**UNIVERSIDAD POLITÉCNICA SALESIANA**

**SEDE QUITO**

**CARRERA:**

**INGENIERÍA DE SISTEMAS**

**Trabajo de titulación previo a la obtención del título de:**

**Ingeniero de Sistemas**

**TEMA:**

**MINERÍA DE OPINIÓN PARA TEXTOS EN ESPAÑOL USANDO PROCESAMIENTO NATURAL DEL LENGUAJE**

**AUTOR:**

**EDWIN FRANCISCO AUQUILLA MOROCHO**

**TUTOR:**

**JULIO RICARDO PROAÑO ORELLANA**

**Quito, junio de 2021**

**CESIÓN DE DERECHOS DE AUTOR**

Yo, Edwin Francisco Auquilla Morocho, con documento de identificación N° 1723555338, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del trabajo de titulación con el tema: “MINERÍA DE OPINIÓN PARA TEXTOS EN ESPAÑOL USANDO PROCESAMIENTO NATURAL DEL LENGUAJE.”, mismo que ha sido desarrollado para optar por el título de INGENIERO DE SISTEMAS en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada.

En concordancia, suscribo este documento en el momento que hago la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

………………………….

EDWIN FRANCISCO

AUQUILLA MOROCHO

CI: 1723555338

Quito, junio de 2021

**DECLARATORIA DE COAUTORÍA DE LA TUTORA**

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Artículo Académico, con el tema: MINERÍA DE OPINIÓN PARA TEXTOS EN ESPAÑOL USANDO PROCESAMIENTO NATURAL DEL LENGUAJE, realizado por Edwin Francisco Auquilla Morocho, obteniendo un producto que cumple con todos los requisitos por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, junio de 2021

………………………………

JULIO RICARDO PROAÑO ORELLANA

CI:

**MINERÍA DE OPINIÓN PARA TEXTOS EN ESPAÑOL USANDO PROCESAMIENTO NATURAL DEL LENGUAJE**

**OPINION MINING FOR TEXTS IN SPANISH USING NATURAL LANGUAGE PROCESSING**

Edwin Auquilla[[1]](#footnote-2), Julio Proaño[[2]](#footnote-3)

|  |  |
| --- | --- |
| **Resumen** | **Abstract** |
| ***Palabras Clave:*** | ***Keywords*:** |

1. **Introducción**
2. **Métodos y materiales**

En la siguiente sección se describe la metodología usada en la investigación. Este consta de siete fases: I) recolección de tweets, II) procesamiento tweets y limpieza de carácteres especiales, III) construcción del corpus, IV) selección de los algoritmos más usados en NLP, V) entrenamiento y pruebas, VI) expansión del corpus, VII) elección del algoritmo que obtuvo un mayor acierto que los demás.

## Recolección de tweets

En esta fase ~~la recolección de tweets~~ se realizó a través del uso de la API (significado) de Twitter y del taller de análisis semántico en SEPNL (TASS) (Significado).

### Api de Twitter

Twitter tiene una gran cantidad de API’s las cuales son utilizadas mayormente por desarrolladores de software para realizar sistemas automatizados que interactúan con su aplicación [1]. Para esta investigación se utilizó solamente la obtención de tweets.

~~Antes de usar dicha Api se debe obtener unas credenciales las cuales son proporcionadas cuando se registra en la página de desarrollador de Twitter, [2] una vez teniendo dichas credenciales se integrarán con el IDE de Python spyder con la ayuda de un script se~~ empezará a descargar una cantidad de tweets que pueden ser filtrados por hashtag o por usuario y almacenados en un archivo con extensión .csv.

### Taller de Análisis Semántico en SEPLN(TASS)

Este taller es realizado desde el año 2012 por el Congreso Internacional de la Sociedad Española del Procesamiento del Lenguaje Natural (SEPLN). TASS fue la primera tarea compartida sobre análisis de sentimientos de tweets en español. [3]

TASS proporciona una gran cantidad de tweets que son exclusivamente para el idioma español, los cuales son obtenidos de países como España, México, Perú, Uruguay, Chile y Costa Rica. [3] Esta colección de tweets al descargarlos tiene una extensión de archivo .csv, se puede clasificar en dataset para entrenamiento y pruebas, además que cada tweet ya viene integrado con su etiqueta de sentimiento que puede ser positivo, negativo o neutro.

## Procesar Tweets

El procesar o limpiar tweets es una de las tareas más importantes en el procesamiento natural del lenguaje ya que facilita el entendimiento entre el usuario final y el propio sistema. Este procesamiento de tweets se realizó teniendo en cuenta los algoritmos de clasificación para NLP que son: Naïve Bayes, TensorFlow y LSMT.

En el primer clasificador se debe realizar los siguientes pasos dentro de los datos:

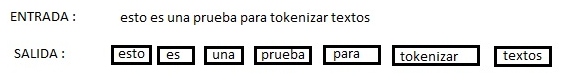
* Tokenizar.
* Normalizar.
* Remover palabras, caracteres especiales no significativos.
* Preparar datos para el algoritmo de clasificación.
* Convertir tokens en un diccionario. [4]

En otros dos algoritmos de clasificación solo se debe remover palabras y caracteres especiales no significativos.

### Tokenizar Tweets para Naïve Bayes

El proceso para realizar la tokenización de tweets es dividir el texto en oraciones y estas en palabras. Para NLP un token es la unidad mínima de procesamiento que puede ser términos o palabras. [5]

Los tokens son separados por cada espacio en blanco que existe entre palabras [6] como muestra la Figura 1, en la cual se puede ver como un texto de entrada luego de un proceso de Tokenización es dividido en términos independientes.



**Figura 1.** Proceso de tokenización

La tokenización se aplica en el algoritmo de clasificación NaïveBayes ya que es un clasificador de textos supervisado que entiende tokens [7] que serán las características de cada uno de los tweets.

### Normalizar

Normalizar es una tarea en la cual se debe poner al texto en igualdad de condiciones. [8] En NLP la normalización es el proceso de convertir una palabra es su forma canónica esto permite que proceda el procesamiento de manera uniforme. [9]

**Tabla 1.** Ejemplo de normalización de texto.

|  |  |
| --- | --- |
| **Texto Original** | **Texto Normalizado** |
| holaa, le pregunto x el anunsio q tiene sobre el empleo | hola le pregunto por el anuncio que tiene sobre el empleo |
|  |  |

### Remover palabras o caracteres especiales no significativos

De acuerdo a [10] al remover cualquier palabra o carácter especial no significativo dentro de un tweet permite una mejor comprensión de los algoritmos de clasificación ya que se tendrá un mejor procesamiento del lenguaje natural.

Los elementos a remover son los siguientes:

* **Hipervínculos**: todos los hipervínculos en Twitter se convierten a la URL acortador.
* **Twitter identifica en las respuestas**: estos nombres de usuario de Twitter están precedidos por un @símbolo, que no transmite ningún significado.
* **Puntuación y caracteres especiales**: si bien estos suelen proporcionar contexto a los datos textuales, este contexto suele ser difícil de procesar. Para simplificar, removerá todos los signos de puntuación y caracteres especiales de los tweets. [10]
  + - 1. **Preparar datos para el algoritmo de clasificación Naïve Bayes**

Se creará un conjunto de datos para el entrenamiento con el 80% del corpus lingüístico y el 20 % para prueba, estos datos irán a los tres algoritmos de clasificación antes mencionados. Se requiere que cada conjunto de datos se asocie a un sentimiento para el entrenamiento, los modelos usarán los sentimientos de positivo, negativo y neutro.

La preparación de datos permite la minería de opiniones convirtiendo los tokens al formulario de diccionario y luego dividirá los datos con fines de entrenamiento y prueba.

* + - 1. **Convertir tokens en un diccionario**

Esta es una parte importante al usar el modelo Naïve Bayes ya que requiere no solo una lista de palabras de un tweet, sino un diccionario de Python con palabras como claves y True valores. [7]

## Construcción del corpus lingüístico

El corpus lingüístico es un conjunto de textos que contiene un número considerable de textos, dichos textos comparten entre si varios rasgos definitorios [11] que en este caso para esta investigación serán divididos en textos que describen opiniones positivas, negativas y neutras.

Para la construcción se uso la data que nos proporciona TASS, esta data esta dividida en cinco archivos con extensión .csv de los países de Costa Rica, España, México, Perú y Uruguay esta data se la une en un solo archivo .csv con tweets obtenidos de Ecuador este conjunto de datos tiene 5147 tweets con su respectivo sentimiento.

## Selección de algoritmos para NLP

Para NLP hay una gran variedad de algoritmos que se pueden utilizar, en esta investigación se seleccionó tres los cuales son:

* NaïveBayes
* TensorFlow
* LSTM

En las siguientes secciones se describirá cada uno de estos algoritmos utilizados con sus respectivas características.

* + 1. **Naïve Bayes**

Es un clasificador bayesiano que hace una suposición simplificadora, es un algoritmo de aprendizaje automático que clasifica o filtra datos se utiliza principalmente en procesamiento natural del lenguaje [12].

Lo que significa que para un documento d, de todas las clases c C el clasificador devuelve la clase que tiene el máximo posterior dando la probabilidad del documento en (1) donde se usa la notación ^ para la estimación de la clase correcta [13].

El clasificador Naive Bayes tiene tres algoritmos diferentes: Guassian Naive Bayes, multinomial Naive Bayes, Bernoulli Naive Bayes [12]. Aunque en esta investigación solo se usara el multinomial que permite la clasificación de texto, se lo utiliza típicamente para modelos de ventos multinomiales como bolsa de palabras [14].

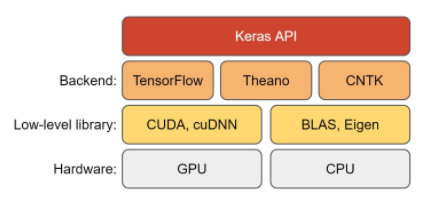
### Características de Naive Bayes

Naive Bayes se clasifica como aprendizaje supervisado y se basa en el teorema de Bayes. Ya que usa aprendizaje supervisado el algoritmo necesita de datos de entrenamiento para la clasificación, en si estos datos de entrenamiento deben tener una buena calidad ya que esto afecta a la precisión de la clasificación [15].

* + 1. **TensorFlow**

Es una librería de código abierto para Deep learning que se utilizó en esta investigación, ofrece programación de flujo de datos que realiza una variedad de tareas de aprendizaje automático. Se complementa con el uso de Keras una biblioteca de red neuronal de código abierto escrita en Python que es ejecutada sobre TensorFlow [16].

En la figura 2 se muestra cómo trabaja la API Keras sobre la librería TensorFlow quien la contiene y además puede usarse con soporte GPU [17].



**Figura 2.** Arquitectura Keras API [17].

### Características del uso de TensorFlow con Keras

Keras es un API de alto nivel que se ejecuta sobre TensorFlow y al ejecutarse las dos al mismo tiempo proporcionan una API de alto nivel que se utiliza para crear y entrenar modelos fácilmente con la gran ventaja que en esta investigación [18]. Keras se complementa perfectamente con Python el lenguaje que se uso en el desarrollo de la herramienta de esta investigación.

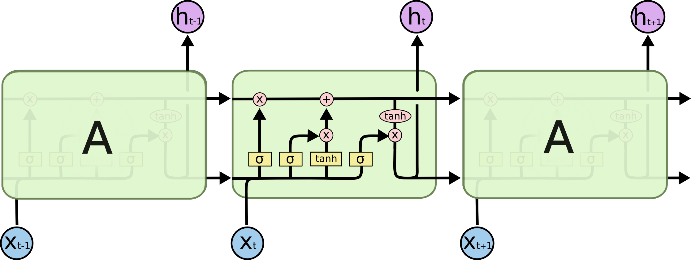
Construido sobre TensorFlow 2.0 Keras es un marco de trabajo fuerte en la industria que es escalable a grandes grupos de GPU [17]. Keras fue utilizado ya que el conjunto de datos usados es pequeño y nos proporciona una implementación rápida del modelo.

* + 1. **LSTM**

LSTM traducido al español significa unidades de memoria a corto plazo o a largo plazo. Son un subconjunto de las redes neuronales recurrentes que son mayormente conocidas y con una gran variedad de aplicaciones en Deep learning [19], capaces de reconocer y predecir secuencias de datos a lo largo del tiempo, como textos, genomas, discurso hablado o series numéricas [20].

Las LSTM están diseñadas para evitar el problema de dependencia a largo plazo es decir que recordarán información durante largos periodos de tiempo [21] según [19] las redes neuronales recurrentes que son antecesoras a las LSTM tenían un problema que poseían un cierto tiempo de memoria y esto hacía que no fueran tan eficaces como las memorias LSTM.

En la figura 3 muestra la arquitectura de LSTM que tienen una estructura tipo cadena lo que cambia de la arquitectura de RNN es el módulo de repetición el cual en vez de tener una sola capa de red neuronal hay cuatro que interactúan entre sí. [19]



**Figura 3.** Arquitectura LSTM. [19]

### Características de LSTM

Ayudan a preservar el error que se puede propagar hacia atrás a través del tiempo y de las capas, esto permite que las redes recurrentes continúen aprendiendo muchos pasos de tiempo [21].

La información que contiene los LSTM fuera de flujo normal de la red neuronal recurrente se puede almacenar, escribir o leer desde una celda, de manera similar a los datos en memoria de una computadora [22].

## Entrenamiento y pruebas

Este apartado describe el procedimiento que se llevó a cabo para el entrenamiento y pruebas de los algoritmos de clasificación antes mencionados en los cuales se uso el corpus creado de tweets.

* + 1. **Entrenamiento**

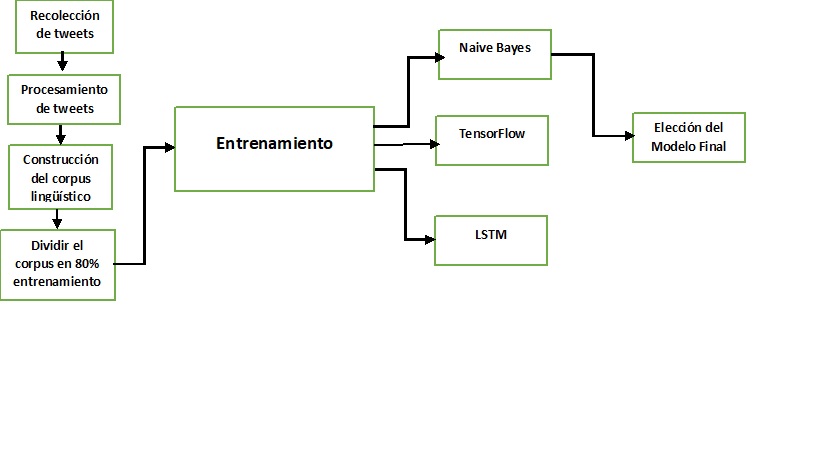
El hardware y software requerido para el entrenamiento de Naive Bayes, TensorFlow y LSTM se muestran en la Tabla 2 con la respectiva versión usada de Python y el IDE Spyder.

**Tabla 2.** Características de hardware y software.

|  |  |  |  |
| --- | --- | --- | --- |
| **Características Especificaciones** | | | |
| Ram |  | 8 GB , 2400 MHz | |
| Cpu |  | 4 núcleos, 4 hilos,2.4 MHz, 6MB en caché |  |
| Disco Duro |  | 500 GB | |
| SO |  | Windows 10 – 1809 x64 | |
| Python |  | V3.6 | |
| Spyder |  | V5.0 | |
| Tensorflow |  | V2.4 | |
| NLTK |  | V3.6.2 | |

Para un correcto funcionamiento de TensorFlow, NLTK y Python se usó la versión de Anaconda 1.10.0 con el respectivo IDE Spyder en su versión 5.0 la cual fue compatible con la última versión de TensorFlow.

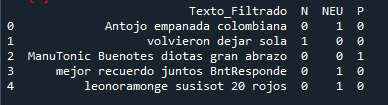
Luego de cumplir con los requisitos de hardware y software se procedió con el siguiente diagrama de la Figura 4.



**Figura 4.** Etapas del Entrenamiento.

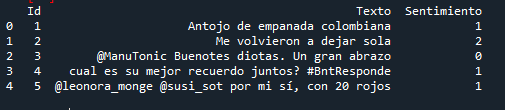
La recolección de tweets se la realizo por medio de la API de Twitter y de los data sets proporcionados por TASS, luego se agrupo estos tweets que contaban con su sentimiento y se los envió a procesar para limpiarlos de caracteres especiales, links etc., a partir de esos tweets ya limpios se construyó el corpus con alrededor de 5.147 tweets con sui respectivo sentimiento: positivo, negativo y neutro. Por último, este corpus lingüístico creado se ocupó el 80% para realizar el entrenamiento.

Los tweets de entrenamiento disponen de su sentimiento el cual al entrenarlo con TensorFlow-Keras y LSTM se debe convertir a un valor numérico en esta investigación al usar TensorFlow-Keras se usó 1 para el tweet que posea uno de los sentimientos mencionados y 0 cuando no tenga uno de los sentimientos descritos como se muestra en la Figura 4.



**Figura 4.** Datos TensorFlow-Keras

En el algoritmo LSMT se debe convertir a un valor numérico en esta investigación al usar TensorFlow-Keras se usó 1 para sentimiento positivo, 0 para Neutro y 2 para Negativo como muestra la Figura 5.



**Figura 5.** Datos LSTM

### Pruebas

Al realizar las pruebas de los tres algoritmos antes mencionados se usó el 20% del corpus el resto del mismo. Se obtuvo un mayor acierto en el algoritmo Naive Bayes ya que tiene mejor procesamiento en su composición por que tokeniza primero las palabras y luego se procesa además de que se divide en verbos, adjetivos, adverbios que dan una mejor compresión al algoritmo de que palabras componen el tweet y el sentimiento que devuelve.

## Expansión del corpus

La expansión del corpus se realizó ya que aunque se obtuvo un mayor acierto en el algoritmo de Naive Bayes no era el porcentaje que se esperaba por esa razón se expandió el corpus a unos 14.666 tweets con su respectivo sentimiento.

Al expandir el corpus se vio una mejora en acierto del algoritmo al entrenarlo con sentimientos de tweets positivo, negativo de igual manera subió el acierto al entrenarlo con los tres sentimientos positivo, negativo y neutro

## Elección del algoritmo que obtuvo un mayor acierto que los demás

Al tener ya un corpus más extenso, se decidió optar por elegir el algoritmo Naive Bayes que proporciono un mayor acierto al realizar el test frente a los otros algoritmos.

Naive Bayes se usó para la minería de opiniones en esta investigación ya que proporciona una mejor comprensión de textos gracias a que el formato que entiende son tokens y esto se complementa con lo que el procesamiento natural del lenguaje.

1. Results and discussion

These two sections usually appear together in many works. We should not confuse this discussion or analysis with the obtaining of conclusions, something that depends as much on the results and their analysis as on the heoretical framework and the objectives.

# Conclusions

The conclusions should therefore be obtained from more than just the recorded data. In fact, some data or results can have one meaning or another and can lead us to conclusions and others, depending on the conceptual framework that justifies our research, the methodology followed, the proposed objectives, etc to different ones La

# **Referencias**

|  |  |
| --- | --- |
| [1] | Pavloski, Mihajlo, «Accessing the Twitter API with Python,» stackabuse, 2019. [En línea]. Available: https://stackabuse.com/accessing-the-twitter-api-with-python/. [Último acceso: 17 Mayo 2021]. |
| [2] | Twitter Developer, «Twitter Developer,» [En línea]. Available: https://developer.twitter.com/en/docs. [Último acceso: 17 Mayo 2021]. |
| [3] | TASS 2020, «TASS 2020,» [En línea]. Available: http://tass.sepln.org/2020/. [Último acceso: 17 Mayo 2021]. |
| [4] | M. Mayo, «Preprocesamiento de datos de texto: un tutorial en Python,» [En línea]. Available: https://medium.com/datos-y-ciencia/preprocesamiento-de-datos-de-texto-un-tutorial-en-python-5db5620f1767. [Último acceso: 18 Mayo 2021]. |
| [5] | F. Murzone, «Procesamiento de Lenguaje Natural: Stemming y Lemmas,» *Medium,* 2020. |
| [6] | P. R. S. Christopher D. Manning, An Introduction to Information Retrieval, England: Cambridge UP, 2009. |
| [7] | M. B. Hernández y J. M. Gómez, «Aplicaciones de Procesamiento de Lenguaje Natural,» *Revista Politécnica,* 2013. |
| [8] | W. J. T. y. A. Crymble, «Normalizar datos de texto con Python,» 03 Julio 2012. [En línea]. Available: https://programminghistorian.org/es/lecciones/normalizar-datos. [Último acceso: 21 Mayo 2021]. |
| [9] | A. Bracco, «Normalización de Texto en Español de Argentina,» *Universidad Nacional de Córdoba,* 2017. |
| [10] | Samuel, «Introducción al procesamiento del lenguaje natural en Python – SitePoint,» 08 Abril 2019. [En línea]. Available: https://stips.wordpress.com/2019/04/08/introduccion-al-procesamiento-del-lenguaje-natural-en-python-sitepoint/. [Último acceso: 21 Mayo 2021]. |
| [11] | G. Parodi, «LINGÜISTICA DE CORPUS: UNA INTRODUCCION AL AMBITO,» *RLA. Revista de lingüística teórica y aplicada,* 2008. |
| [12] | eiki, «Natural Language Processing: Naive Bayes Classification in Python,» Marzo 2019. [En línea]. Available: https://medium.com/@eiki1212/natural-language-processing-naive-bayes-classification-in-python-e934365cf40c. [Último acceso: 24 Mayo 2021]. |
| [13] | D. J. &. J. H. Martin, «Naive Bayes and Sentiment,» *Speech and Language Processing,* 2020. |
| [14] | A. Rai, «Text Classification in NLP — Naive Bayes,» Julio 2017. [En línea]. Available: https://theflyingmantis.medium.com/text-classification-in-nlp-naive-bayes-a606bf419f8c. [Último acceso: 25 Mayo 2021]. |
| [15] | Sitio Big data, «Sitio Big data,» Abril 2019. [En línea]. Available: https://sitiobigdata.com/2019/12/24/clasificacion-de-aprendizaje-automatico-supervisado/. [Último acceso: 25 Mayo 2021]. |
| [16] | L. Llamas, «MACHINE LEARNING CON TENSORFLOW Y KERAS EN PYTHON,» 03 Febrero 2019. [En línea]. Available: https://www.luisllamas.es/machine-learning-con-tensorflow-y-keras-en-python/. [Último acceso: 25 Mayo 2021]. |
| [17] | ShawnHymel, «Introducción al aprendizaje automático mediante TensorFlow y Keras,» 2020. [En línea]. Available: https://www.digikey.com/en/maker/projects/getting-started-with-machine-learning-using-tensorflow-and-keras/0746640deea84313998f5f95c8206e5b. [Último acceso: 25 Mayo 2021]. |
| [18] | J. Terra, «Keras vs Tensorflow vs Pytorch: comprensión de los marcos de aprendizaje profundo más populares,» 13 Mayo 2021. [En línea]. Available: https://www.simplilearn.com/keras-vs-tensorflow-vs-pytorch-article. [Último acceso: 25 Mayo 2021]. |
| [19] | A. Mañas, Notas sobre pronóstico del flujo de tráfico en la ciudad de Madrid, Madrid: BookDown, 2019. |
| [20] | C. Nicholson, «Una guía para principiantes sobre LSTM y redes neuronales recurrentes,» *Wiki de IA,* 2020. |
| [21] | S. Hochreiter, «Long Short-Term Memory.” Neural Computation,» 1997. |
| [22] | B. Ripley, Pattern Recognition and Neural Networks, Cambridge, 2007. |

1. Estudiante de Ingeniería de Sistemas – Universidad Politécnica Salesiana, Egresado – UPS – sede Quito. Autor para correspondencia: eauquillam@est.ups.edu.ec [↑](#footnote-ref-2)
2. [↑](#footnote-ref-3)