Intelligent methods for intrusion detection in local area networks

Dmitry Valerievich Pantiukhin
Master of Science,
Senior lecturer, National Research University Higher School of Economics, Moscow, Russian Federation
142800, Russia, Moscow Region, Stupino, Chaikovskogo str. 52, apt. 32
dpantiukhin@hse.ru

Ilya Michailovich Voronkov
Master of Science,
Head of Departament, International Centre of Informatics and Electronics, ICIE, Moscow, Russian Federation
Invited lecturer, National Research University Higher School of Economics, Moscow, Russian Federation
iliav_scn@mail.ru

Alexey Nickolaevich Nazarov
Doctor of Technical Science, Professor
Professor, National Research University Higher School of Economics, Moscow, Russian Federation
a.nazarov06@bk.ru

Annotation: Review of intelligent methods for intrusion detection in local area networks is presented. Publically available datasets of intrusions are shortly described. A problem of imbalanced classes appointed and approach for batch training of a neural network intrusion classifier with imbalanced classes is presented. In computer simulation, it is shown that such approach helps to train on classes with small amount of examples by the cost of larger classes.

Keywords: information security, intrusion detection, neural network classifier, imbalanced classes, intrusion datasets, UNSW-NB15 dataset

1 Introduction

Today machine learning and neural network methods has a quite wide application area and come more and more popular. Of course, most important results made in image processing and there convolutional neural networks play a great role. However, it is also interesting to try application of such methods in different areas, such as information security, and particular in intrusion detection and classification. Then intrusion detection systems (IDS) try to apply neural network to intrusion classification problem we see that there exist a problem, that number of examples of intrusion classes is quite different and classifier need to be trained taking this into consideration.

Researchers can develop different intrusion classifiers, based on different approach. There exist approaches based on:

– signatures of intrusions [1], [2];
– protected system state analysis [3];
– scenario graphs and attack graphs [4], [5];
– expert systems [6], [7];
– multivariate adaptive regression splines [8];
– support vector machines [9], [10], [11];
– immune networks and genetic algorithms [12];

– neural networks, including recurrent [13], [14], [15], convolutional [16], [17], [18] and others [19], [20], [21], [22].

Most of these works use a KDD99 intrusion dataset that is quite old and has no modern examples of intrusions. However, today exists large set of publically available labeled datasets with intrusions examples. We use UNSW-NB15 dataset.

2 Publically available intrusion datasets

Intrusion detection systems and datasets, which they use to training, can be classified in two big classes: network-based and host-based. Host based systems make a decisions using information from defended host machines, like CPU usage, processes launched, errors occurrence etc. Network-based systems make a decisions using information about network traffics between defended hosts and\or outside networks. There are exist examples of both type of systems and combinations of them. Datasets, also can be of these two types, here we deal only with network-based ones, but always remember that for practice better to combine various approaches and use hybrid IDS systems and corresponding datasets. Below short description of existing publically available datasets is presented.

One of the first and still most popular intrusion dataset was a KDD99Cup [23], available since 1998-99 and based on DARPA Intrusion Dataset. It is open, free for researchers, dataset made in Information and Computer Science University of California. It consists from about 5 million records about network transaction. Each record includes 41 parameters of network traffic (for example destination IP\port, source IP\port etc.) of three type: categorical, logical (flags) and numeric. Dataset includes information about 22 types of intrusion in four main classes: Denial of Service (DoS, 3883370 records), Remote to User (R2L, 1126 records), User to Root (U2R, 52 records), Probe (41102 records) and one class of Normal packets (972780 records). As we can see, number of records of different classes is quite different. Although dataset was criticized [24], it is yet the most popular dataset. Main disadvantages are:

- large number of duplicated, redundant records (about 80%!),
- lack of records of some intrusion classes (U2R, R2L),
- moral obsolescence – dataset was collected in 1998-99 and does not contain information about modern attacks.

In 2009, ten years after, KDD99Cup was modified in NSL-KDD [24], redundant records was dropped, some work for refining was made and dataset reduced 4 times, see Table 1. Many authors try to modify this dataset, for example PU-IDS [25] in which statistics of data was used to generate new, unreal but similar, records (about 200 000 new records), and generic framework to generate new similar records was presented.

Table 1 – Comparison of training part of NSL-KDD with respect to KDD CUP 99 [24]

	Original Records	**Distinct Records**	**Reduction Rate**
Attacks	3925650	262178	93.32%
Normal	972781	812814	16.44%
Total	4898431	1074992	78.05%

In 2012, the UNB ISCX 2012 Intrusion Detection Evaluation Data Set [26] was created and announced at the Canadian Institute for Cybersecurity (CIC). This dataset, as well as KDD99, contains information about network connections, but also offers records of all traffic. Traffic was received within 7 days from a test computer network (6 local subnets, NAT server, main and secondary server, traffic monitoring tools), under various attack scenarios. Includes 2450324 connections (about 90GB of traffic), of which 68792 are attacks. CIC continue works on this dataset and, in 2017, the UNB ISCX 2017 (known also as CICIDS2017) dataset was announced. Dataset containing the latest attack scenarios. A tool for traffic analysis was also presented [27]. This dataset was used in [28] for training of different types of neural network classifiers. It is shown that recurrent neural networks like LSTM or GRU achieve a good results (accuracy >97%).

In 2015, the UNSW-NB15 dataset [29] was created at the Australian Center for Cyber Security. Dataset contains both records about network connections, and the traffic itself (about 100GB) from the test computer network (3 servers). Dataset contains 9 types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms. Each records contains 47 fields - information about network connections and 2 fields - information about the type of intrusion (or normal packets). The total number of records is about 2 million. Information about network connections is divided into 4 text files (csv format). Files for training and testing of classifiers, containing respectively 175341 and 82332 records, are separately presented. We will use this dataset in our work. Records amount and description of used intrusions are shown in Table 2 [29].

It is important to note, than mostly intrusion features, measured in the UNSW-NB15 dataset are comparable with ones in the NSL-KDD dataset, so combinations of these datasets can be used for increasing of amount of available data.

Comprehensive comparison of another existing dataset can be found in [30].

Also it can be mention about ADFA dataset from Australian Defense Force Agency which works in different principle, it consist not from records about network traffic, but from information about running processes on host machine, so can be used in host-based intrusion detection systems, which is beyond of our work now.

Table 2 – Statistics of UNSW-NB15 dataset [29]

Type	No. Records	Description
Normal	2,218,761	Natural transaction data.
Fuzzers	24,246	Attempting to cause a program or n-etwork suspended by feeding it the randomly generated data.
Analysis	2,677	It contains different attacks of port scan, spam and html files penetrations.
Backdoors	2,329	A technique in which a system security mechanism is bypassed stealthily to access a computer or its data.
DoS	16,353	A malicious attempt to make a server or a network resource unavailable to users, usually by temporarily interrupting or suspending the services of a host connected to the Internet.
Exploits	44,525	The attacker knows of a security problem within an operating system or a piece of software and leverages that knowledge by exploiting the vulnerability.
Generic	215,481	A technique works against all block-ciphers (with a given block and key size), without consideration about the structure of the block-cipher.
Reconnaissa-nce	13,987	Contains all Strikes that can simulate attacks that gather information.
Shellcode	1,511	A small piece of code used as the payload in the exploitation of software vulnerability.
Worms	174	Attacker replicates itself in order to spread to other computers. Often, it uses a computer network to spread itself, relying on security failures on the target computer to access it.

3 Short review of UNSW-NB15 dataset applications in machine learning approaches to intrusion detection and classification

Recent time UNSW-NB15 dataset is used to create machine learning classifiers by some authors. Some results are described below[1]. Hereafter we use the definition and abbreviation according to [31], "false alarm rate" is synonymous to "false positive rate".

Firstly, in 2015, creators of UNSW-NB15 dataset propose and compare significant feature selection approach to UNSW-NB15 and KDD99 datasets and build classifiers based on Naive Bayes and EM-clustering approaches [32]. They show that such simple approaches works with accuracy about 30-40% but some classes are not recognized at all. In [33] (2017) other authors combine Random forest classification method and feature selection approach, and reach better results both for KDD99 and UNSW-NB15 dataset. In same 2017, another author [34] combine Random Forest method with "Logitboost" [35] boosting algorithm and show better results on both datasets compared to clear Random forest methods [36] and some previous approaches, see Table 3 (for Random Forest application in intrusion detection see also [37]). Boosting is quite attractive approach to create classifiers, as example in [38] authors compare a set of boosting methods (Bagged Tree, AdaBoost, GentleBoost, LogitBoost and RUSBoost algorithms) on UNSW-NB15 dataset and show that Bagged Tree and GentleBoost classifiers show superior performance (see [38] for details). Another comparison of boosting presented in [39] and ensembling methods in [40].

[1] This review based on IEEEXPLORE publications search engine on the July 2019.
https://ieeexplore.ieee.org/search/searchresult.jsp?queryText=unsw%20nb15&highlight=true&returnFacets=ALL&returnType=SEARCH&sortType=newest

Table 3 – Comparison of intrusion detection methods over UNSW-NB15 dataset [34].

Algorithms	Model Build (s)	Test Time (s)	False Alarm Rate (%)	Detection Rate (%)	Accuracy (%)
Naïve Bayes (NB)	0.02	0.06	1.54	98.20	98.33
Support Vector Machine (SVM)	0.21	0.23	0.21	95.87	97.86
Multi Layer Perceptron (MLP)	3.42	0.05	0.42	97.92	98.76
Decision Tree (J48)	0.06	0.04	0.21	98.24	99.03
Random Forests (RF)	0.16	0.08	0.21	98.38	99.10
Adaboost + Random Forests (RF)	0.27	0.17	0.18	98.64	99.24
Logitboost + Random Forests (RF)	**0.35**	**0.21**	**0.18**	**99.10**	**99.45**

In 2016 researches apply genetic search to select appropriate features to each class and use Support Vector Machine to classify intrusions with such features [41]. They achieve high accuracy, more than 90% for all classes except of "Exploits" (~80%), and false positive rates less then 0.1%.

In 2018 quite interesting approach was introduced in [42]. Authors use multiscale wavelet transform and perceptron-like neural network with Hebbian learning for anomaly detection in records with HTTP protocol. They achieve Mean Accuracy 93.56% (wherein Mean TPR is 73.55%, Mean FPR is 4.46%, Mean TNR is 95.53%, Mean FNR: is 26.44%).

Creators of UNSW-NB15 continues their work and propose a Collaborative Anomaly Detection Framework, which is based on modification of Gaussian Mixture Model [43]. This system installed on each node of cloud network and each node in training phase create a statistical model of normal data by approximation of probability distribution function using Gaussian Mixture Model. Then, in testing phase, such model can be used to detect an anomaly, which is interpreted as intrusion. Such approach can be applied only to detection but not to classification of intrusion. Authors achieve results 96% in accuracy and detection rate and 4-9% in false positive rate. Similar approaches but with Hidden Markov Models also was applied in [44], with Beta Mixture Model in [45], with Dirichlet Mixture Mechanism in [46].

Some comparison of machine learning approaches for intrusion classification over UNSW-NB15 dataset are made in [47]. Authors compare Support Vector Machine, Multilayer Perceptron, Restricted Boltzmann Machine, Sparse Autoencoder and deep learning architecture with embedding (like word2vec approach). They show that deep learning approach is better in average than others, but, unfortunately, does not provide information about performance on each classes for UNSW-NB15 dataset. Autoencoders is also interesting approach to intrusion detection, in [48] authors present a two stage approach with autoencoders then second stage uses a results (score – output of classification unit) from first stage. Such approach shows follow results: 89.134% in accuracy and a 0.7495% in FAR for the UNSW-NB15 dataset. Combination of deep autoencoder, Support Vector Machine and Artificial Bee Colony searching method was used in [49] and show detection accuracy about 90% and FAR about 5%.

In [50] authors provide using a Long-Short-Term-Memory (LSTM) neural network, which is a type of Recurrent Neural Network, for intrusion detection over UNSW-NB15 dataset. They achieve quite high results (Precision=98.02%; Accuracy=99.41%; TPR=97.97%; TNR=99.53%, FNR=2.03%, and FPR=0.47%) in detection and show that such approach works better than, for example, Support Vector Machine. Bidirectional LSTM was used in [51] and show average 85% in precision and 88% in recall metrics. It is also shown that some classes are not recognized at all, due to imbalance amount of data.

In [52] deep learning architecture (16 layers, including fully connected with ReLU activation, dropout, and batch normalization layers) was created and tested on various intrusion datasets, including UNSW-NB15. Results not so high as in previous researches, but this architecture in used for large set of different datasets, and clearly show a problem of imbalanced classes.

Authors of [53] provide a set of techniques (Bootstrap Aggregation, Synthetic Minority Over-sampling, Under-sampling, and Class Balancer) to deal with imbalanced classes in intrusion detection system. They show minor advantages in term of area under ROC-curve to whole dataset, but, unfortunately, did not provide any information for classes.

Latest article [54] use a multiple-layer approach consisting of a coarse layer and a fine layer, in which the coarse layer with the deep convolutional neural network model focuses on identification of N abnormal classes and a normal class. While in the fine layer, an improved model based on gcForest (caXGBoost) further classifies the abnormal classes into N-1 subclasses. The proposed framework has been compared with the existing deep learning models using dataset NBC, a combination of UNSW-NB15 and CICIDS2017 including 101 classes. The experimental results show that method outperforms other single deep learning methods in terms of accuracy, detection rate and FAR and works well with imbalanced classes.

4 Neural network structure

Convolutional neural network consists from some number of layers of the next basic types: convolutional layers, fully connected layers, activation layers. In addition, network can contain different layers, such as normalization layers, dropout layers etc.

Convolutional layer performs a convolution of input data and some kernel – set of parameters also known as weights. Such kernels may be adjustable during training or not. In the last case weights is not changing during training and example of such layer is pooling layer which is widely use in image processing to decrease number of adjustable parameters, but for our work number of parameter is not large, so we will not use a convolution with nonadjustable weights. Convolution can be applied to data of different dimension, for example, 2D convolution is more usable in image processing. In our work, we will use 1D convolution, because input data presented as a vector, 1D array. In this case, mathematically convolution can be described as follow. Let x_i be the i-th element of the input vector, $i = 1 \dots N$, where N is the number of input parameters, w_k is the k-th element of the kernel (which is also represented as a vector), $k = 1..K$; w_0 is separate coefficient which is called bias, then the output of a one-dimensional convolutional layer y_j defined as (no stride, no padding):

$$y_j=\sum_k (x_{i+k}*w_k)+w_0 \, , \, j=1..N-K$$

Schematically this is shown in Fig.1 (top).

Layer can contain some number of kernels, ordinary with the same number of parameters. In this case, output of layer is an union of outputs of each kernel.

Fully connected layer consists of neurons, each of them multiplies corresponding element of input by its own weight, summarizes results and adds bias. Inputs to all neurons in a layer is the same. Let x_i be the i-th element of the input vector, $i=1...N$, where N is the number of input parameters, $w_{i,j}$ is the i-th weights of the j-th neuron, $w_{0,j}$ is the bias of the j-th neuron, then the output y_j of the fully-connected layer is defined as:

$$y_j=\sum_i (x_i*w_{i,j})+w_{0,j}$$

Schematically this is shown in Fig.1 (bottom).

Comparing convolutional and fully connected layers we can understand that convolutional layer is some particular case of fully connected one with some weights equal to zero and some weights in different neurons are the same (shared between neurons). Example of such equivalence for small layer is shown in Fig.1. Convolutional layer uses less number of weights but still able to perform complex transformation from input to output.

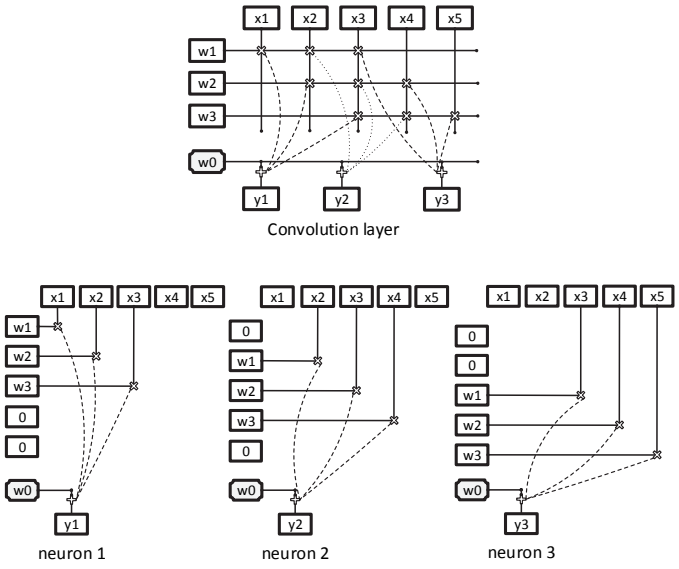

Fig. 1 – Example of equivalence of convolutional (top) and fully connected (bottom) layers with 5 inputs and 3 outputs. ‘x’ – multiplication, ‘+’ – addition.

Activation layer performs some nonlinear transformation from its inputs to outputs. Both, convolutional and fully connected layers perform, as we can see, a linear transformation of inputs. Because of this, if we use two of such layers sequentially, this equivalent to use only one layer (linear transformation of linear transformation is a linear transformation). Therefore, in practice, convolutional or fully connected layers alternates with activation layers. Activation layers apply some activation function to inputs. For example, so called 'sigmoid' calculates each output y as $y=1/(1+e^{-x})$, x – input. Piecewise linear activation function ReLU calculates output as a maximum between 0 and input x. 'Softmax' activation function calculates output y_j as $y_j=exp(x_j)/[\sum_i exp(x_i)]$, x_i - *i-th input.* There are exist large number of different activation functions. Overall, convolutional neural network consists of alternated convolutional and activation layers and has some number of fully connected layers at the end, also alternated with activation.

Structure of network, which we use in our work, is shown in Fig.2. It consist from 5 convolutional layers with 'sigmoid' activations and 3 fully connected layers, first and second with 'sigmoid' and last with 'softmax' activation. Number of inputs – 190 (see Section IV), number of outputs – 10 (number of classes, 9 intrusions and Normal).

To use chosen dataset we need preprocess data to be suitable to neural network. We made it in three stage: choosing of relevant features, coding of nonnumeric data and scaling of numeric data.

Firstly, we need to drop some irrelevant features. UNSW-NB15 has 47 features field, some of them relates only to that computer network which used for dataset collection. Namely, we drop following 7 fields from dataset:

- 'srcip' (source IP address),
- 'dstip' (destination IP address),
- 'sport' (source port number),
- 'dsport' (destination port number),
- 'stime' (record start time),
- 'ltime' (record last time),
-'res_bdy_len' because this field does not change in dataset.

Among other fields, dataset contains 3 categorical (string) fields:

- 'proto' (transaction protocol), has 129 different values ('udp', 'tcp', arp' etc.);
- 'servis', has 8 values ('http', 'ftp', 'smtp', 'ssh', 'dns', 'ftp-data', 'ir and '-' if not used);
- 'state', depend on transaction protocol and has 16 values (ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN and '-' if not applicable).

We code this categorical fields using one-hot encoding scheme then a sparse vector with '0' and only one '1' show the value of fields. Therefore, for categorical fields we have output vector of 153 (129+8+16) length. 15 of 37 numerical fields scaled by division to appropriate value to make range almost equal and remaining 22 fields unchanged.

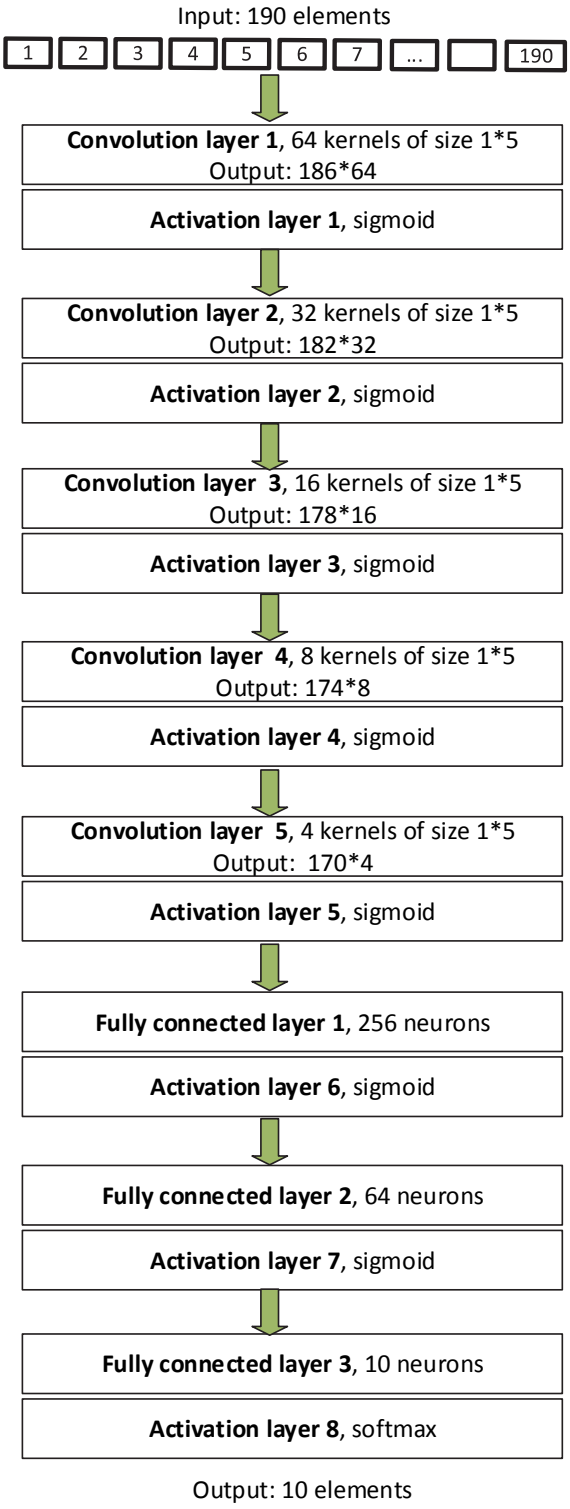

Fig. 2 – Structure of used neural network.

5 Class-balanced training

Desired outputs of neural network classifier is 10-dimensional vectors showed class of input data. We have 9 classes of intrusions (‘Fuzzers’, ‘Analysis’, ‘Backdoors’, ‘DoS’, ‘Exploits’, ‘Generic’, ‘Reconnaissance’, ‘Shellcode’, ‘Worms’) and one class of normal packets (‘Normal’). The same one-hot encoding scheme used for outputs.

Neural network trained using ‘RMS-prop’ [55] method, which is one of the modification of gradient-based method realized in ‘keras’ [28] library. Training is an iterative process in which loss-function minimizes. On each iteration, examples of inputs and corresponding desired outputs processed in neural network.

Common choice for large datasets is to use so called ‘batches’ – combination of inputs\outputs of some length. This batches varies in each iteration and random input\output examples taken to a batch (see Fig.3 top).

In our work, we show that for situations then amount of examples in classes is very different better to use another approach for batch forming, namely predefine amount of examples of each class in batch (see Fig.3 bottom). For example, if batch size is 300 and we have 10 classes then we can form a batch that consist of 30 random examples of each class.

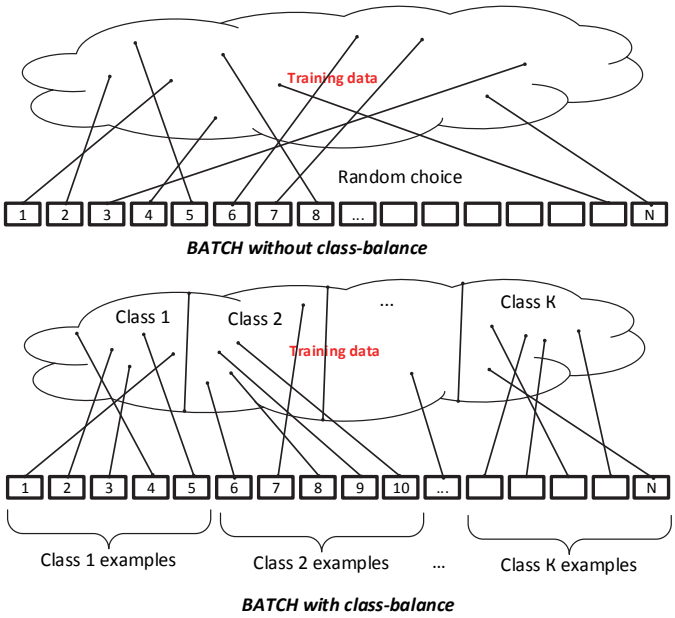

Fig. 3 – Variants of batch forming. Top – random selection, bottom – class-balanced selection.

6 Results and discussion

For computational experiments we use next software products:

- programming language: Python 3.5 with Anaconda 4.2.0;
- Microsoft Visual Studio 2017, v15.5.2;
- neural networks training: : keras 2.0.2 with tensorflow (GPU version) 1.1.0;
- additional: pandas 0.20.1 for text file processing, numpy 1.13.1 for calculations, scikit-learn 0.18.1 for data encoding, matplotlib 2.0.2 for visulization.

and hardware:

- computer with IntellCore i7, 3.4GHz, 32 Gb
- graphical processing unit Nvidia Quadro K600, 1Gb

We use the following configuration of neural networks:

- structure showed in Fig.2;
- batch size: 256;
- learning rate lr=0.001;
- maximum allowed training epochs: 200;
- other parameters are by default of keras library [55].

We train and compare neural network in two variants: with random batches (default) and with class-balanced batches. Test data differ from train data, experiments repeat several times, mean results are shown.

Fig. 4 and 5 shows results of experiment. Fig. 4 shows confusion matrix on train and test data. Confusion matrix show how much data recognized well (diagonal elements) and how many mistakes (non-diagonal elements). Fig.5 shows value of loss function and accuracy during training both on train and test data. Loss-function is a categorical cross-entropy [55].

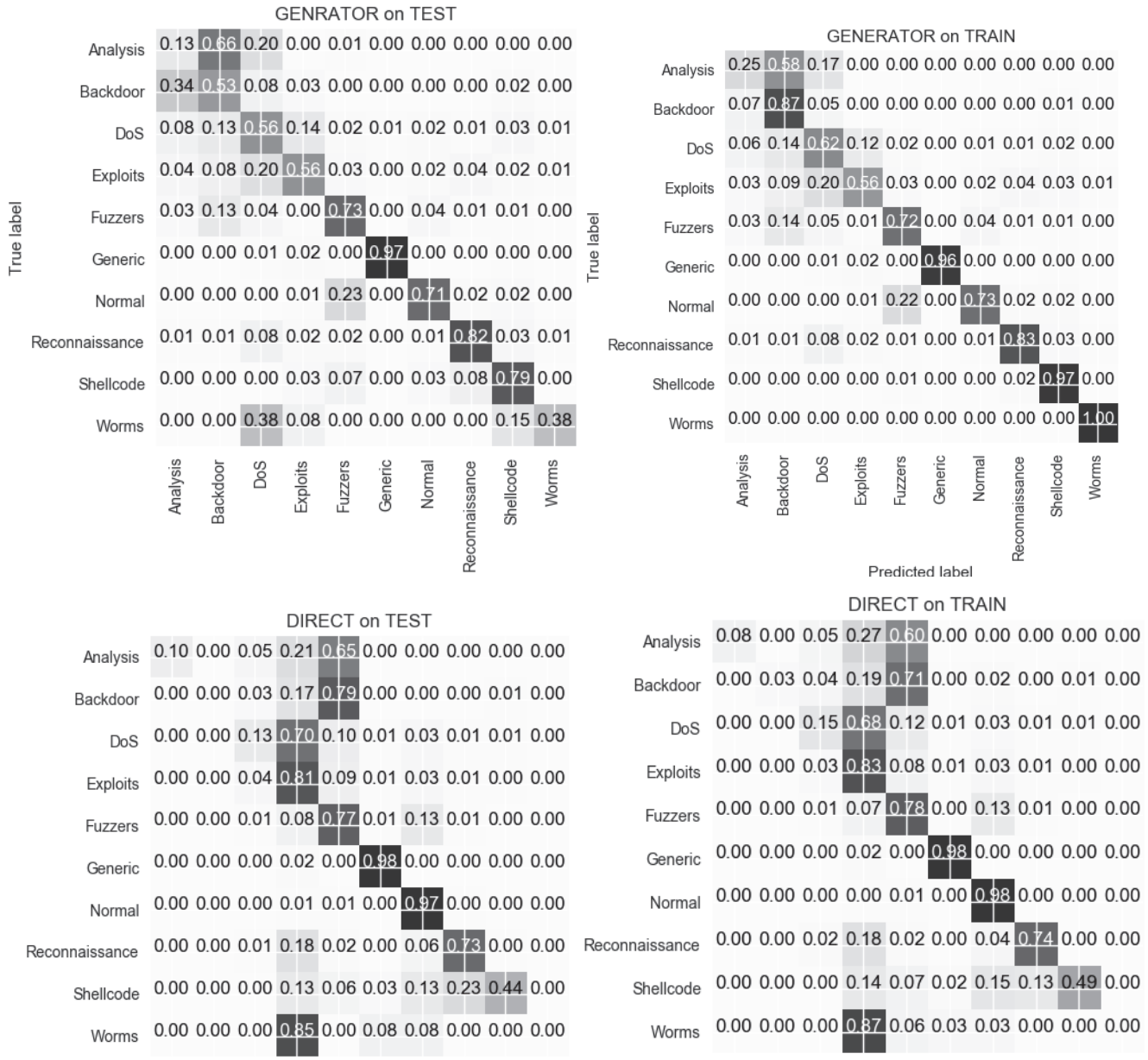

Fig. 4. Confusion matrix on train and test sets for random (‘direct’) batch forming and class-balanced (‘generator’) one.

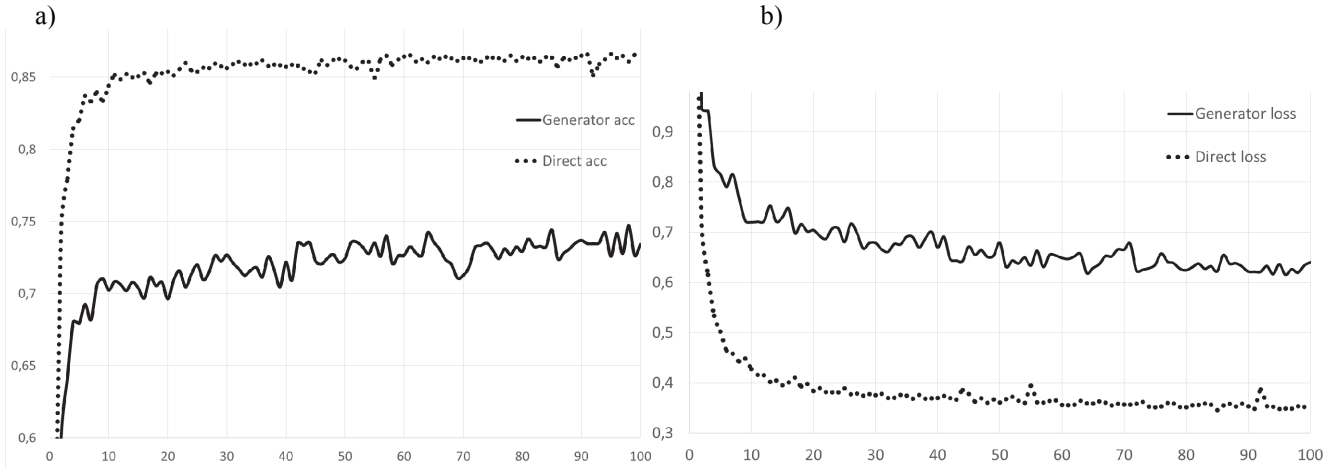

Fig.5 – a) Accuracy and b) loss-function values on test set for random (‘direct’) batch forming and class-balanced (‘generator’) one.

Analyzing the results we can see that class-balanced approach to batch forming can improve recognition performance of classes with small number of examples. Without such balance it is possible that some classes will be absolutely ignored by neural network (‘worms’, ‘backdoors’ in our experiment). In addition, we can see that better accuracy (and loss) does not guaranteed that classification will be better (see Fig.4 and 5 that shows that accuracy is better for random batch forming but some classes does not recognized at all).

But it is need to understand that better recognition of classes with small number of examples achieved at the expense of worse recognition of classes with large number of examples (compare ‘Normal’ and ‘Exploits’ for both cases). Therefore, such approach allow to control, is some degree, recognition performance of different classes and can be used in systems with different importance of classes. Note also that usage of class-balanced approach is not a guarantee of good recognition and only a one means to increasing of such recognition (see for example on class ‘Analysis’ which is not recognized nor by random batch forming nor by class-balanced one).

7 Conclusion

In this work we show that for intrusion detection tasks it is a common situation then amount of examples of data of various classes is quite different. In this case training process of neural networks for intrusion recognition need to be modified to achieve better recognition of classes with small amount of examples. We describe an approach of class-balanced batch forming and show in experiment that it can improve recognition performance of classes with small number of examples by the expense of decreased recognition performance of classes with large number of examples.

Approach can be enlarged to control relative importance of classes by varying proportion of classes presented in batch.

Previously reported UNSW-NB15 dataset were used to support this study and it is available at 10.1080/19393555.2015.1125974, and can be downloaded from https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/ .These prior studies (and datasets) are cited at relevant places within the text as references [29].

8 Acknowledgement

Support from the Basic Research Program of the National Research University Higher School of Economics is gratefully acknowledged.

References

[1] S. Kumar, E. H. Spafford, “A pattern matching model for misuse intrusion detection,” in Proc. of the 17th National Computer Security Conference, Baltimore, MD, USA, 1995, pp. 11–21.

[2] Erlacher and F. Dressler, "FIXIDS: A high-speed signature-based flow intrusion detection system," NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, 2018, pp. 1-8. doi: 10.1109/NOMS.2018.8406247

[3] S. T. Eckmann, G. Vigna, R. A. Kemmerer, “STATL: An attack language for state-based intrusion detection,” in Journal of Computer Security, vol. 10, № 1-2, pp. 71-103, 2002.

[4] S. Roschke, F. Cheng and C. Meinel, "High-quality attack graph-based IDS correlation," in Logic Journal of the IGPL, vol. 21, no. 4, pp. 571-591, Aug. 2013. doi: 10.1093/jigpal/jzs034

[5] O. M. Sheyner, “Scenario Graphs and Attack Graphs,” Ph.D. dissertation, School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, USA, 2004.

[6] M. M. Sebring, “Expert systems in intrusion detection: a case study,” in Proc. 11th National Computer Security Conference, Baltimore, Maryland, USA, 1988, pp. 74-81.

[7] B. Peralta, A. Saavedra and L. Caro, "A proposal for mixture of experts with entropic regularization," 2017 XLIII Latin American Computer Conference (CLEI), Cordoba, 2017, pp. 1-9. doi: 10.1109/CLEI.2017.8226425

[8] S. E. Smaha, “Haystack: An intrusion detection system,” in IEEE Aerospace Computer Security Applications Conference, Washington, DC, USA, 1988, pp. 37-44.

[9] S. Mukkamala, A. H. Sung, A. Abraham, “Intrusion detection using an ensemble of intelligent paradigms,” in Journal of Network and Computer Applications, vol. 28, .№ 2, pp. 167-182, 2005.

[10] C. Zhang, M. Ni, H. Yin and K. Qiu, "Developed Density Peak Clustering with Support Vector Data Description for Access Network Intrusion Detection," in IEEE Access. doi: 10.1109/ACCESS.2018.2866128

[11] I. Ahmad, M. Basheri, M. J. Iqbal and A. Rahim, "Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection," in IEEE Access, vol. 6, pp. 33789-33795, 2018. doi: 10.1109/ACCESS.2018.2841987

[12] M. Zielinski, L. Venter, “Applying mobile agents in an immune-system-based intrusion detection system: reviewed article,” in South African Computer Journal, vol. 2005, №.34, pp. 76-83, 2005.

[13] W. Anani and J. Samarabandu, "Comparison of Recurrent Neural Network Algorithms for Intrusion Detection Based on Predicting Packet Sequences," 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), Quebec, QC, Canada, 2018, pp. 1-4. doi: 10.1109/CCECE.2018.8447793

[14] C. Xu, J. Shen, X. Du and F. Zhang, "An Intrusion Detection System Using a Deep Neural Network with Gated Recurrent Units," in IEEE Access. doi: 10.1109/ACCESS.2018.2867564
[15] Y. Fu, F. Lou, F. Meng, Z. Tian, H. Zhang and F. Jiang, "An Intelligent Network Attack Detection Method Based on RNN," 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), Guangzhou, 2018, pp. 483-489. doi: 10.1109/DSC.2018.00078
[16] U. Çekmez, Z. Erdem, A. G. Yavuz, O. K. Sahingoz and A. Buldu, "Network anomaly detection with deep learning," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, 2018, pp. 1-4. doi: 10.1109/SIU.2018.8404817
[17] W. Lin, H. Lin, P. Wang, B. Wu and J. Tsai, "Using convolutional neural networks to network intrusion detection for cyber threats," 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba, 2018, pp. 1107-1110. doi: 10.1109/ICASI.2018.8394474
[18] R. Vinayakumar, K. P. Soman and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, 2017, pp. 1222-1228. doi: 10.1109/ICACCI.2017.8126009
[19] H. Debar, M. Becker, D. Siboni, “A neural network component for an intrusion detection system,” in Proc. of 1992 IEEE Computer Society Symposium on Research in Security and Privacy, IEEE, 1992, pp. 240-250.
[20] R. Jalili et al., “Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks,” in International Conference on Information Security Practice and Experience, Springer, Berlin, Heidelberg, 2005, pp. 192-203.
[21] A. Nisioti, A. Mylonas, P. D. Yoo and V. Katos, "From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods," in IEEE Communications Surveys & Tutorials. doi: 10.1109/COMST.2018.2854724
[22] M. Zaman and C. Lung, "Evaluation of machine learning techniques for network intrusion detection," NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, 2018, pp. 1-5. doi: 10.1109/NOMS.2018.8406212
[23] “KDD Cup 1999 Data.” [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99
[24] M. Tavallaee et al., “A detailed analysis of the KDD CUP 99 data set,” in 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), IEEE, 2009, pp. 1-6.
[25] R. Singh, H. Kumar, and R. Singla, “A Reference Dataset for Network Traffic Activity Based Intrusion Detection System,” International Journal of Computers Communications & Control, vol. 10, no. 3, pp. 390–402, 2015.
[26] A. Shiravi et al, “Toward developing a systematic approach to generate benchmark datasets for intrusion detection,” in Computers & Security, vol. 31, № 3, pp. 357-374, 2012.
[27] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. of 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, Jan. 2018, pp.108-116.
[28] Le T. T. H. et al. Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks //Applied Sciences. – 2019. – Т. 9. – №. 7. – p. 1392.
[29] N.Moustafa, J.Slay, “The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set,” in Information Security Journal: A Global Perspective, vol. 25, № 1-3, pp. 18-31, 2016.
[30] Ring M. et al. A Survey of Network-based Intrusion Detection Data Sets //arXiv preprint arXiv:1903.02460. – 2019.
[31] Precision and recall, Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Precision_and_recall
[32] N. Moustafa and J. Slay, "The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems," /2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)/, Kyoto, 2015, pp. 25-31. doi: 10.1109/BADGERS.2015.014.
[33] T. Janarthanan and S. Zargari, "Feature selection in UNSW-NB15 and KDDCUP'99 datasets," /2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)/, Edinburgh, 2017, pp. 1881-1886. doi: 10.1109/ISIE.2017.8001537.
[34] M. H. Kamarudin, C. Maple, T. Watson and N. S. Safa, "A LogitBoost-Based Algorithm for Detecting Known and Unknown Web Attacks," in /IEEE Access/, vol. 5, pp. 26190-26200, 2017. doi: 10.1109/ACCESS.2017.2766844.
[35] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: A statistical view of boosting,” Ann. Stat., vol. 28, no. 2, pp. 337-374, 2000.
[36] H. Tribak, B. L. Delgado-Márquez, P. Rojas, O. Valenzuela, H. Pomares, and I. Rojas, “Statistical analysis of different artificial intelligent techniques applied to intrusion detection system,” in Proc. Int. Conf. Multimed. Comput. Syst., 2012, pp. 434-440.
[37] R. Primartha and B. A. Tama, "Anomaly detection using random forest: A performance revisited," /2017 International Conference on Data and Software Engineering (ICoDSE)/, Palembang, 2017, pp. 1-6. doi: 10.1109/ICODSE.2017.8285847.
[38] V. Timčenko and S. Gajin, "Ensemble classifiers for supervised anomaly based network intrusion detection," /2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)/, Cluj-Napoca, 2017, pp. 13-19. doi: 10.1109/ICCP.2017.8116977.

[39] B. Patel, Z. Somani, S. A. Ajila and C. Lung, "Hybrid Relabeled Model for Network Intrusion Detection," /2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)/, Halifax, NS, Canada, 2018, pp. 872-877. doi: 10.1109/Cybermatics_2018.2018.00167.
[40] N. Moustafa, B. Turnbull and K. R. Choo, "An Ensemble Intrusion Detection Technique Based on Proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things," in /IEEE Internet of Things Journal/, vol. 6, no. 3, pp. 4815-4830, June 2019. doi: 10.1109/JIOT.2018.2871719.
[41] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," /2016 8th International Symposium on Telecommunications (IST)/, Tehran, 2016, pp. 139-144. doi: 10.1109/ISTEL.2016.7881798
[42] S. Siddiqui, M. S. Khan and K. Ferens, "Multiscale Hebbian neural network for cyber threat detection," /2017 International Joint Conference on Neural Networks (IJCNN)/, Anchorage, AK, 2017, pp. 1427-1434. doi: 10.1109/IJCNN.2017.7966020
[43] N. Moustafa, G. Creech, E. Sitnikova and M. Keshk, "Collaborative anomaly detection framework for handling big data of cloud computing," /2017 Military Communications and Information Systems Conference (MilCIS)/, Canberra, ACT, 2017, pp. 1-6. doi: 10.1109/MilCIS.2017.8190421.
[44] N. Moustafa, E. Adi, B. Turnbull and J. Hu, "A New Threat Intelligence Scheme for Safeguarding Industry 4.0 Systems," in /IEEE Access/, vol. 6, pp. 32910-32924, 2018. doi: 10.1109/ACCESS.2018.2844794.
[45] N. Moustafa, J. Slay and G. Creech, "Novel Geometric Area Analysis Technique for Anomaly Detection using Trapezoidal Area Estimation on Large-Scale Networks," in /IEEE Transactions on Big Data/. doi: 10.1109/TBDATA.2017.2715166.
[46] N. Moustafa, K. R. Choo, I. Radwan and S. Camtepe, "Outlier Dirichlet Mixture Mechanism: Adversarial Statistical Learning for Anomaly Detection in the Fog," in /IEEE Transactions on Information Forensics and Security/, vol. 14, no. 8, pp. 1975-1987, Aug. 2019. doi: 10.1109/TIFS.2018.2890808.
[47] J. Yan, D. Jin, C. W. Lee and P. Liu, "A Comparative Study of Off-Line Deep Learning Based Network Intrusion Detection," /2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)/, Prague, 2018, pp. 299-304. doi: 10.1109/ICUFN.2018.8436774.
[48] F. A. Khan, A. Gumaei, A. Derhab and A. Hussain, "A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection," in /IEEE Access/, vol. 7, pp. 30373-30385, 2019. doi: 10.1109/ACCESS.2019.2899721.
[49] Q. Tian, J. Li and H. Liu, "A Method for Guaranteeing Wireless Communication Based on a Combination of Deep and Shallow Learning," in /IEEE Access/, vol. 7, pp. 38688-38695, 2019. doi: 10.1109/ACCESS.2019.2905754.
[50] S. Xiao, J. An and W. Fan, "Constructing an Intrusion Detection Model based on Long Short-term Neural Networks," /2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)/, Singapore, 2018, pp. 355-360. doi: 10.1109/ICIS.2018.8466445.
[51] S. Yang, "Research on Network Behavior Anomaly Analysis Based on Bidirectional LSTM," /2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)/, Chengdu, China, 2019, pp. 798-802. doi: 10.1109/ITNEC.2019.8729475.
[52] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," in /IEEE Access/, vol. 7, pp. 41525-41550, 2019. doi: 10.1109/ACCESS.2019.2895334.
[53] C. Wheelus, E. Bou-Harb and X. Zhu, "Tackling Class Imbalance in Cyber Security Datasets," /2018 IEEE International Conference on Information Reuse and Integration (IRI)/, Salt Lake City, UT, 2018, pp. 229-232. doi: 10.1109/IRI.2018.00041.
[54] X. Zhang, J. Chen, Y. Zhou, L. Han and J. Lin, "A Multiple-layer Representation Learning Model for Network-Based Attack Detection," in /IEEE Access/. doi: 10.1109/ACCESS.2019.2927465.
[55] "Keras: The Python Deep Learning Library." [Online]. Available: https://keras.io