MAE	66
Validation RMSE	1848
Validation MAE	230
Test RMSE	729
R2-score	0.32

Table 14: Results for the TTE forecasting with variable defense strategy and graph

The results are still disappointing. Even though the R2-score is a bit better, the model can't accurately predict the TTE of the propagation.

3.2.4 Everything variable

Finally, everything is variable like in part 3.1.5:



(a) Results (b) Results close-up

Figure 12: Results for the TTE forecasting with everything variable

Metric	Value
Train RMSE	432
Train MAE	37
Validation RMSE	599
Validation MAE	123
Test RMSE	1574
R2-score	-1.08

Table 15: Results for the TTE forecasting with everything variable

We do an SVR to compare the results:

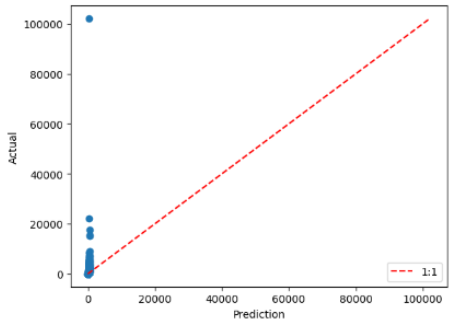


Figure 13: Results for TTE prediction by SVR with everything variable

Metric	Value
Test RMSE	2244
R2-score	0.01

Table 16: Results for TTE prediction by SVR with everything variable

Even when everything is variable, the results of the predictions are not accurate, the R2-score is low, and even with the SVR method which was a bit better in general for epidemic peak forecasting, the results are bad. The predictions seem to be inaccurate because of the large scope of possible values for the TTE.

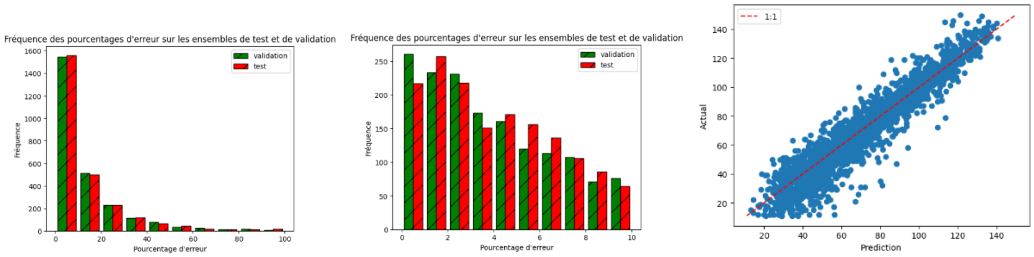
These results show that the only way we currently have to predict the TTE of a malware infection is to know every parameter of the infection and the graph. As shown in part 3.2.1, when everything is known, the R2-score of the predictions is above 0.9 but when one parameter is missing, then it is complicated to get a good result and the R2-score of the predictions is always below 0.35.
Another observation one can make is that the SVR method provides poor results, almost always worse than the NN method. It was the complete opposite when forecasting the epidemic peak.

3.3 More results

An interesting result to observe is that even if some features are removed, predictions of the epidemic peak can be pretty accurate.

3.3.1 Probabilities

When we remove the probabilities of the SIR model of the features (i.e. I $\rightarrow$ S, and S $\rightarrow$ R probabilities), we obtain these results:



(a) Network's results (b) Network's results close-up (c) SVR's results

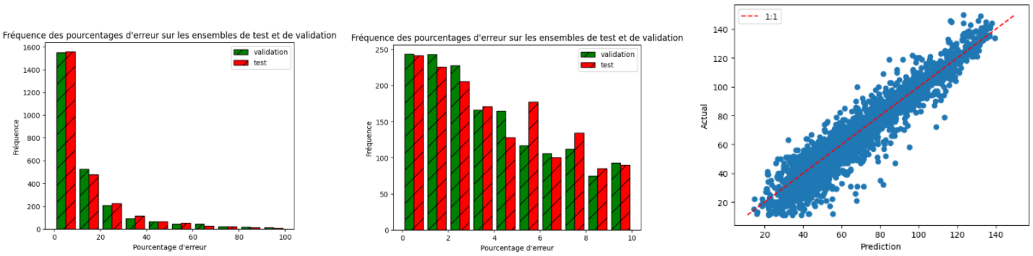
Figure 14: Results for the epidemic peak forecasting with a new set of features

Metric	Network	SVR
Train RMSE	9.38	$\emptyset$
Train MAE	6.23	$\emptyset$
Validation RMSE	10.11	$\emptyset$
Validation MAE	7.21	$\emptyset$
Test RMSE	9.88	9.38
R2-score	0.86	0.89

Table 17: Results for the epidemic peak forecasting with a new set of features

3.3.2 Source node's degree

When we remove the probabilities of the SIR model of the features and the source node's degree, in a context where we cannot identify the source of the propagation, we obtain these results:



(a) Network's results (b) Network's results close-up (c) SVR's results

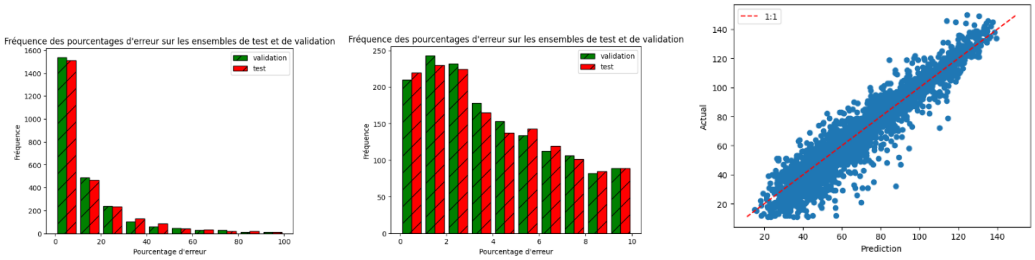
Figure 15: Results for the epidemic peak forecasting with a new set of features

Metric	Network	SVR
Train RMSE	9.97	∅
Train MAE	6.86	∅
Validation RMSE	10.18	∅
Validation MAE	7.32	∅
Test RMSE	10.13	9.15
R2-score	0.84	0.9

Table 18: Results for the epidemic peak forecasting with a new set of features

3.3.3 Propagation dynamics

When we remove the probabilities of the SIR model of the features, the source node's degree, and the propagation dynamics, we obtain these results:



(a) Network's results (b) Network's results close-up (c) SVR's results

Figure 16: Results for the epidemic peak forecasting with a new set of features

Metric	Network	SVR
Train RMSE	11.24	∅
Train MAE	7.70	∅
Validation RMSE	10.89	∅
Validation MAE	7.76	∅
Test RMSE	11.35	9.49
R2-score	0.79	0.89

Table 19: Results for the epidemic peak forecasting with a new set of features

Even when removing some features of the attacks, in contexts where access to this data is complex or impossible, for instance, the epidemic peak of an infection can be pretty much accurately predicted. The more we remove data, the less accurate predictions are, even if the difference is very slight when using the SVR method.

4 Conclusion

In this work, we study the forecasting of the epidemic peak and the time to extinction of a malware infection in a network of interconnected devices using machine-learning methods (neural networks and SVR). We use for the neural networks, a feedforward architecture.
We can predict the epidemic peak of an infection accurately. We get the best accuracy when all the parameters of the simulations and the graph are fixed and when using broadcast propagation. We obtain an R2-score of 0.93 and an RMSE of 3.42 when using an SVR. Even when not everything is fixed, we get good results with R2-scores above 0.85. The features used to obtain these results are some features of the graph, the probabilities of the SIR model, the propagation dynamics, and the defense strategy. The details are presented in the previous parts. However, if we remove some of these features, and we train the model or do the SVR without them, we still get accurate predictions with R2-scores above 0.8. So, even if one doesn't have all the information he wants about an epidemic, he could still predict its peak quite accurately.
Despite the good accuracy for the epidemic peak, the time to extinction has not been predicted with the same precision. The only good prediction was when every setting was known and fixed. In these conditions, the R2-score is 0.92, but if one parameter is not fixed, then the accuracy decreases very much, and the predictions are no longer reliable.
The difficulty of predicting the TTE must be due to the large scale of values it can take combined with the chaos of the simulations. One could consider training a model with massive quantities of data to pursue this training, maybe a huge amount of simulations will help the model learn better patterns and predict more accurately the TTE of a malware infection when not all the features are known.