Report
Machine Learning Assignment

Assignment by:-
Udit Varshney (1706017)
Suprabhat Kumar (1706018)
Shubhankit Bansal (1706048)

Paper

Name : A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks

Date of publication : October, 12, 2017

Paper by : CHUANLONG YIN , YUEFEI ZHU, JINLONG FEI, AND XINZHENG HE

Dataset Used

In this assignment we have used NSL-KDD dataset which has been widely used in intrusion detection experiments.

Dataset Description:

The dataset contains 41 different features.

It contains 34 continuous data feature and 7 discrete feature.

The dataset has 5 target feature naming: Normal, DoS, R2L, U2R, Probe.

Table of description of dataset:

No.	Features	Types	No.	Features	Types
1	duration	Continuous	22	is_guest_login	Symbolic
2	protocol_type	Symbolic	23	count	Continuous
3	service	Symbolic	24	srv_count	Continuous
4	flag	Symbolic	25	serror_rate	Continuous
5	src_bytes	Continuous	26	srv_serror_rate	Continuous
6	dst_bytes	Continuous	27	rerror_rate	Continuous
7	land	Symbolic	28	srv_rerror_rate	Continuous
8	wrong_fragment	Continuous	29	same_srv_rate	Continuous
9	urgent	Continuous	30	diff_srv_rate	Continuous
10	hot	Continuous	31	srv_diff_host_rate	Continuous
11	num_failed_logins	Continuous	32	dst_host_count	Continuous
12	logged_in	Symbolic	33	dst_host_srv_count	Continuous
13	num_compromised	Continuous	34	dst_host_same_srv_rate	Continuous
14	root_shell	Continuous	35	dst_host_diff_srv_rate	Continuous
15	su_attempted	Continuous	36	dst_host_same_src_port_ra	Continuous
16	num_root	Continuous	37	dst_host_srv_diff_host_rat	Continuous
17	num_file_creations	Continuous	38	dst_host_serror_rate	Continuous
18	num_shells	Continuous	39	dst_host_srv_serror_rate	Continuous
19	num_access_files	Continuous	40	dst_host_rerror_rate	Continuous
20	num_outbound_cmds	Continuous	41	dst_host_srv_rerror_rate	Continuous
21	is_host_login	Symbolic			

In the above table No1 to No10 are the basic features, No11 to No22 are content features, No23 to No41 are traffic features.

Data Preprocessing

Numericalisation: In the dataset there are three features naming *protocol_type, service, flag* having non-numerical values. For operating on the data we need to convert these text feature values into numerical feature values.

protocol_type has 3 feature values(tcp, udp, icmp) which is converted to binary vectors as (1,0,0) for tcp (0,1,0) for udp (0,0,1) for icmp due to which the now the total number of feature also increases.

As service has 70 different values and flag has 11 different values total number of feature increases to 122. Thus finally feature vector becomes 122-dimensional.

Normalisation: In the dataset range of values of the data is varying very widely such as duration varies from 0 to 58329.

We do it in two process:

First we apply a logarithmic scaling method for large range values only and thus our duration comes in range of 0 to 4.77

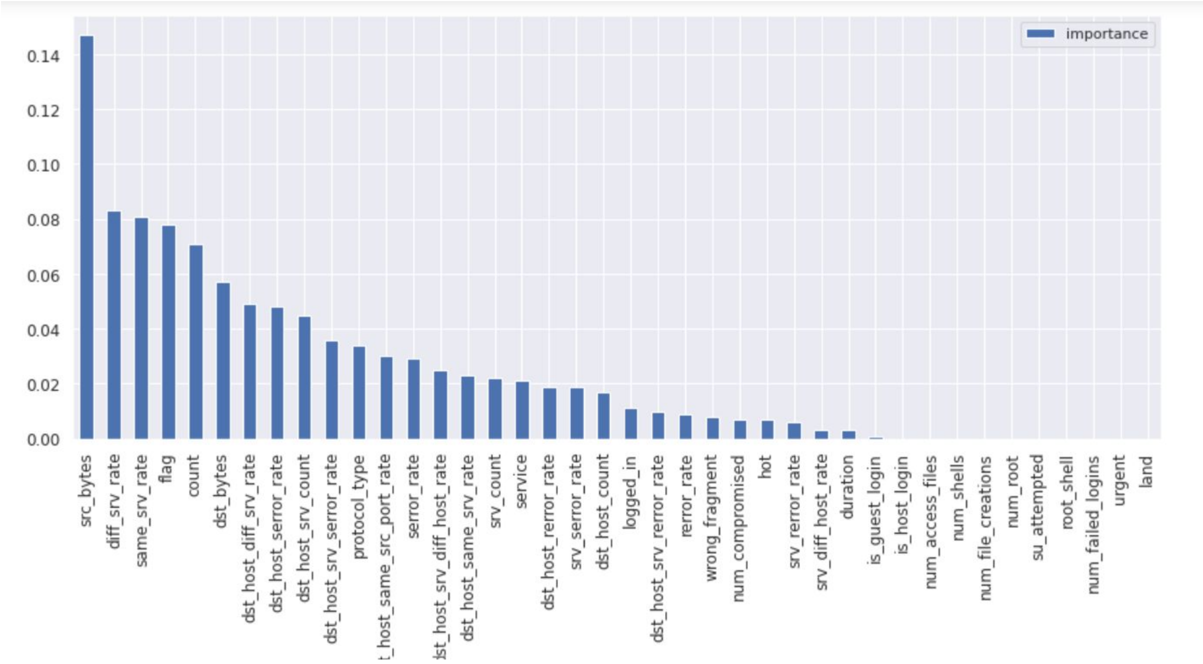
Second we apply simple normalisation on all the feature as follows:

$$x_i = \frac{x_i - Min}{Max - Min}$$

here, xi is our feature value, and Max and Min are the corresponding highest feature value and lowest feature value.

Dimensionality reduction using Recursive Features Elimination Technique: The number of features is very high due to which training and testing takes a lot of time and also it increases the noise in the dataset.

So using recursive feature elimination we are removing those features whose significance is very less.



As in the above graph we can see many of the the features are very insignificant so we remove those features.

Model Used

Recurrent Neural Network is a generalization of feedforward neural network that has an internal memory.

RNN can model sequence of data so that each sample can be assumed to be dependent on previous ones

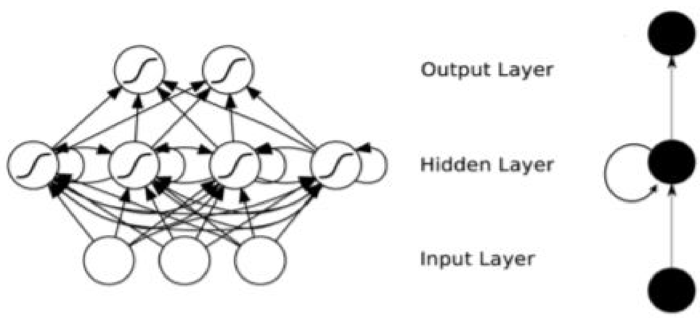


FIGURE 1. Recurrent Neural Networks (RNNs).

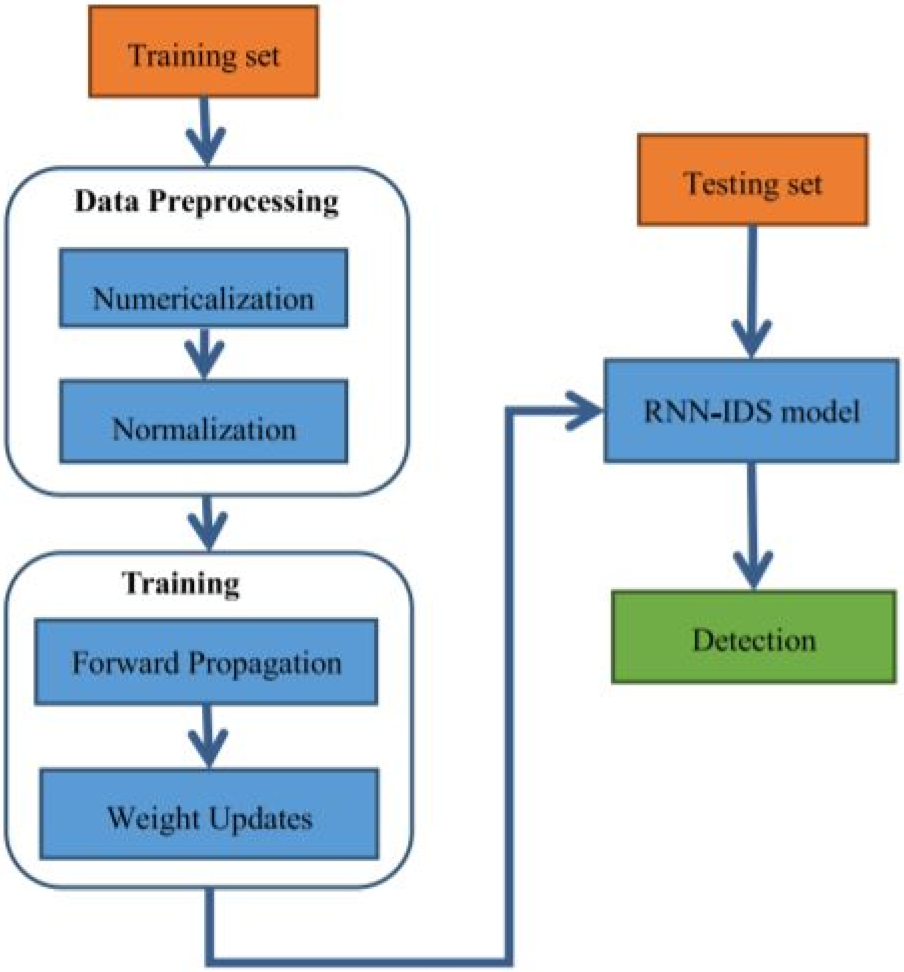
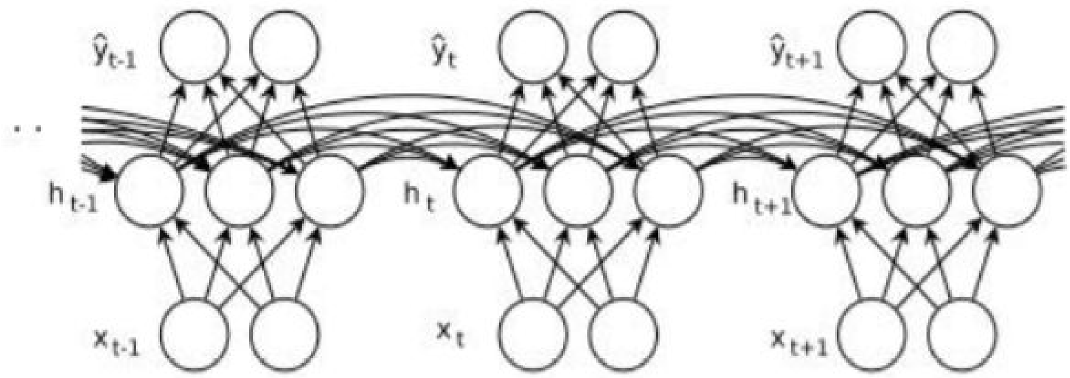


FIGURE 2. **Block diagram of proposed RNN-IDS.**

Methodology: The diagram below is of unfolded RNN.



Given,

training samples xi(i = 1, 2, . . ., m)

a sequence of hidden states hi (i = 1, 2, . . ., m)

a sequence of predictions yˆi(i =1, 2, . . ., m).

Whx is the input-to-hidden weight matrix, Whh is the hidden-to-hidden weight matrix, Wyh is the hidden-to-output weight matrix

The vectors bh and by are the biases .

The activation function e is a sigmoid, and the classification function g engages the SoftMax function.

L(yi : ˆyi) is a distance function which measures the deviation of the predictions yˆi from the actual labels yi .

Let η be the learning rate and k be the number of current iterations.

RNN-IDS consist of two process: Forward Propagation and backward propagation.

Forward Propagation algorithm:

Algorithm 1 Forward Propagation Algorithm

Input x_i(i = 1, 2, . . ., m)

Output $\hat{y}_i$

for *i* from *1* to *m* do

$ti = \mathbf{W}_{hxxi} + \mathbf{W}_{hhhi-1+bh}$

$hi = \text{sigmoid}\,(ti)$

$si = \mathbf{W}_{yhhi+by}$

$\hat{y}i = \text{SoftMax}\,(si)$

end for

Back Propagation Algorithm:

Algorithm 2 Weights Update Algorithm

Input $\langle y_i, \hat{y}_i \rangle (i = 1, 2, \ldots, \text{m})$

Initialization $\theta = \{\mathbf{W}_{hx}, \mathbf{W}_{hh}, \mathbf{W}_{yh}, b_h, b_y\}$

Output $\theta = \{\mathbf{W}_{hx}, \mathbf{W}_{hh}, \mathbf{W}_{yh}, b_h, b_y\}$

1: for i from k downto *1* do

2: Calculate the cross entropy between the output value and the label value: $\text{L}(y_i: \hat{y}_i) \leftarrow -\sum_{\text{i}} \sum_j \text{y}_{ij} \log (\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij})$

3: Compute the partial derivative with respect to θ_i : $\delta_i \leftarrow dL/d\theta_i$

4: Weight update: $\theta_i \leftarrow \theta_i \eta + \delta_i$

5: end for

Evaluation metrics:

First we build a confusion matrix and find True Positive, False Positive, True Negative and False Positive as follows.

Actual Class \ Predicted Class	anomaly	normal
anomaly	TP	FN
normal	FP	TN

Then from this we measure three metrics:-

Accuracy: : the percentage of the number of records classified correctly versus total the records shown

$$\text{AC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Training

- The KDD Dataset has target label “attack_type” which we have classified into binary and multi classes.
- Firstly, dimension of input is converted into 3-Dimension so that it can pass to LSTM layer.
- Sequential model is used here on which first layer is of LSTM having output units of 128 and a hidden layer of 256 units.
- In between the layer, we have used the Dropout layer to prevent overfitting.
- The optimizer used is Adaptive Moment Estimation(Adam)
- The loss function used is binary cross-entropy (for binary classification) and categorical cross entropy (for multi-class classification).
- Batch Size = 32 and Epochs = 25

Result

Binary classification: In this we have classified only on anomaly vs normal.

Confusion matrix obtained:

```
array([[3473,   43],
       [  18, 4024]])
```

Model accuracy in evaluation:

Model	Accuracy
ANN Classifier	0.979
Keras Classifier	0.978
RNN Classifier	0.992
MLP Classifier	0.993
Logistic Regression	0.943
Naive Bayes Classifier	0.886
Decision Tree Classifier	1
Random Forest Classifier	1
KNeighbors Classifier	0.990
Support Vector Classifier	0.979

Model accuracy in validation:

Model	Accuracy
ANN Classifier	0.978
Keras Classifier	0.978
RNN Classifier	0.992
MLP Classifier	0.990
Logistic Regression	0.945
Naive Bayes Classifier	0.895
Decision Tree Classifier	0.994
Random Forest Classifier	0.998
KNeighbors Classifier	0.986
Support Vector Classifier	0.978

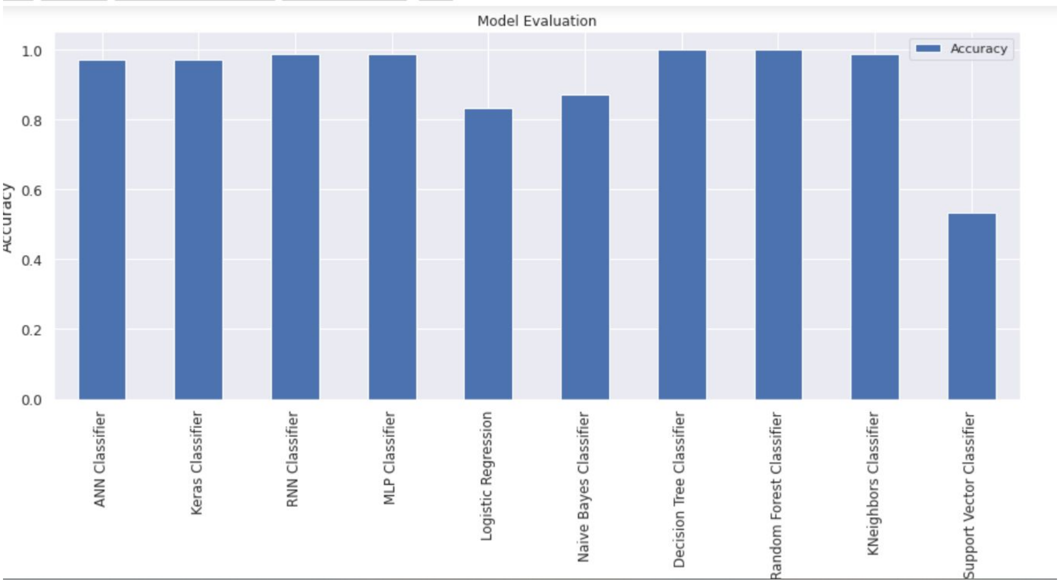
Multiclass classification: In this we are classifying on 5 target label which contains normal and four anomalies naming: DoS (Denial of Service attacks), R2L (Root to Local attacks), U2R (User to Root attack), and Probe (Probing attacks).

Confusion matrix obtained:

```
array([[2749,   62,    5,    0,    0],
       [   2, 4033,    7,    0,    0],
       [   0,   13,  637,    0,    0],
       [   0,   44,    0,    3,    0],
       [   0,    3,    0,    0,    0]])
```

Model accuracy evaluation:

Model	Accuracy
ANN Classifier	0.971
Keras Classifier	0.970
RNN Classifier	0.986
MLP Classifier	0.985
Logistic Regression	0.832
Naive Bayes Classifier	0.871
Decision Tree Classifier	1
Random Forest Classifier	1
KNeighbors Classifier	0.988
Support Vector Classifier	0.534



Model accuracy validation:

Model	Accuracy
ANN Classifier	0.974
Keras Classifier	0.970
RNN Classifier	0.987
MLP Classifier	0.981
Logistic Regression	0.844
Naive Bayes Classifier	0.877
Decision Tree Classifier	0.994
Random Forest Classifier	0.997
KNeighbors Classifier	0.986
Support Vector Classifier	0.536