Analysis of anomalies in random permutations using recurrent neural networks

1st Fabian Fries
Fachbereich Technik
Hochschule Trier
Trier, Germany
fabianfries93@googlemail.com

2nd PROF. Dr. Ernst Georg Haffner
Fachbereich Technik
Hochschule Trier
Trier, Germany
e.haffner@hochschule-trier.de

***Abstract*—This paper is about detecting the difference between fully-random and semi-random shuffleing data sets, with the use of unsupervised learning algorithms. Because of the limits of the k-means algorithm alone, a recurrent autoencoder is used for feature extraction to improve the results of k-means. In the next step the autoencoder alone is used for clustering.**

I. Introduction

In the last years, machine learning has been used more and more in different areas and it is also appropriate for for pattern recognition in data. Random data is characterized through the missing of defined patterns. Permutations without repetitions have the highest amount of entropy for a sequence of its length, which is similar to random data according to Andrei Kolmogorov [4], who states that random data have the highest amount of information and can't be compressed. Therefore, this paper analyses the difference between random permutations and good shuffled permutations, which have some remaining patterns left. This is done via a recurrent autoencoder. [1]

II. Data to cluster

A. Generation of data

In the first step, two data sets of permutations are created, where the clustering is performed. The first set is created with the fisher-yates-shuffle [2] to generate random permutations. It should be mentioned that these are also only determined by a deterministic system, so they are actually semi-random permutations. However, since the models used do not crack the underlying pseudo-random generator, their results are random from their point of view, which is why they will continue to be referred to as random permutations in the further course.

The other dataset are created by shuffling the permutations along the lines of a deck of cards. The Riffle Shuffle is used for this, in which the deck is divided into two stacks and these are then alternately interleaved by both hands. This is described as an algorithm with the Gilbert-Shannon-Reeds model [3]. A deck of 52 cars is used and the point where the deck is split in two sets, left (L) and right (R), is determined with a binomial distribution Bin(52, 0.5). Then, the two sets are merged together by building a new deck through drawing cards from one of the two sets, where the next card is determined with a probability distribution depended of the size of the right and left sets. So, the probability that the next card is chosen from the left set, is $P_L = L/(L+R)$ and the probability for the right set is $P_R = R/(L+R)$. This algorithm can model the human riffle shuffle really well.

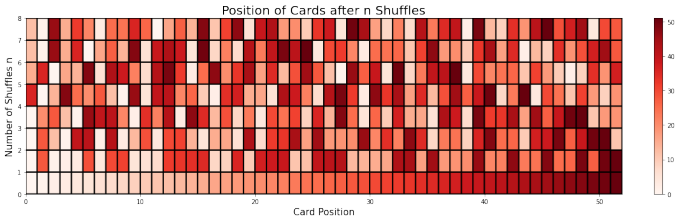

Fig. 1. Iterations of the Riffle Shuffle

Figure 1 shows, that the riffle shuffle is a strong shuffle, which can randomize the deck with 6-8 shuffles (depending on the definition of random). [7] [8]

B. Measurement criteria for absence of randomness

Nature of randomness disallows in general an exact measurement for a single sample, because this would imply that randomness could be mathematical described. But it is possible to calculate different types of absence of randomness. Andrei Kolmogorov states that a sequence is random, if it is not possible to compress the sequence into a program shorter than the sequence itself [4]. A random sequence has the highest possible amount of information. The redundancy score bases on the entropy, which indicates the information content of a sequence [5]. It is calculated with:

$$R = 100 \cdot \left(1 - \frac{H_{single}}{H_{max}}\right) \tag{1}$$

with:

$$H_{single} = \log_2(n) - \frac{1}{n}\left(\sum n_i \log_2(n_i)\right) \tag{2}$$

and:

$$H_{max} = \log_2(a) \tag{3}$$

where n is the number of different random responses in the sequence, n_i is how often the ith response alternative appears in the sequence and a is the maximum possible amount of different alternatives in the sequence.

Because a permutation without repetitions has the maximum amount of information for a sequence of this length, the redundancy score is not calculated for the permutation, but for each index of the set of permutations and then the mean is calculated. If a random response appears more often in a certain index, absence of randomness can be derived.

This can't be the only measurement criteria, because it is possible, that each random response has the same probability at each index, but the sequence of the permutation is nevertheless not random. Therefore, the random number generation score (RNG-Score) is used, which indicates the dependencies between one random variable and its following random variable in the sequence [6]. All these possible response pairs, so called digrams, are count and tabulated in a $a \cdot a$ matrix, where n_{ij} is the number of appearances of a digram where i follows j and n_i is the number of appearances of the single random variable. With this, the RNG-score can be calculated as follows:

$$RNG = 100 \cdot \frac{\sum n_{ij} \log(n_{ij})}{\sum n_i \log(n_i)} \tag{4}$$

Because in permutations without repetition each digram would appear only once, this is calculated for the whole set of digrams. Permutations without repetitions the score still can't show the whole range form 0 to 100. Therefore, another measurement criteria is used, to calculate the absence of randomness from the sequence of random variables. The adjacency score calculates the ratio of the original digrams n_a to the total number of digrams n_r:

$$A = 100 \cdot \frac{n_a}{n_r} \tag{5}$$

This is a good indicator how good a permutation is shuffled from the original sequence.

C. Measurements

For each number of riffle shuffle iterations a set of permutations is made, also a set of total random permutations with the fisher-yates-shuffle is generated. Figure 2 shows how the probability for card zero at the index zero decreases with each iteration of the riffle shuffle.

Number Shuffles	Redundancy	RNG-Score	Adjacency-Score
1	18.66	78.06	50.01
2	7.14	64.86	24.99
3	1.72	59.57	12.48
4	0.49	57.92	6.57
5	0.178	57.47	3.93
6	0.093	57.37	2.82
7	0.069	57.34	2.34
random	0.064	57.34	1.98

TABLE I
SCORES OF DIFFERENT SHUFFLE ITERATIONS

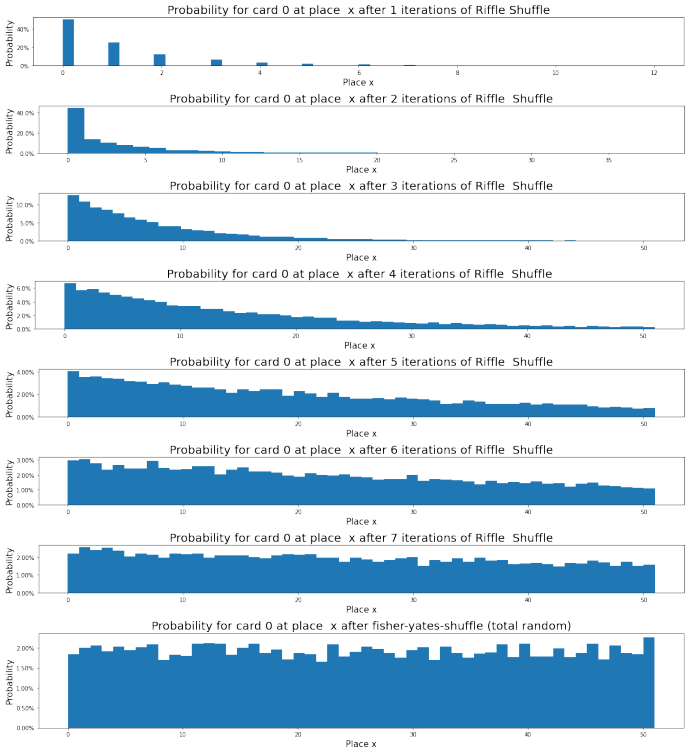

Fig. 2. Probability for the first card after x iterations of riffle shuffle

This can be seen in the redundancy score, which also decreases with each iteration in table I. Also the RNG score and the Adjacency decreases with each shuffle iteration. This shuffle process can be seen in Figure 1.

III. Recurrent Autoencoder

The k-means algorithm alone can only distinguish semi-random and random patterns when the redundancy differs a lot. It comes to its limits, when each random variable has a similar frequency for each index even if the sequence is shuffled really bad. Therefore, an unsupervised learning algorithm is needed, which is able to cluster the data sets based on the sequential order of the permutations. For this task a recurrent autoencoder is used like in figure 3. This is special architecture of an artificial neural network which consists of two connected networks. The first network is the encoder, which learns to compress the input data, and the second network, the decoder, learns to restore the original input data from the compressed encoder output. The autoencoder gets the permutation as x-data and y-data in the training process. According to Kolmogorov random data can't be compressed, so the encoded data from the random permutations should differ from the encoded data from the shuffled semi-random permutations. Thus, the k-means algorithm can distinguish between them.

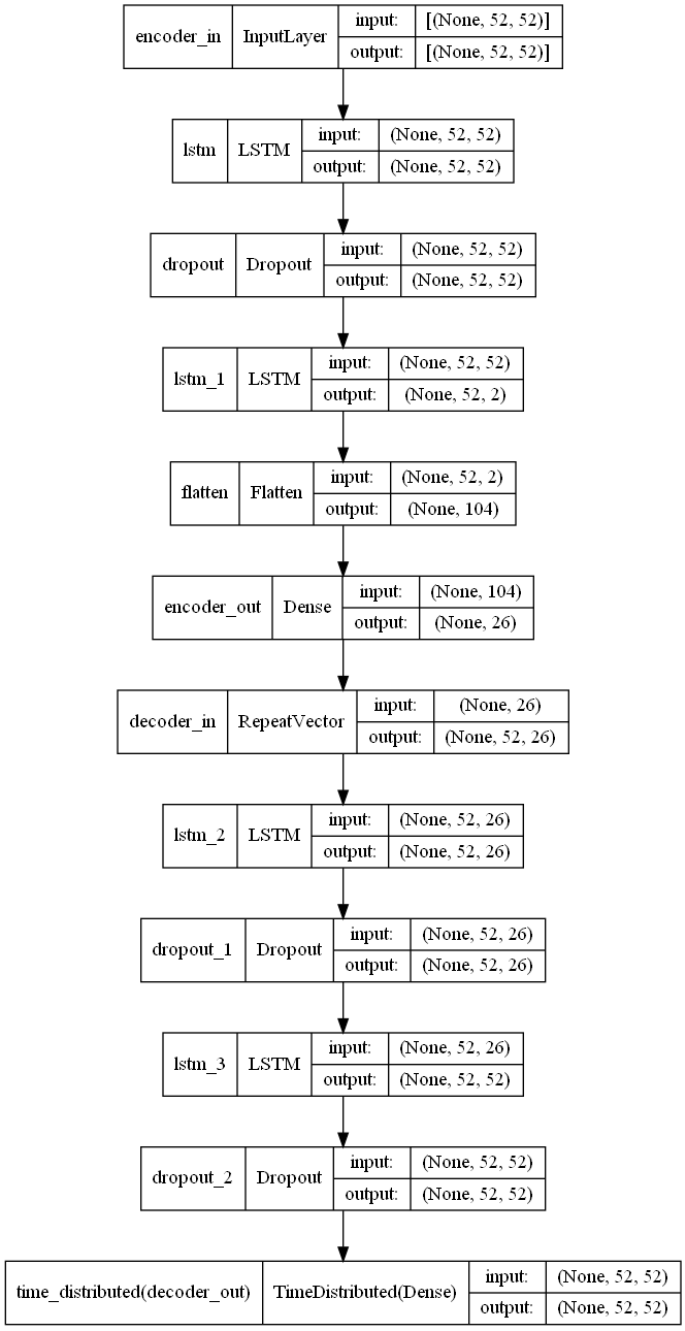

Fig. 3. Autoencoder architecture

Both, the encoder and the decoder consist of two LSTM-Layers, which bring knowledge of the sequential order of the permutations. The sequential data is flattened and compressed for the encoder output. The autoencoder model is trained with a data set, which contains half each random permutations and three times shuffled permutations and the k-mean algorithm is trained with the encoded results from this model.

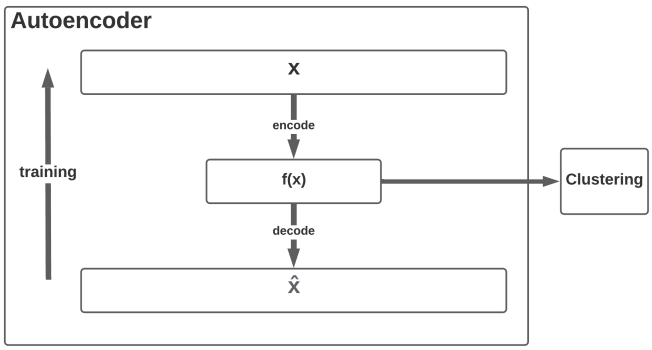

Fig. 4. Autoencoder

In the next step they are tested for a data set which contains half each random permutations and half shuffled permutations. This is done for different numbers of shuffles:

Number Shuffles	Accuracy k-means alone	Accuracy k-means with autoencoder
1	97	97
2	82	92
3	65	81
4	52	60
5	52	57
6	50	55
7	51	51

TABLE II
CLUSTER RESULTS

This model can distinguish differences between random permutations and shuffled permutations very good in the first three shuffles and then it recognizes still differences at four and five riffle shuffles, while the k-means alone can't see a difference. This shows that recurrent autoencoders improve the clustering for sequential data.
The next step shows, that clustering without the k-means and only with the auotencoder is possible. Therefore, the dense layer of the encoder output is reduced from 26 units to one unit with a sigmoid activation function.

The idea is, that the encoder only can show if and how much the input signal is compressible or has an non random looking order. The histogram of the encoder output in figure 5 shows that this is possible, and the encoder alone can distinguish between random and semi-random permutations. The less random a permutation is, the higher is the possibility that the encoder output is near the number one and the random permutations have a higher possibility to have a encoder result near zero. The encoder is a small, but not perfect, indicator for the absence of randomness.

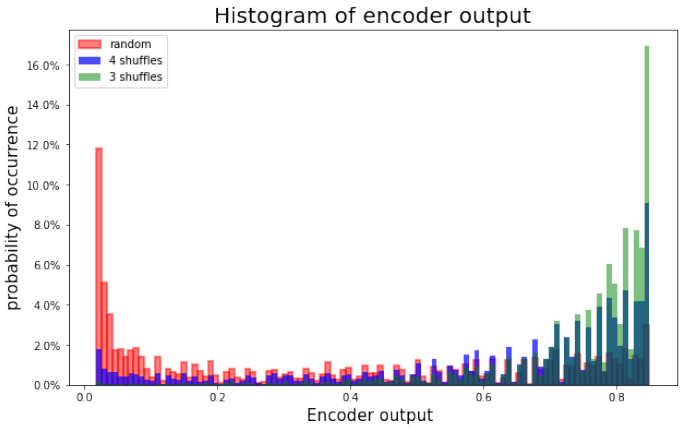

Fig. 5. Histogram of encoder output with one output unit for riffle shuffles

IV. Conclusion

These results proof that, that recurrent autoencoders can be used to distinguish between random and semi random permutations and that they achive more knowledge of the sequential order of permutations to cluster them with the k-mean algorithm. Also they can be used without the autoencoder to distinguis between them.

References

[1] Fabian Fries, Analysis of anomalies in permutations using recurrent neural networks and clustering algorithms after feature extraction with autoencoders, Masterthesis , University of Applied Sciences, Trier, 2022

[2] R. A. Fisher and F. Yates. Statistical Tables for Biological, Agricultural and Medical Research. OliverAndBoyd, 1938.

[3] Edgar Gilbert. Theory of shuffling. Technical report, Technical memorandum, Bell Laboratories, Murray Hill, 1955.

[4] Andrei N. Kolmogorov. On tables of random numbers. Sankhya, the Indian Journal of Statistics, Series A 25, 1963.

[5] John Towse and Derek Neil. Analyzing human random generation behavior, Behavior Research Methods, Instruments, Computers 30, 1998.

[6] Frederick J. Evans. Monitoring attention deployment by random number generation: An index to measure subjective randomness. Bulletin of the Psychonomic Society, 38, Jul 1978

[7] D. Bayer and P. Diaconis. Trailing the dovetail shuffe to its lair, The Annals of Applied Probability, 2(2):294-313, 1992.

[8] L. N. Trefethen and L. M. Trefethen. How many shuffes to randomize a deck of cards?, Proceedings: Mathematical, Physical and Engineering Sciences, 2000.