The Evaluation of Randomness of Integer Sequence: The Traditional and Machine Learning Approaches

Haoran Bai
University of Electronic Science and Technology of China
2017020911030@std.uestc.edu.cn

Huairui Chu
University of Electronic Science and Technology of China
2017020908026@std.uestc.edu.cn

Jing Liang*
University of Electronic Science and Technology of China
liangjing@uestc.edu.cn

Abstract: People nowadays care more about the justice of the lottery system than ever before, especially when it comes to topics related to people's livelihood including housing and car purchasing. Lottery systems have been widely applied in such administrative process in China. However, researchers payed little attention to address the fairness of such systems through systematic study. In this work, we attempt to study the justice evaluation system of the lottery system in Chengdu's housing market. We present our methods to approach this problem, both in traditional way by improving the Approximate Entropy and machine learning way based Long Short-Term Memory model.

1. Introduction

Nowadays, there exists a strong need of justice evaluation on multiple topics related to people's livelihood in society, above which the housing market has stood out for it consists of multiple stages and the ambiguity when defining randomness. In previous study, complexity of series has been studied with the intention to learn how to interpret randomness, tools and models including approximate entropy [1] have been proposed to tackle such problem, which has been applied to the application in medical research, such as schizophrenia [2] epilepsy [3] and addiction [4]. However, approximate entropy suffers from two main disadvantages, its dependence on record length and the lack of consistency [5]. Meanwhile, as machine learning merges with tools and models to solve the problems related to series analysis, there is little work that tries to remedy this in a machine learning way. In this work, we propose two methods to approach this problem, by introducing machine learning method to the area and improving the traditional way of how approximate entropy calculation is conducted.

2. Machine Learning Approach

The first method adapted in this work is through machine learning, our model is primarily based on LSTM [6].

2.1 Task

Randomness evaluation is the task of evaluating the randomness of a list of integers N with length of L.

2.2 Method

The method we adapted in this work is quite different from a traditional machine learning approach, three stages are evolved in this process.

The first stage is the training stage, where a training dataset is derived from the actual input sequence that is to be evaluated and fed into the model to train.

The second stage is the evaluating stage, where the trained model will be used to test the actual sequence and outputs a result vector O.

The last stage is the output stage, where the result vector O would be further manipulated and used to calculate the final output score of the evaluation system.

In conclusion, our method approaches the problem of evaluating sequence randomness by first training a model based on sequence being evaluated and predict the rest of the sequence that has not been seen by the model, the final score would be calculated based on the prediction vector (vector O) under the following intuition:

If vector O tends to be fluctuating and does not adapt any form of pattern, we say the sequence has a relatively higher degree of randomness.

If vector O tends to be adapting a form of pattern, we say

* Corresponding author

the sequence has a relatively lower degree of randomness.

2.3 Models

The model is straightforward with only a simple LSTM followed by an affine layer to output the result. There are 3 main modules in the presented model:

* A preprocessor that cut the input integer sequence into S smaller subsequences as the training data.

* A sequence encoder in f_{enc} that encodes the input subsequences into a feature vector $V = \{v_1, v_2, \dots, v_S\}$

* An output layer that converts features V into an output vector $O = \{o_1, o_2, \dots, o_{L+1}\}$

We employ LSTM as the subsequence encoder and a simple affine layer as the output layer.

2.4 Experimenting Setup

Sequence Preprocessing. In order to obtain enough training data to train on the input sequence, sequence has to be cut into subsequence to form a thorough training dataset, the subsequence must cover each location in the complete sequence equally. i.e., for the sequence N with length of 100, if we assign each subsequence with a length of 20, the training dataset would therefore consist the following subsequence:

$$x^1 = \mathrm{N}[0:20], x^2 = \mathrm{N}[1:21], x^3 = \mathrm{N}[2:22],$$

$$\dots$$

$$, x^{80} = \mathrm{N}[79:99], x^{81} = \mathrm{N}[80:100]$$

Implementation Details. Different from traditional machine learning task, this method does not require a thorough hyperparameters tuning process. We recommend setting parameters ahead of training process to maintain a equal-training process. Our experiment adapts the following values for an input of a sequence N with length L:

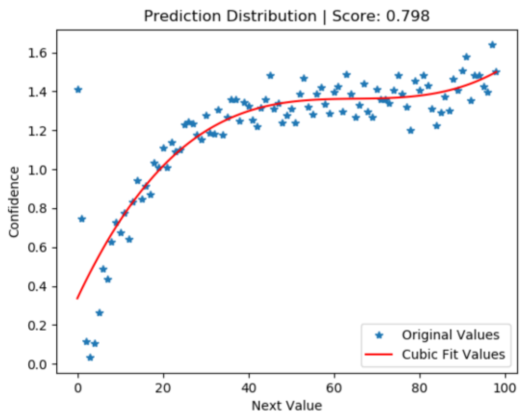
The sequence encoder is a LSTM with input size of 10 and hidden size L/2. The output affine layer has the input size of L/2 and output size of L+1. Model is trained with cross entropy loss, optimized with Adam optimizer [6]. Model adapts a batch size of 10, learning rate of 0.0001 and uses 20000/L epochs.

Evaluation Details. After acquiring the output vector, we first take the abstract value of the direct result and sample the interested point from the distribution, then we perform a linear fit to the distribution using a third power polynomial to smooth the curve and calculate the variance of the result. This could be further interrupted as the final output of the evaluation system.

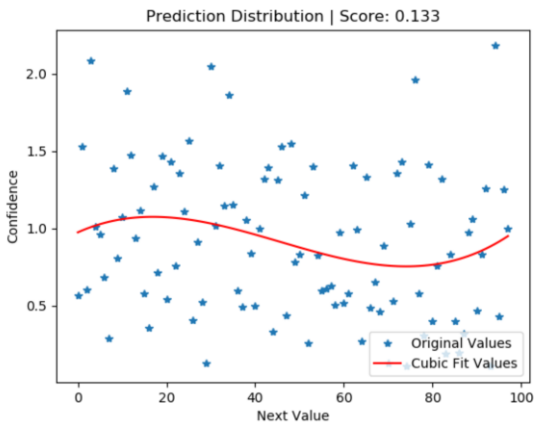
2.5 Results

We perform our test by manually changing the proportion of the random numbers within a test sequence that is originally sequenced. This way we obtain a series of test sequences with different levels of degree of randomness.

Specifically, our test sequences are derived from sequenced integer sequence with length of 1000, based on the methods above, we would be able to obtain a test set with the randomness degree varies from 1 to 20. Here we show the first (the least random) and last (the most random) results:



(a) Result distribution of degree 1



(b) Result distribution of degree 20

Figure 1: Demonstration of testing results. (a) A result showing the performance of the proposed model on a testing sequence with the smallest degree of randomness. (b) A result showing the performance of the proposed model on a testing sequence with the largest degree of randomness. Note that we adapted a polynomial fit to the original output data with a cubic polynomial.

2.6 Discussion and Limitations

Flexibility. The proposed approach and model have offered a new way of approaching the problem where machine learning method has not yet been introduced in related area, due to the flexibility in training a model, the model could cover more in terms of 'regular sequence'. This property is particularly helpful due to the ambiguity of defining a sequenced series and a random series, which turns out to be hard to capture by traditional model. Meanwhile, by adjusting the hyper parameter in the model, we could easily reshape the model to adapt to other related problems.

Reverse Machine Learning. By using the 'reverse' version of a typical machine learning approach formation, we proposed a different angle of approaching machine learning problem.

One seemingly limitation of the current model is the fact that LSTM has been uneasy to train hence the model is lack of consistency, that is, if the randomness of one testing sequence is higher than that of another, the corresponding score should, but does not always, remain smaller for all tested tests. This phenomenon could also be identified as a sign of lack of useful tools to interpret the result precisely. Also, to address the task, the model must adopt to a certain degree of complexity in structure, this could serve to be a disadvantage when it comes to a sequence that has shorter length. In our tests, I model has been unstable when length is less than 200.

3. Traditional Approach

In this section we introduce the other method. This method is an elaborated version of the previously purposed computational frame work [9]. The intuition of block comparing was proposed in [9] before. In this section, we first formulize the task and define some notions, then present our method by showing the procedure and the intuition behind it.

3.1 Task

A shuffled sequence L is a permutation of a series set of integers: 1, ..., N, where N is the length of the sequence. $L(1), L(2) \dots L(N)$ are used to denote each element in L. We propose a method to test the irregularity of such sequences. A block $B(i,j)$ is the subsequence extracted from the initial sequence, which is $L(i), L(i+1) \dots L(j)$. An exchange $E(L,i)$ is an operation for L that exchanges $L(i)$ and $L(i+1)$, we equivalently call it a basic operation. Some other mathematical definitions will be explained in the following subsections.

3.2 Intuition

This method is based on two intuitions. The first one is that a regular sequence is usually constructed by repeating some single rules, so that there are probably similar blocks in the sequence.

The other intuition is about how to define the similarity of two blocks, so that we can therefore evaluating similarity of blocks to exam the regularity of a sequence. We have noticed that the sequence is limited when it comes to the possibilities of all permutation, during which exchanging two adjacent elements accounts for a basic operation. If a permutation can be transformed into another one by applying a small number of basic operations, then these two permutations are probably similar.

However, there is a detail in the second intuition. We can't ignore the difference between similarity and regularity. In fact, if two permutations need a great number of exchanges to transform into each other, they will be considered not similar but predictable, which means regular. We exploit this in our method.

3.3 Method

To make the intuitions into application, we define a series of functions and transformations. The final evaluating method will combine them and give out a value denoting the regularity of the shuffled sequence.

To compare blocks in the basic-operation-counting way, we need to transform blocks into permutations, which always consist of same elements. For an arbitrary shuffled sequence L, $BT(i,j)$ is a permutation of $1, 2, \dots j-i+1$ obtained form $B(i,j)$. Let $P = B(i,j)$ that is $L(i), L(i+1), \dots L(j)$. Let P_s be the sequence that is the sorted version of P in ascending order. So P_s is $L(s_1), L(s_2), \dots L(s_{j-i+1})$ where $s_1, s_2, \dots s_{j-i+1}$ is a permutation of $i, i+1, \dots j$. We approach $BT(i,j)$ by replace each element in P by the index of the equal element in P_s. For example, if $s_k = m$, we replace $L(m)$ in P by k.

We define the distance between two permutations of $1, 2, 3, \dots, N$, L_1 and L_2, as the minimum number of exchanges needed to transform L_2 into L_1. We denote the number by $D(L_1, L_2)$. Trivially we have $D(L_1, L_2) = D(L_2, L_1)$. This number can be easily counted by calculating N values $GN_1, GN_2, \dots GN_N$. GN_i is the number of elements before and greater than $L_1(i)$ in L_2. Consider a procedure that first move $L_1(1)$ in L_2 to the 1st position, then $L_1(2)$ to the 2nd position…, until L_2 is completely transformed into L_1. After calculating $GN_1, GN_2, \dots GN_N$, we can get $D(L_1, L_2)$ by the following formula. We omit the proof of it.

$$D(L_1, L_2) = \sum_{i=1}^{N} GN_i$$

Now naturally we can define the distance between any two blocks $B(i_1, j_1)$ and $B(i_2, j_2)$ in L by $D(TB(i_1, j_1), TB(i_2, j_2))$. We shortly use $D(i_1, j_1, i_2, j_2)$ to denote that in the following context. We now explain how to apply distance to examine the regularity of the whole sequence. We've mentioned that intuitively, regular sequences have some blocks similar, or predictable by each other. We suppose to examine the predictability of the whole sequence when choosing each block in it as prior information. This value is denoted by $Pr(i, j)$.

To show the detailed function in $Pr(i, j)$, we need some more signs at first. If we randomly shuffle a permutation P with length N into P', there will be $N!$ Possible results. Let $x_{M/N}$ be the number of "P'" s that $D(P, P') = M$. Then we turn to the initial shuffled sequence L with length N. We choose a common length N_B for the blocks we observe. For each two blocks $B(i, i + N_B - 1)$ and $B(j, j + N_B - 1)$, $\frac{x_{D(i,i+N_B-1,j,j+N_B-1)/N_B}}{N_B!}$ is the probability that their distance is $D(i, i + N_B - 1, j, j + N_B - 1)$ when the whole sequence is a random shuffle of $1,2,3 \dots N$. We denote this probability as $Ps(i, j)$. Let $A(i)$ be a set that $A(i) = \{x | x + N_B - 1 \leq N, x \geq 0, x \equiv i (mod\ N_B), x \neq i\}$. Actually, $A(i)$ contains every start index of the possible blocks in L which differs from i in a multiply of N_B. $Pr(i, i + N_B - 1)$ is defined as follows. Note that this value is the probability that the distance between $B(i, i + N_B - 1)$ and $B(j, j + N_B - 1)$ is $D(i, i + N_B - 1, j, j + N_B - 1)$ for each j in $A(i)$ simultaneously when the whole sequence is a random shuffle.

$$Pr(i, i + N_B - 1) = \prod_{j \in A(i)} Ps(i, j)$$

For an arbitrary block $B(i, i + N_B - 1)$ in L, $\frac{1}{|A(i)|} \ln Pr(i, i + N_B - 1)$ is the answer of this question: how much can L be predicted by $B(i, i + N_B - 1)$. Finally, we choose the block that can predict the whole sequence best and its Pr value to present the regularity of the whole sequence Reg. And we take a logarithm to make the result clear.

$$Reg = -min_{1 \leq i \leq N+1-N_B} \left(\frac{1}{|A(i)|} \ln Pr(i, i + N_B - 1) \right)$$

The greater the Reg value is, the more regular L presents.

4. Experiment & Analysis

4.1 Traditional Approach

4.1.1 Validity Verification

In this section, we test the effectiveness of our method. In fact, it is not easy to test as there is no common standard of regularity. We construct a series of shuffled sequence which seem to be more and more irregular and apply our method on it. To construct a shuffled sequence L with length N, the only difficulty is that each number in $1,2, \dots, N$ must appear and only appear once. A natural way is to choose a number N_p co-prime with N. Then construct L as follows.

$$L(i) = [(i - 1) \times N_p\ mod\ N] + 1$$

For experiment, we constructed 9 sequences with $N = 300$ and $N_p = 1,7,13,19,29,37,59,89,131$ respectively. Note that when $N_p = 1$ the sequence is the ascending order of positive integers. And we added a sequence which is randomly shuffled by the random permutation function in Python[2]. Their Reg value are demonstrated in figure 2. We set $N_B = 6$ for this calculation.

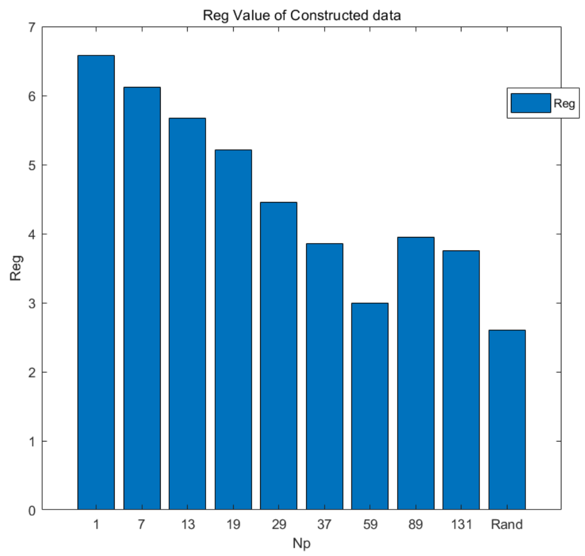


Figure 2: Reg values of Constructed data. The last value is the Reg value of the randomly shuffled sequence.

The value is decreasing when the N_p value increases, which matches our intuition. It confirms our method. Note that the a local minimum value appears when $N_p = 59$, but the

[2] This kind of functions shuffles the input with equal probability for each permutation. They can be found in C++, MATLAB and so on. The randomly shuffle function mentioned below refers to functions with this characteristic but not limited in functions in Python.

value is still greater than the randomly shuffled case by 0.393908. We will show a method to judge whether a sequence is completely irregular by comparing with randomly shuffled sequences. This sequence will be defined not completely irregular in the judging method.

4.1.2 Advantages & Limitation

Parameters. The only parameter in this method is N_B which denotes the length of windows. After setting this parameter, this method is effective for comparing shuffled sequences in different length. For the same sequence, N_B will influence the value Reg. So comparing two Reg values of two sequences with different N_B value setting makes no sense. We suggest that N_B can be set to $3 \sim 0.05N$ so that it will not be trivial and the number of blocks is enough. A test about how N_B influence a shuffled sequence with length of 300 is shown in figure 3.

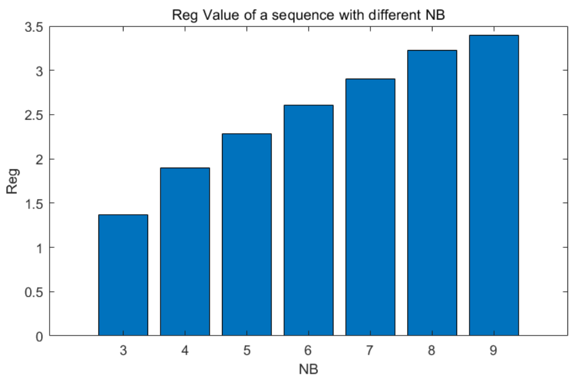


Figure 3: Reg Value of a Sequence with Different N_B

Our method is more suitable for shuffled sequence while the Approximation Entropy method has an extra parameter denoting the threshold and the parameter is hard to determine when comparing shuffled sequences with different lengths.

Independence with length. This method is applicable for comparing sequences with different length. To verify this, we find a series of sequences with intuitively similar degree of regularity but different lengths. They are the sequences with length of $100,200,300\ldots,2000$ produced by random permutation function. For each length we produced 10 samples, the average Reg value for each length is demonstrated in figure 4.

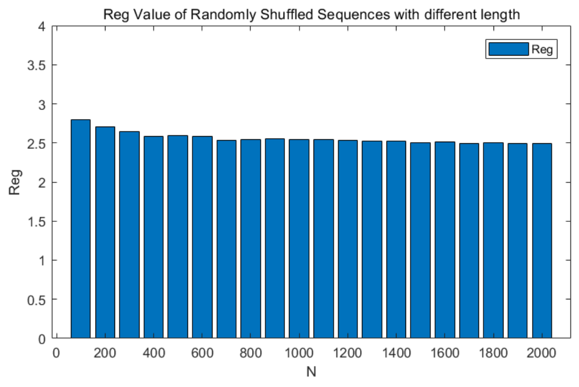


Figure 4: Reg Value of Randomly Shuffled Sequences with Different Length

As we can see, the average value goes slightly smaller when the sequence goes longer. This is because longer sequences are more complex, so they will be more irregular. The value seems to converge when the length goes very large and this also convince the validity of our method in some way.

4.1.3 Discussion

This method calculates a probability with specific meaning, and the intuitions about block comparing and exchanging adjacent elements can be used to construct more models about sequence processing. A feasible modification of this method is to calculate the distance of two blocks in two different sequences. This will develop a model for testing the similarity (if simply by calculating the distance between blocks) or predictability (if calculating the probability that the certain distance value appears) between two sequences.

A shortage of this method which can be considered in the future is that there still exists a parameter that is the block length. We didn't come up with a strategy to eliminate this parameter. Fortunately, although the parameter influence Reg value, we can set the parameter to a fixed value in application and what this value is doesn't influence a lot. We reset $N_B = 5,7,8$ in the validity test and it still works. The result is shown in figure 5.

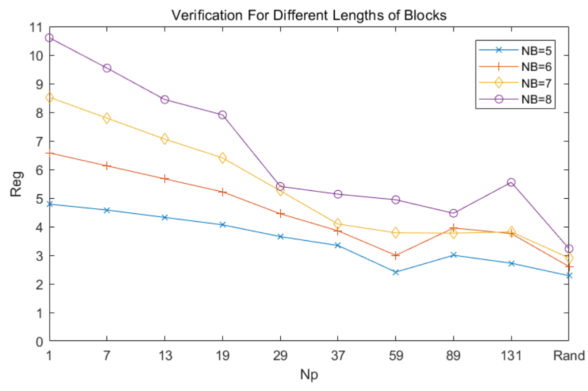


Figure 5: Verification for Different Lengths of Blocks

The result shows that for any proper N_B value, the Reg value decreases when the sequence becomes more and more irregular, which means our method is still valid.

5. Application

Lottery system for house purchasing has been a major concern of people in need of buying a housing estate. Briefly speaking, although lottery system in China nowadays enjoys a quite complex process, which consists of different types of housing source and involves the identities of purchasers. Nevertheless, the lottery system can be break down into smaller pieces, which turns out to be a system that performs a task of generating random sequence.

To simplify this application, we only consider a lottery system that performs the simple task as follows:

Generating a sequence of integer numbers with length of L starts from 1 to L with no repeated elements in the result sequence. The goal is to achieve certain degree of randomness within the sequence produced.

We performed our two methods on the open sources of the lottery system results in Chengdu housing market in the year of 2019[10], the result of which has been shown below.

5.1. Application

We selected 15 lottery system outputs, each of which is a shuffle of a sequenced series of participant numbers.

5.1.1 Traditional Method

To be able to make a conclusion about whether the lottery system is fair, we are supposed to test whether the shuffle it produces is irregular. We approach this objective by comparing those 15 shuffles with randomly shuffled sequences. We assume that if the system produces irregular shuffles, it must be at least better than the output produced by random shuffle function. So, we estimated the expected Reg value of the randomly shuffled sequences and their standard deviation with the same length as each tested sequence of lottery result respectively. If the Reg value of a sequence of lottery result is greater than the expected Reg value plus its standard deviation of the randomly shuffled sequence of the same length, we conclude that this sequence is regular. Here, for each length, the Reg value of the lottery result, and the randomly shuffled sequences are demonstrated in figure 6. For each length we test 10 randomly shuffled sequences to estimate the expected Reg value and its standard deviation.

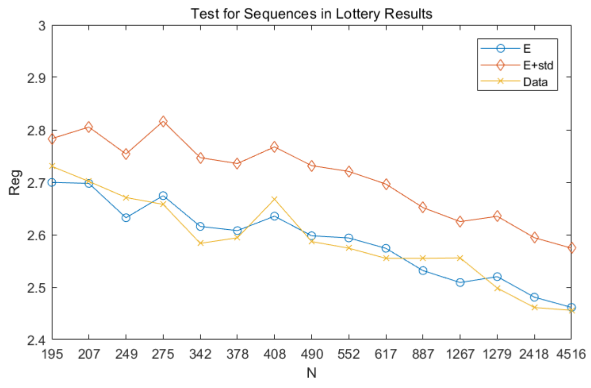


Figure 6: Application of Traditional Method. E denotes the expected value and std denotes the standard deviation.

The results show that each sequence in the lottery results is irregular. So, our conclusion is that the lottery system has been fair.

5.1.2 Machine Learning Method

For the purpose of testing the machine learning method, we apply the evaluation process to the real-life data, the same as 5.1.1 section, along with a traditional shuffle algorithm documented by Python, and compare the result of two shuffle outputs. The result has been shown below:

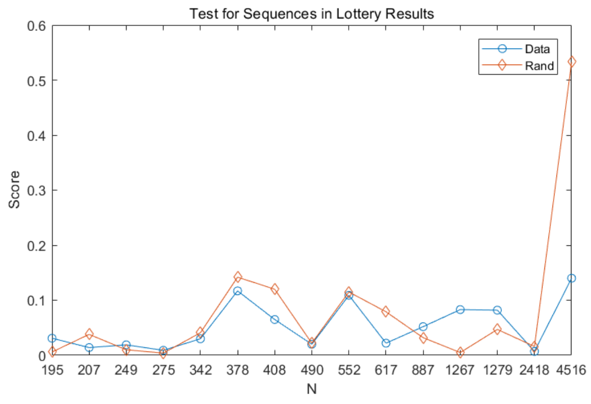


Figure 7: Application of the machine learning approach. The result demonstrates the comparison between 2 algorithms, the curve labeled with Rand refers to the algorithm by Numpy

documentation while the curve labeled Data has been the shuffle results provided by real estate developer. Notice that the smaller the score is, the better randomness the algorithm achieves.

In this application, we may conclude that most of the lottery systems are fair in possessing the randomness in their outputs.

References

[1] Pincus, S. M., Gladstone, I. M., & Ehrenkranz, R. A. (1991). A regularity statistic for medical data analysis. *Journal of clinical monitoring*, 7(4), 335-345.

[2] Sabeti, M., Katebi, S., & Boostani, R. (2009). Entropy and complexity measures for EEG signal classification of schizophrenic and control participants. *Artificial intelligence in medicine*, 47(3), 263-274.

[3] Yuan, Q., Zhou, W., Li, S., & Cai, D. (2011). Epileptic EEG classification based on extreme learning machine and nonlinear features. *Epilepsy research*, 96(1-2), 29-38.

[4] S.Hochreiter and J.Schmidhuber. Longshort-termmemory. Neural Computation, 9:1735–1780, 1997.

[5] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[6] Yun, K., Park, H. K., Kwon, D. H., Kim, Y. T., Cho, S. N., Cho, H. J., ... & Jeong, J. (2012). Decreased cortical complexity in methamphetamine abusers. Psychiatry Research: Neuroimaging, 201(3), 226-232.

[7] Richman, J.S.; Moorman, J.R. (2000). "Physiological time-series analysis using approximate entropy and sample entropy". American Journal of Physiology. Heart and Circulatory Physiology. 278 (6): 2039–2049. PMID 10843903.

[8] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[9] Pincus, S., & Singer, B. H. (1996). Randomness and degrees of irregularity. *Proceedings of the National Academy of Sciences*, 93(5), 2083-2088.

[10] cdgzc.com