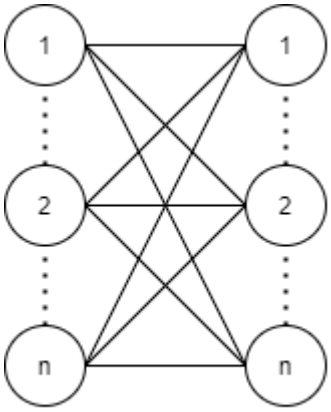
CNN vs NN: Data transformation to improve neural networking model training on malware classification context

Summary

Some researchers use neural networks model to classify malware types after a program analysis. In the last decade, benefits in this field are huge and models used by researchers and/or analysts provide good results. A convolutional neural network is an improvement of neural network model and has made its arrival at the end of the 1990s.



Neural Network

However, these models used a large dataset to make a classification, and a fine detection, with good accuracy. To obtain this condition, these models also required a significant amount of time to be trained. To measure this training with CNN and/or NN models, we should observe two graphs, the learning curve based on the continuous measurement of loss provided by these models and the accuracy curve also based on the accuracy provided by these models while the training step.

To observe these elements, models use datasets to make a supervised classification without features engineering as opposed to models based on machine learning algorithms.

Goals

This research is based on the observing of training model's performances during their training step. Our research tends to provide a good way not to improve accuracy performance but to improve training performance to set up efficient classification models with a reasonable timeframe. We focus our attention to build a process to build a classification model able to reach an optimal accuracy value faster than usual model with a sustainable accuracy.

Methodology

This work is based on three different android malware datasets: “CiCMal2017”, “CidreProject”, and “CiCAndMal 2020”. The last dataset is used to construct two sub-datasets with distinct data. First, we focus our work to make an automatic transformation of dataset records in image construction. This process is also use in biological fields in cellular analysis and tumor detection. However, in our process records are converted in 2-Dim graph with usual matplotlib library. Here is the structure of the two deep learning models:

Damien Strullu – 16/08/2022

Attribute	CNN	NN
Batch Size	16	16
Epochs	30	30
Loss Mesure	Categorical Cross Entropy	Categorical Cross Entropy
Optimizer	Adam	Adam

NN Layer Structure (1)

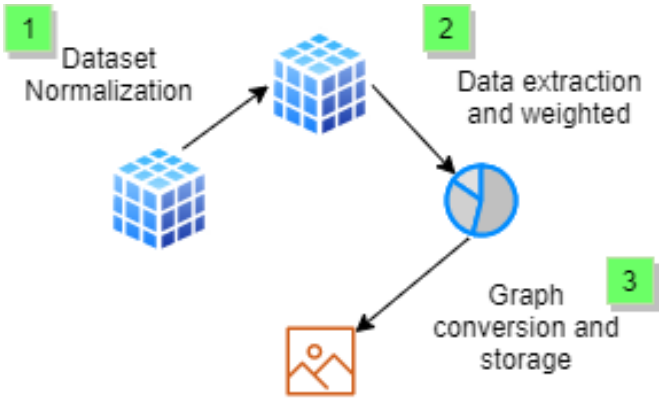
Attribute	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
Node Numbers	Number of classes x 2	Number of classes x 2	-	Number of classes x 2	Number of class	Number of class
Activation	Elu	Elu	-	Relu	Relu	Softmax
Type of layer	Dense	Dense	BatchNormalization	Dense	Dense	Dense

CNN Layer Structure (2)

Attribute	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	Layer 8
Activation	-	Relu	-	Relu	-	-	Relu	-
Type of layer	Rescaling	Conv2D	Max Pooling	Conv2D	Max Pooling	Flatten	Dense	Dense

Transformation process

After each dataset is rebuilt, these datasets are exploited in NN (1) configuration without transformation. Afterward, each record is converted with the following algorithm.



Step 1: Dataset is normalization

Step 2: Relative weight of each data included is calculated and these are stored on a tuple

Step 3: The tuple is converted into "PieChart" and stored on the repository corresponding to the type of malware program.

After this processing, this set of the picture is used with the CNN (2) configuration.

All datasets are processed one by one and separately.

Results

As we see in figure (3), the model learning curve with the NN (1) model starts with a started value much lower than CNN(2) model. Of course, these results are obtained with only four

datasets, but they show an average improvement of 6 percent for this value. This improvement is stable. Furthermore, the end value improvement shows an average of 10% but this result isn't stable (lowest value: 2% - highest value: 20%). We conclude this improvement is highly volatile and it depends on the dataset structure.

As we see in figure (4), the loss model curve with the NN (1) model starts with started value much higher than CNN (2) model. This improvement is shown by a reduction average of 12% of the stated value. This improvement isn't stable (lowest value: 9% - highest value: 22%). Furthermore, the end value benefits from a constant reduction but the amount of this reduction is highly volatile. The average of this reduction is 30,5% (lowest value: 6% - highest value: 79%). Here again, this improvement depends on the dataset structure.

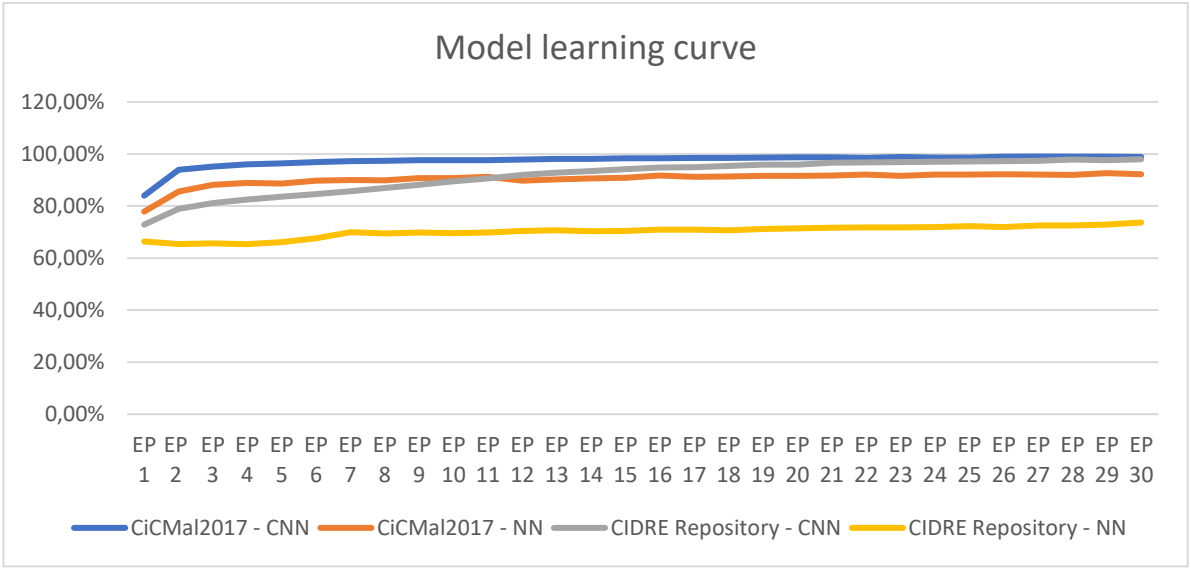
We can conclude this process improve significantly the started value of the learning model curve and loss model curve. Also, the CNN model can grow up more quickly than NN Model with a malware classification model. That's could allow analysts and/or researchers to build deep learning models more efficiently and faster to learn.

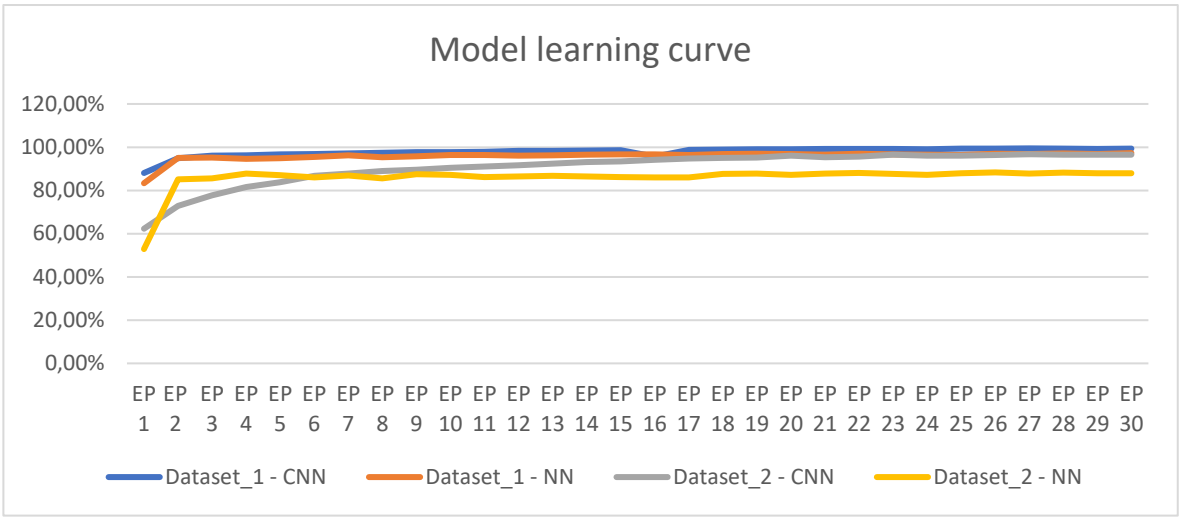
Perspective

Results show a sustainable way to build a more effective android malware classification model able to reach an optimal accuracy quickly than the usual neural network model. This technic is firstly used in genetic and biological field and it is effective in android malware detection. This improvement could allow researcher and/or analyst to exploit CNN (2) model on computer system where resources aren't unlimited or sufficient.

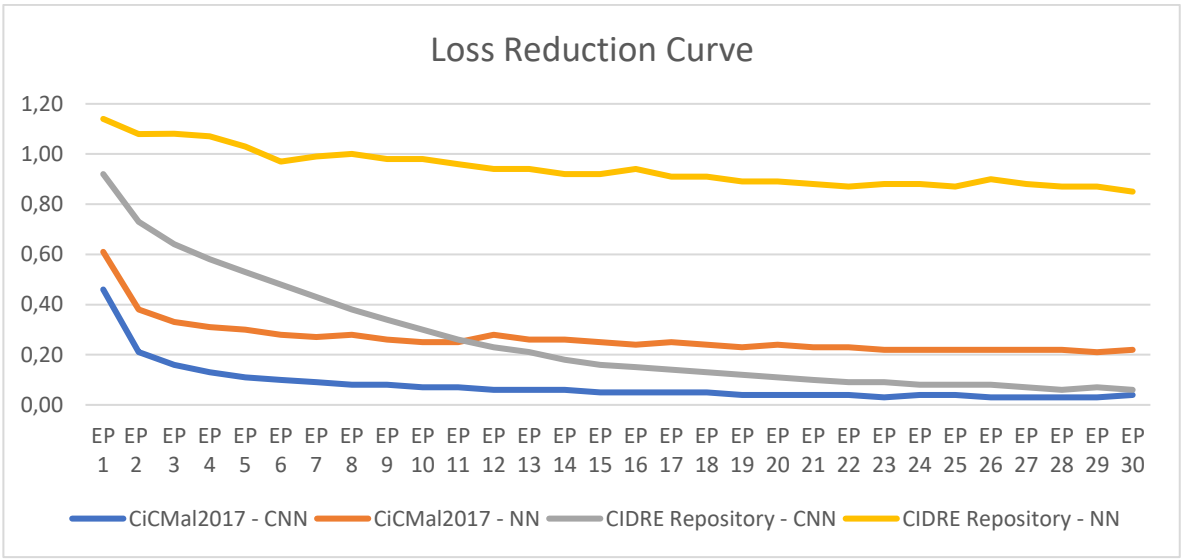
To verify this assertion, we test this implementation on IoT architecture in future research.

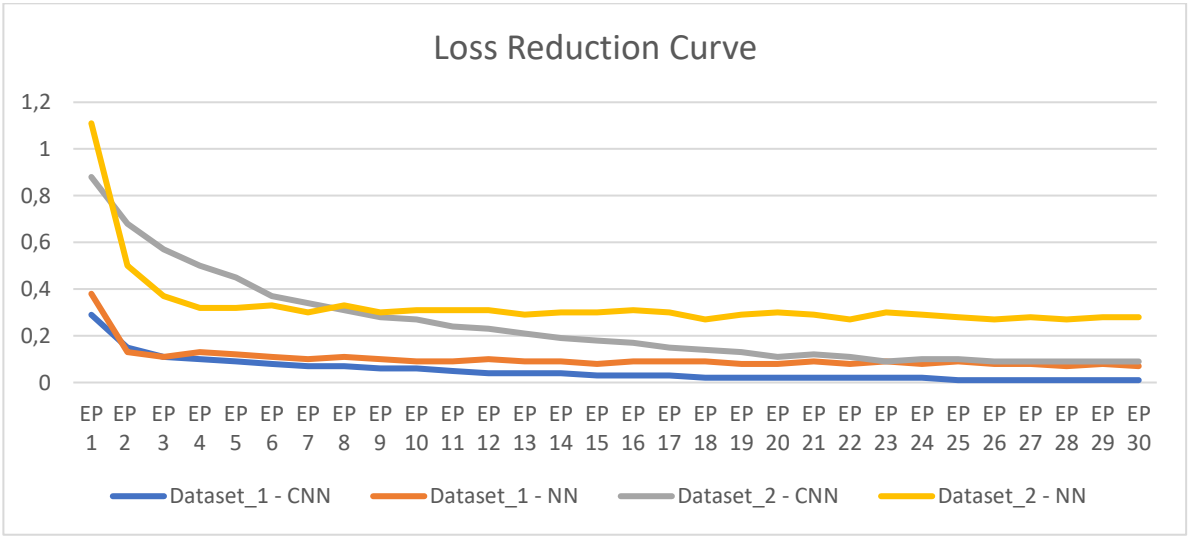
Model learning curve (3)





Loss reduction curve (4)





Damien Strullu – 16/08/2022

References

01 - CidreProject - https://gitlab.inria.fr/cidre-public/dada

02 - CiCAndMal2017 - https://www.unb.ca/cic/datasets/andmal2017.html

03 - CiCAndMal2020 - https://www.unb.ca/cic/datasets/andmal2020.html