Challenge Report

In this challenge we were given a dataset of 10 users that contain 15,000 bash commands for each one.

Every 100 commands defined as a segment and our mission was to distinguish, for each user, if a segment was his own or was executed by a masqueraders.

At first, we examined the data manually, to see if we can find patterns of segments that classified as '1' (masqueraders).

We saw that in some segments that classified as 1 there are a set of commands that coming in a certain order and repeat throughout the entire segment.

That gave us the idea that not only a command itself needed to be taken in consideration when we classify a segment, but also the order and amount of time that a set of commands are appearing.

To work with the data, we first loaded the entire vocabulary and gave each command an index (e.g. sh -> 571). After that we collected all segments from all users and built a data frame with the following structure: (segment id, data\_int, data\_string).

data\_int and data\_string are the commands of the segment in a form of index and string respectively. For our train set we only took the first 50 segments of each user and for the validation set we took the 100 last segments of the first 10 users.

After creating the train and validation sets, we started to extract features for each segment.

The features were: most common command, the most common command count, the rarest command, unique commands count, the longest command, longest command length.

On top of all that, for each segments we collected **ngrams** of 1 and 2 words from all segments, and counted the amount of times that each ngram appears in the segment and we also added ngram of 3 words that were taken from only the segments that classified as 1 and also counted the amount of time that each ngram appears in a segments.

Now our data frame looks like that:

(segment id, data\_int, data\_string, 1gram1, 1gram2, 1gram3 …. 2gram1, 2gram2, …,3gram1, 3gram2, 3gram3...)

1gram1 – the first ngram of 1 word.

3gram1 – the first ngram of 3 words.

Each row contains the amount of time that each n-gram appears in the segment.

Finally, we dropped the data\_int and data\_string columns and normalized the data.

To classify each segment, we built a fully connected Dense neural network with 1 hidden layer of size 150 with activation function ‘relu’, and the output layer was of size 40 with activation function ‘softmax’, so the shape of the model is (num\_of\_features, 150, num\_of\_users).

To train the model we feed the network with our train set and the target was the user that each segment belongs to.

The model output was 40 probabilities that mean what is the probability that the segment belongs to each user. To determine if a segment belongs to the user or not, we took the user that the model predicted with the highest probability and if it was the same as the target, we classified it as 0 otherwise 1. To test the model, we tried to predict our validation set segments and compare it to the actual classifications.

At first, we saw that the model gives us a lot of false negative (predicts a segment as 1 when it needs to be 0) so we made sure that it will classify segments as 1 only for the segments that he is most confident that is correct (with the highest probability) and we limited it to classify only 25 1’s at most.

After limiting the model, the results were much better, and we managed to classify the segments in a very good precision.