International Journal on Advanced Science Engineering Information Technology

Vol.11 (2021) No. 6
ISSN: 2088-5334

Generation of a Synthetic Dataset for the Study of Fraud through Deep Learning Techniques

Marco Sánchez [a,*], Verónica Olmedo [a], Carlos Narvaez [a], Myriam Hernández [a], Luis Urquiza-Aguiar [b]

[a] *Department of Informatics and Computer Science, Escuela Politécnica Nacional, Quito, 170517, Ecuador*
[b] *Department of Electronics, Telecommunications and Information Networks, Escuela Politécnica Nacional, Quito, 170517, Ecuador*
Corresponding author: [*]marco.sanchez01@epn.edu.ec

***Abstract*— Fraud is defined as any purposeful or deliberate act including cunning, deception, or other unfair means to deprive someone of property or money. Nowadays, fraud-related activities are growing at a dizzying rate, causing substantial economic losses every year. For an adequate analysis of this phenomenon, it is necessary to have data that evidences this behavior. Even so, given that these data are scarce and difficult to find, generating synthetic data for their study is a viable option. We designed two algorithms to generate text to create a synthetic data set that allows fraud analysis. These algorithms rely on the Fraud Triangle Theory proposed by Donald R. Cressey and use Recurrent Neural Network (RNN) and Long Short-Term Memory Networks (LSTM), respectively. The datasets generated were analyzed from the semantic point of view, giving a score about their readability and grammar consistency. The results obtained from this evaluation indicate that the data generation architecture proposed using the LSTM algorithm provides better performance in sentence readability (efficiency greater than 70%) than RNN (less than 40%). With LSTM, it was possible to synthesize a comprehensive data set related to the fraud triangle's vertices. This will make it easier to investigate fraudulent actions that are linked to human behavior. We will present a fraud predictor system based on machine learning techniques in the future.**

***Keywords*—Fraud triangle theory; machine learning; deep learning; LSTM; RNN.**

Manuscript received 29 Jan. 2021; revised 2 Apr. 2021; accepted 18 Aug. 2021. Date of publication 31 Dec. 2021.

I. INTRODUCTION

Fraud includes any intentional act to deprive another of property or money through cunning, deception, or other unfair acts. According to the Association of Certified Fraud Examiners (ACFE), fraud is defined as using one's occupation for personal enrichment through the deliberate misuse or misapplication of the employing organization's resources or assets [1]. The Fraud Triangle Theory explains the factors to be considered within human nature that create fraud conditions. Cressey [2], a leading expert in criminal psychology, investigates the reasons behind the question of why do people commit fraud? Besides, he determines that fraud commitment is motivated by the following three elements: pressure, opportunity, and rationalization, which must be present consecutively to provoke the desire to commit fraud. Based on his research, he introduces the concept "Fraud Triangle Theory." Information with evidence of fraudulent activities associated with the fraud triangle, in which communications related to pressure, opportunity, and rationalization are observed, is incipient in the scientific community, except for studies carried out by private entities such as the Federal Bureau Investigation (FBI) and ACFE. They have managed to obtain data related to these topics from their investigations.

Fraud-related data is necessary to generate fraud mitigation strategies. Violations of copyright and intellectual property have limited the availability of actual datasets. A valid option for obtaining fraud data is synthetic data generation due to the difficulty in obtaining this sensitive information. According to many experts, synthetic data is the key to make ML and AI faster and their algorithms more accurate in their predictions of fraudulent behavior, especially when real data is expensive to obtain or difficult to access [3].

Therefore, we will analyze deep learning techniques to show the application of the Long Short-Term Memory (LSTM) and Recurrent Neural Network (RNN) algorithms, commonly used for the generation of specific synthetic datasets, practically and efficiently. The datasets obtained will be subjected to performance tests. Specifically, they will be compared using the Readability tool, which evaluates the text's coherence and provides score values between 0 to 100.

Afterward, we will apply the arithmetic means to all the fraud triangle vertices (Pressure, Opportunity, and Rationalization) to identify the most accurate and efficient algorithm.

Many areas of study use synthetically generated data, from data mining to software engineering and artificial intelligence. For example, Demillio and Offut [4] presented a fault-based technique to create data for a software module's unit tests. In the field of evolutionary computing, it is also possible to find works on genetic algorithms to generate data suitable for tests [5], [6].

Albuquerque *et al.* [7] presented a framework for generating high-dimensional data. Using the graphical interface, the user can build a unified database for his application through statistical distributions with user-defined properties. Wang *et al.* [8] present a new approach to generating synthetic data, in which the user designs the desired data by hand, and the system calculates the generator model from the user's designs. Kwon *et al.* [9] used drawing interactions to direct the visualization of high-dimensional data according to the users' domain knowledge. Liu [10] created a synthetic data generator to evaluate the learning of classification rules. Similarly, researchers have proposed database synthesizers to analyze data mining tools [11]–[13]. However, these generators are specific to a problem tool or context, limiting large-scale use in other areas. Other works created a free dataset generation system for a wide range of areas as an alternative to systems from market applications [14].

Some approaches are dedicated to generating synthetic network data [15] [16]. For example, Brodkorb proposed a network data generator with geographical locations attached to nodes. In this way, the generated data can be displayed interactively on a map, where the user can explore the developed network and adjust the results later. Can Yang et al. [17] propose a custom channel recommendation framework with dynamic data provisioning through deep learning of historical channel switching sequences in systems IPTV Internet Protocol Television (Internet Protocol Television), for the dynamic generation of a list of recommended channels for each user through an LSTM Long Short-Term Memory network (Long-Short Memory Networks Term).

Regarding multimedia applications, Hosler *et al.* [18] use Convolutional Neural Networks (CNN) plug-ins to merge activations of multi-patch neurons to represent the camera model´s identification at the video level. So, the video authentication and camera identification data are used as a training base to generate a carefully constructed collection of videos to develop and evaluate algorithms to identify video camera models. Zhou *et al.* [19] create a dataset that includes fixations of 10 observers on 1,900 images degraded by 19 types of transformations. It uses the latest data on the transformed images, called data augmentation transformation (DAT), to train deep salience models, Altaheri *et al.* [20] use techniques based on deep learning to classify the fruit according to the date of harvest with an accuracy of 99.2%. They create a dataset of four types of dates by using the Google search engine. Furthermore, they propose a real-time machine vision framework for fruit picking robots based on the picking date in an orchard environment based on deep learning.

Regarding medical applications, Argha *et al.* [21] used machine learning techniques to estimate systolic and diastolic blood pressure (SBP and DBP). They design a deep neural network (DNN) classification model to extract artificial features to estimate SBP and DBP. Zhu *et al.* [22] create a large-scale biomedical key phrase dataset to assess system performance. The semantic web results are merged into the biomedical dataset to participate in the neural network training process, and further related information is considered to generate key phrases. This proposal is the closest to the research topic but with a different approach.

II. Materials and Method

This section provides an overview of three essential components to generate a synthetic dataset for the study of fraud: The fraud triangle, the advantage of synthetic datasets, and a short review of neural networks as a tool to generate new text.

A. Fraud Triangle Theory

Criminal psychologist, Cressey [23] proposes a model that explains the possible factors that cause someone to commit fraud. This theory, known as the Fraud Triangle, comprises three elements: pressure, opportunity, and rationalization (Fig. 1), which determine possible fraudulent behavior. Combining these three factors, known as pressure, opportunity, and rationalization increases the probability of committing fraud. There must be a "pressure" or "incentive" commonly related to financial or other needs to commit fraud. When conditions are right, there is the "opportunity" for fraud to occur. A lack of security, weak internal control systems, or unclear policies are examples of situations that lead to these circumstances. Commonly people can "rationalize" committing a fraudulent act [33].

Some people possess an attitude, character, or set of ethical values that allow them to commit a dishonest act knowingly and intentionally. However, even honest individuals can commit fraud in an environment that places sufficient pressure on them. The greater the incentive or stress, the more likely a person will be able to rationalize the acceptability of committing fraud.

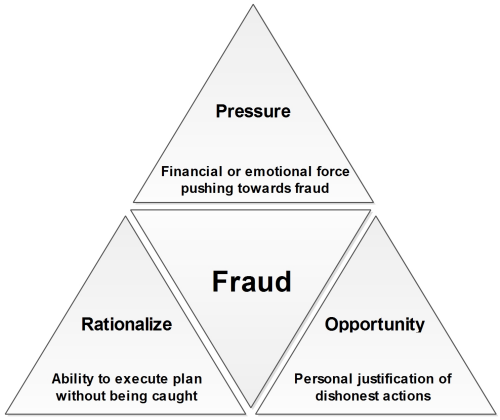

Fig. 1 Fraud Triangle Theory proposed by Donald R. Cressey (Pressure, Opportunity, and Rationalization)

To obtain data from frauds is challenging because most of the information is kept confidential. Therefore, *synthetic datasets* must be considered to develop tools to prevent this crime.

B. Synthetic Dataset

Rubin [34] was the one who initially proposed the concept of synthetic data through the multiple imputations of a complete set of data, with the objective that no real data would be published. As an alternative to this proposal, a method was introduced that replaces only some observed data characteristics, called partially synthetic data. Because the fully and partially synthetic data lacks original data, it is much less likely that sensitive information will be revealed compared to the original data [35].

The main purpose of a synthetic dataset is to be versatile and robust enough to be useful in training ML models, as the term "synthetic" suggests the *synthetic datasets* are generated through computer programs rather than being made up of documentation of real-world events. The creation of these is more profitable than the data collection of the real most of the time since it minimizes the time, cost, and risk of the operations. Besides, some research shows that it is possible to obtain the same results using *synthetic data* as real-world data [24].

An essential feature in generating a synthetic dataset is to guarantee the proper representation of all types of elements. Therefore, we need a tool to balance a dataset. Random subsampling Pine aims to balance the distribution of classes by randomly eliminating examples of majority classes. The goal is to balance an unbalanced dataset. The main drawback of random subsampling is that this method can discard potentially useful data that could be important for text generation, which has to do with training a logic model representing a known dataset. As long as the sample is obtained at random, it can be used to estimate the distribution of data where they were obtained. Therefore, by learning from the sample, it is feasible to approximate the target distribution. However, once we subsample the majority class, the sample can no longer be considered random.

C. Neural Networks

Deep learning (DL) is a subfield within machine learning, inspired by the biological process of neural networks that outperforms conventional deep learning algorithms. DL is a computational model composed of multiple layers of processing that learn from data representations with multiple abstraction levels. It extracts more abstract features from a more extensive training data set, mainly without human supervision [26]. Deep neural networks (DNNs) can perform a profound hierarchical transformation of input data. As a result, they have been found to have better performance and more rendering power than shallow neural networks [36].

Neural Networks (NN) are computational models that emulate humans' specific characteristics, such as memorizing and associating facts. They are nothing more than an artificial and simplified human brain model, which is the perfect example of defining a system capable of acquiring knowledge through experience [27].

1) Recurrent Neural Network: Recurrent Neural Network (RNN) appeared in the 1980s. One of these networks' most famous applications is neural machine translation; around 2014, it was a fantastic breakthrough. The NN only acts in a forward direction, from the input layer to the output layer, without remembering previous values. The RNN is similar but includes connections that point backward, which is feedback between neurons within the layers [28]. The simplest RNN is composed of a single neuron that receives an input and produces an output sends that output to itself.

RNN is a neural network designed to analyze data streams using hidden units. The output depends on previous calculations in applications such as word processing, speech recognition, and DNA sequences. Since RNNs deal with sequential data, they are well suited for processing vast amounts of data. RNN is the first algorithm to remember the input due to internal memory, making it perfectly suitable for machine learning problems involving sequential data. They have a meaningful representation to keep information about the past tense. The output produced in time t_i affects the parameter available in time t_{i+1}. In this way, the RNNs maintain two input types as the current and the recent past, to produce the new data output. RNNs also face the problem of the disappearance or explosion gradient (a problem is found in the learning process that occurs in networks with a certain number of hidden layers (intermediate layers, that is, that are between the input data and the final output or response from the network) [29].

2) Long-Short Term Memory: Long-Short Term Memory (LSTM) is an extension of recurrent neural networks proposed to solve the RNN scatter gradient problem. These networks have more benefits than traditional RNNs because they can maintain long-term relationships by expanding their memory to learn from meaningful experiences that have happened a long time ago. LSTMs allow remembering the entries over a long period. Because it contains its information in memory, which can be considered as the memory of a computer, in the sense that a neuron of an LSTM can read, write, and erase information from its memory [30].

There are two areas where an LSTM cell differs from the standard recurring layer. First, the cell state is divided into two parts, the short-term $h_{(t)}$, and the long-term $c_{(t)}$. Second, three control gates are added along with the state path: the forget gate, the input gate, and the output gate regulating the cell states. The forget gate $f_{(t)}$ controls the long-term removal of information from the previous long-term state $c_{(t-1)}$.

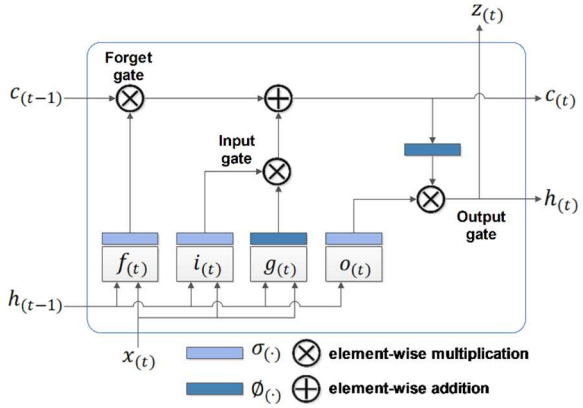

Fig. 2 Structure of an LSTM cell

Input gate $i_{(t)}$ controls the addition of information from the current output $g_{(t)}$ to the current long-term state $c_{(t)}$. The output gate $o_{(t)}$ controls the formation of the current short-term state $h_{(t)}$ using the long-term current state information $c_{(t)}$ [37], as can be seen in Fig. 2.

D. Methodology

In certain circumstances, it is preferable to have real data to show fraud. However, either due to the absence of data or insufficient data for the analysis, synthetic data generated by a simulating system becomes a suitable alternative. The generation of synthetic data is a complicated task. It requires much time for its execution, so it is necessary to use an adequate methodology that optimizes the work and establishes a procedure that enables the tasks to be executed. In our case, the activities and steps necessary to carry out the data generation process are directly related to the behavior of a person who intends to commit fraud, so that we will use the methodology proposed by Lundin *et al.* [31] to adapt it to our needs, as can be seen in Fig. 3.

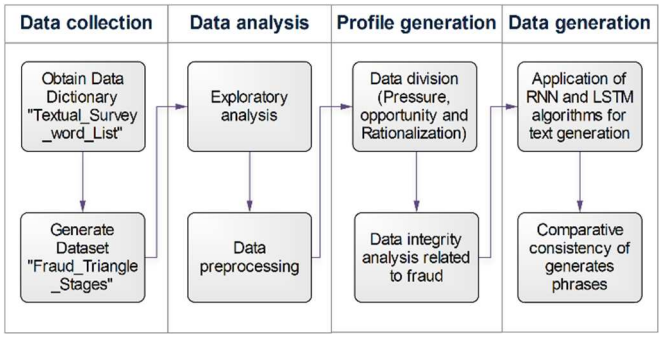

Fig. 3 Flow diagram of the methodology used for the generation of a synthetic dataset through the use of deep learning algorithms.

In the first instance, data is collected to serve as a baseline for subsequent analysis and use; these data must include the necessary characteristics representing the expected behavior in the target system or phrase generator. The selected information may consist of accurate reference data, valid antecedents of similar systems, verified and authentic attacks related to the object of study, and other information collections related to the topic. The second phase consists of analyzing the collected data and identifying essential properties such as fraud classes, usage statistics, attack characteristics, and system behavior statistics. Next, the information analysis will be used to identify the parameters preserved in the sentences' generation. Besides, we create profiles based on the fraud triangle that adjusts to the established parameters' statics.

1) Data collection: The initial information is the starting point for the generation of synthetic data; they must represent data samples of human behavior related to the fraud triangle. Authentic data helps improve the effectiveness of the data creation process. For our purpose, a data dictionary was acquired Textual Survey Word List 103115, related to the fraud triangle that was built by the company Audinet [32], which contributes to the financial community by offering resources online where auditors, accountants, and finance professionals share tools and experiences on audit work programs. This dictionary is a valuable source of information for generating text related to the fraud triangle.

2) Analysis of data: The next step is to analyze the collected data, using exploratory data analysis (EDA) [31], an existing set of ideas on how to study datasets to discover the underlying structure, find important variables, and detect anomalies. Besides, essential characteristics must be identified, that is, parameters that are useful for fraud detection. The data collected should be examined to determine if they are adequate.

3) Profile generation: The next step is to identify the relevant parameters in the input data's behavior. One way to identify these parameters is to study the characteristics necessary to detect fraud. These characteristics must have properties directly related to the fraud triangle in the data generated to be later used in detection processes. Additionally, the correlation between parameters can be accurate indicators of potential fraud. The output at this stage will allow us to identify a suitable profile for the analysis of fraudulent activities that contain values for all the necessary parameters in the generation of sentences.

4) Generation of the dataset: In this phase, the initial dataset's sub-sampling will be carried out to balance the minority class with the majority class. Our initial dataset, composed of phrases identified as fraud, will be the input for the generation text algorithms by applying the RNN and LSTM. In the data simulation actions, only the interest data must be considered to cope with the complexity in the dataset's generation. In general, it is easier to model a specific and well-delimited behavior with prior knowledge of its approach than to do it blindly, so the division of the test dataset by vertex (Pressure, Opportunity, and Rationalization) was performed to delimit the results.

III. Results and Discussion

This section analyzes the results obtained from the execution and comparison of the algorithms used, applying ML techniques by organizing data, learning representation, fitting the model, and evaluating data. It is essential to have large amounts of data, so that deep learning techniques can be better developed and produce good results, particularly in applications where human interpretation is difficult. With large amounts of data, these techniques lead to the generation of text quickly and intelligently, improving the decision-making process.

A. Analysis and Debugging of the Test Set

In this section, we provide the results for the three first steps of the methodology.

Textual Survey Word List 103115 dictionary comprises 2,154 words; It serves as a starting point in the data generation method since they represent human behavior related to fraud. In a controlled environment, the Escuela Politécnica Nacional (EPN) personnel generated an initial dataset using a dictionary [32]. This dataset, named ***FraudTriangle_Stages***, consists of 7,879 sentences, and it is the input in the synthetic data generation process. The information obtained must be relevant to obtain consistency in applying the algorithms and avoid directing the result.

The Audinet data [32] has inconsistencies related to spelling, repetition, and sense, affecting model performance. It is better to identify these anomalies in the input data in advance, which will allow us to correct them for proper processing and analysis. Therefore, exploratory analysis is carried out that consists of applying additional recording mechanisms such as manual analysis of the text, debugging, and information retrieval, used to obtain consistency and accuracy in the data. Manual analysis of the text involves checking the spelling, meaning, and vertex of the fraud triangle to which each of the words in the analyzed data dictionary belongs. The debugging proceeds with the elimination of distorted and repeated information. Finally, the retrieval of relevant information allows the generation of sentences that consist of a second analysis of the data to be eliminated.

Fig. 4 details that 566 words were found from the Textual Survey Word List 103115 dictionary after debugging. Concerning the ***FraudTriangle_Stages*** dataset, Figure 5 shows that it is composed of 7358 sentences after debugging. Finally, the vertex division of the test data was performed, as shown in Fig. 6.

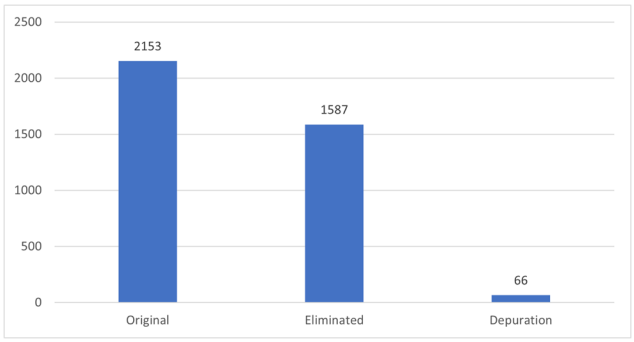

Fig. 4 Data dictionary of Textual Survey Word List 103115

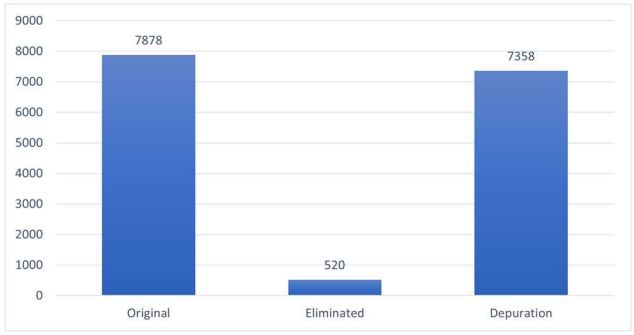

Fig. 5 Data phrases of FraudTriangle_Stages

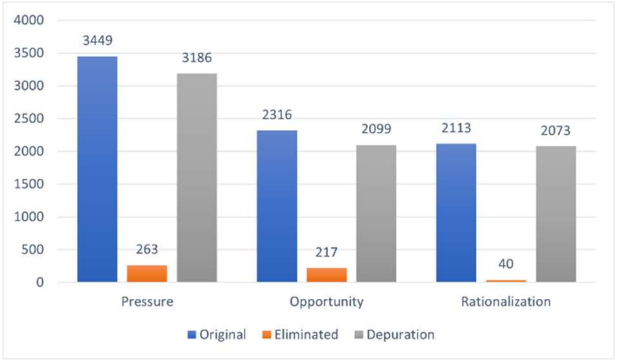

Fig. 6 Vertex FraudTriangle_Stages

B. Development of Tools

The algorithms developed were executed on the ***Google Colab*** platform, which was used as a development environment, allowing programming and debugging tasks to be carried out using a browser in Python programming language. This tool was selected due to its advantages (No local configuration required, free access to CPU's in the cloud, and ease of sharing content).

C. Script's Structure

The structure of the developed code and the ML algorithms used are detailed below. Each of these is made up of various code blocks. The following describes the functions to predict the next word based on the input words.

1) RNN architecture: The function generate name generates the sentences with the purified and trained data. This function requires as input parameters: a training model "particularly a model with RNN networks," a data dictionary of all the words that the data contains, a reverse dictionary to decode the generated phrases, the size of the alphabet that contains the data, which in this case is 30 and finally the number of neurons that in this case have been used 25.

Algorithm 1 RNN algorithm for text generation

Require: String of all Phrases
Input: model (variables: Xt and at-1), char_to_int (Data dictionary), size_alphabet (Characters that make up the data), neurons_number
Output: Phrase generated
1: **Initialization** x = n.zeros [1, 1, 'alphabet_size']
2: **Initialization** a = np.zeros[1, 'n_a']
3: **Initialization** phrase_generated =' '
4: **Initialization** line_break ="\n"
5: **Initialization** comparator = -1
6: **Initialization** count = 0
7: **while** (*comparator* ! = *line_break* & *counter*! = 50) **do**
8: *a*, _ = *recurring_cell*(*K.constant*(*x*), *initial_state* = *K.constant*(*a*))
9: *y* = *layer_out*(*a*)
10: *prediction* = *K.eval*(*y*)
11: *x* = *np.random.choice*(*list*(*range*(*alphabet_size*)), *p* = *prediction.ravel*())
12: *car* = *int_to_char*[*ix*]
13: *generated_name* + = *car*
14: *x* = *to_categorical* (*ix, alphabet_size*). *reshape* (1, 1, *alphabet_size*)
15: *a* = *K.eval*(*a*)
16: counter += 1
17: **if** (counter == 50): **then**
18: *generated name* +='*n*'
19: *return (generated name)*
20: **end if**

A while loop will also be executed, which ends when the next predicted character is a line break or the sentence reaches 50 words and can increase this quantity; meanwhile, it will continue to produce characters to generate sentences. For the generation of sentences with the previously trained model, the recurring cell and the output layer will be used. To start the

prediction, zero values are entered both for the input X and the previous hidden state (Xt, at-1); after this, the resulting activation will be sent to the output layer to generate the prediction. Finally, the inputs (Xt, at-1) are updated, this data becoming the input for the next instant of time, and the process is repeated iteratively until the while loop ends.

Algorithm 2 LSTM algorithm for text generation

```
Require: String of all phrases
Input: model, seed_textQ, next_words,
       max_sequence_len
Output: Phrase generated
Initialization
    start=np.random.randint(0,len(all_phrases)-1)
Initialization seed CNAT = all_phrases[start]
Initialization number=random.randint(2,15)
for _in range(next_words) do
    token_list=tokenizer.texts_to_sequences([seed
        textQ])[0]
    token_list = pad_sequences ([token_list],
    maxlen =max_sequence_len - 1, padding ='pre')
    predicted = model.predict_classes(token_list,
verbose=0)
    output_word = " "
        for word, index in tokenizer.word_index.items():
            if index == predicted: then
                output_word = word
                print (output_word)
                seed_textQ + = " " + output_word
                returnseed_textQ.title()
            end if
        end for
end for
```

2) LSTM Architecture: The generate_phrases function requires as input parameters a training model "particularly a model with LSTM networks", a word that will serve as a seed, and an integer that indicates how many words the generated phrase will contain. It should be noted that the mentioned parameters are automatically generated randomly. The function takes the entered seed and is based on the previously trained model to predict the next word that matches the seed, a process that is repeated repeatedly until reaching the number of requested words.

D. Results

The text generation is carried out, vertex to vertex, regardless of the algorithm used, to avoid inconsistency in the data collected and avoid adverse effects on its operation.

1) RNN algorithm results: The generation of text through the RNN algorithm, in which a neural network model is trained to predict sentences from a sequence of words, thereby generating longer text sequences by calling the model repeatedly through a loop that allows the output of the network or part of it to serve as input to the network itself at the next moment; besides, the number of sentences to obtain is indicated, in this case, 1500 sentences for each vertex (Pressure, Opportunity, Rationalization).

TABLE I
RNN ALGORITHM RESULTS

Text generated by RNN
olaunss
niwrelhznad y nt te i hiehle oinw sepaauo t ymqli aid
nfrpymeadsiotayoliy ugnimosey nmnpu ywdint ia ah
oayautnimeuni ivos
espedemuwatd inalyea anyteabui
awiynuaiw ierve n ib
ene
slpzsdowepwaacubhineltn aleveu a eaceefnla
rvmduqlthmftpfy lestanr sz mumhece
teeagcrckot ushnhwebhz cenmtaoveopltplu viaas tifz
eelfsv iillliumejsca oasnavibui wcue hmdifsheyaew

The RNN algorithm is developed on the Google Colab platform. At the end of the text generation process, the results are stored locally to facilitate access to the information obtained. Table 1 shows some results.

2) LSTM Algorithm Results Presentation: The algorithm developed with LSTM networks has a more complex structure. It expands its memory to learn what to remember and forget. The data is already pre-processed in the format required to be used in the neural network training; the implemented model learns in a few iterations, which are the sentences oriented to each vertex (Pressure, Opportunity, Rationalization).

TABLE II
LSTM ALGORITHM RESULTS

Text generated by LSTM
Not Have to Pay for The Transportation of Me
Where I Get Money to Lend You.
Money to Pay for The Clothes I Will Be Sanctioned Profits This
For the Bank to Give Me the Loan, I Will Have to Mortgage
My House.
My Salary Is Not Enough to Cover with These Fats
We Must Take Out an Express Loan to Pay
They Owe Me Money and I Have Nothing. Money to Pay.
From the Company and Are Expensive Time and I Do Not
It Is Impossible to Deal with This Situation with The
Tell Them You are Sick, and You Can Not Be in The Audit.

In the LSTM algorithm, the same instructions were developed to be executed that were carried out in the RNN algorithm to present the results in Table 2.

E. Discussion

The Readable tool was used to obtain the metrics, which is an online platform that allows checking the legibility, spelling, and grammar of a text, giving a score related to the consistency of the data analyzed. This readability score will allow us to evaluate and compare the RNN and LSTM algorithms' sentences. Ranges from 0-100 were entered until reaching 1000 data. Once this amount was reached, the range from 0-500 was increased, having a maximum of 1500 data entered, due to working with a tool's trial license, which does not allow more data entry.

Initially, the original data were analyzed to determine the tool's score to this data set to generate a baseline and establish a metric to compare the results obtained from the deep learning algorithms used. Table 3 presents the percentages obtained when evaluating the consistency of the original text based on the parameters established for its measurement, identifying the averages of each vertex (Pressure = 75.83%,

Opportunity = 82.82%, and Rationalization = 80.78%) that will help us to compare the averages of the data generated with RNN and LSTM.

TABLE III
RESULTS IN PERCENTAGES OF ANALYSIS USING READABLE TOOL ON THE SOURCE TEXT

Amount	Pressure	Opportunity	Rationalization
100	70.10	89.10	88.00
200	71.90	88.40	84.90
300	75.10	87.60	83.90
400	75.90	85.60	83.20
500	78.20	85.30	83.10
600	78.30	85.50	81.30
700	77.10	81.80	78.80
800	77.10	79.40	76.60
900	77.20	77.70	75.10
1000	77.50	74.30	75.20
1500	75.70	76.30	78.50
Avg.	**75.83**	**82.82**	**80.78**

The consistency results obtained by the RNN and LSTM algorithms applied in each of the vertices of the Fraud Triangle (Pressure (I), Opportunity (II), and Rationalization (III)) are described below, based on the scores obtained by run this analysis on the data processed by said algorithms in the Readable tool. How it can be seen in Table 4. Additionally, the averages of each vertex are identified in both RNN and LSTM (I RNN = 28.43%, I LSTM = 77.71%, II RNN = 16.15%, II LSTM = 70.46%, III RNN = 24.8% and III LSTM = 77.05%) respectively.

We obtain the results of the metrics applied to the text generated by the RNN - LSTM algorithms. We can identify these sentences' consistency versus the original data representing the baseline against which those results will be compared to identify which technique is the most appropriate to build phrases related to fraud.

TABLE IV
RESULTS IN PERCENTAGES OF ANALYSIS USING TOOL READABLE ON DATA GENERATED BY RNN AND LSTM ALGORITHMS

Amount	I RNN	I LSTM	II RNN	II LSTM	III RNN	III LSTM
100	18.70	76.70	14.30	71.00	23.30	77.70
200	31.50	78.10	19.50	70.80	25.60	77.20
300	28.30	77.60	17.90	69.80	20.90	77.10
400	31.60	77.70	16.70	70.00	21.70	77.40
500	29.10	77.40	16.40	70.30	23.90	76.80
600	31.20	77.80	16.20	70.80	23.80	76.70
700	31.40	77.80	15.10	70.80	25.20	76.80
800	30.70	77.80	15.60	70.40	25.90	76.80
900	27.00	77.80	15.90	70.40	25.70	76.90
1000	27.00	78.10	15.40	70.40	26.70	77.00
1500	26.20	78.00	14.60	70.40	25.50	77.20
Avg.	**28.43**	**77.71**	**16.15**	**70.46**	**24.8**	**77.05**

Fig. 7 shows the comparison of the obtained metrics; the scores related to the original data corresponding to pressure, opportunity, and rationalization are represented by orange graphs. For the data generated by the RNN algorithm in the three mentioned vertexes, they are yellow graphs. Furthermore, finally, the data generated by the LSTM algorithm by blue graphics. First, we can see the comparison between the results obtained when analyzing each of the vertexes of the fraud triangle for the RNN Algorithm and the original dataset, the data collected by the algorithm have a score below 40%, and the original dataset a score above 70%, which shows that specifically, the RNN algorithm is inefficient. Regarding the scores obtained by the LSTM Algorithm for each vertex of the fraud triangle, we can see that the data collected by the algorithm has a score above 70%, like the whole of original data. Therefore, the text generated by the LSTM algorithm has consistency.

Let us leave aside the score for a moment and focus on the number of sentences analyzed. We can affirm that a deep learning algorithm provides better results when working with large amounts of data. If we observe Fig. 7, we can see that from the 1000 data, the scores stabilize and do not make significant variations, while before the 1000 data, these scores are highly variable.

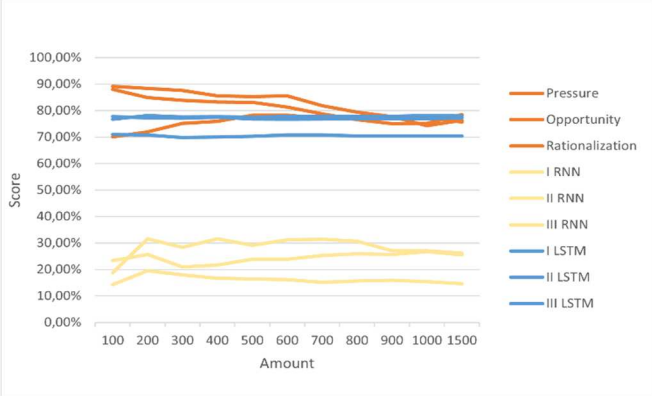

Fig. 7 Score comparison between the data generated by RNN and LSTM algorithms against the original data.

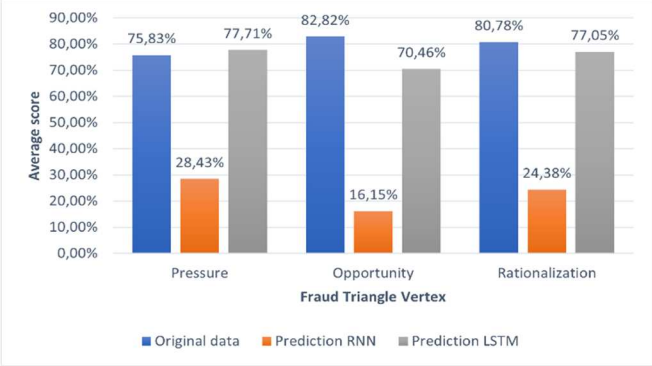

Fig. 8 Comparison of averages obtained by the algorithms (RNN-LSTM) and the original data at the vertices of the fraud triangle.

The average of the scores obtained by the different data sources (Original, RNN, and LSTM) at the fraud triangle's vertices indicate that the data generated by the RNN algorithm (orange color) is inefficient with percentages that do not exceed 28.43 % in the best case. The opposite occurs with the LSTM algorithm (gray color) with satisfactory results and even surpasses the original data (blue color) with 77.71% for Pressure. In comparison, Opportunity, and Rationalization above 70%, as can be seen in Fig. 8.

F. Implementation Recommendations

Under the constrained resources, such as the hosts used in the development and testing phase, the Google Colab platform provides resources that allow the storage and processing of large amounts of data, used in this research. The

number of interactions to achieve optimal results varies between the different proposed algorithms; however, the model results' difference without oversampling is significant. For this reason, for a small dataset or with unbalanced data, it is best to perform oversampling before training the algorithm.

Another case to consider is not to overtrain the Neural Network since a more significant number of iterations does not always mean greater precision in the results. For example, in the LSTM algorithm developed in this project, when trying 150 iterations, the results did not have much difference than with 100. The variety of optimizers available for implementing neural networks can make developers hesitate between them for their purposes. We corroborate that the Adam algorithm (Adaptive moment estimation) works well for the generation of text. Adam combines the benefits of the AdaGrad and RMSProp algorithms.

IV. Conclusion

The limited availability of data sets for the analysis and study of fraud presents a challenge in developing tools for its detection. This paper has presented a methodology for the generation of uniformly distributed synthetic data based on the fraud triangle theory. For the generation of sentences related to fraud, we used RNN, and LSTM, an original data set, and a data dictionary built on the fraud triangle's vertices (pressure, opportunity, and rationalization).

We compared the consistency of original and synthetically generated data distributions based on their readability and grammar. Initially, the original data's consistency was analyzed, obtaining a score higher than 70% as a baseline. Our results show that the original data's consistency score is higher than 70% as a baseline, which serves as the baseline. The synthetic data set generated with the RNN algorithm is deficient and has a consistency below 40%. On the contrary, the LSTM algorithm maintains a consistency level higher than 70% and similar to the original data's score. As future work, we will propose a fraud predictor system that employs machine learning algorithms.

References

[1] Acfe-spain.com, “Acfe association of certified fraud examiners capítulo españa,” Available: https://acfe-spain.com, Accessed: 06-11-2019.

[2] D. Cressey, Other people's money. Montclair, NJ: Patterson Smith, 1973.

[3] S. Y. J. Guan, R. Li and X. Zhang, "A method for generating synthetic electronic medical record text, IEEE/acm transactions on computational biology and bioinformatics," Available: 10.1109/tcbb.2019.2948985, accedido 06-11-2019.

[4] R. DeMilli and A. Offutt, "Constraint-based automatic test data generation, IEEE transactions on software engineering, vol. 17, no. 9, pp. 900-910, 1991." Available: 10.1109/32.92910, Accessed: 06-11-2019.

[5] P. T. M. Mann, O. P. Sangwan and S. Singh, "Automatic goal-oriented test data generation using a genetic algorithm and simulated annealing", in international conference-cloud system and big data engineering confluence. IEEE, 1 2016.

[6] S. Rani and B. Suri, "An approach for test data generation based on genetic algorithm and delete mutation operators", in international conference on advances in computing and communication engineering. IEEE, 2015.

[7] T. L. G. Albuquerque and M. Magnor, "Synthetic generation of high-dimensional datasets", IEEE transactions on visualization and computer graphics, vol. 17, no. 12.

[8] P. R. Bing Wang and K. Mueller, "Sketchpadn-d: Wydiwyg sculpting and editing in high-dimensional space," IEEE transactions on visualization and computer graphics, vol. 19, no. 12.

[9] E. W. B. C. Kwon, H. Kim and A. E. J. Choo, H. Park, "Axisketcher: Interactive nonlinear axis mapping of visualizations through user draw- ings", IEEE transactions on visualization and computer graphics, vol. 23, no. 1, pp. 221–230, 1 2017.

[10] Y. Y. T. R. Liu, B. Fang and P. P. Chan, "Synthetic data generator for classification rules learning," in international conference on cloud computing and big data (ccbd). IEEE, 11 2016, pp. 357–361.

[11] B. S. P. Lin, D. J. A. Cipolone, C. R. S. Cox, and R. X. D. Holt, "Development of a synthetic dataset generator for building and testing information discovery systems," in international conference on information technology: New generations (itng). IEEE, 2006, pp. 707–712.

[12] P. L. D. Jeske, R. X. C. Rendon, and B. Samadi, "synthetic data generation capabilities for testing data mining tools", in milcom. IEEE, 2006, pp. 1–6.

[13] M. A. A. M. Pasinato, C. E. Mello and G. Zimbrao, "Generating synthetic data for context-aware recommender systems", in 2013 brics congress on computational intelligence and 11th brazilian congress on computational intelligence. IEEE, 9 2013, pp. 563–567.

[14] D. Garcia and M. Millan, "A prototype of synthetic data generator", in colombian computing congress (ccc). IEEE, 5 2011, pp. 1–6.

[15] M. K. F. Brodkorb, A. Kuijper, and T. V. Landesberger, "a modular rule- based visual interactive creation of tree-shaped geo-located networks," in international conference on signal-image technology & internet-based systems (sitis). IEEE, 2016, pp. 397–403.

[16] X. Ying and X. Wu, "graph generation with prescribed feature constraints," in siam international conference on data mining. philadelphia, pa: Society for industrial and applied mathematics, 4 2009, pp. 966–977.

[17] Y. L. Can Yang, Sixuan Ren and G. H. Houwei Cao, Qihu Yuan, "personalized channel recommendation deep learning from a switch sequence - IEEE journals & magazine", Available: https://IEEExplore.IEEE.org/document/8458124, Accessed: 07-07-2020.

[18] X. Z. Brian C. Hosler, C. C. Owen Mayer, and M. C. S. James A. Shack- leford, "the video authentication and camera identification database: A new database for video forensics - IEEE journals & magazine", Available: https://IEEExplore.IEEE.org/document/8734060, Accessed: 07-07-2020.

[19] Y. Z. Hangxia Zhou, K. Y. Lingfan Yang, Qian Liu, and Y. Du, "short-term photovoltaic power forecasting based on long short-term memory neural network and attention mechanism - IEEE journals magazine", Available: https://IEEExplore.IEEE.org/document/8736879, Accessed: 07-07-2020.

[20] M. A. Hamdi Altaheri and G. Muhammad, "date fruit clas- sification for robotic harvesting in a natural environment using deep learning - IEEE journals & magazine", Available: https://IEEExplore.IEEE.org/document/8807111, Accessed: 07-07-2020.

[21] S. W. S. Ahmadreza Argha, Ji Wu and B. G. Celler, "blood pressure estimation from beat-by-beat time-domain features of oscillometric waveforms using deep-neural-network classification models - IEEE journals & magazine", Available: https://IEEExplore.IEEE.org/document/8789404, Accessed: 07-07-2020.

[22] C. L. Xun Zhu and D. Ji, "keyphrase generation with copy- net and semantic web - IEEE journals & magazine", Available: https://IEEExplore.IEEE.org/document/9019613, accedido 07-07-2020.

[23] N. M. Rabiu Abdullahi, "Fraud triangle theory and fraud diamond theory. understanding the convergent and divergent for future research," Available: 10.6007/IJARAFMS/v5-i4/1823, Accessed: 06-11-2019.

[24] A. R. Fernández, “los datos sintéticos, la clave para mejorar la inteligencia artificial”, Addison Wesley college, 1997.

[25] P. P. S. Kotsiantis, "mixture of expert agents for handling imbalanced datasets", annals of mathematics, computing tele informatics, vol 1, no 1 (46-55), 2003.

[26] I. Mufti Mahmud, Senior Member, I. Mohammed Shamim Kaiser, Senior Member, I. Amir Hussain, Senior Member, and S. Vassanelli, "Applications of deep learning and reinforcement learning to biological data, IEEE transactions on neural networks and learning

systems, vol. 29, no. 6, 2018," Available: 10.1109/TNNLS.2018.2790388, Accessed: 06-11-2019.
[27] D. J. Matich, “Redes neuronales: Conceptos básicos y aplicaciones.”, informática aplicada a la ingeniería de procesos – orientación i.
[28] J. Z. Ruiqin Bai, X. L. Dengao Li, and B. Z. Qiang Wang, "Rnn- based demand awareness in smart library using crfid, IEEE china communications, vol. 17, no. 5 2020)," Available:10.23919/JCC.2020.05.021, Accessed: 11-12-2020.
[29] F. D. D. Rav´ı, C. Wong, J. A. M. Berthelot, and G. Y. B. Lo, "deep learning for health informatics", IEEE journal of biomedical and health informatics 21 (1) (2017) 4–21.
[30] S. D. S.Dhananjay Kumar, "Prediction of depression from eeg sig- nal using long short term memory(lstm), IEEE 3rd international conference on trends in electronics and informatics (icoei)," Available: 10.1109/ICOEI.2019.8862560, Accessed: 06-11-2019.
[31] H. K. E. Lundin and E. Jonsson, "a synthetic fraud data generation methodology", Accessed: 07-07-2020.
[32] AuditNet, "using key word analysis of an organization's big data for error and fraud detection". urlhttps://www.auditnet.org/key-word-analytics.
[33] Huang, Shaio Yan & Lin, Chi-Chen & Chiu, Ann & Yen, David. (2017). Fraud detection using fraud triangle risk factors. Information Systems Frontiers. 19. 10.1007/s10796-016-9647-9.
[34] Raghunathan, T., Reiter, J. and Rubin, D. (2003). Multiple Imputation for Statistical Disclosure Limitation. Journal of Official Statistics, 19(1), pp.1-16.
[35] Taub, Jennifer & Elliot, Mark & Sakshaug, Joseph. (2020). The Impact of Synthetic Data Generation on Data Utility with Application to the 1991 UK Samples of Anonymised Records. Journal of Transactions on Data Privacy. 13. 1-23.
[36] Golovko, Vladimir & Kroshchanka, A. & Treadwell, D. (2016). The nature of unsupervised learning in deep neural networks: A new understanding and novel approach. Optical Memory and Neural Networks. 25. 127-141. 10.3103/S1060992X16030073.
[37] Zhang, Jianjing & Wang, Peng & Yan, Ruqiang & Gao, Robert. (2018). Long short-term memory for machine remaining life prediction. Journal of Manufacturing Systems. 48. 10.1016/j.jmsy.2018.05.011.