# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA DE SISTEMAS

## GENERACIÓN DE UN CONJUNTO DE DATOS SINTÉTICOS MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO PARA ANÁLISIS DE FRAUDE.

**PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

**VERÓNICA ELIZABETH OLMEDO VÉLEZ**
veronica.olmedo@epn.edu.ec
**CARLOS ÁNDRES NARVÁEZ TELLO**
carlos.narvaez@epn.edu.ec

**DIRECTOR: M.Sc. MARCO POLO SANCHEZ AGUAYO**
marco.sanchez01@epn.edu.ec

**CODIRECTOR: Dra. MYRIAM BEATRIZ HERNANDEZ ALVAREZ**
myriam.hernandez@epn.edu.ec

**Quito, marzo 2021**

# CERTIFICACIÓN

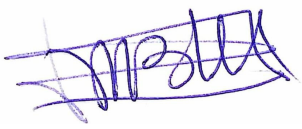
Certifico que el presente trabajo fue desarrollado por **Verónica Elizabeth Olmedo Vélez** y **Carlos Andrés Narvaéz Tello,** bajo mi supervisión

**M.Sc. Marco Polo Sanchéz Aguayo**

**DIRECTOR DE PROYECTO**

# CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por **Verónica Elizabeth Olmedo Vélez** y **Carlos Andrés Narváez Tello**, bajo mi supervisión



**Dra. Myriam Beatriz Hernández Álvarez**

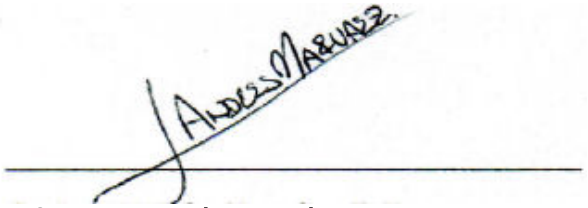
**CODIRECTOR DE PROYECTO**

# DECLARACIÓN

Nosotros,Carlos **Andrés Narváez Tello** y **Verónica Elizabeth Olmedo Vélez**, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

**Verónica Elizabeth Olmedo Vélez**



**Carlos Andrés Narváez Tello**

# DEDICATORIA

A mis padres, Aracely y Abundio, a mi hermanas y mis amigos, por ser testigos de mi esfuerzo y dedicación durante toda la mi vida académica, por su gran apoyo, buenos momentos y amor incondicional hasta el final.

A mi familia quienes nunca dejaron de creer en mí.

**Verónica E. Olmedo V.**

# DEDICATORIA

A mis padres, Marco y Marlene por su esfuerzo, ya que día a día supieron brindarme su apoyo, a mi hermana Camila por su cariño y motivación, como familia fueron los que estuvieron en toda mi vida académica, a pesar de las dificultades y adversidades vividas.

**Carlos A. Narváez T.**

# AGRADECIMIENTOS

Agradezco a Dios, por ayudarme a culminar esta importante etapa de mi vida, por hacer que este sueño se cumpla.

A mi madre Aracely, mi padre Abundio, los cuales me brindaron su amor y comprensión incondicional, además, de todo su apoyo, durante esta larga travesía. Fueron mi pilar para seguir adelante y no darme por vencida. A mi madre, por estar siempre pendiente de mí, con sus tiernas palabras ayudarme a seguir. A mi padre, por brindarme consejos en momentos difíciles y ser el mejor padre. A mis hermanas, por estar siempre a mi lado en las buenas y en las malas. A todos ellos les agradezco porque nunca dejaron de creer en mí, siempre estuvieron allí para mí, siendo participes de mi crecimiento personal.

Una persona importante en mi vida formó parte de esta etapa, ahora ya no está conmigo, sin embargo, tuve la dicha de compartir gratos momentos y agradezco su apoyo brindado.

A mis amigos, por compartir momentos inolvidables conmigo, nunca me dieron la espalda, fueron de gran ayuda en toda mi vida académica, gracias por su apoyo moral, amor y amistad, son parte de mi vida.

A mi director M.Sc. Marco Sánchez y codirector Dra. Myriam Hernández, por hacerme participe de este proyecto, además por el apoyo recibido durante el desarrollo del mismo, por sus consejos, sabiduría y tiempo.

Sobretodo agradezco a mi compañero Carlos Narváez, por su amistad, colaboración, tiempo compartido y conocimientos para la realización del presente trabajo, fue incondicional en los momentos difíciles.

Finalmente, agradezco a la Escuela Politécnica Nacional y a sus autoridades por haberme brindado la oportunidad de haber culminado mis estudios y una etapa más de mi vida.

**Verónica E. Olmedo V.**

# AGRADECIMIENTOS

Agradezco a Dios, por cuidar y guardar mis pasos en todo el tiempo que duró la carrera y de esa manera poder culminar esta etapa de mi vida.

A mi familia, por ser las personas que estuvieron paso a paso, dándome ánimo para seguir adelante, a mis padres por brindarme ese apoyo incondicional y estar al pendiente de las cosas que necesitaba a lo largo de mi vida académica y también a mi hermana Camila por motivarme en todo momento.

Al director de la tesis, el M.Sc. Marco Polo Sánchez Aguayo, por hacerme partícipe de este importante proyecto, por compartir su conocimiento con nosotros y de esa manera poder obtener los objetivos planteados.

Así también a la codirectora de tesis, Dra. Myriam Beatriz Hernández Álvarez, por ser parte del proyecto y aportar con su valioso conocimiento.

Agradezco a mi compañera Verónica Olmedo la cual colaboró en el proyecto y ha sido una parte esencial dentro de la realización de este trabajo, además de ser de gran apoyo a lo largo de mi vida universitaria.

A mis amigos, quienes estuvieron acompañándome día a día, con quienes tuve la dicha de compartir momentos dentro y fuera del aula, por su cariño y amistad de manera desinteresada.

Por último, pero no menos importante a los maestros que transmitieron su conocimiento y habilidades obtenidas a lo largo de estos años dentro de la Escuela Politécnica Nacional.

**Carlos A. Narváez T.**

# CONTENIDO

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# RESUMEN

En la actualidad, actividades relacionadas con fraude crecen a un ritmo vertiginoso, causando enormes pérdidas económicas cada año. Para un adecuado análisis de este fenómeno es necesario disponer de datos que evidencien dicho comportamiento, pero debido a que estos son escasos y difíciles de encontrar la generación de datos sintéticos para su estudio es una opción que se debe considerar. Para la generación de texto, se usaron técnicas de Machine Learning (ML), específicamente modelos de aprendizaje profundo como ***Recurrent Neural Network (RNN)*** y ***Long Short Term Memory Networks (LSTM)*** apoyados en la teoría del triángulo del fraude propuesta por Donald R. Cressey, para construir un conjunto de datos sintético que permita su análisis. La RNN trabaja con múltiples copias de sí misma, cada una emite un mensaje a su sucesor, lo que resta precisión al momento de generar frases, además, enfrenta el problema del gradiente de desaparición. El modelo LSTM se propone para resolver este problema, porque estas redes son capaces de mantener interrelaciones a largo plazo al ampliar su memoria para aprender de experiencias pasadas, lo que las hace perfectas para generar texto. Los resultados obtenidos indican que la arquitectura de generación de datos propuesta mediante el algoritmo LSTM proporciona un mejor rendimiento en la generación de frases y la legibilidad de los datos es superior con una eficiencia del 70 % en comparación con el enfoque del algoritmo RNN la cual alcanzó un 40 %. Mediante esta técnica (LSTM) se logró sintetizar un conjunto de datos entendible y relacionado con el triángulo del fraude que permitirá realizar el estudio del fraude con efectividad.

# ABSTRACT

Today, 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, but since these are scarce and difficult to find, the generation of synthetic data for its study is a viable option. Machine Learning (ML) techniques were used for the generation of text, specifically deep learning models such as ***Recurrent Neural Network (RNN)*** and ***Long Short Term Memory Networks (LSTM)*** supported by the theory of the fraud triangle proposed by Donald R. Cressey, to build a synthetic data set that allows its analysis. The RNN works with many copies of itself; each sends a message to its successor, which reduces precision when generating sentences; besides, it faces the disappearance gradient problem. The LSTM model aims to solve this problem; these networks can maintain long-term interrelationships by expanding their memory to learn from past experiences, making them perfect for generating text. The results obtained indicate that the data generation architecture proposed using the LSTM algorithm provides better sentence generation performances. The data's readability is superior with an efficiency of 70 % compared to the RNN algorithm approach reached 40 %. Using this technique (LSTM), it was possible to synthesize a set of understandable data related to the fraud triangle to allow the fraud study to be carried out effectively.

# 1 INTRODUCCIÓN

El presente trabajo de titulación aplica la teoría del triángulo del fraude que, dentro de la naturaleza humana, crean condiciones para cometer fraude. El fraude incluye cualquier acto intencional o deliberado de privar a otro de propiedad o dinero mediante astucia, engaño u otros actos injustos, de acuerdo con la Asociación de Examinadores de Fraude Certificados (ACFE). Según ACFE, el fraude se define como el uso de la propia ocupación para el enriquecimiento personal a través del uso indebido deliberado o la mala aplicación de los recursos o activos de la organización empleadora [1].

Donald R. Cressey [2], un destacado experto en sociología del crimen investiga las razones detrás de la pregunta ¿por qué las personas cometen fraude? y determina la respuesta en los siguientes tres elementos críticos: presión percibida, oportunidad percibida y racionalización que deben estar presentes consecutivamente para provocar el deseo de cometer fraude. A partir de su investigación presenta el concepto ***"The Fraud Triangle Theory".***

Información con evidencias sobre actividades fraudulentas asociadas al triángulo del fraude, en la cual se observe comunicaciones relacionadas con presión, oportunidad y racionalización, es incipiente en la comunidad científica; a excepción de estudios realizados por entidades privadas como el Buró Federal de Investigaciones (FBI) y ACFE, quienes han logrado obtener datos relacionados con estos tópicos a partir de sus investigaciones a importantes empresas con carácter reservado. En este contexto, una opción válida es la de generar datos sintéticos para utilizar la predicción automática y detección temprana del fraude, los cuales, según muchos expertos, aseguran que son la clave para hacer que el aprendizaje automático, dentro de la Inteligencia Artificial (IA), sea más rápido y que los algoritmos aumenten la precisión en sus predicciones con el objetivo de identificar comportamientos de carácter fraudulento, sobre todo cuando los datos reales son muy costosos de conseguir o tienen difícil acceso [3].

Las violaciones de derechos de autor y los derechos de propiedad intelectual han limitado la disponibilidad de conjuntos de datos reales. Por lo tanto, analizaremos algunas técnicas de Machine Learning (ML) para evidenciar la aplicación de los algoritmos Long Short Term

Memory (LSTM) y Recurrent Neural Network (RNN), que se utilizan con la intención de generar de manera eficiente conjuntos de datos sintéticos específicos, los resultados obtenidos serán sometidos a pruebas de rendimiento. Las métricas se obtuvieron mediante la utilización de la herramienta Readability, la cual evalúa la coherencia del texto ingresado y proporciona valores de puntuación entre 0 a 100; A continuación se aplica una media aritmética a todos los vértices del triángulo del fraude (Presión, Oportunidad, Racionalización), **con la finalidad de identificar el algoritmo más preciso y eficiente**.

En el segundo apartado de este documento se encuentra el fundamento teórico bajo el cual se realizó el trabajo de titulación. En la tercera parte se presenta la metodología. En el cuarto apartado se encuentran los resultados y discusión. La quinta sección contiene las conclusiones y recomendaciones. Finalmente, se presentan las referencias bibliográficas y, los anexos.

## 1.1 PREGUNTA DE INVESTIGACIÓN

¿Se pueden generar datos textuales sintéticos, con un buen nivel de legibilidad y consistencia, en ambientes controlados, como base para el aprendizaje automático para la identificación de comportamientos de carácter fraudulento?

## 1.2 OBJETIVO GENERAL

Generar un conjunto de datos sintéticos basados en la teoría del triángulo del fraude, utilizando algoritmos relacionados con Inteligencia Artificial para la generación de texto mediante técnicas de Machine Learning.

## 1.3 OBJETIVO ESPECÍFICOS

- ❒ Identificar recursos para la generación de datos y técnicas asociadas a ML.
- ❒ Desarrollar un modelo para el análisis de datos y técnicas necesarias para la construcción de datos sintéticos.
- ❒ Identificar frases o palabras claves asociadas a la teoría del triángulo del fraude que permitirá establecer una línea base para la generación del conjunto de datos sintético.
- ❒ Obtener datos relevantes para la realización de las pruebas y obtención de resultados.

## 1.4 HIPÓTESIS

Un conjunto de datos sintéticos generado en un ambiente controlado puede contribuir con la investigación en muchos campos donde la obtención de datos es un problema, como evidencias tangibles de fraude.

# 2 MARCO TEÓRICO

## 2.1 JUSTIFICACIÓN TEÓRICA

Muchas áreas de estudio utilizan datos generados sintéticamente, desde la minería de datos hasta la ingeniería de software y la inteligencia artificial. Por ejemplo, Demillio y Offut [4] presentaron una técnica basada en fallas para generar datos para el módulo de software o pruebas unitarias. En el campo de la computación evolutiva, también es posible encontrar trabajos sobre algoritmos genéticos para generar datos adecuados para las pruebas[5][6].

Albuquerque, Lowe & Magnor [7] presentan un marco de referencia para generar datos de alta dimensión. Usando la interfaz gráfica, el usuario puede construir una base de datos adecuada para su aplicación a través de distribuciones estadísticas con propiedades definidas por el usuario.

Wang, Ruchikachorn & Mueller [8] presentan un nuevo enfoque para generar datos sintéticos, en los cuales el usuario diseña los datos deseados a mano, y el sistema calcula el modelo generador a partir de los diseños del usuario. Las interacciones similares son presentadas por [9] que hicieron uso de las interacciones de dibujo para dirigir la visualización de datos de alta dimensión de acuerdo con el conocimiento del dominio de los usuarios.

Liu [10], creó un generador de datos sintéticos para evaluar el aprendizaje de reglas de clasificación. Del mismo modo, los investigadores han propuesto sintetizadores de bases de datos para analizar herramientas de minería de datos [11] [12] [13]. Estos tipos de sistemas generan bases de datos con el objetivo de analizar herramientas de minería de datos ya que a menudo obtener datos reales puede ser muy costoso o limitado por derechos legales o privacidad de datos. Sin embargo, estos generadores son específicos de una herramienta o de un contexto problemático, lo que limita el uso a gran escala en otras áreas.

García & Millán [14] crearon un sistema de generación de conjuntos de datos para una amplia gama de áreas. Han comparado su propuesta con los sistemas ya existentes en el mercado, pero su aplicación propuesta es un software libre diferente de las aplicaciones de

mercado.

Algunos enfoques se dedican a generar datos de redes sintéticas [15] [16]. Por ejemplo, Brodkorb propuso un generador de datos de red con ubicaciones geográficas adjuntas a los nodos, de esta manera, los datos generados se pueden mostrar interactivamente en un mapa, donde el usuario puede explorar la red generada y también puede ajustar los resultados más adelante.

Can Yang, Sixuan Ren, Yong Liu, Houwei Cao, Qihu Yuan, and Guoqiang Han [17] proponen un marco de recomendación de canal personalizado con aprovisionamiento dinámico de datos a través del aprendizaje profundo de secuencias de conmutación de canales históricos en sistemas IPTV Internet Protocol Television*(Televisión por Protocolo de Internet)*, para la generación dinámica de una lista de canales recomendados para cada usuario por medio de una red LSTM Long Short-Term Memory *(Redes de Memoria Larga-Corto Plazo)*.

Brian C. Hosler, Xinwei Zhao, Owen Mayer, Chen Chen, James A. Shackleford and Matthew C. Stamm [18], utilizan una técnica de aprendizaje profundo avanzada, concretamente complementos Convolutional Neural Networks (CNN), para fusionar las activaciones de neuronas de múltiples parches para representar la identificación del modelo de cámara a nivel de vídeo, por lo que se emplea como base de entrenamiento datos de autenticación de vídeos e identificación de cámara para generar una colección de vídeos cuidadosamente construida con el propósito de desarrollar y evaluar algoritmos de identificación de modelos de cámaras de vídeo.

Hangxia Zhou, Yujin Zhang, Lingfan Yang, Qian Liu, Ke Yan and Yang Du [19], crean un nuevo conjunto de datos de relevancia que incluye fijaciones de 10 observadores sobre 1900 imágenes degradadas por 19 tipos de transformaciones. Utiliza los nuevos datos sobre las imágenes transformadas, llamadas transformación de aumento de datos (DAT), para entrenar modelos de saliencia *(define el mecanismo que utiliza nuestro cerebro para priorizar ciertos estímulos, en este caso de tipo visual*)profunda.

Ahmadreza Argha, Ji Wu, Steven W. Su and Branko G. Celler [20], utiliza técnicas de aprendizaje automático para la estimación de la presión arterial sistólica y diastólica (SBP y DBP). Diseñan un modelo de clasificación de redes neuronales profundas (DNN) para la extracción de características artificiales para estimar SBP y DBP.

Hamdi Altaheri, Mansour Alsulaiman and Ghulam Muhammad [21], utilizan técnicas basada en el aprendizaje profundo para la clasificación de la fruta según la fecha de recolección con una precisión del 99,2 %. Crean un conjunto de datos de cuatro tipos de fechas mediante la adquisición del motor de búsqueda de Google, además, propone un marco de visión

artificial en tiempo real para robots de recolección de fruta según la fecha de recolección en un entorno de huerto basado en el aprendizaje profundo. En cuanto al trabajo futuro, mejorarán el conjunto de datos al incluir imágenes de prueba capturadas de diferentes huertos según la fecha.

Xun Zhu, Chen Lyu and Donghong Ji [22], crean un conjunto de datos de frase clave biomédica a gran escala para evaluar el rendimiento del sistema, los resultados de la web semántica se fusionan en el conjunto de datos biomédicos para participar en el proceso de entrenamiento de la red neuronal y se considera que más información relacionada genera frases clave. Esta propuesta es la más cercana al tema de investigación, pero con un enfoque totalmente diferente.

Zhaohui Che, Ali Borji, Guangtao Zhai, Xiongkuo Min, Guodong Guo and Patrick Le Callet [23], esta investigación se basa en la creación de un nuevo conjunto de datos de seguimiento ocular utilizando algunas transformaciones de preservación de etiquetas para impulsar modelos de prominencia basados en aprendizaje profundo.

## 2.2 JUSTIFICACIÓN METODOLÓGICA

El presente proyecto de titulación plantea, mediante la utilización de técnicas de ***Machine Learning***, generar un conjunto de datos sintéticos basado en la teoría del triángulo del fraude. La ausencia de investigaciones sobre datos auténticos para el análisis de detección de fraude es un gran inconveniente, sobre todo cuando los datos reales son muy costosos de conseguir o de difícil acceso. Con el fin de contrarrestar este inconveniente, se utilizará la metodología propuesta por Emilie Lundin, Håkan Kvarnström, & Erland Jonsson [24], para aplicarla y adaptarla en este proyecto de titulación.

Para que los resultados de este proyecto sean concluyentes es necesario aplicar la técnica de experimentación, que implica la manipulación de pequeños conjuntos de datos auténticos, para crear datos sintéticos apropiados en grandes cantidades. Las propiedades de los parámetros en los datos iniciales se conservan o pueden adaptarse para satisfacer las necesidades de prueba [25].

## 2.3 JUSTIFICACIÓN PRÁCTICA

### 2.3.1 Triángulo del Fraude

El psicólogo criminal Donald R. Cressey [26], descubrió que existen tres factores que deben estar presentes para que el fraude se materialice. Primero, la gerencia u otros empleados tienen un incentivo o están bajo presión, lo que proporciona una razón para cometer fraude. En segundo lugar, existen circunstancias (por ejemplo, la ausencia y la e ineficacia de controles o la capacidad de la administración por invalidarlos) que brindan la oportunidad para perpetuar un fraude. En tercer lugar, los involucrados son capaces de racionalizar el cometer un acto fraudulento. Algunas personas poseen una actitud, carácter o un conjunto de valores éticos que les permiten cometer un acto deshonesto a sabiendas e intencionalmente. Sin embargo, incluso los individuos honestos pueden cometer fraude en un entorno que les impone suficiente presión. Cuanto mayor sea el incentivo o la presión, más probable será que una persona pueda racionalizar la aceptabilidad de cometer fraude.


**Figura 2.1:** Triángulo del Fraude *Donald R. Cressey*

### 2.3.2 Conjunto de Datos Sintéticos

El propósito principal de un *conjunto de datos sintéticos* es ser lo suficientemente versátil y robusto para ser útil en el entrenamiento de modelos de ML, como sugiere el término "sinté-

tico"los *conjuntos de datos sintéticos* se generan a través de programas de computadoras, en lugar de componerse de documentación de eventos del mundo real. La creación de estos es más rentable que la recopilación de datos del reales la mayoría de las veces, ya que minimiza el tiempo, el coste y el riesgo de las operaciones, además algunas investigaciones evidencian que es posible obtener los mismos resultados utilizando*datos sintéticos* que datos del mundo real. [27].

### 2.3.3 Machine Learning

Machine Learning (ML), se encuentra dentro del mundo de la IA. Es una metodología de aprendizaje muy utilizada que proporciona a los sistemas la capacidad de aprender y mejorar automáticamente a partir de la experiencia sin necesidad de intervención humana. Refiriéndonos al método o la forma de aprendizaje podemos hablar de dos campos principales, *aprendizaje supervisado y no supervisado*, y un tercero conocido como *aprendizaje por refuerzo* [28].

***Aprendizaje Supervisado***

A través del análisis de los datos de entrada, se le asigna a cada uno de ellos la correspondiente etiqueta de salida, es decir, se conoce las etiquetas de salida de los datos. El *aprendizaje supervisado* se compone de dos partes: ***el entrenamiento***, donde el algoritmo aprende los patrones o métodos para clasificar y ***las pruebas***, donde se comprueba si el algoritmo implementado realmente aprende los patrones y funciona.

***Aprendizaje no Supervisado***

Este método se enfoca en generar agrupaciones de datos en función de características que los hacen similares entre sí, se distingue del aprendizaje supervisado por el hecho de que sus valores de salida son desconocidos.

***Aprendizaje por Refuerzo***

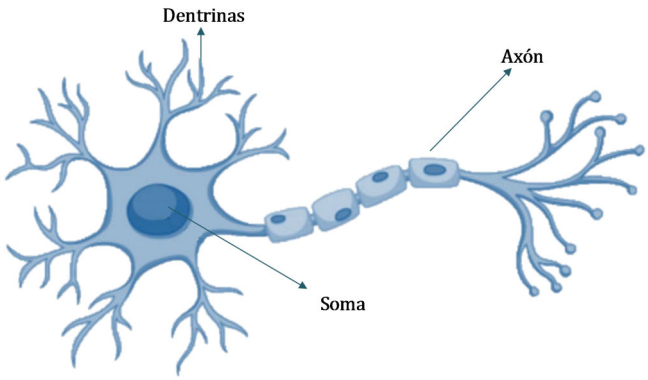
Puede ser similar al *aprendizaje supervisado* ya que utiliza una retroalimentación o recompensa para mejorar, pero tiene claras diferencias. El aprendizaje por refuerzo se enfoca, sobre todo, en la posibilidad de hacer que las máquinas aprendan por ensayo y error a lograr un objetivo claro, y que, en lugar de utilizar etiquetas para los resultados como el aprendizaje supervisado, utiliza una misma dinámica [28].

### 2.3.4 Deep Learning

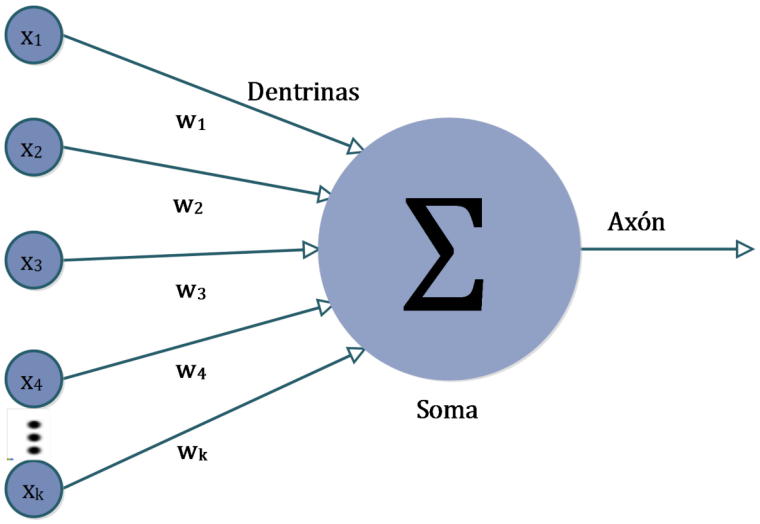
Deep Learning (DL), es un sub-campo dentro del ML, se basa en la forma de aprendizaje de un modelo por capas, es decir, extrae características más abstractas de un conjunto más amplio de datos de entrenamiento, en su mayoría sin supervisión humana. Este sistema de aprendizaje por capas está representado esencialmente, por el modelo conocido como ***"Neural Networks"*** [29].

### 2.3.5 Neural Networks

Neural Networks (NN), son modelos computacionales que emulan ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos, es decir, no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo perfecto para definir a un sistema que es capaz de adquirir conocimiento a través de la experiencia. Las neuronas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta se hacen muy poderosas. Lo que básicamente ocurre en una neurona biológica es lo siguiente: la neurona es estimulada o excitada a través de sus entradas (inputs) y cuando se alcanza un cierto umbral, la neurona se dispara o activa, pasando una señal hacia el axon *(es una especie de conducto por el que pasan impulsos nerviosos a toda velocidad; actúa como canal de comunicación entre la parte central de la neurona y otra parte del sistema nervioso al que ha de llegar este estímulo eléctrico)*. Posteriores investigaciones condujeron al descubrimiento de que estos procesos son el resultado de eventos electroquímicos [30].




(a) Neurona Biológica




(b) Neurona Artificial

**Figura 2.2:** Neurona Biológica vs Neurona Artificial

### 2.3.6 Recurrent Neural Network

Recurrent Neural Network (RNN) aparecieron en la década de 1980, una de las aplicaciones más famosas de estas redes es la traducción automática neuronal, alrededor del 2014 fue un avance asombroso. La NN solo actúa en una dirección “hacia delante”, desde la capa de entrada hacia la capa de salida sin recordar valores previos, la RNN es parecida, pero incluye conexiones que apuntan “hacia atrás”, una especie de retroalimentaciones entre las neuronas dentro de las capas [31], es decir, la RNN más simple, está compuesta por una sola neurona que recibe una entrada, produciendo una salida, y enviando esa salida a sí misma, como se muestra en la siguiente figura 2.3.

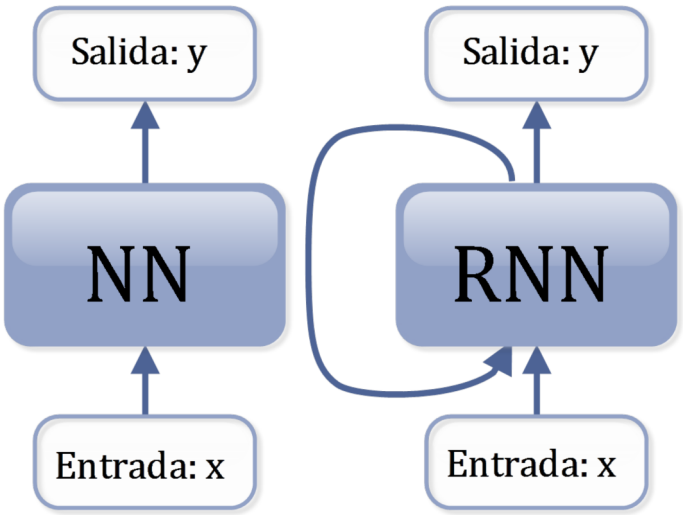



**Figura 2.3:** Red Neuronal vs Red Neuronal Recurrente

RNN es una red neuronal diseñada para analizar flujos de datos mediante unidades ocultas. En aplicaciones, como el procesamiento de texto, el reconocimiento de voz y las secuencias de ADN, la salida depende de cálculos anteriores. Dado que las RNN se ocupan de datos. A continuación se muestra en la figura 2.4 un modelo de RNN.

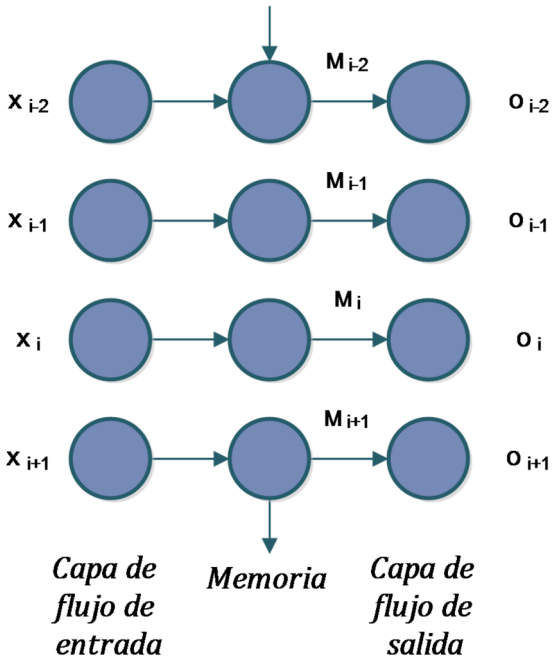



**Figura 2.4:** Modelo RNN

En general, RNN es el primer algoritmo que recuerda su entrada, debido a una memoria interna, lo que lo hace perfectamente adecuado para problemas de aprendizaje automático que involucran datos secuenciales. Tienen una representación significativa para mantener la información sobre el tiempo pasado. La salida producida en el tiempo ***t1*** afecta el parámetro disponible en el tiempo ***t1 + 1***. De esta manera, los RNN mantienen dos tipos de entrada como la actual y la pasada reciente para producir la salida de los nuevos datos [32].

Las RNN también enfrentan el problema del gradiente de desaparición *(se encuentra un problema en el proceso de aprendizaje que se da en las redes con cierto número de capas ocultas (capas intermedias, es decir, que se encuentran entre la entrada de datos y la salida o respuesta final de la red))*. El gradiente de desaparición tiene como propósito reducir el error del modelo RNN, como muestra la figura 2.5, mediante la evaluación del error de dicho modelo en el punto en el que se encuentra, para posteriormente calcular las derivadas parciales en dicho punto, así se obtiene un vector de direcciones, que indica la pendiente de la función hacia donde el error se incrementa, cuanto menor sea el gradiente, más difícil será para la red actualizar los pesos y tardará más en llegar al resultado final [33].

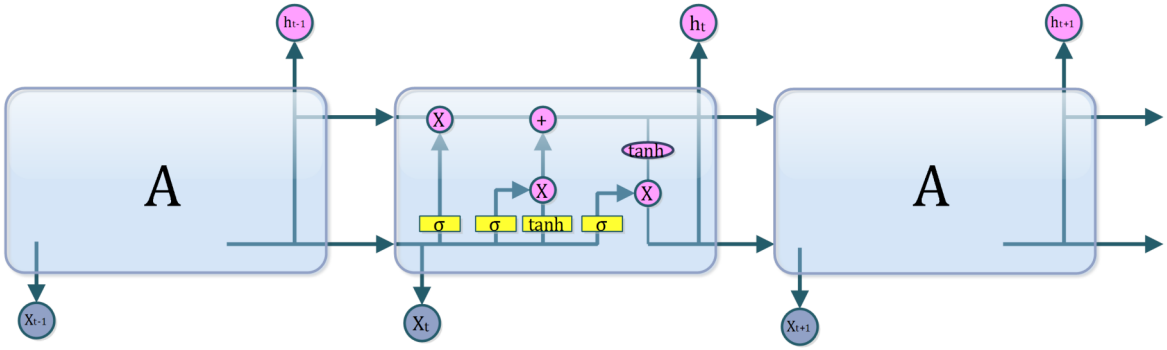



**Figura 2.5:** Resolución del gradiente de desaparición

### 2.3.7 Long-Short Term Memory

Long-Short Term Memory (LSTM) son una extensión de las redes neuronales recurrentes, que básicamente se propusieron para resolver el problema del gradiente de desaparición de las RNN. Estas redes tienen más beneficios que las RNN tradicionales, porque son capaces de mantener interrelaciones a largo plazo al ampliar su memoria para aprender de experiencias importantes que han pasado hace mucho tiempo. Las LSTM permiten recordar sus entradas durante un largo período de tiempo. Esto se debe a que contiene su información en la memoria, que puede considerarse similar a la memoria de un computador, en el sentido que una neurona de una LSTM puede leer, escribir y borrar información de su memoria [34], la diferencia entre RNN y LSTM se presenta en la figura 2.6.

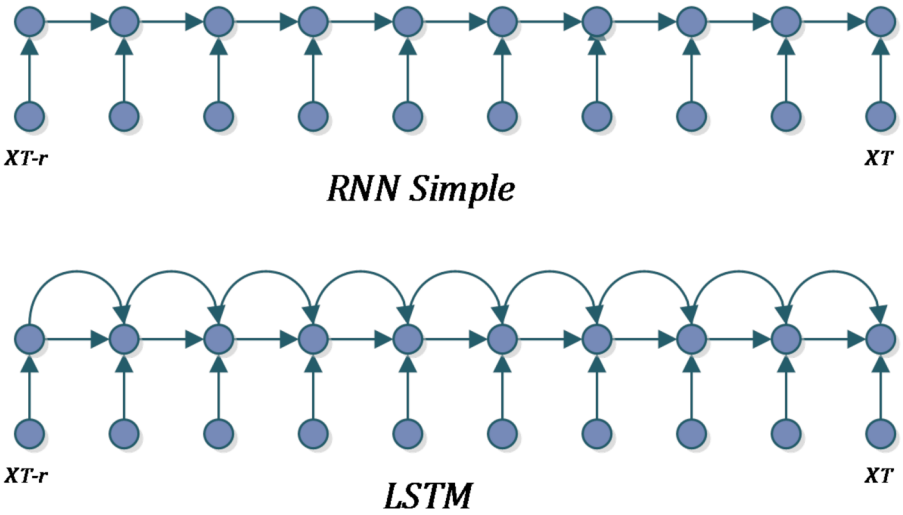



**Figura 2.6:** RNN vs LSTM

En una neurona LSTM hay tres puertas a estas “celdas” de información: puerta de entrada ***(input gate)***, puerta de olvidar ***(forget gate)*** y puerta de salida ***(output gate***). Estas puertas

determinan si se permite o no una nueva entrada, se elimina la información porque no es importante o se deja que afecte a la salida en el paso de tiempo actual como se puede observar en la figura 2.7.

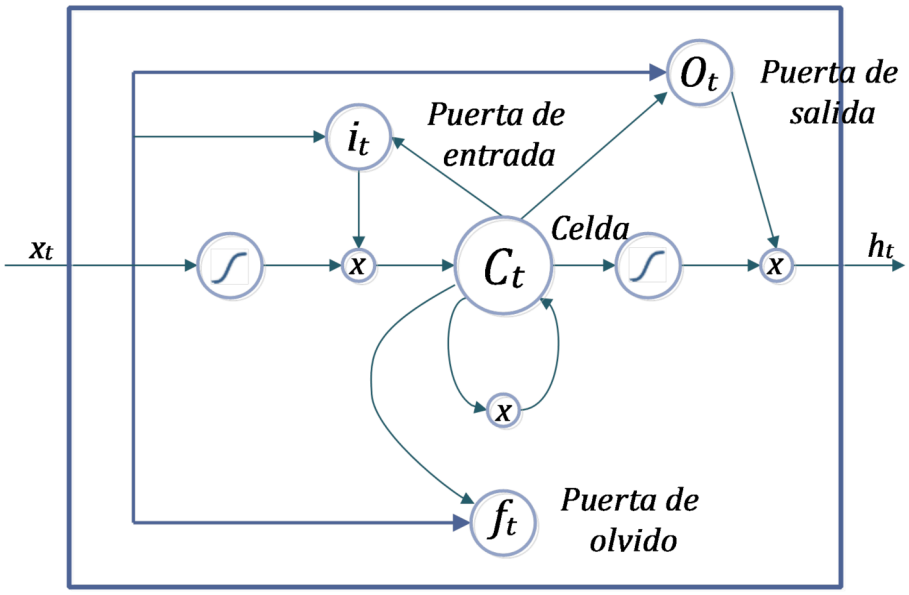



**Figura 2.7:** Celdas de Información LSTM

### 2.3.8 Submuestreo Aleatorio

El sub-muestreo aleatorio [35], tiene como objetivo equilibrar la distribución de clases mediante la eliminación aleatoria de ejemplos de clases mayoritarias. La razón detrás de esto es tratar de equilibrar un conjunto de datos. El principal inconveniente del sub-muestreo aleatorio es que este método puede descartar datos potencialmente útiles que podrían ser importantes para la generación de texto, tiene que ver con el entrenamiento de un modelo lógico que represente un conjunto de datos conocidos.

Siempre que la muestra se extraiga al azar, la distribución de la muestra se puede utilizar para estimar la distribución de los datos de donde se extrajo. Por lo tanto, al aprender la distribución de la muestra, podemos aprender a aproximarnos a la distribución objetivo. Sin embargo, una vez que realizamos un sub-muestreo de la clase mayoritaria, la muestra ya no puede considerarse aleatoria. Enlaces Tomek *(son pares de instancias opuestas que están muy juntas)* se pueden utilizar como método de sub-muestreo o como método de limpieza de datos. Como método de sub-muestreo, solo se eliminan los ejemplos que pertenecen a la clase mayoritaria y como método de limpieza de datos, se eliminan los ejemplos de ambas clases.

# 3 METODOLOGÍA

En determinadas circunstancias es preferible disponer de datos reales, con la probabilidad de no encontrar evidencias de fraude ya sea por la ausencia de este comportamiento en el conjunto de datos o la cantidad de datos para el análisis sea insuficiente. Otra alternativa es utilizar datos sintéticos los cuales pueden ser generados por un sistema simulando. También se puede incluir personas que realicen actividades simuladas en un determinado sistema, estas actividades significan que las personas realizan acciones de acuerdo con un procedimiento creado por los organizadores del experimento y obviamente no actúan de acuerdo con su comportamiento normal. Depende de la situación y las restricciones inherentes tal vez será mejor simular el comportamiento de personas utilizando un software autómata o contratar personas para generar datos que en nuestro caso reflejen actividad fraudulenta. Debido a la naturaleza y sensibilidad de la información se utiliza la primera opción para generar la data requerida mediante técnicas de aprendizaje profundo en un ambiente controlado [24].

La generación de datos sintéticos es una tarea complicada y demanda mucho tiempo para su ejecución, por lo que es necesario utilizar una metodología adecuada que permita optimizar el trabajo y además de establecer un procedimiento que viabilice las tareas a ejecutar. Las actividades y pasos necesarios para realizar el proceso de generación de datos se relacionan directamente con el comportamiento de una persona que tenga la intencionalidad de realizar fraude, por lo que utilizaremos la metodología propuesta por Emilie Lundin, Håkan Kvarnström, & Erland Jonsson [24], para adaptarla a nuestras necesidades, como se puede observar en la figura 3.1.

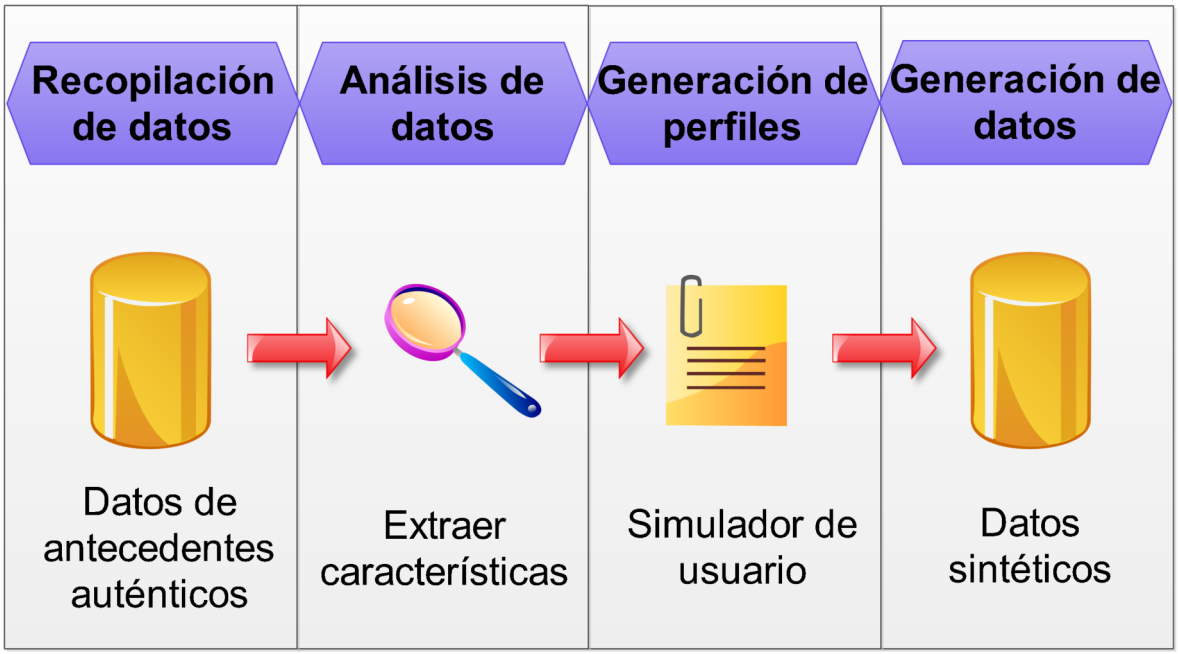



**Figura 3.1:** Metodología

## 3.1 METODOLOGÍA GENERAL

En primera instancia se recopilan datos que sirvan de línea base para el posterior análisis y utilización, los datos deben contemplar las características necesarias que representen el comportamiento previsto en el sistema destino o generador de frases. La información seleccionada puede consistir en datos de referencia auténticos, antecedentes validos de sistemas similares, ataques verificados y auténticos relacionados con el objeto de estudio, y otras colecciones de información referentes al proyecto. La segunda fase consiste en analizar los datos recopilados e identificar las propiedades importantes como clases asociadas al fraude, estadísticas de uso, características de ataques y estadísticas de comportamiento del sistema. A continuación, se utiliza la información obtenida anteriormente, la misma que se usará para identificar los parámetros que deben preservarse y poder generar frases, además de crear perfiles en base al triángulo del fraude que se ajusten a las estadísticas de los parámetros establecidos [24].

### 3.1.1 Recolección de Datos

La información inicial es el punto de partida en el proceso para la generación de datos sintéticos, deben ser muestras de datos representativas del comportamiento humano relacionado con el triángulo del fraude. Esta información debe ser producida u obtenida de

acuerdo con las características requeridas en función del comportamiento deseado para los datos de salida en la generación de frases. Por lo tanto, es conveniente tener muestras disponibles de datos auténticos, debido a que las obtenciones de los datos relacionados con el tema de estudio deben ser representativos. Los datos auténticos ayudan a mejorar la efectividad en el proceso de creación de datos. Si es demasiado pequeña la cantidad de datos obtenida, es posible que se deba determinar algún método alternativo para aumentar la información, por lo tanto, hay que determinar si la información es aplicable a nuestro caso de estudio. Para el presente proyecto se realizó la adquisición de un diccionario de datos *Textual Survey Word List 103115*, relacionado con el triángulo del fraude que lo construyó la empresa Audinet [36], la cual contribuye con la comunidad financiera ofreciendo recursos en línea donde auditores, contadores y profesionales de finanzas comparten herramientas y experiencias sobre programas de trabajo de auditoría. Este diccionario es una fuente valiosa de información para la generación de texto relacionada con el triángulo del fraude.

### 3.1.2 Análisis de Datos

El siguiente paso es analizar los datos recolectados. Una tarea es identificar clases de usuarios con un comportamiento similar. Análisis exploratorio de datos (EDA) [24], es un conjunto existente de ideas sobre cómo estudiar conjuntos de datos para descubrir la estructura subyacente, encontrar variables importantes, detectan anomalías, etc. Las técnicas basadas en estas ideas serán utilizadas en la sección 5 de este documento. Esto puede incluir el uso de herramientas de visualización y / o métodos de agrupación. Es importante usar la teoría del triángulo del fraude para determinar el comportamiento de los usuarios y obtener relevancia en los datos generados.

Otra tarea es identificar características importantes, es decir, parámetros que son útiles para la detección de fraude. Los datos generados por el sistema de destino deben ser examinados para determinar si son adecuados, caso contrario puede ser necesario implementar otros mecanismos.

### 3.1.3 Generación de Perfil

El siguiente paso es identificar los parámetros relevantes en el comportamiento de los datos de ingreso. Una manera de identificar estos parámetros es el estudio de las características necesarias para detectar fraudes. Estas características deben tener propiedades directamente relacionadas con el triángulo del fraude en los datos generados, para luego

ser utilizados en procesos de detección. Además, la correlación entre parámetros pueden ser indicadores precisos de un posible fraude.

Existen herramientas semiautomáticas disponibles para el cálculo de valores de los parámetros, sin embargo, utilizaremos la teoría del triángulo del fraude para su definición y análisis. La salida en esta etapa nos permitirá identificar un perfil adecuado para el análisis de actividades fraudulentas que contienen valores para todos los parámetros necesarios en la generación de frases. En esta fase también realizaremos el sub-muestreo del conjunto de datos de prueba para balancear la clase minoritaria con la mayoritaria a fin de equilibrar los datos de entrada.

### 3.1.4 Generación de Data

En esta fase se realizará el sub-muestreo del conjunto de datos de prueba para balancear la clase minoritaria con la mayoritaria a fin de equilibrar el conjunto de datos de prueba, que está compuesto de frases identificadas como fraude, que será la entrada para la generación de texto, mediante la aplicación de los algoritmos RNN y LSTM. Para limitar la complejidad en la generación del conjunto de datos, hay que tener en cuenta sólo los datos de interés para las acciones de simulación de datos, por eso es importante depurar los datos de entrada. En general es más sencillo modelar un comportamiento específico y bien delimitado con un previo conocimiento de su enfoque que realizarlo a ciegas, por lo que se realizó la división del conjunto de datos de prueba por vértice (Presión, Oportunidad y Racionalización) para delimitar los resultados.

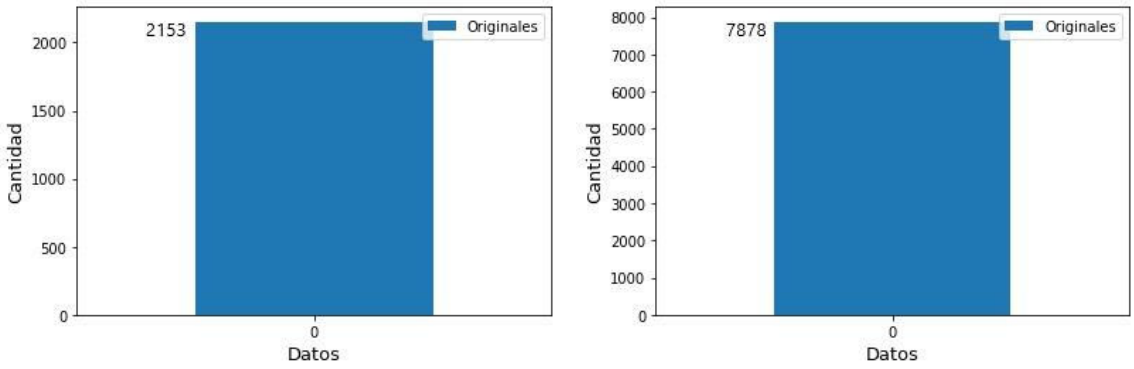
# 4 RESULTADOS Y DISCUSIÓN

En esta sección se presentan y analizan los resultados obtenidos de la ejecución y comparación de los algoritmos utilizados, aplicando técnicas de ML, mediante la organización de datos, aprendizaje de representación, ajuste del modelo y evaluación de datos. Es fundamental disponer de grandes cantidades de datos, para que técnicas de aprendizaje profundo pueden desarrollarse mejor y producir buenos resultados, particularmente en las aplicaciones donde la interpretación humana es difícil, es decir, estas técnicas conducen a la generación de texto de forma rápida e inteligente y también mejoran el proceso de toma de decisiones[37]. Por lo tanto, diseñar modelos eficientes de aprendizaje profundo que permitan producir buenos resultados predictivos es un problema desafiante en esta investigación.

## 4.1 ANÁLISIS Y DEPURACIÓN DEL CONJUNTO DE PRUEBA

*Textual Survey Word List 103115*, este diccionario está compuesto de 2,154 palabras (Figura 4.1); sirve como punto de partida en el método de generación de datos, ya que son palabras representativas del comportamiento humano relacionadas con el fraude.

La participación de personal de la Escuela Politécnica Nacional fue indispensable para el proyecto, a fin de que en un ambiente controlado y con ayuda del diccionario de datos [36], se realizo la generación de un conjunto de datos de prueba, el cual consta de 7,879 frases (Figura 4.1) y lleva de nombre: ***FraudTriangle_Stages***, este conjunto de datos es la entrada en el proceso de generación de datos sintéticos. La información obtenida debe ser relevante con el objetivo de obtener consistencia en la aplicación de los algoritmos y evitar el direccionamiento del resultado. Esta es la fuente más valiosa para la generación del conjunto de datos, por eso, es importante determinar si está información es aplicable a través de un análisis detallado de los datos recogidos.




(a) Textual Survey Word List 103115 (b) FraudTriengle_Stages

**Figura 4.1:** Conjuntos de Datos Originales

Los datos de Audinet [36] tienen inconsistencias, lo cual puede afectar el rendimiento del modelo. Es mejor identificar estas anomalías en los datos de entrada, lo que nos permitirá su corrección para un procesamiento y análisis más sencillo, por ello se realiza el análisis exploratorio que consiste en aplicar mecanismos de registro adicionales como: análisis manual del texto, depuración y recuperación de información, utilizados para obtener consistencia y veracidad en los datos. El análisis manual del texto implica revisar ortografía, significado y el vértice del triángulo del fraude al cual pertenece cada una de las palabras del diccionario de datos para determinar su consistencia. En la depuración de datos se procede a eliminar información distorsionada y repetida. Por último la recuperación de información relevante para generar frases que consiste en un segundo análisis de los datos a eliminar.

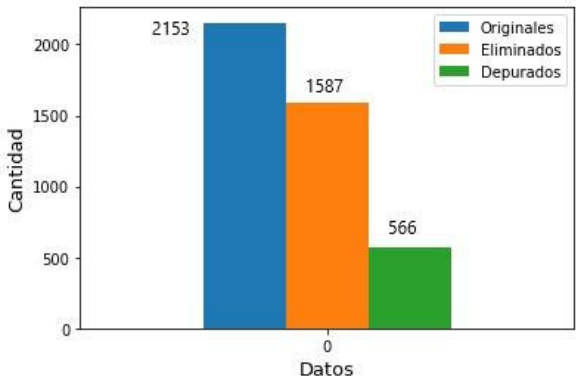



**Figura 4.2:** Textual Survey Word List 103115

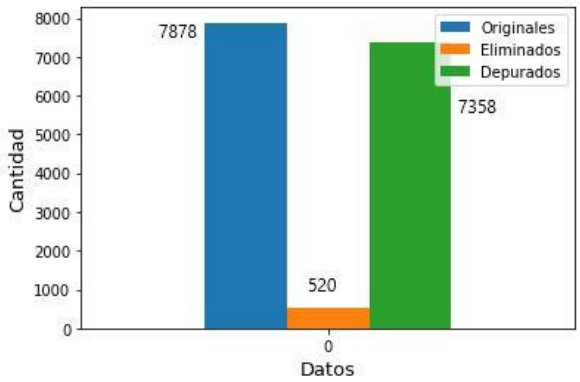



**Figura 4.3:** FraudTriangle_Stages

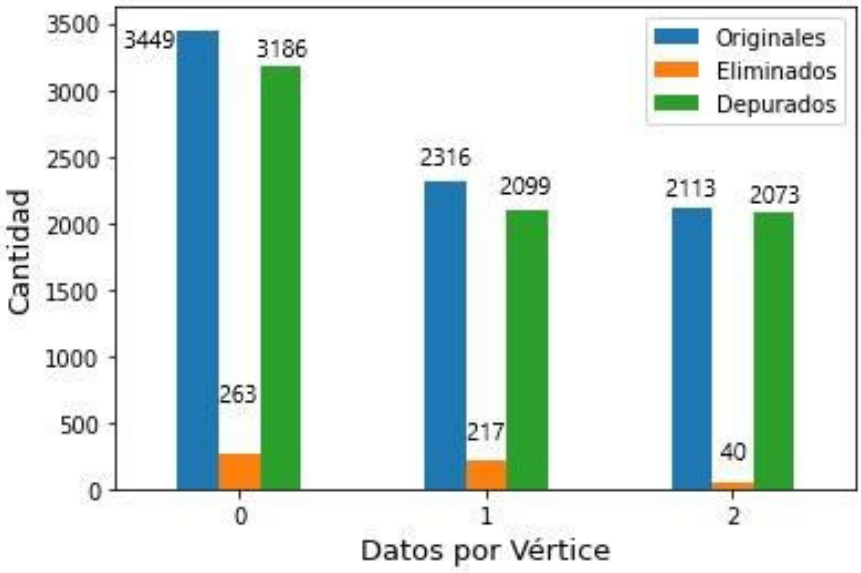



**Figura 4.4:** FraudTriangle_Stages Vértices

En la figura 4.2, se detalla que de las 2153 palabras que conforman el diccionario de datos, en la depuración se registran 1587 palabras inconsistentes que tras su eliminación, las palabras funcionales de este conjunto de datos son 566 [36]. El siguiente paso es generar el conjunto de datos de pruebas ***FraudTriangle_Stages***, a partir de las palabras funcionales del diccionario de datos. En este conjunto de datos también se realiza el análisis exploratorio y los resultados se muestra en la figura 4.3, en el análisis manual del texto el conjunto consta de 7878 frases, en la depuración se eliminan 506 frases inconsistentes, es decir, 7358 frases conformarán el conjunto de datos de prueba. Por último se realizó la división del conjunto de datos de prueba por vértice como se puede observar en la figura 4.4.

## 4.2 HERRAMIENTAS DE DESARROLLO

Los algoritmos desarrollados fueron ejecutados en la plataforma de ***Google Colab*** , la cual fue utilizada como entorno de desarrollo, misma que mediante un navegador permite realizar tareas de programación y depuración de código en lenguaje de programación Python. Se selecciono esta herramienta debido a las ventajas que brinda (No se requiere configuración local, libre acceso a CPUs en la nube y facilidad al compartir el contenido)

Python es un lenguaje de programación que permite trabajar rápidamente e integrar sus herramientas de manera eficiente.

## 4.3 ESTRUCTURA DE LOS SCRIPTS

A continuación, se detalla la estructura del código desarrollado y los algoritmos de ML utilizados. Cada uno de estos está constituido por diversos bloques de código.

### 4.3.1 Arquitectura con RNN

1. **Importar las bibliotecas.-** Como primer paso, se realizó la importación de varios módulos que se encuentran integrados en la biblioteca estándar de Python, que brindan acceso a la funcionalidad del sistema o brindan soluciones estandarizadas. Esta biblioteca es parte de cada instalación de Python, se accede a los módulos mediante la instrucción ***import***.

2. **Cargar el conjunto de datos.-** El acceso a los archivos se realiza utilizando el montaje de Google Drive, es decir, configurar la cuenta como una unidad virtual para que se pueda acceder a los recursos de la unidad como si fuera un disco duro local, en este caso previamente se realizó la carga del conjunto de datos relacionados con el triángulo del fraude analizados y depurados.

3. **Crear diccioario de datos.-** Un diccionario de datos es fundamental para que la investigación sea más reproducible porque permite comprender los datos. En Python, se puede crear un diccionario colocando una secuencia de elementos entre llaves{}, separados entre sí. El diccionario contiene un par de valores, uno es la clave y el otro elemento, el cual toma el nombre de tupla. Los valores de un diccionario pueden ser de cualquier tipo de datos y se pueden duplicar, mientras que las claves no se pueden repetir y deben ser inmutables.

4. **Modelo de la red RNN.-** RNN está diseñada para analizar flujos de datos mediante unidades ocultas. En este caso se utilizan 25 unidades ocultas para el procesamiento de texto, la salida depende de los cálculos anteriores, puede constar de una sola capa de neuronas y cada neurona devuelve su señal de salida a las entradas de todas las demás neuronas, no existen bucles de auto-retroalimentación en la red; auto-retroalimentación se refiere a una situación en la que la salida de la neurona se retroalimenta en su propia entrada.

5. **Generación de datos de entrenamiento.-** Para generar los datos de entrenamiento de la red neuronal recurrente, se utiliza la función random de NumPy *(es una biblioteca*

*de Python que se utiliza para trabajar con matrices)* que proporciona métodos convenientes para seleccionar frases aleatorias del conjunto de datos de prueba ***Fraud-Triengle_Stages***, estas frases servirán como datos de entrada en el entrenamiento.

6. **Entrenamiento del modelo.-** Se requiere un proceso de entrenamiento para que el algoritmo RNN pueda "aprender" a generar frases automáticamente que tengan consistencia. Este proceso de aprendizaje se realiza utilizando un modelo de la serie de modelos disponibles, que minimice el criterio de costo.

7. **Generación de texto.-** Una vez que se tiene la arquitectura modelo ya lista, se la puede entrenar usando nuestros datos. A continuación, se describe la función para predecir la siguiente palabra en función de las palabras de entrada.

```
1  ********************************************************************************
   -                    Algoritmo RNN para la generación de texto
   -  ********************************************************************************
   -   Require: String of all phrases
   5
   -   Inputs:
   -      model (variables:  Xt && at-1),
   -      char_to_int (Data dictionary)
   -      size_alphabet (Characters that make up the data)
  10      neurons_number
   -
   -   Output: Phrase generated
   -
   -   Initialization
  15      x = np.zeros((1,1, size_alphabet ,))
   -      a = np.zeros((1, n_a))
   -      phrase_generated = ''
   -      line_break = '\n'
   -      comparator = -1
  20      count = 0
   -
   -    while (comparator != line_break  and count != 50)
   -    do {
   -              a, _ =  recurring_cell (K.constant(x), initial_state=K.constant(a))
  25              y =  output_layer (a)
   -              prediction = K.eval(y)
   -              x = to_categorical(ix, size_alphabet).reshape(1,1, size_alphabet)
   -              a = K.eval(a)
   -              count += 1
```

```
30
-             if (count == 50):
-                     phrase_generated  += '\n'
-
-              return (phrase_generated )
35 }
```

La función ***generar_nombre*** es la encargada de generar las frases con la data depurada y entrenada. Esta función requiere como parámetros de entrada un modelo de entrenamiento "particularmente un modelo con redes RNN, un diccionario de datos de todas las palabras que contiene la data, un diccionario inverso para decodificar las frases generadas, el tamaño del alfabeto que contiene la data que en este caso es 30 y finalmente el número de neuronas que en este caso se han utilizado 25, además se ejecutará un ciclo while que termina cuando el siguiente carácter predicho sea un salto de línea o la frase llegue a las 50 palabras pudiendo aumentar esta cantidad, mientras tanto seguirá produciendo caracteres para generar frases. Para la generación de frases con el modelo previamente entrenado, se hará uso de la celda recurrente y la capa de salida, para inicial la predicción de introduce valores de ceros tanto para la entrada X y para el estado oculto anterior (Xt, at-1), posterior a esto se pasará la activación resultante a la capa de salida para generar la predicción. Finalmente se actualiza las entradas (Xt, at-1), convirtiéndose estos datos en la entrada para el siguiente instante de tiempo y el proceso se repite iterativamente hasta cumplir la condición del ciclo while.

### 4.3.2 Arquitectura con Redes de Memoria a Corto y Largo Plazo

1. **Importar las bibliotecas.-** Los módulos que están integrados en la biblioteca estándar de Python, brindan acceso a la funcionalidad del sistema o brindan soluciones estandarizadas. Estas bibliotecas de Python son parte de cada instalación, se accede a los módulos mediante la instrucción ***import***.

2. **Cargar el conjunto de datos.-** El acceso a los archivos se realiza utilizando el montaje de Google Drive, que significa configurar la cuenta como una unidad virtual para que podamos acceder a los recursos de la unidad como si fuera un disco duro local, en el que previamente se cargaron los datos relacionados con el triángulo del fraude analizados y depurados.

3. **Sobre muestreo.-** Debido a la poca cantidad de datos disponibles, se ha optado por implementar el método de sobre muestreó estadístico para de esta manera poder

ampliar el conjunto de datos de entrada y mejorar los resultados buscados.

4. **Preparación del conjunto de datos:**

   - **Limpieza del conjunto de datos.-** En el proceso de preparación de los datos, primero realizaremos la limpieza del texto de los datos, lo que incluye la eliminación de signos de puntuación y las palabras en minúscula.
   - **Generación de secuencia de tokens.-** El siguiente paso es la tokenización. La tokenización es un proceso de extracción de términos / palabras de un corpus (Conjunto de datos previamente cargado). La biblioteca de Python, Keras, tiene un modelo incorporado para la tokenización que se puede usar para obtener los tokens y su índice en el corpus. Después de este paso, todos los documentos de texto del conjunto de datos se convierten en una secuencia de tokens.

5. **LSTM para la generación de texto.-** Se implementa una red LSTM debido a la gran ventaja de la memoria ya que dicho modelo puede recordar u olvidar las inclinaciones de forma más selectiva, utilizando las siguientes capas:

   - Capa de entrada: toma la secuencia de palabras como entrada.
   - Capa LSTM: calcula la salida utilizando unidades LSTM. Se agregó 100 unidades en la capa, pero este número se puede ajustar más adelante.
   - Capa de abandono: una capa de regularización que apaga aleatoriamente las activaciones de algunas neuronas en la capa LSTM. Ayuda a prevenir el ajuste excesivo.
   - (Capa opcional) Capa de salida: calcula la probabilidad de la mejor palabra siguiente posible como salida.

6. **Entrenar el modelo.-** Requiere un proceso de entrenamiento para que el algoritmo LSTM pueda “aprender” a generar frases automáticamente que tengan consistencia. Este proceso de aprendizaje se realiza utilizando un modelo de la serie de modelos disponibles, que minimice el criterio de costo.

7. **Generación de texto.-** Una vez que se tiene la arquitectura modelo ya lista, se puede entrenarla usando nuestros datos. A continuación, se escribe la función para predecir la siguiente palabra en función de las palabras de entrada.

```
*****************************************************************************
                    Algoritmo LSTM para la generación de texto
*****************************************************************************
 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)
    seedCNAT = all_phrases[start]
    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() do {
                if index == predicted:
                    output_word = word
                    break
          }
        seed_textQ += " "+output_word
        return seed_textQ.title()
    }
```

La función ***generate_phrases*** requiere como parámetros de entrada un modelo de entrenamiento “particularmente un modelo con redes LSTM", una palabra que servirá como semilla y un entero que indique cuantas palabras va a contener la frase generada, cabe recalcar que los parametros mencionados se generan automáticamente de manera aleatoria. La función toma la semilla ingresada y se basa en el modelo previamente entrenado para predecir la siguiente palabra que concuerde con la semilla, proceso que se repite en reiteradas ocasiones hasta llegar al número de palabras solicitadas.

## 4.4 RESULTADOS

La generación de texto se lleva a cabo, vértice a vértice independientemente del algoritmo utilizado, con el propósito de evitar inconsistencia en los datos recopilados, así mismo, evitar efectos adversos en su funcionamiento.

### 4.4.1 Presentación de Resultados del Algoritmo RNN

La generación de texto mediante el algoritmo RNN, en el cual, se entrena un modelo de red neuronal para predecir frases a partir de una secuencia de palabras, con lo que se consigue generar secuencias de texto más largas llamando al modelo repetidamente a través de un bucle que permite que la salida de la red o parte de ella sirva como entrada de la propia red en el siguiente momento, además, se indica la cantidad de frases a obtener, en este caso, 1500 frases orientadas a cada vértice (Presión, Oportunidad, Racionalización).

| ***Resultados del Algoritmo RNN*** |
|---|
| olaunss |
| niwrelhznad y nt te i hiehle oinw |
| sepaauo t y mqli aid |
| ahohustek |
| ahv |
| nfrpymeadsiotayoliy ugnimosey nmnpu ywdint ia ah |
| oayautnimeuni ivos |
| espedemuwatd inalyea anyteabui |
| awiynuaiw ierve n ib |
| t n un |
| hpmy cn wse l |
| htemihhiteof |
| r fely lahric |
| e a lle |
| dm |
| ee tlnayeiye hea |
| elaay c |
| ilayqyue |
| nthvimo meme |
| enwhmntsezpe ei gnk |
| slpzsdowepwaacubhineltn aleveu a eaceefnla |
| s |
| rvmduqlthmftpfy lestanr sz mumhece |
| teeagcrckot ushnhwebhz cenmtaoveopltplu viaas tifz |
| eelfsv iillliumejsca oasnavibui wcue hmdifsheyaew |
| ohsfmmeanemh |
| aluoy ltsuenl yo iaciiitinlarimems |
| anueatcfheethln |
| ene |

**Tabla 4.1:** Resultados del Algoritmo RNN.

El algoritmo es ejecutado en la plataforma Google Colab, al finalizar la ejecución del programa que genera las frases y se exportan los resultados a una hoja Excel con el fin de almacenar los datos de forma local para facilitar el acceso a la información como se muestra en la tabla 4.1.

### 4.4.2 Presentación de Resultados del Algoritmo LSTM

El algoritmo desarrollado con redes LSTM tienen una estructura un poco más complicada, amplía su memoria para aprender que recordar y que olvidar, los datos ya están preprocesados en el formato que se requiere para ser usado en el entrenamiento de la red neuronal, el modelo aprendió en pocas iteraciones, cuales son las frases orientadas a cada vértice (Presión, Oportunidad, Racionalización).

| ***Resultados del Algoritmo LSTM*** |
|---|
| I Got It Wrong What I Did. Not Have To Pay For The Transportation Of My |
| 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. So I Will Have To Mortgage My |
| My Salary Is Not Enough To Cover With These Fats. But Those Holidays Are |
| For The Bank To Give Me The Loan, I Will Have To Mortgage My House. So I Will Have To Mortgage My |
| I Ran Out Of Internet For Not Canceling The Monthly Fee For The Secret Business |
| They Owe Me Money And I Have Nothing. Money To Pay |
| We Must Take Out An Express Loan To Pay |
| I Need Those Products For The Weekend However I Do Not Have Money |
| We'Ll Just Ask Him For A Little Money. From The Company And Are Expensive Time And I Do Not |
| We Have To Be Careful With This. Month I Have No Money To |
| Someone Knows What I'M Doing. Someone Down The Police |
| Time Is Measured In The |
| It Is Impossible To Deal With This Situation With The |
| Good Handling That You Did With The Manager, We Managed To Keep The Business. Because I |
| Tell Them You'Re Sick And You Can Not Be In The Audit. To Pay The Debt But I |
| You Don'T Think Bosses Are Going To Coax You For What You'Re Doing Possible And |
| I Pawned My Stuff In Exchange For Money. From The Company Is Quite Complex To Give Me The |
| You Must Collect The Debt To Invest |
| His Behavior Is Different Things In The Inventory Report That The Manager Of The Company |
| Are Disproportionate Time To The Office For The Party That I Do |
| The Salary I Have Is Not Enough To Buy A New Car. I Will Be In The Audit Center Due To The |
| If I Don'T Do What The Boss Tells Me, He Will Fire Me Out With The Other Expenses Of The House |
| I Want The Best Cheerleader For My Party. And Work Is Very Expensive And |
| I'M Having A Really Bad Time. Of The Software That |
| The Lease Of The House Went Up Again. I Do Not Have |
| This Is Not Coherent. Promoted Of The Loan Nobody Will Fire Me For The House But I Have A |
| It'S Quick And Easy Money. To Pay The Debt I Have To Pay |
| Seem Mysterious Gets Something By Accident Is |

**Tabla 4.2:** Resultados del Algoritmo LSTM

En el algoritmo LSTM se efectuaron las mismas actividades técnicas que se realizaron en el algoritmo RNN como presenta la tabla 4.2.

## 4.5 DISCUSIÓN

### 4.5.1 Herramientas de evaluación del texto generado

Para realizar una correcta comparación, enfocada en la eficiencia de los algoritmos desarrollados, se necesita métricas para evaluar el desempeño de cada uno de estos algoritmos, así como, métricas que sustenten la legibilidad de los datos de entrada utilizados en cada algoritmo, por lo tanto, se ha resuelto utilizar la herramienta ***Readable*** para obtener métricas orientadas principalmente a la ortografía y la gramática.

Readable es una plataforma en línea que permite una revisión rápida de la legibilidad, ortografía y gramática de texto que mostrará cómo y dónde realizar mejoras. El mayor aporte de esta herramienta para el presente proyecto es que la herramienta indica el puntaje que amerita el texto ingresado como muestra la figura 4.5.

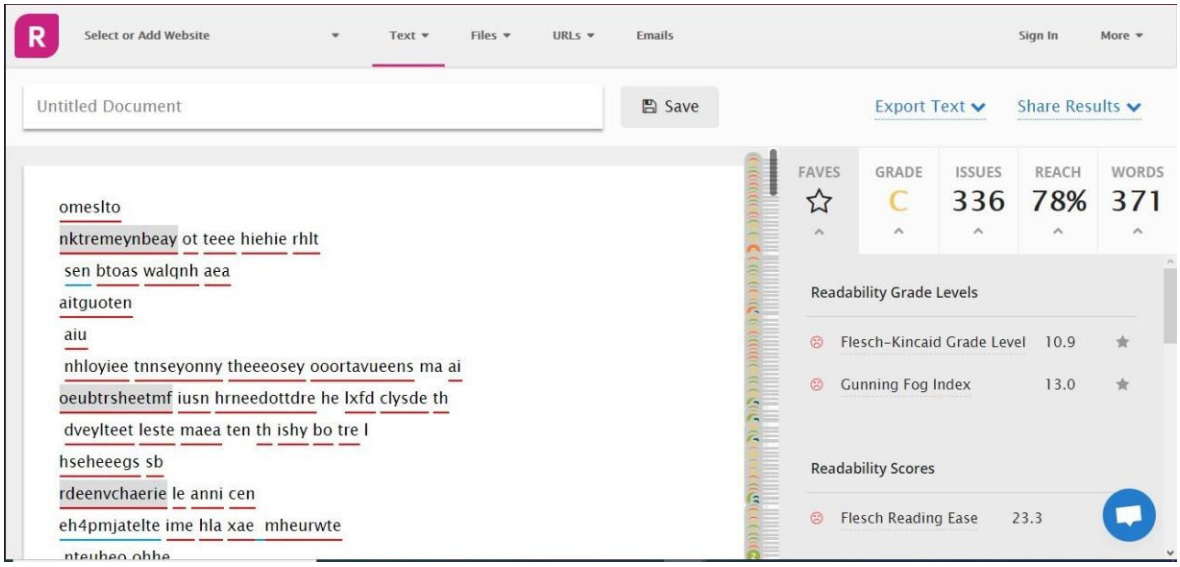



**Figura 4.5:** Legibilidad Algoritmo RNN

El puntaje que indican la plataforma, servirá para llevar a cabo el análisis y comparación entre los algoritmos desarrollados (RNN y LSTM), con respecto al conjunto de datos original, y de esa manera poder saber cuál es más óptimo al momento de generar frases orientadas al triángulo del fraude.

### 4.5.2 Obtención de métricas

Para el proceso de obtención de métricas, se ingresan datos a la herramienta ***Readable***, en rangos de 0-100 hasta llegar a los 1000 datos, una vez que se llegaron a 1000 datos se aumentó el rango de 0-500, teniendo como máximo 1500 datos ingresados a las herramientas, debido a que solo se obtuvo una licencia trial, la cual no permitía más datos para analizar.

En cada iteración se fueron recaudando los resultados arrojados por las herramientas, y almacenando en una hoja Excel para posteriormente realizar las gráficas comparativas correspondientes a cada uno de los algoritmos.

#### 4.5.2.1 RNN

**Presentación de Resultados**

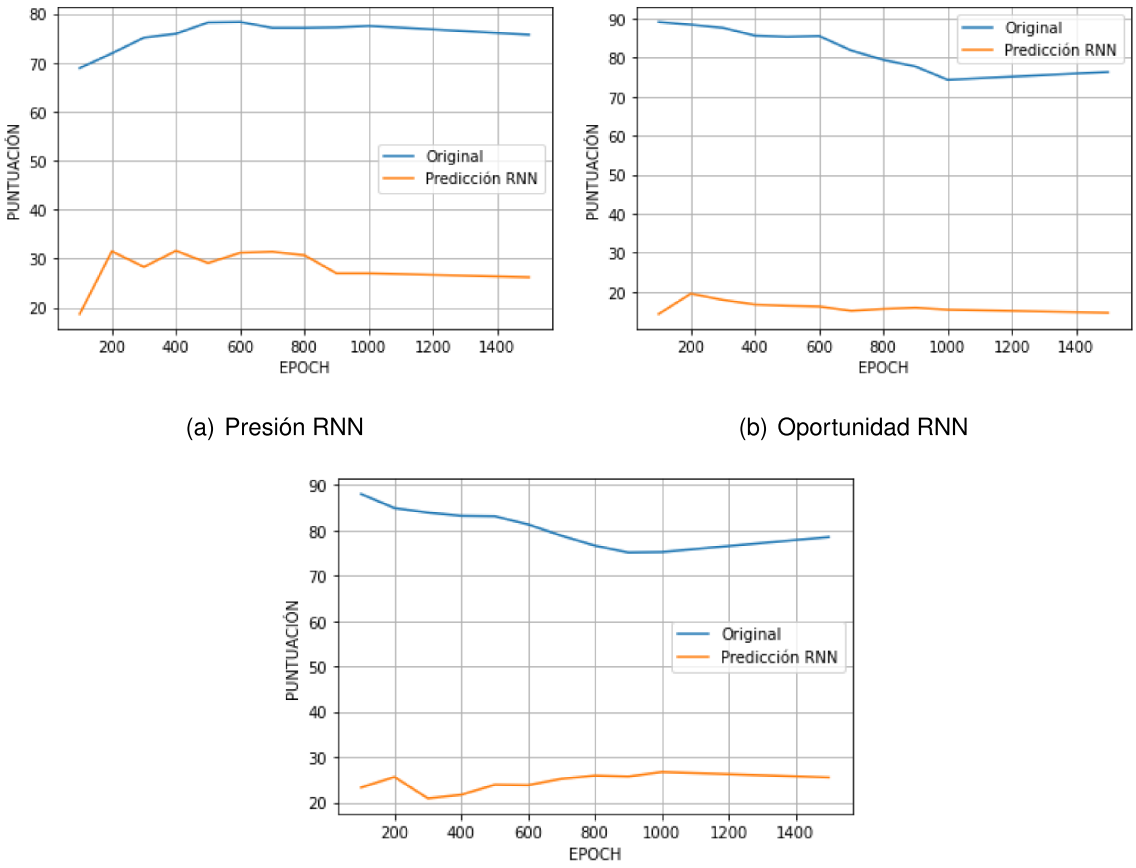
A continuación, se describen los resultados de la consistencia del algoritmo RNN aplicado en cada uno de los vértices del Triángulo del Fraude (Presión (Vértice I), Oportunidad (Vértice II) y Racionalización (Vértice III)), mediante la herramienta Readable. Esta herramienta fue utilizada para realizar el análisis de la consistencia de los datos, ya que analiza rápidamente la legibilidad, la ortografía y la gramática del conjunto de datos obtenido tras la utilización del algoritmo RNN como se muestra en la tabla 4.3.

| Cantidad | Vértice I | Vértice I RNN | Vértice II | Vértice II RNN | Vértice III | Vértice III RNN |
|---|---|---|---|---|---|---|
| 100 | 70.1 % | 18.7 % | 89.1 % | 14.3 % | 88 % | 23.3 % |
| 200 | 71.9 % | 31.5 % | 88.4 % | 19.5 % | 84.9 % | 25.6 % |
| 300 | 75.1 % | 28.3 % | 87.6 % | 17.9 % | 83.9 % | 20.9 % |
| 400 | 75.9 % | 31.6 % | 85.6 % | 16.7 % | 83.2 % | 21.7 % |
| 500 | 78.2 % | 29.1 % | 85.3 % | 16.4 % | 83.1 % | 23.9 % |
| 600 | 78.3 % | 31.2 % | 85.5 % | 16.2 % | 81.3 % | 23.8 % |
| 700 | 77.1 % | 31.4 % | 81.8 % | 15.1 % | 78.8 % | 25.2 % |
| 800 | 77.1 % | 30.7 % | 79.4 % | 15.6 % | 76.6 % | 25.9 % |
| 900 | 77.2 % | 27 % | 77.7 % | 15.9 % | 75.1 % | 25.7 % |
| 1000 | 77.5 % | 27 % | 74.3 % | 15.4 % | 75.2 % | 26.7 % |
| 1500 | 75.7 % | 26.2 % | 76.3 % | 14.6 % | 78.5 % | 25.5 % |

**Tabla 4.3:** Concistencia del texto usando la Herramienta Readable para el Algoritmo RNN

En las gráficas 4.6. se puede observar la comparativa entre los resultados obtenidos al

analizar cada uno de los vértices del triángulo del fraude para el Algoritmo RNN y el conjunto de datos original, los datos recogidos por el algoritmo, tienen un score por debajo del 40 % y el conjunto de datos original una puntuación por encima del 70 %, con lo cual, se demuestra que específicamente el algoritmo RNN es ineficiente, por lo cual no será utilizado en este proyecto.




(a) Presión RNN

(b) Oportunidad RNN

(c) Racionalización RNN

**Figura 4.6:** Gráficas del Algoritmo RNN utilizando Readable

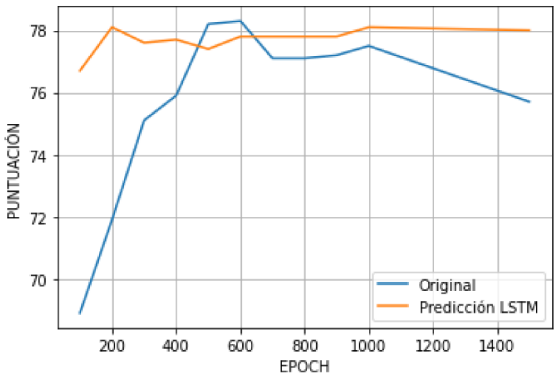
#### 4.5.2.2 LSTM

**Presentación de Resultados**

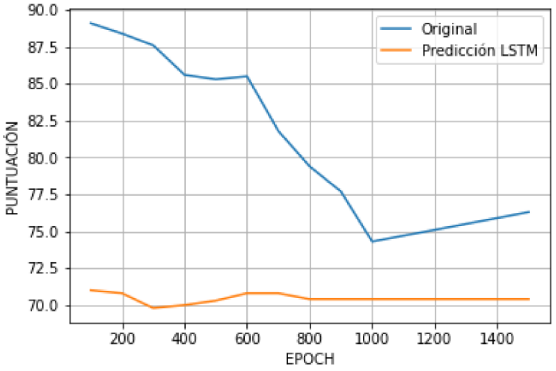
La herramienta Readable se utilizó para obtener resultados de la legibilidad de texto obtenido tras la aplicación del Algoritmo LSTM a cada vértice del Triángulo del Fraude (Presión (Vértice I), Oportunidad (Vértice II) y Racionalización (Vértice III)) con respecto al conjunto de datos original, para luego proceder a realizar una comparativa entre los dos algoritmos utilizados en este proyecto.

| Cantidad | Vértice I | Vértice I LSTM | Vértice II | Vértice II LSTM | Vértice III | Vértice III LSTM |
|---|---|---|---|---|---|---|
| 100 | 70.1 % | 76.7 % | 89.1 % | 71 % | 88 % | 77.7 % |
| 200 | 71.9 % | 78.1 % | 88.4 % | 70.8 % | 84.9 % | 77.2 % |
| 300 | 75.1 % | 77.6 % | 87.6 % | 69.8 % | 83.9 % | 77.1 % |
| 400 | 75.9 % | 77.7 % | 85.6 % | 70 % | 83.2 % | 77.4 % |
| 500 | 78.2 % | 77.4 % | 85.3 % | 70.3 % | 83.1 % | 76.8 % |
| 600 | 78.3 % | 77.8 % | 85.5 % | 70.8 % | 81.3 % | 76.7 % |
| 700 | 77.1 % | 77.8 % | 81.8 % | 70.8 % | 78.8 % | 76.8 % |
| 800 | 77.1 % | 77.8 % | 79.4 % | 70.4 % | 76.6 % | 76.8 % |
| 900 | 77.2 % | 77.8 % | 77.7 % | 70.4 % | 75.1 % | 76.9 % |
| 1000 | 77.5 % | 78.1 % | 74.3 % | 70.4 % | 75.2 % | 77 % |
| 1500 | 75.7 % | 78 % | 76.3 % | 70.4 % | 78.5 % | 77.2 % |

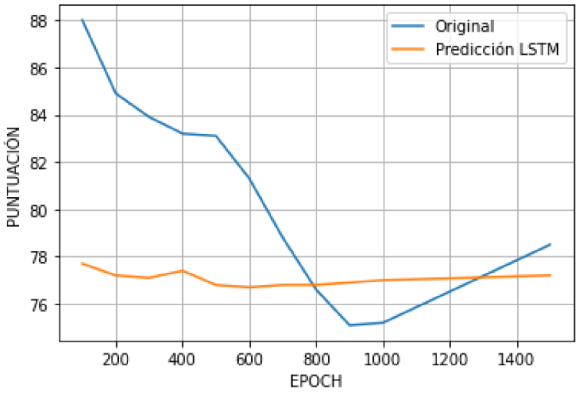
**Tabla 4.4:** Consistencia del texto usando la Herramienta Readable para el Algoritmo LSTM




(a) Presión LSTM




(b) Oportunidad LSTM




(c) Racionalización LSTM

**Figura 4.7:** Gráficas del Algoritmo LSTM utilizando Readable

Las gráficas 4.7 presentan la comparativa entre las puntuaciones obtenidas por el Algoritmo LSTM para cada vértice del triángulo del fraude y el conjunto de datos original, los datos recogidos por el algoritmo tienen un score por encima del 70 % al igual que el conjunto de datos original, con lo cual, se asume que el texto generado mediante el algoritmo LSTM tiene consistencia.

#### 4.5.2.3 RNN vs LSTM

**Comparativa**

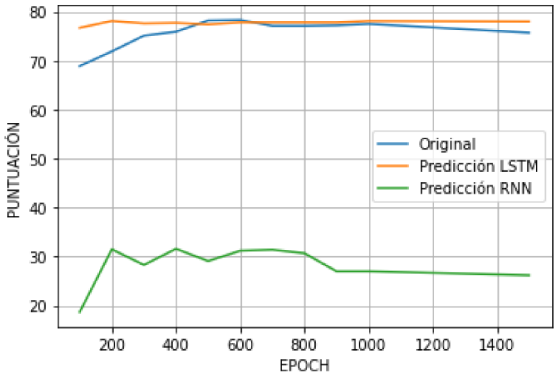
A continuación en la tabla 4.5, se muestran los resultados obtenidos por los algoritmos RNN y LSTM, basándose en los puntajes obtenidos al ejecutar los datos procesados por dichos algoritmos en la herramienta ***Readable***, se realiza una comparativa entre los resultados obtenidos para cada algoritmo y el conjunto de datos original, los puntajes se encuentran comprendidos entre 0 y 100 puntos, versus la cantidad de frases analizadas que se encuentran entre 100 y 1500 frases.

| Cantidad | Presión RNN | Presión LSTM | Oportunidad RNN | Oportunidad LSTM | Racionalización RNN | Racionalización LSTM |
|---|---|---|---|---|---|---|
| 100 | 18.7 % | 76.7 % | 14.3 % | 71 % | 23.3 % | 77.7 % |
| 200 | 31.5 % | 78.1 % | 19.5 % | 70.8 % | 25.6 % | 77.2 % |
| 300 | 28.3 % | 77.6 % | 17.9 % | 69.8 % | 20.9 % | 77.1 % |
| 400 | 31.6 % | 77.7 % | 16.7 % | 70 % | 21.7 % | 77.4 % |
| 500 | 29.1 % | 77.4 % | 16.4 % | 70.3 % | 23.9 % | 76.8 % |
| 600 | 31.2 % | 77.8 % | 16.2 % | 70.8 % | 23.8 % | 76.7 % |
| 700 | 31.4 % | 77.8 % | 15.1 % | 70.8 % | 25.2 % | 76.8 % |
| 800 | 30.7 % | 77.8 % | 15.6 % | 70.4 % | 25.9 % | 76.8 % |
| 900 | 27 % | 77.8 % | 15.9 % | 70.4 % | 25.7 % | 76.9 % |
| 1000 | 27 % | 78.1 % | 15.4 % | 70.4 % | 26.7 % | 77 % |
| 1500 | 26.2 % | 78 % | 14.6 % | 70.4 % | 25.5 % | 77.2 % |

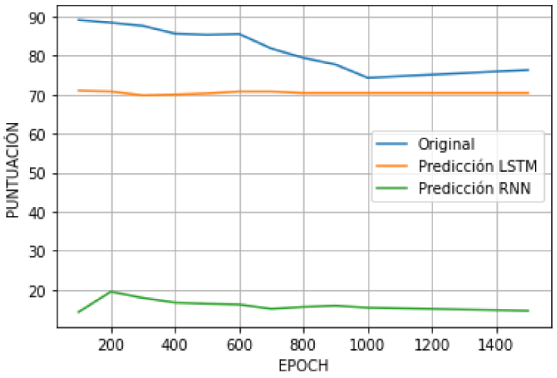
**Tabla 4.5:** Comparativa entre la consistencia del texto usando el Algoritmo RNN vs LSTM

Como muestra la figura 4.8, se realiza una gráfica por separado para cada vértice del triángulo de fraude, en cada figura se representa con la línea de color naranja el algoritmo LSTM, con la línea dibujada de color verde el algoritmo RNN y la data original con la línea en color azul, además si dejamos de lado por un momento la puntuación y nos centramos en la cantidad de frases analizadas, podemos confirmar que para que un algoritmo arroje resultados

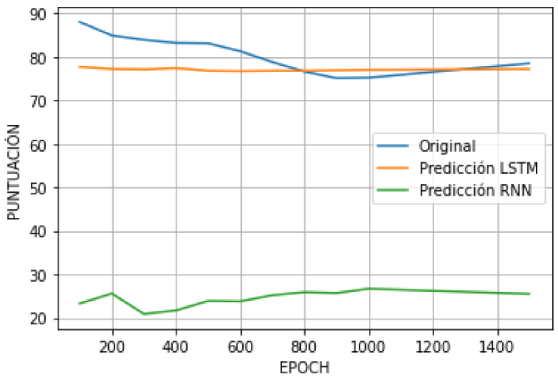
más precisos se necesitan grandes cantidades de datos, por lo que si nos fijamos en la gráfica, podemos observar que a partir de los 1000 datos, los puntajes no hacen variaciones significantes, por el contrario, antes de los 1000 datos, dichos puntajes son muy variables.




(a) Presión RNN vs LSTM




(b) Oportunidad RNN vs LSTM




(c) Racionalización RNN vs LSTM

**Figura 4.8:** Comparativa entre los Algoritmos RNN vs LSTM

Finalmente, después de analizar las gráficas se puede observar claramente que, en este caso específico correspondiente al procesamiento de texto, el algoritmo LSTM tiene mayor efectividad al momento de generar texto consistente, por lo que este proyecto ayuda al sustento de dicha teoría, junto con la parte práctica. Por otra parte, se observa que sin importar el vértice al que corresponda, el algoritmo LSTM presenta una notable mejora del score al momento de la generación de frases relacionadas con el triángulo del fraude.

# 5 CONCLUSIONES Y RECOMENDACIONES

## 5.1 CONCLUSIONES

Los algoritmos de aprendizaje profundo dependen principalmente de grandes cantidades de datos de entrenamiento, por eso en ocasiones, la red neuronal recibe muestras de datos sintéticos o fabricados, debido a que es muy difícil conseguir datos reales de entidades que hayan sido víctimas de algún ataque fraudulento, por miedo a que su reputación se vea afectada, indisponibilidad de datos y lo más importante por preservar la privacidad. Sin embargo, como se ha demostrado en este proyecto, mediante la utilización del algoritmo LSTM se logra conseguir obtener datos relevantes para el análisis de fraude.

En comparación con el algoritmo LSTM, la red neuronal RNN produce errores relativamente grandes sin entrenamiento, esto muestra que la consideración de datos históricos (memoria), no tienen ninguna influencia en el entrenamiento del modelo. El costo de la adquisición de datos es prohibitivo e inevitablemente nos enfrentamos al desafío al generar datos coherentes y tomar decisiones bajo información parcial.

El conjunto de datos sintéticos obtenidas mediante el desarrollado del algoritmo LSTM en el presente proyecto, contribuye a la creación de una base de conocimiento para futuras investigaciones, debido a que está sustentada bajo la teoría del triángulo del fraude. La cual aporta elementos para el desarrollo de un sistema predictor de fraude a nivel empresarial.

Los conjuntos de datos recopilados mediante una fase de análisis, permitieron la identificación de incoherencias y anomalías, las mismas que se corrigieron o eliminaron en la fase de depuración. De esta manera, fue posible obtener conjuntos de datos base limpio para poder entrenar la red y de esa manera evitar el direccionamiento hacia un lado determinado de los datos, generando frases más claras y coherentes.

Las figuras comparativas, entre los algoritmos RNN y LSTM, permiten visualizar el rendimiento de ambos con respecto a la generación de texto, para determinar qué Red Neuronal es mejor para este propósito en específico. La red LSTM con respecto a RNN obtuvo mejo-

res resultados en la herramienta Readable, es decir, según las métricas obtenidas se puede asegurar que la red más idónea para la generación de texto es LSTM. Sin embargo, cabe recalcar que los mejores resultados se obtienen con cantidades grandes de datos.

## 5.2 RECOMENDACIONES

El número de interacciones para alcanzar resultados óptimos varía entre los diferentes algoritmos propuestos, sin embargo, la diferencia con los resultados del modelo sin sobremuestreo es significativa. Por dicha razón, cuando se dispone de un conjunto de datos pequeño o con datos des-balanceados lo más óptimo es realizar sobremuestreo antes de entrenar al algoritmo.

Otra caso a tener en cuenta es no sobreentrenar la Red Neuronal, ya que no siempre una mayor cantidad de iteraciones significa mayor precisión en los resultados. Por ejemplo, en el algoritmo LSTM desarrollado en el presente proyecto, al intentar con 150 iteraciones los resultados no tuvieron gran diferencia que con 100.

Tener en cuenta las limitaciones de los recursos empleados en el proyecto, como son los host utilizados en la fase de desarrollo y pruebas, ya que pueden presentar obstáculos al momento de realizar el proyecto. Por esta razón se utilizó la plataforma de Google Colab ya que facilita recursos de almacenamiento los que permiten guardar y procesar cantidades grandes de datos, que se utilizaron en la presente investigación.

Hay que tener en cuenta las limitaciones de las herramientas utilizadas para la evaluación de la eficiencia de las frases generadas por los algoritmos, ya que las licencias trial que otorgan dichas empresas, tienen capacidad limitada de ingreso de texto para procesar lo que representa una mayor inversión de tiempo y recursos para poder obtener los resultados deseados.

La variedad de optimizadores disponibles para la implementación de redes neuronales puede hacer que el desarrollador tienda a confundirse entre cual es mejor o peor para la red, por lo que en este proyecto se puede aportar que para la generación de texto el algoritmo Adam (Adaptive moment estimation) responde bien al momento del entrenamiento de los datos ya que combina las bondades de los algoritmos AdaGrad y RMSProp.

# 6 REFERENCIAS BIBLIOGRÁFICAS


[1] Acfe-spain.com, *ACFE Association of Certified Fraud Examiners Capítulo España*, Available: https://acfe-spain.com, Accedido 06-11-2019.

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

[3] S. Y. J. Guan R. Li y 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 y 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, Accedido 06-11-2019.

[5] P. T. M. Mann O. P. Sangwan y 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 y 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 y M. Magnor, *Synthetic Generation of High-Dimensional Datasets, IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 12.*

[8] P. R. Bing Wang y 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 y A. E. J. Choo H. Park, *AxiSketcher: Interactive Nonlinear Axis Mapping of Visualizations through User Drawings, IEEE Transactions on Visualization and Computer Graphics, vol. 23, no. 1, pp. 221–230, 1 2017*.

[10] Y. Y. T. R. Liu B. Fang y 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 y R. X. D. Holt, *Development of a Synthetic Data Set 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 y B. Samadi, *"Synthetic Data Generation Capabilties for Testing Data Mining Tools,"in MILCOM. IEEE, 2006, pp. 1–6.*

[13] M. A. A. M. Pasinato C. E. Mello y 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 y 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 y 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 y 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 y 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, Accedido 07-07-2020.

[18] X. Z. Brian C. Hosler, C. C. Owen Mayer y M. C. S. James A. Shackleford, *"The Video Authentication And Camera Identification Database: A New Database For Video Forensics - IEEE Journals & Magazine"*, Available: https://ieeexplore.ieee.org/document/8734060, Accedido 07-07-2020.

[19] Y. Z. Hangxia Zhou, K. Y. Lingfan Yang Qian Liu e 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, Accedido 07-07-2020.

[20] S. W. S. Ahmadreza Argha Ji Wu y 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, Accedido 07-07-2020.

[21] M. A. Hamdi Altaheri y G. Muhammad, *"Date Fruit Classification For Robotic Harvesting In A Natural Environment Using Deep Learning - IEEE Journals & Magazine"*, Available: https://ieeexplore.ieee.org/document/8807111, Accedido 07-07-2020.

[22] C. L. Xun Zhu y D. Ji, *"Keyphrase Generation With Copynet And Semantic Web - IEEE Journals & Magazine"*, Available: https://ieeexplore.ieee.org/document/9019613, Accedido 07-07-2020.

[23] G. Z. Zhaohui Che Ali Borji y P. L. C. Xiongkuo Min Guodong Guo, *"How Is Gaze Influenced By Image Transformations? Dataset And Model - IEEE Journals & Magazine"*, Available: https://ieeexplore.ieee.org/document/8866748, Accedido 07-07-2020.

[24] H. K. E. Lundin y E. Jonsson, *"A Synthetic Fraud Data Generation Methodology"*, Accedido 07-07-2020.

[25] C. R. d. S. Y. dos Santos Brito y S. Mendonça, *"A Prototype Application to Generate Synthetic Datasets for Information Visualization Evaluations"*, Accedido 07-07-2020.

[26] N. M. Rabi'u Abdullahi, *Fraud Triangle Theory and Fraud Diamond Theory. Understanding the Convergent and Divergent For Future Research*, Available: 10.6007/IJARAFMS/v5-i4/1823, Accedido 06-11-2019.

[27] A. R. Fernández, *"Los datos sintéticos, la clave para mejorar la Inteligencia Artificial", Addison Wesley College, 1997*.

[28] A. G. Huerta, *"Algoritmos de Clasificación para Datasets Desequilabrados: Análisis y Comparativa", Proyecto de Fin de Grado en Ingeniería de Computadores*.

[29] I. Mufti Mahmud Senior Member, I. Mohammed Shamim Kaiser Senior Member, I. Amir Hussain Senior Member y S. Vassanelli, *Applications of Deep Learning and ReinforcementLearning to Biological Data, IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 6, 2018*, Available: 10.1109/TNNLS.2018.2790388, Accedido 06-11-2019.

[30] D. J. Matich, *"Redes Neuronales: Conceptos Básicos y Aplicaciones.", Informática Aplicada a la Ingeniería de Procesos – Orientación I*.

[31] J. Z. Ruiqin Bai, X. L. Dengao Li y 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, Accedido 11-12-2020.

[32] F. D. D. Raví C. Wong, J. A. M. Berthelot y G. Y. B. Lo, *"Deep learning for health informatics", IEEE Journal of Biomedical and Health Informatics 21 (1) (2017) 4–21*.

[33] S. Team, *Recurrent Neural Networks (RNN) - The Vanishing Gradient Problem*, Available: https://www.superdatascience.com/blogs/recurrent-neural-networks-rnn-the-vanishing-gradient-problem, Accedido 06-11-2019.

[34] S. D. S.Dhananjay Kumar, *Prediction of depression from EEG signal using Long Short Term Memory(LSTM), IEEE 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Available: 10.1109/ICOEI.2019.8862560, Accedido 06-11-2019.

[35] P. P. S. Kotsiantis, *"Mixture of Expert Agents for Handling Imbalanced Data Sets", Annals of Mathematics, Computing  TeleInformatics, Vol 1, No 1 (46-55), 2003.*

[36] AuditNet, *"Using Key Word Analysis of an Organization's Big Data For Error and Fraud Detection".* urlhttps://www.auditnet.org/key-word-analytics.

[37] D. Q. H. H. Xue y M. Reynolds, *"SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction", Department of Computer Science and Software Engineering The University of Western Australia, Perth, Australi, 2018.*

# 7 ANEXOS

## 7.1 ANEXO I. ARTÍCULO CIENTÍFICO

### 7.1.1 NOTIFICACIÓN DE RECEPCIÓN CONFERENCIA

**De:** SAM'21 <sam21@easychair.org>
**Enviado:** miércoles, 3 de marzo de 2021 17:41
**Para:** VERONICA ELIZABETH OLMEDO VELEZ
**Asunto:** SAM'21 submission 1

Dear authors,

We received your submission to SAM'21 (The 2021 International Conference on Security and Management):

Authors : Veronica Olmedo, Carlos Narvez, Marco Sanchez and Myriam Hernadez
Title : Construction of a synthetic dataset applyingMachine Learning techniques
Number : 1

The submission was uploaded by Marco Sanchez <marco.sanchez01@epn.edu.ec>. You can access it via the SAM'21 EasyChair Web page

https://easychair.org/conferences/?conf=sam21

Thank you for submitting to SAM'21.

Best regards,
EasyChair for SAM'21.

**Figura 7.1:** Notificación de recepción a veronica.olmedo@epn.edu.ec

From: SAM'21 <sam21@easychair.org>
Date: Wed, Mar 3, 2021, 5:41 PM
To: CARLOS ANDRES NARVAEZ TELLO <carlos.narvaez02@epn.edu.ec>
Subject: SAM'21 submission 1

Dear authors,

We received your submission to SAM'21 (The 2021 International Conference on Security and Management):

Authors : Veronica Olmedo, Carlos Narvez, Marco Sanchez and Myriam Hernadez
Title : Construction of a synthetic dataset applyingMachine Learning techniques
Number : 1

The submission was uploaded by Marco Sanchez <marco.sanchez01@epn.edu.ec>. You can access it via the SAM'21 EasyChair Web page

https://easychair.org/conferences/?conf=sam21

Thank you for submitting to SAM'21.

Best regards,
EasyChair for SAM'21.

**Figura 7.2:** Notificación de recepción a carlos.narvaez02@epn.edu.ec

### 7.1.2 ARTÍCULO ENVIADO

# Construction of a synthetic dataset applying Machine Learning techniques


Verónica Olmedo
*Facultad de Ingeniería en Sistemas*
*Escuela Poltécnica Nacional*
Quito, Ecuador
veronica.olmedo@epn.edu.ec

Carlos Narvaez
*Facultad de Ingeniería en Sistemas*
*Escuela Poltécnica Nacional*
Quito, Ecuador
email address

Marco Sánchez
*Facultad de Ingeniería en Sistemas*
*Escuela Poltécnica Nacional*
Quito, Ecuador
marco.sanchez01@epn.edu.ec

Myriam Hernández
*Facultad de Ingeniería en Sistemas)*
*Escuela Poltécnica Nacional*
Quito, Ecuador
myriam.hernandez@epn.edu.ec



***Abstract*—Today, fraud-related activities are growing at a dizzying rate, causing huge economic losses every year. For an adequate analysis of this phenomenon, it is necessary to have data that evidences this behavior, but since these are scarce and difficult to find, the generation of synthetic data for its study is a viable option. Using Machine Learning (ML) techniques, specifically deep learning models such as *Recurrent Neural Network (RNN)* and *Long Short Term Memory Networks (LSTM)* were used for the generation of text and supported by the theory of the fraud triangle proposed by Donald R. Cressey, to build a synthetic data set that allows its analysis. The RNN works with many copies of itself; each one sends a message to its successor, which reduces precision when generating sentences; besides, it faces the disappearance gradient problem. While the LSTM model aims to solve this problem, these networks can maintain long-term interrelationships by expanding their memory to learn from past experiences, making them perfect for generating text. The results obtained indicate that the data generation architecture proposed using the LSTM algorithm provides better performance in the generation of sentences. The data's readability is superior with an efficiency of 70% compared to the RNN algorithm approach reached 40%. Using this technique (LSTM), it was possible to synthesize a set of understandable data related to the fraud triangle that will allow the fraud study to be carried out effectively.**

***Index Terms*—Fraud triangle, Machine learning, Deep learning, LSTM, RNN.**


## I. Introduction

The triangle theory explains the factors to be taken into account within human nature that create fraud conditions. Fraud includes any intentional or deliberate act to deprive another of property or money through cunning, deception, or other unfair acts, according to the Association of Certified Fraud Examiners (ACFE). According to ACFE, fraud is defined as the use of one's occupation for personal enrichment through the deliberate misuse or misapplication of the resources or assets of the employing organization [1].

Donald R. Cressey [2], a leading crime psychologist expert, investigates the reasons behind the question why do people commit fraud? Moreover, it determines the answer in the following three critical elements: perceived pressure, perceived opportunity, and rationalization that must be present consecutively to provoke the desire to commit fraud. Based on his research, he presents 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 of Investigation (FBI) and ACFE, who have managed to obtain data related to these topics from their investigations to important companies on a confidential basis.

A valid option for the prediction and early detection of fraud is the generation of synthetic data, which, according to many experts, they claim is the key to making machine learning, within Artificial Intelligence (AI), faster and algorithms increase the precision of their predictions to identify fraudulent behavior, especially when the real data is costly to obtain or difficult to access [3]. Violations of copyright and intellectual property rights have limited the availability of actual data sets. Therefore, we will analyze some Machine Learning (ML) techniques to show the application of the Long Short Term Memory (LSTM) and Recurrent Neural Network (RNN) algorithms, which are used to generate data sets specific synthetics efficiently. Metrics will be applied to estimate which of them best fits our project.

The rest of this article is organized as follows. In section II, a theoretical foundation is made under which the work was carried out. In section III are the works related to our topic. In section IV, the data used and the method applied for the generation and comparison of the algorithms used are described. Then, in section V, results and comparisons between the models developed are presented. Finally, in section VI, the respective conclusions are presented.

**Figura 7.3:** Página uno artículo científico

## II. THEORICAL BACKGROUND

### A. Fraud Triangle

Criminal psychologist Donald R. Cressey [4] discovered that three factors must be present for the fraud to materialize:

1) The management or other employees have an incentive or are under pressure, which provides a reason to commit fraud.
2) There are circumstances (for example, the absence and ineffectiveness of controls or management's ability to override them) that offer the opportunity to perpetrate fraud.
3) Those involved can rationalize committing a fraudulent act.

Some people possess an attitude, character, or set of ethical values that allow them to knowingly and intentionally commit a dishonest act. 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.

### B. Synthetic Data Set

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* than real-world data [5].

### C. Machine Learning

Machine Learning (ML) is within the world of AI. It is a widely used learning methodology that provides systems with the ability to learn and improve automatically from experience without human intervention. Referring to the method or form of learning, we can talk about two main fields, *supervised and unsupervised learning*, and a third known as *reinforcement learning* [6].

### D. Deep Learning

Deep Learning (DL) is a subfield within ML, is based on the learning form of a layered model. It extracts more abstract characteristics from a broader set of training data, mostly without human supervision [7].

### E. Neural Networks

Neural Networks (NN) are computational models that emulate humans' specific characteristics, such as the ability to memorize and associate facts. They are nothing more than an artificial and simplified model of the human brain, which is the perfect example of defining a system capable of acquiring knowledge through experience [8].

### F. 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," a kind of feedback between neurons within of the layers [9], that is, the simplest RNN is composed of a single neuron that receives an input, produces an output, and sends that output to itself.

RNN is a neural network designed to analyze data streams using hidden units. In applications such as word processing, speech recognition, and DNA sequences, the output depends on previous calculations. Since RNNs deal with sequential data, they are well suited for processing vast amounts of data. RNN is the first algorithm to remember your 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 ***(t1)*** affects the parameter available in time ***(t1 + 1)***, in this way, the RNNs maintain two types of input 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))* [10].

### G. 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 similar to the memory of a computer, in the sense that a neuron of an LSTM can read, write and erase information from its memory [11].

In an LSTM neuron there are three gates to these information "cells": ***(input gate)*** gate, ***(forget gate)*** gate, and exit gate ***(output gate)***. These gates determine whether or not a new input is allowed. The information is eliminated because it is not important or is allowed to affect the output in the current time step.

### H. Random Subsampling

Random subsampling [12] is a non-heuristic method that aims to balance classes' distribution by randomly eliminating examples of majority classes. The reason behind this is trying to balance a data set. The main drawback of random subsampling is that this method can discard potentially useful data

**Figura 7.4:** Página dos artículo científico

that could be important to the induction process, which has to do with training a logic model representing a known data set.

## III. Related Work

Many areas of study use synthetically generated data, from data mining to software engineering to artificial intelligence. For example, Demillio and Offut [13] presented a fault-based technique to create data for the software module or unit tests. In the field of evolutionary computing, it is also possible to find works on genetic algorithms to generate data suitable for tests [14] [15].

Albuquerque, Lowe & Magnor [16] present 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, Ruchikachorn & Mueller [17] 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 presents similar interactions, Kim, Wall, Choo, Park & Endert [18] who made use of drawing interactions to direct the visualization of high-dimensional data according to the domain knowledge of the users.

Liu [19], created a synthetic data generator to evaluate the learning of classification rules. Similarly, researchers have proposed database synthesizers to analyze data mining tools [20] [21] [22]. These systems generate databases to analyze data mining tools since obtaining real data can often be very expensive or limited by legal rights or data privacy. However, these generators are specific to a problem tool or context, limiting large-scale use in other areas.

García & Millán [23] created a data set generation system for a wide range of areas. They have compared their proposal with the systems already on the market, but their proposed application is free software different from market applications.

Some approaches are dedicated to generating synthetic network data [24] [25]. 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, Sixuan Ren, Yong Liu, Houwei Cao, Qihu Yuan, and Guoqiang Han [26] 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 textit (Long-Short Memory Networks Term).

Brian C. Hosler, Xinwei Zhao, Owen Mayer, Chen Chen, James A. Shackleford, and Matthew C. Stamm [27], use an advanced deep learning technique, specifically Convolutional Neural Networks (CNN) plug-ins, to merge activations of multi-patch neurons to represent the identification of the camera model 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 for the identification of video camera models.

Hangxia Zhou, Yujin Zhang, Lingfan Yang, Qian Liu, Ke Yan and Yang Du [28], create a new relevancy data set 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 *(defines the mechanism our brain uses to prioritize certain stimuli, in this case visual)* deep.

Ahmadreza Argha, Ji Wu, Steven W. Su, and Branko G. Celler [29], uses 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.

Hamdi Altaheri, Mansour Alsulaiman, and Ghulam Muhammad [30], 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 data set of four types of dates by acquiring 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.

Xun Zhu, Chen Lyu, and Donghong Ji [31], create a large-scale biomedical keyphrase data set to assess system performance, the semantic web results are merged into the biomedical data set 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. It proposes a recommendation framework for a personalized channel with dynamic provisioning of data through deep learning and dynamically generates a list of recommended channels for each user. Through an LSTM Long Short-Term Memory network *(Long-Short-Term Memory Networks)*.

Zhaohui Che, Ali Borji, Guangtao Zhai, Xiongkuo Min, Guodong Guo, and Patrick Le Callet [32], this research is based on the creation of a new eye-tracking dataset using some tag preservation transformations to drive models based on deep learning, the SALICON data set is used as input data for training. Besides, the new salience data set includes fixations of 10 observers on 1900 images degraded by 19 types of transformations.

## IV. Methodology

In certain circumstances, it is preferable to have real data to show fraud. However, either due to the absence of data or the amount of data is insufficient for the analysis, and a simulating system can generate the alternative of using synthetic data. 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 allows optimizing the work and establishing a procedure that enables the tasks to be executed. The activities and steps necessary to carry out the

**Figura 7.5:** Página tres artículo científico

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 Emilie Lundin, Håkan Kvarnström, &Erland Jonsson [33] to adapt it to our needs, as can be seen in the Figure 1.

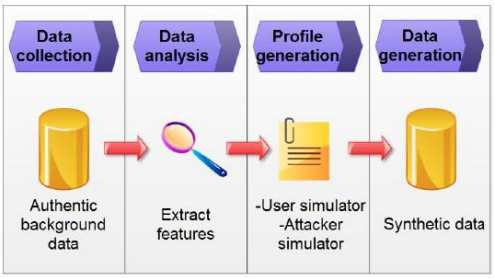



Fig. 1. Methodology

### A. General Methodology

In the first instance, data are collected to serve as a baseline for subsequent analysis and use. The data must include the necessary characteristics representing the expected behavior in the target system or phrase generator. The selected information can consist of authentic reference data, good history of similar systems, verified and authentic attacks related to the object of study, and other collections of information related to the project. The second phase is to analyze the collected data and identify the essential properties related to human behavior. Next, the information obtained previously is used, the same that will be used to identify the parameters that must be preserved and to generate phrases and create profiles based on the fraud triangle that adjusts to the statistics of the established parameters.

*1) Data collection:* The initial information is the starting point for the generation of synthetic data, and they must be representative data samples of human behavior related to each vertex of the fraud triangle. This information must be produced or obtained according to the required characteristics depending on the desired behavior for the output data in the generation of sentences. Accurate data helps improve the data creation process's effectiveness, so the data dictionary *Textual Survey Word List 103115* is a valuable source of information at this stage.

*2) Analysis of data:* The next step is to analyze the collected data, using exploratory data analysis (EDA), an existing set of ideas on how to study data sets to discover the underlying structure, find important variables, detect anomalies, etc. 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; otherwise, it may be necessary to implement other mechanisms.

*3) Profile Generation:* The next step is to identify the relevant parameters in the behavior of the input data. 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 Data Set:* Generating phrases with suspicious behavior characteristics related to fraud can be very complex to model. To limit the complexity only the data of interest must be taken into account for the data simulation actions. In general, it is easier to model a specific and well-defined behavior with prior knowledge of its approach than to do it blindly.

## V. Results and Discussion

In this section, the results obtained in the execution of the algorithms used are presented and analyzed, applying ML techniques, to carrying out four fundamental steps: data organization, representation learning, model fitting, and data evaluation. 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; that is, these techniques lead to the generation of text quickly and intelligently, and they also improve the decision-making process. Therefore, designing efficient deep learning models to produce good predictive results is challenging in this research.

### A. Analysis and Debugging of the Test Set

For this project, the data dictionary related to the fraud triangle was acquired by the company Audinet [34], which contributes to the financial community by offering online resources where auditors, accountants, and finance professionals share tools and experiences on audit work programs. This dictionary comprises 2,154 words; It serves as a starting point in the data generation method since they represent human behavior.

The participation of personnel from the National Polytechnic School (EPN) was necessary, so that in a controlled environment and with the help of the data dictionary, a test data set will be generated, which consists of 7,879 phrases and is named: ***FraudTriengle_Stages***, this data set will be the input in the synthetic data generation process. The information obtained must be relevant to obtain consistency for applying the algorithms and avoid directing to a specific side of the data; this can be the most valuable source for the generation of the data set.

Low-quality data can significantly affect the model's performance, so it is essential to detect anomalies in the input data since it is much easier to process and analyze it. The data is processed manually in order to analyze its consistency. The

**Figura 7.6:** Página cuatro artículo científico

next step is the purification of data before its automation; the data is examined in detail to determine if it is adequate and belongs to one vertex of the fraud triangle; then, we proceed to eliminate inappropriate and repeated data.

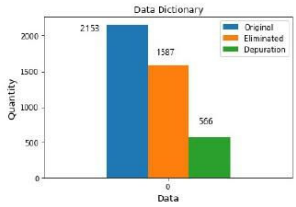



Fig. 2. Textual Survey Word List 103115

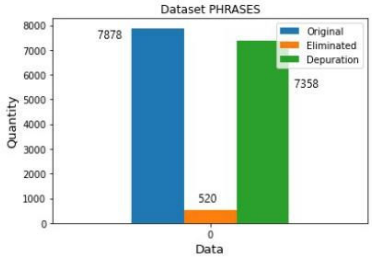



Fig. 3. FraudTriengle_Stages

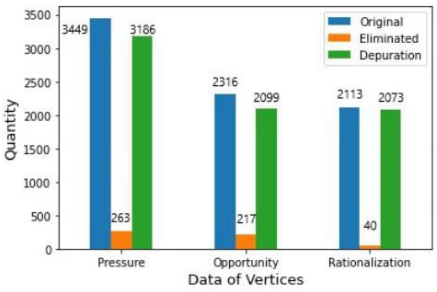



Fig. 4. Vertex FraudTriengle_Stages

In the Figure 2, it is detailed that 566 words from the data dictionary were used for the generation of sentences from the set of tests ***FraudTriengle_Stages***, which, in the Figure 3 shows that after debugging, it is composed of 7358 sentences. Finally, the test data set was divided by vertex, as shown in Figure 4.

*B. Results*

The generation of text 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 Presentation :* 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

| ***RNN Algorithm Results*** |
|---|
| olaunss |
| niwrelhznad y nt te i hiehle oinw |
| sepaauo t y mqli aid |
| ahohustek |
| ahv |
| nfrpymeadsiotayoliy ugnimosey nmnpu ywdint ia ah |
| oayautnimeuni ivos |
| espedemuwatd inalyea anyteabui |
| awiynuaiw ierve n ib |
| ene |

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. In table I, you can see some results.

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

TABLE II
LSTM ALGORITHM RESULTS

| ***LSTM Algorithm Results*** |
|---|
| I Got It Wrong What I Did. |
| Not Have To Pay For The Transportation Of My |
| 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. |
| So I Will Have To Mortgage My |
| My Salary Is Not Enough To Cover With These Fats. |
| But Those Holidays Are |
| To Pay The Debt But I |

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 the table II.

**Figura 7.7:** Página cinco artículo científico

*C. Discussion*

For the process of obtaining metrics, data is entered into the tool ***Readable***, in ranges from 0-100 until reaching 1000 data, once 1000 data were reached, the range of 0-500, with a maximum of 1500 data entered into the tools, because only a trial license was obtained, which did not allow more data to be analyzed.

*1) RNN:* The following describes the results of the consistency of the RNN algorithm applied in each of the vertices of the Fraud Triangle (Pressure (I), Opportunity (II), and Rationalization (III)), using the Readable tool. This tool was used to perform the data consistency analysis, as it quickly analyzes the readability, spelling, and grammar of the data set obtained after using the RNN algorithm as shown in table III.

TABLE III
TEXT CONSISTENCY USING THE READABLE TOOL FOR THE RNN ALGORITHM

| Amount | I | I RNN | II | II RNN | III | III RNN |
|---|---|---|---|---|---|---|
| 100 | 71.6% | 18.1% | 81.3% | 16.6% | 81.5% | 24.2% |
| 200 | 75.9% | 27.5% | 74.8% | 18.6% | 78.3% | 23.8% |
| 300 | 74.8% | 30.4% | 83.4% | 15.0% | 81.3% | 21.5% |
| 400 | 76.2% | 23.0% | 85% | 16.8% | 81.3% | 20.8% |
| 500 | 71.1% | 26.9% | 79.0% | 18.4% | 81.6% | 21.4% |
| 600 | 70.0% | 25.5% | 84.3% | 19.1% | 78.2% | 23.4% |
| 700 | 76.3% | 27.8% | 76.5% | 16.3% | 78.6% | 23.4% |
| 800 | 75.9% | 23.8% | 74.7% | 14.1% | 81.3% | 24.8% |
| 900 | 72.4% | 25.3% | 80.2% | 18.2% | 77.7% | 24.8% |
| 1000 | 70.1% | 30.7% | 82.2% | 16.2% | 78.5% | 21.9% |
| 1500 | 75.7% | 18.3% | 75.7% | 18.3% | 79.5% | 21.9% |

In Figure 6, we can see the comparison between the results obtained when analyzing each of the vertices of the fraud triangle for the RNN Algorithm and the original data set, the data collected by the algorithm have a score below 40%, and the original data set a score above 70%, which shows that specifically the RNN algorithm is inefficient, so it will not be used in this project.

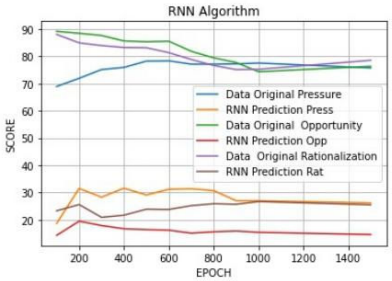



Fig. 5. Graphs of the RNN Algorithm using Readable

*2) LSTM :* **Results presentation**

The Readable tool was used to achieve readability results of the text g after generated with the LSTM Algorithm at each vertex of the Fraud Triangle (Pressure (I), Opportunity (II), and Rationalization (III)) concerning the original data set. And then proceed to make a comparison between the two algorithms used in this project.

TABLE IV
TEXT CONSISTENCY USING THE READABLE TOOL FOR THE LSTM ALGORITHM

| Cantidad | I | I LSTM | II | II LSTM | III | III LSTM |
|---|---|---|---|---|---|---|
| 100 | 70.1% | 76.2% | 80.5% | 69.2% | 78.5% | 76% |
| 200 | 73.1% | 76.2% | 88.3% | 70.6% | 86% | 76.6% |
| 300 | 74.3% | 77.1% | 80% | 70.5% | 84.2% | 76% |
| 400 | 71.8% | 77.7% | 74.2% | 69.5% | 82.1% | 76% |
| 500 | 74.5% | 76.1% | 79.2% | 69.2% | 81% | 77% |
| 600 | 76.9% | 76.7% | 88.9% | 69.2% | 81% | 77% |
| 700 | 76.9% | 77.5% | 81.4% | 69% | 78.3% | 76.7% |
| 800 | 74.1% | 77.9% | 86.5% | 70% | 80.1% | 76.3% |
| 900 | 70.1% | 76.1% | 87.6% | 70.3% | 77.7% | 76.5% |
| 1000 | 70% | 77.2% | 75.4% | 70.6% | 78.5% | 76.5% |
| 1500 | 71.4% | 77.1% | 75.7% | 70.6% | 79.3% | 76.2% |

Figure 7 shows the comparison between the scores obtained by the LSTM Algorithm for each vertex of the fraud triangle and the original data set, the data collected by the algorithm has a score above 70% like the whole of original data, with which it is assumed that the text generated by the LSTM algorithm has consistency.

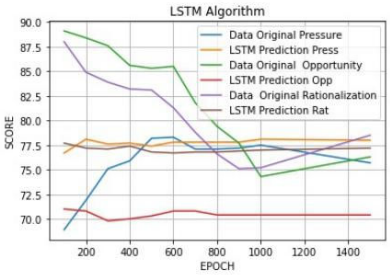



Fig. 6. Graphs of the LSTM Algorithm using Readable

*3) RNN vs LSTM :* **Comparative**

Below is the table V, the results obtained by the RNN and LSTM algorithms are shown, based on the scores obtained when executing the data processed by said algorithms in the tool ***Readable***, a comparison is made between the results obtained for each algorithm and the original data set, the scores are between 0 and 100 points, versus the number of analyzed sentences that are between 100 and 1500 sentences.

**Figura 7.8:** Página seis artículo científico

TABLE V
COMPARISON BETWEEN TEXT CONSISTENCY USING THE RNN VS. LSTM ALGORITHM

| Cantidad | I RNN | I LSTM | II RNN | II LSTM | III RNN | III LSTM |
|---|---|---|---|---|---|---|
| 100 | 28% | 76.2% | 16% | 70.1% | 21.6% | 76.9% |
| 200 | 25.3% | 77.5% | 15.7% | 70.1% | 26% | 76.7% |
| 300 | 25.2% | 76% | 15.3% | 69.2% | 21.2% | 76.7% |
| 400 | 26.1% | 76.5% | 18.2% | 70.7% | 24.7% | 76% |
| 500 | 18.7% | 76.3% | 15.7% | 70.3% | 20% | 76.9% |
| 600 | 29% | 77.6% | 15.4% | 70.1% | 24.3% | 76.4% |
| 700 | 27.8% | 77.7% | 15.6% | 69.9% | 24.4% | 76.3% |
| 800 | 25.5% | 77.9% | 16.9% | 70.6% | 22.7% | 76.1% |
| 900 | 22.4% | 76.9% | 15.9% | 69.5% | 23.4% | 76.2% |
| 1000 | 18.7% | 76.8% | 18.8% | 69.2% | 25.7% | 76.1% |
| 1500 | 30.2% | 76.4% | 18.5% | 70.7% | 22.4% | 76% |

In Figure 8, the LSTM, RNN algorithms are represented with various colors, and the original data as well. If we leave aside the score for a moment and focus on the number of sentences analyzed, we can confirm that for an algorithm to give more accurate results, large amounts of data are needed, so if we look at From the graph, we can see that from the 1000 data, the scores do not make significant variations, on the contrary, before the 1000 data, these scores are highly variable.

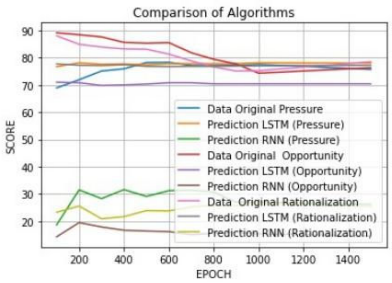



Fig. 7. Comparison between RNN vs LSTM Algorithms

Finally, after analyzing the graphs, it can be observed that, in this specific case corresponding to text processing, the LSTM algorithm is more effective when generating consistent text, so this project helps support this theory, together with the practical part. Nevertheless, it is observed that regardless of the vertex to which it corresponds, the LSTM algorithm presents a notable improvement in the score at the time of generating sentences related to the fraud triangle.

## VI. CONCLUSIONS AND RECOMMENDATIONS

### A. Conclusions

Deep learning algorithms depend on large amounts of training data, so sometimes the neural network receives samples of synthetic or manufactured data because it is complicated to get real data from entities that have been victims of a fraudulent attack, for fear that its reputation is affected, data unavailability and most to preserve privacy. However, as demonstrated in this project, by using the LSTM algorithm, it is possible to get relevant data for fraud analysis.

Compared to the LSTM algorithm, the RNN neural network produces large errors without training; this shows that the consideration of historical data (memory) does not influence the model's training. The cost of acquiring data is prohibitive, and we face the challenge of generating consistent data and making decisions under partial information.

The set of synthetic data obtained through the development of the LSTM algorithm in this project contributes to creating a knowledge base for future research since the theory of the fraud triangle supports it, which provides elements for the development of a fraud predictor system at the business level.

The datasets collected through an analysis phase allowed identifying inconsistencies and anomalies, which were corrected or eliminated in the debugging phase. In this way, it was possible to get a clean base data set to train the network and avoid directing to a particular side of the data, generating more precise and more coherent sentences.

The comparative figures between the RNN and LSTM algorithms allow us to visualize the performance of both concerning the text generation to determine which Neural Network is better for this specific purpose. The LSTM network concerning RNN obtained better results in the Readable tool; that is, according to the metrics obtained, it can be ensured that the most suitable network for text generation is LSTM. However, it should be noted that the best results are obtained with large amounts of data.

### B. recommendations

The number of interactions to achieve optimal results varies between the different proposed algorithms; however, the difference with the model results without oversampling is significant. For this reason, when you have a small data set 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.

Take into account the limitations of the project's resources, such as the hosts used in the development and testing phase, since they can present obstacles when carrying out the project. For this reason, the Google Colab platform was used as it provides storage resources that allow the storage and processing of large amounts of data, which were used in this research.

The limitations of the tools used to evaluate the algorithms' efficiency generated by the algorithms must be taken into account since the trial licenses granted by said companies have limited text input capacity to process, representing a more significant investment of time and resources to obtain the desired results.

**Figura 7.9:** Página siete artículo científico

The variety of optimizers available for the implementation of neural networks can make the developer tend to be confused between which is better or worse for the network, so in this project, it can be contributed that for the generation of text, the algorithm Adam (Adaptive moment estimation) responds well when training the data since it combines the benefits of the AdaGrad and RMSProp algorithms.

## REFERENCES


[1] Acfe-spain.com, "Acfe association of certified fraud examiners capítulo españa," Available: https://acfe-spain.com, accedido 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] N. M. Rabi'u Abdullahi, "Fraud triangle theory and fraud diamond theory. understanding the convergent and divergent for future research," Available: 10.6007/IJARAFMS/v5-i4/1823, accedido 06-11-2019.

[5] A. R. Fernández, ""los datos sintéticos, la clave para mejorar la inteligencia artificial", addison wesley college, 1997."

[6] A. G. Huerta, ""algoritmos de clasificación para datasets desequilabrados: Análisis y comparativa", proyecto de fin de grado en ingeniería de computadores."

[7] I. Mufti Mahmud, Senior Member, I. Mohammed Shamim Kaiser, Senior Member, I. Amir Hussain, Senior Member, and S. Vassanelli, "Applications of deep learning and reinforcementlearning to biological data, ieee transactions on neural networks and learning systems, vol. 29, no. 6, 2018," Available: 10.1109/TNNLS.2018.2790388, accedido 06-11-2019.

[8] D. J. Matich, ""redes neuronales: Conceptos básicos y aplicaciones.", informática aplicada a la ingeniería de procesos – orientación i."

[9] 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, accedido 11-12-2020.

[10] F. D. D. Raví, 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."

[11] S. D. S.Dhananjay Kumar, "Prediction of depression from eeg signal using long short term memory(lstm), ieee 3rd international conference on trends in electronics and informatics (icoei)," Available: 10.1109/ICOEI.2019.8862560, accedido 06-11-2019.

[12] P. P. S. Kotsiantis, ""mixture of expert agents for handling imbalanced data sets", annals of mathematics, computing teleinformatics, vol 1, no 1 (46-55), 2003."

[13] 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, accedido 06-11-2019.

[14] 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."

[15] 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."

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

[17] 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."

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

[19] 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."

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

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

[22] 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."

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

[24] 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."

[25] 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."

[26] 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, accedido 07-07-2020.

[27] X. Z. Brian C. Hosler, C. C. Owen Mayer, and M. C. S. James A. Shackleford, ""the video authentication and camera identification database: A new database for video forensics - ieee journals & magazine"," Available: https://ieeexplore.ieee.org/document/8734060, accedido 07-07-2020.

[28] 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, accedido 07-07-2020.

[29] 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, accedido 07-07-2020.

[30] M. A. Hamdi Altaheri and G. Muhammad, ""date fruit classification for robotic harvesting in a natural environment using deep learning - ieee journals & magazine"," Available: https://ieeexplore.ieee.org/document/8807111, accedido 07-07-2020.

[31] 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.

[32] G. Z. Zhaohui Che, Ali Borji and P. L. C. Xiongkuo Min, Guodong Guo, ""how is gaze influenced by image transformations? dataset and model - ieee journals & magazine"," Available: https://ieeexplore.ieee.org/document/8866748, accedido 07-07-2020.

[33] H. K. E. Lundin and E. Jonsson, ""a synthetic fraud data generation methodology"," accedido 07-07-2020.

[34] AuditNet, ""using key word analysis of an organization's big data for error and fraud detection"." urlhttps://www.auditnet.org/key-word-analytics.


**Figura 7.10:** Página ocho artículo científico

## 7.2 ANEXO II. CÓDIGO

### 7.2.1 CÓDIGO DEL ALGORITMO RNN

```
#*****************************************************************************
#                    Algoritmo RNN para la generación de texto
#*****************************************************************************
#generador de frases haciendo un diccionario de frases no del alfabeto

import numpy as np
np.random.seed(5)

from keras.layers import Input, Dense, SimpleRNN
from keras.models import Model
from keras.optimizers import SGD
from keras.utils import to_categorical
from keras import backend as K
from google.colab import drive
from numpy.random import seed

import pandas as pd
import numpy as np
import string, os
import warnings

warnings.filterwarnings("ignore")
warnings.simplefilter(action='ignore', category=FutureWarning)

# ============================================================
# 1. LECTURA DEL SET DE DATOS
# ============================================================

drive.mount('/content/drive')
curr_dir = '/content/drive/My Drive/LSTM_FRAUDE/'
nombres = []
filename='Oportunidad.xlsx'
for filename in os.listdir(curr_dir):
    if 'Oportunid' in filename:
        article_df = pd.read_excel(curr_dir + filename)
        nombres.extend(list(article_df.message_oportudidad.values))
        break
nombres = [h for h in nombres if h != "Unknown"]
```

```
len(nombres)

import random
i=0
while (i<1):
  i=i+1
  output = random.sample(nombres,1679)

len(output)
output[:10]

input_dataCNAT = output + nombres
len(input_dataCNAT)
input_dataCNAT[:10]

alfabeto = list(set(input_dataCNAT))
tam_datos, tam_alfabeto = len(input_dataCNAT), len(alfabeto)
print("En total hay %d caracteres, y el diccionario tiene un tamaño de %d
    caracteres." %(tam_datos, tam_alfabeto))

car_a_ind = { car:ind for ind,car in enumerate(sorted(alfabeto))}
ind_a_car = { ind:car for ind,car in enumerate(sorted(alfabeto))}
print(car_a_ind)
print(ind_a_car)

# ============================================================
# 2. MODELO
# ============================================================

n_a = 25
entrada  = Input(shape=(None,tam_alfabeto))
a0 = Input(shape=(n_a,))

celda_recurrente = SimpleRNN(n_a, activation='tanh', return_state = True)
capa_salida = Dense(tam_alfabeto, activation='softmax')

salida = []
hs, _ = celda_recurrente(entrada, initial_state=a0)
salida.append(capa_salida(hs))
modelo = Model([entrada,a0],salida)
modelo.summary()

opt = SGD(lr=0.0005)
modelo.compile(optimizer=opt, loss='categorical_crossentropy')
```

```
-
- # ============================================================
- # 3. EJEMPLOS DE ENTRENAMIENTO
85 # ============================================================
-
- ejemplos = output
- ejemplos = [x.lower().strip() for x in ejemplos]
- np.random.shuffle(ejemplos)
90
- len (ejemplos)
-
- # ============================================================
- # 4. CREAR EJEMPLOS DE ENTRENAMIENTO USANDO UN GENERADOR
95 # ============================================================
-
- def train_generator():
-     while True:
-         ejemplo = ejemplos[np.random.randint(0,len(ejemplos))]
100
-         X = [None] +[car_a_ind[c] for c in ejemplo]
-         Y = X[1:] + [car_a_ind['\n']]
-
-         x = np.zeros((len(X),1,tam_alfabeto))
105          onehot = to_categorical(X[1:],tam_alfabeto).reshape(len(X)-1,1,
                 tam_alfabeto)
-         x[1:,:,:] = onehot
-         y = to_categorical(Y,tam_alfabeto).reshape(len(X),tam_alfabeto)
-
-         a = np.zeros((len(X), n_a))
110          yield [x, a], y
-
- modelo.fit_generator
-
- # ============================================================
115 # 5. GENERACIN DE NOMBRE USANDO EL MODELO ENTRENADO
- # ============================================================
-
- def generar_nombre(modelo,car_a_num,tam_alfabeto,n_a):
-
120     x = np.zeros((1,1,tam_alfabeto,))
-     a = np.zeros((1, n_a))
-
-     fin_linea = '\n'
-     car = -1
```

```
125
-       while (car != fin_linea and contador != 50):
-
-           a, _ = celda_recurrente(K.constant(x), initial_state=K.constant(a))
-           y = capa_salida(a)
130        prediccion = K.eval(y)
-
-           ix = np.random.choice(list(range(tam_alfabeto)),p=prediccion.ravel())
-
-           car = ind_a_car[ix]
135        nombre_generado += car
-
-           x = to_categorical(ix,tam_alfabeto).reshape(1,1,tam_alfabeto)
-           a = K.eval(a)
-
140        contador += 1
-           if (contador == 50):
-             nombre_generado += '\n'
-
-       return (nombre_generado)
145
- # CICLO PARA CREAR FRASES
- i=1
- while i<=1500:
-
150 # GENERADOR
-   print(generar_nombre(modelo,car_a_ind,tam_alfabeto,n_a))
-   i=i+1
```

### 7.2.2 CÓDIGO DEL ALGORITMO LSTM

```
1  ********************************************************************************
-                    Algoritmo LSTM para la generación de texto
-  ********************************************************************************
-
5  # keras module for building LSTM
- from keras.preprocessing.sequence import pad_sequences
- from keras.layers import Embedding, LSTM, Dense, Dropout
- from keras.preprocessing.text import Tokenizer
- from keras.callbacks import EarlyStopping
10 from keras.models import Sequential
```

```
import keras.utils as ku
from google.colab import drive
from numpy.random import seed

import tensorflow as tf
tf.random.set_seed(2)

import pandas as pd
import numpy as np
import string, os
import warnings

warnings.filterwarnings("ignore")
warnings.simplefilter(action='ignore', category=FutureWarning)


drive.mount('/content/drive')
curr_dir = '/content/drive/My Drive/LSTM_FRAUDE/'
all_phrases = []
filename='Racionalizacion.xlsx'
for filename in os.listdir(curr_dir):
    if 'Racionalizacion' in filename:
        article_df = pd.read_excel(curr_dir + filename)
        all_phrases.extend(list(article_df.message_racionalizacion.values))
        break
all_phrases = [h for h in all_phrases if h != "Unknown"]
len(all_phrases)

import random

i=0
while (i<1):
  i=i+1
  output = random.sample(all_phrases,1658)

len(output)
output[:10]

input_dataCNAT = output + all_phrases
len(input_dataCNAT)

def clean_text(txt):
    txt = "".join(v for v in txt if v not in string.punctuation).lower()
    txt = txt.encode("utf8").decode("ascii",'ignore')
```

```
    return txt

corpus = [clean_text(x) for x in input_dataCNAT]
corpus[:10]

tokenizer = Tokenizer()

def get_sequence_of_tokens(corpus):
    tokenizer.fit_on_texts(corpus)
    total_words = len(tokenizer.word_index) + 1

    input_sequences = []
    for line in corpus:
        token_list = tokenizer.texts_to_sequences([line])[0]
        for i in range(1, len(token_list)):
            n_gram_sequence = token_list[:i+1]
            input_sequences.append(n_gram_sequence)
    return input_sequences, total_words

inp_sequences, total_words = get_sequence_of_tokens(corpus)
inp_sequences[:5]

def generate_padded_sequences(input_sequences):
    max_sequence_len = max([len(x) for x in input_sequences])
    input_sequences = np.array(pad_sequences(input_sequences, maxlen=
        max_sequence_len, padding='pre'))

    predictors, label = input_sequences[:,:-1],input_sequences[:,-1]
    label = ku.to_categorical(label, num_classes=total_words)
    return predictors, label, max_sequence_len

predictors, label, max_sequence_len = generate_padded_sequences(inp_sequences)

def create_model(max_sequence_len, total_words):
    input_len = max_sequence_len - 1
    model = Sequential()

    # Add Input Embedding Layer
    model.add(Embedding(total_words, 10, input_length=input_len))

    # Add Hidden Layer 1 - LSTM Layer
    model.add(LSTM(100))
    model.add(Dropout(0.1))

```

```
    # Add Output Layer
    model.add(Dense(total_words, activation='softmax'))

    model.compile(loss='categorical_crossentropy', optimizer='adam')

    return model

model = create_model(max_sequence_len, total_words)
model.summary()

model.fit(predictors, label, epochs=100, batch_size=128)

# FUNCION MODIFICADA
def generate_phrasesQ(seed_textQ, next_words, model, max_sequence_len):
    for _ in range(next_words):
        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:
                output_word = word
                break
        seed_textQ += " "+output_word
    return seed_textQ.title()

# CICLO PARA CREAR FRASES
i=1
while i<=5000:
  start = np.random.randint(0, len(all_phrases)-1)
  seedCNAT = all_phrases[start]

  number= random.randint(2,15)

# GENERADOR
  print(generate_phrasesQ(seedCNAT, number, model, max_sequence_len))
  i=i+1
```