Using Recurrent Neural Network for Hash Function Generation

Michal Turčaník

Departament of Informatics
Armed Forces Academy of gen. M. R. Štefánik
Liptovský Mikuláš, Slovak Republic
michal.turcanik@aos.sk

***Abstract* – Effective generation of hash function is very important for an achievement of a security of today networks. A cryptographic hash function is a transformation that takes an input and returns a fixed-size value, which is called the hash value. A recurrent neural network, as a possible approach, could be used for the hash function generation. The performance of the recurrent neural network (RNN) was validated by software implementation of RNN for given network configuration. The principles of recurrent neural networks and the possibility of using recurrent neural network for hashing are presented in the article. For analysis of RNN were created testing sets on the base of several examples of general texts. In the paper were tested RNNs with three layers.**

Keywords- hash function; recurrent neural network

I. INTRODUCTION

The spreading of the Internet connections over the world creates need to establish secure communications of the human. A very important part of the internet security is secure communications. Commercial communication systems are easy to use but they can be misused by adversary side. There are plenty of examples of attacks to communication infrastructure. Coordinated attack of opponent against friendly force during operations is one of them. Secure communication channel is key factor to achieve success not only in military operations.

II. SECURITY AND COMUNICATION

Evolution of new technologies goes together with human kind evolution. Communication technology is one of the most essential technological industries. People like to communicate from early day of human kind. Without communication it is difficult to imagine life and perhaps without it even any existed. Communication has changed through history and is still better and better. The amount of messages and volume of information transmitted by the technical means of the communication technologies is growing. The messages and information are represented not by word of mouth, but in the form of ones and zeros. They are transmitted by wire or wireless and they are much more vulnerable to abuse and they can be also captured. Information security as an answer to previous threats tackles the safe transfer, storage, data processing and manipulation. One specific branch of information security is cryptography.

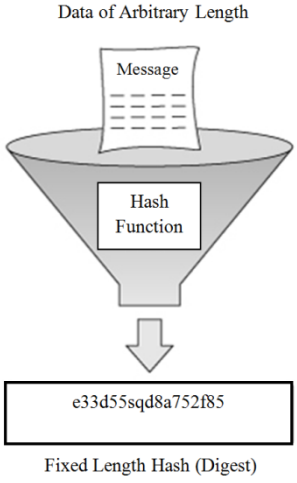

Figure 1. Hash algorithm.

A hash function transforms a data string into a numeric string output of fixed length. A hash function must be a one way function. The output string has generally shorter length than the original data (Fig. 1). This is the reason why hash algorithm is also called message digest or message compression algorithm. Hash algorithms are created to be collision-resistant. It means that there is a very low probability that the same output string would be created for distinct data. One of the most examples of used hash algorithms are the MD5 (Message-Digest algorithm 5) and the SHA-1 (Secure Hash Algorithm). The validation of data integrity after data files transfer or storing is realised by MD5 Message Digest checksums.

ISBN 978-80-261-0642-5,

III. PREPARE YOUR PAPER BEFORE STYLING

A function F is a one-way function if it is easy to compute on every input, but hard to recompute input value on the base of output. A definition of function F is generally known. To compute has function it's not required any secret information. In other word for any given x, it is easy to compute $F(x)$. For given y, in the range of F, it is hard to find an x such

$$F(x)=y \tag{1}$$

An efficient algorithm can be successful to solve a P-problem of inverting F with small probability. Mathematical prove of the one-way functions doesn't exist.

A cryptographic hash function must have several features. First of them is that hash function must be a one-way function. It must be also a collision-free function. Some hash functions are also a trapdoor one-way hash function.

A function H that maps an arbitrary length message M to a fixed length message digest is One-Way Hash Function. Message digest is a one-way hash function if it is a one-way function and for given M and $H(M)$, it is hard to find a message $M' \neq M$ such that $H(M')=H(M)$.

Collision-Free Hash Function is a function H that maps a message M with random length to a message digest with fixed length. The message digest must be a collision-free hash function and also a one-way hash function. It is difficult to create two different input messages (M',M) that have the same result of hash $H(M')=H(M)$. An efficient algorithm (solving a P-problem) could be created to find such a collision.

Trapdoor One-Way Function is following

$$F:\{0,1\}^{l(n)} \times \{0,1\}^{n} \rightarrow \{0,1\}^{m(n)} \tag{2}$$

Let's have function F which is a trapdoor one-way function. For given message M and message digest $F(M)$, it is difficult to find other message $M' \neq M$ for which message digests are the same $F(M')=F(M)$.

A trapdoor one-way function is a particular type of one-way function thanks to which computation is easy in one direction and almost impossible in the opposite direction. A trapdoor one-way function has secret information. On the base of the secret information it is possible to easy compute the function in the opposite direction. It is not so difficult to compute $F(M)$ for given message M, and hard to compute message M for given $F(M)$. However, there is some secret information, Y, such that given $F(M)$ and Y it is easy to compute M.

An attack resistance of the hash function is affected by length of message digest. Today one-way hash functions are typically using 128-bit hashes and more.

A very small change in the input string must cause the hash of the function to change dramatically. Even if only one bit of message is changed to opposite value in the input string, at least half of the bits of the message digest should flip as a result. This is called avalanche effect. Same hash value of two different messages almost doesn't have collision free or collision resistant functions. A hash of the document can be used as a cryptographic equivalent of the document. If different input messages create results with the same hash value hash collision happens.

Hashes functions are used for password hashing, file integrity checks and message authentication.

Sometimes unskilled computer users have only one password for several domains. After they lost password to one domain attacker can gain access to others. Better method from security point of view than storing open password in computer is storing a hash of the password.

After transmission of file the best way to verify file content it could be used hash function. Digital signature of e-mail communication is another application of hash function in real life.

IV. RECURRENT ARTIFICIAL NEURAL NETWORK

Artificial neural networks (ANN) are capable to realize complex functions in different areas (classification, identification, pattern recognition, speech, vision, and automation) [2]. Neural networks are composed from simple elementary units (described as a formal neuron) which work in concurrent manner. Network function is mostly defined by connections (synapses) between these units which have defined some weights. Learning of a neural network for specific function is done by setting of the values of the weights between units. Learning of artificial neural networks is realized by presenting a set of inputs and set of target outputs. [1].

The most known architecture of neural network is a perceptron. Parameters of the perceptron (weights, biases) are adjusted to generate a proper output vector to the equivalent input vector. The simplest one is a single-layer perceptron. The most interesting features of perceptron are generalization and pattern classification. The structure of feed-forward networks can consist of one or several hidden layers of neurons with sigmoid function and one output layer (linear function) [2].

Another class of artificial neural networks is recurrent neural network. Recurrent neural networks with feedback connections can be attractive choice for a number of reasons. The basic feature making recurrent neural networks fascinating and for some applications useful is that they have nonlinear dynamical behaviour, which can be time-varying.

Recurrent networks have a much bigger set of options to represent some internal state of the analyzed system. Recurrent neural networks can form delay time structures when it is needed because their representation of the internal state is versatile (not stable). They can be used to create basic memory elements or other more complex memory structures which can preserve a state for a user defined time period [3].

ISBN 978-80-261-0642-5, © University of West Bohemia, 2017

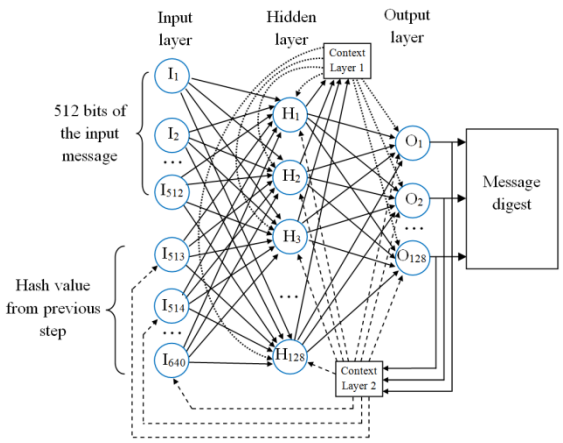

Figure 2. The structure of the reccurent neural network for hash function generation.

A recurrent neural network (RNN) is composed from three layers of neurons and synapses between them. Every neuron can be characterised by an input, activation and output function. Every synapse between two neurons has a value which is called weight. [4].

The final result is calculated by realization of computation which is done layer by layer. At the first, the input values are applied to the input neurons. In the input layer every neuron computes his output function. Computed values are sent to the other layer. The process is finished when the output layer is achieved.

The number of the input layer neurons is 512 and it corresponds to the 512 bits of input message which is hashed by recurrent neural network. The number of the neurons of the hidden layer is a target of the optimization. The number of the neurons of the output layer is 128 which represent the number of bits of the hash function.

The structure of the recurrent neural network for hash function calculation is shown in Fig. 2. The input to the recurrent neural network is represented by 512 bits segment of input data from message. Other group of input neurons is represented by 128 bits of hash function from previous calculation. These values will be used in the next time step. The number of neurons in the hidden layers is goal of the optimization.

The values of hidden neurons are stored in the first context memory. These values are then used in the next step as an input back to the hidden and output layer neurons. The values of output neurons are stored in the second context memory. These values are also used in the next step as an input to the input, hidden and output layer neurons.

After whole message is processed the output of the RNN consists of 128 bits of computed hash function.

V. Hash Function Calculation Steps

The calculation of the hash function is realised in three steps. First step begins with transformation of message into sequence of bits. This sequence of bits is split into 512 bits long blocks. If message has lower number of bits than 512 or after splitting of message a last block is lower than 512 bits padding is added.

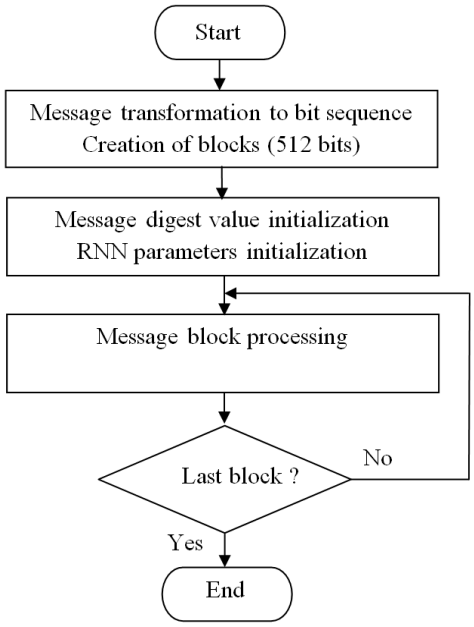

Figure 3. The algorithm of the hash function calculation.

The second step starts with generation of initial message digest value which is used as input for hash function calculation (128 bits). This operation is realized only in first step of calculation because we don't have any hash value yet. After realization of first step hash value is used in the next step of calculation together with next 512 bits block of the massage. In Fig. 2 you can see how 128 bits hash value is used as input to the neural network. The RNN parameters are generated in the in the same step and they are values of synapses between network layers and bias values of each neuron. Their number is not variable and it is not changed during using of the neural network.

At the last part of hash function calculation the message which is divided onto blocks is sent to the input of the recurrent neural network. The hash function computation is finished when all blocks of the message is processed. Last value on the output of the recurrent neural network is final hash function value. Hash function algorithm is shown on Fig. 3.

VI. Selection of RNN for Hash Function Calculation

To find recurrent neural network structure, an analysis of the setting of RNN from point of view of hash function suitable computation [5, 6] must be done. To find optimal structure of RNN several testing sets and several models of neural network were created.

The number of neurons in all three layers is same during using of RNN, because the size of input block and also the size of hash function are static. Testing sets were created on the base of 10 examples of general texts. The number of characters of testing texts is from 3 to 65 thousand.

The quality of hash function generation by a given RNN is evaluated on the base of these testing sets. In the next step the random change of characters in the analyzed texts was realized (5 and 10 characters were changed). The comparison of given RNNs is done on the base of number of changed bits after generation of

hash function for original and modified text. The higher values of changed bits represent better results for a given network. The results for analyzed RNNs are shown in the Table I.

Parameters of all analyzed networks were generated randomly. Parameters of RNN are weights of synapses and biases for whole RNN. No learning algorithm is used to train RNN so no training and validation sets are needed.

TABLE I. COMPARISON OF RANDOMLY GENERATED RNN'S FOR 5 AND 10 CHANGED CHARACTERS IN ANALYZED TEXTS

Randomly generated RNN	Number of changed bits in the hash value	
	5 changed characters in the text	10 changed characters in the text
1	112	116
2	128	128
3	124	128
4	116	116
5	100	104
6	92	92
7	116	120
8	124	128
9	128	128
10	128	128

In the Table I there are summarized results of using of RNN for hash function generation for all general texts. The value in the table represents average value of changed bits in hash value for all texts. The number of changed bits is from interval 0 to 128 which is maximal number of bits of hash value computed by RNN. The value is rounded to integer value without any decimal part.

The minimal number of changed characters in the analyzed texts is 92 bits for 6th RNN. The maximal number of changed bits in the analyzed texts is 128 bits for 2nd, 9th and 10th RNN.

VII. CONCLUSION

The structural complexity of the artificial neural networks creates big obstacles for their using in the real applications. The structural complexity ANN goes together with network structure. Optimisation of the ANN for actual application has of great significance. The structure and type of neural network is determined by given application so the analysis of neural network must be done from point of view of application criteria.

The hash function generation based on recurrent neural networks was presented in this paper. The principal aim was to build RNN for hash function generation which could be easily used in the modern communication systems. The core of the system for hash function generation was presented. Complete system would require application specific modules for a given technology.

Based on the results of RNN for hash function generation given proposal has no dependence on language and average number of changed bits of hash function with change of original text for 5 and 10 characters is more than 91 %.

The main advantage of using RNN for hash function generation is security. If we use same secret parameters of RNN (synapses and bias values) for sender and for receiver it is very difficult or almost impossible to reconstruct them from transferred messages. Very important is also speed of hash function generation because proposed solution can be realized in parallel manner. Behaviour of ANN also could be easily changed by random generation of new network parameters.

Presented artificial neural network was used for 128 bits hash function output. Change of the number of bits of hash function can be changed to 160, 192, 256 or other value. This will cause the change of the number of output neurons of the ANN and of course the number of hidden layer neurons. But this change won't be so huge. The real number of hidden neurons will be result of other ANN optimization and it is also possible way how to realize future research.

Common hash function eq. MD5, SHA-1, SHA-2 are using simple operation (rotations, and, or, xor, ...). Hash function generation by ANN is also based on simple operation. Computation of the partial value of the hash function in this approach is realised in several steps. Each step is represented by one layer of the ANN. So for three layers ANN we need 3 steps. Computation of all neurons on the each layer can be realised by parallel way. For today computers it is an easy task and so won't be so notable.

The ANN parameters (weights, thresholds, activation function, ...) must be transmitted to the all users of the hash function. Without these parameters is impossible to compute hash value. We can publish these parameters or keep them in secret. If we want use this hash function as secret we will use secure channel for ANN parameters transfer.

REFERENCES

[1] J. L. Elman, "Finding Structure in Time," Cognitive Science, Vol. 14, 1990, pp. 179-211.

[2] D. Pattreson, Artificial Neural networks-Theory and Applications, Prentice Hall, 1996.

[3] Y. Chauvin, and D. E. Rumelhart, "Backpropagation: theory, architectures, and applications" Hillsdale, NJ, USA, ISBN: 0-8058-1259-8, 1995.

[4] S. Melacci, L. Sarti, M. Maggini, and M. Bianchini, "A Neural Network Approach to Similarity Learning," Lecture Notes in Computer Science, Artificial Neural Networks in Pattern Recognition, June 30, 2008, pp. 133-136.

[5] I. Mokriš, and M. Turčaník,"Contribution to the analysis of multilayer perceptrons for pattern recognition," Neural Network World, Vol. 10, No. 6, 2000, pp. 969-982.

[6] M. Turčaník, and I. Mokriš,"Possible Approach to Optimization of Neural Network Structure," Proc. of Conf. SECON 97, Warsaw, 1997, pp. 326 - 335.

ISBN 978-80-261-0642-5, © University of West Bohemia, 2017