**APRENDIZAJE AUTOMÁTICO PARA EL ANÁLISIS DE TEXTO**

*Karen Andrea Amaya 1*a*, Laura Camila Agudelo 2*b

*Luisa María Acosta 3*c *, Estefanía Echeverry 4*d *,*

*Camila Ospina 5*e*, Heber Esteban Bermúdez 6*f

*Email: kaamayam@unal.edu.co, lcagudeloo@unal.edu.co, lumacostaor@unal.edu.co, eecheverryf@unal.edu.co, cospinap@unal.edu.co, hebermudezg@unal.edu.co*

**Resumen**

El objetivo de este trabajo es el desarrollo de una aplicación que facilite a los usuarios la lectura y el análisis de textos por medio de resúmenes abstractivos y gráficos descriptivos, obtenidos mediante análisis de minería de texto, procesamiento de lenguaje natural, aprendizaje profundo y métodos de clusterización.

**Palabras clave:** Minería de texto, aprendizaje profundo, PLN, resumen de texto .

**MACHINE LEARNING FOR TEXT ANALYSIS**

*Karen Andrea Amaya 1*a*, Laura Camila Agudelo 2*b

*Luisa María Acosta 3*c *, Estefania Echeverry 4*d *,*

*Camila Ospina 5*e*, Heber Esteban Bermúdez 6*f

*Email: kaamayam@unal.edu.co, lcagudeloo@unal.edu.co, lumacostaor@unal.edu.co, eecheverryf@unal.edu.co, cospinap@unal.edu.co, hebermudezg@unal.edu.co*

**Abstract**

The objective of this work is the development of an application that facilitates to the users the reading and the analysis of texts by means of abstract summaries and descriptive graphs, obtained by means of analysis of text mining, natural language processing, deep learning and methods of clustering

**Key words:** Text mining, deep learning, NLP, summarizer.

**Introducción**

Los textos son un tipo de dato no estructurado, que contiene información que nos interesa extraer de manera eficaz y conservando su significado general para la toma de decisiones. Debido a la gran cantidad de datos producidos en la era digital, nace la necesidad de desarrollar metodologías que nos permitan resumir automáticamente estos grandes volúmenes de texto y brindarnos una visualización que facilite su análisis, reduzca el tiempo del mismo y nos permita generar conocimiento de manera rápida, eficaz y oportuna.

Más del 80% de los datos no estructurados que se encuentran en internet son de tipo texto, los datos generados en las redes sociales, revistas, artículos, noticias y demás permite a las organizaciones mejorar la toma de decisiones, hacer seguimiento a sus productos, perfilar clientes, observar tendencias y generar información crucial para las empresas lo cual se traduce en una ventaja competitiva en la nueva era digital.

**Metodología**

Para el desarrollo de esta aplicación hacemos uso del lenguaje de programación Python y en específico de la librería *Spacy* y *kivy*. En dicha aplicación se integran las siguientes etapas:

1. **Pre-procesamiento:**

Primero obtenemos un texto, ya sea desde una url, utilizando la técnica de web scraping sobre formatos HTML, con ayuda de la librería *beautifulsoup,* o directamente de un fichero *txt.*

Una vez obtenida esta información en formato de cadenas de texto procedemos con la limpieza de las mismas, comenzamos removiendo de palabras de parada (stopwords) y caracteres especiales que no aportan información relevante, como bien lo son los números, signos de puntuación, pronombres, artículos, preposiciones, etc. Esto haciendo uso tanto de la librería *Spacy* como de *nltk* y otras como *re.* Limpias las cadenas de texto se define un formato de texto ordenado como una tabla, con el texto dividido en unidades, las unidades pueden ser palabras o frases, denominadas como tokens.

1. **Resumen extractivo y abstractivo:**

Hay principalmente dos tipos de resumen: El primero es el extractivo, que toma las frases más relevantes del documento de entrada por medio de una ponderación y luego, sin alterarlas, las concatena. El segundo es el abstractivo que interpreta y examina el texto para generar uno más corto.

Para el resumen abstractivo utilizamos el modelado *seq2seq*, esto requiere un entrenamiento con una base de datos de dos columnas: una con un texto original y otra con el resumen de dicho texto. Utilizamos un90% de los datos para entrenar el modelo y un 10% de validación con ayuda del paquete *sklearn.model\_selection*. Para realizar este resumen hacemos uso de la librería *keras* y *Tensor Flow,* con 50 épocas de entrenamiento*.*

1. **Gráficos descriptivos:**

Una vez limpias las cadenas de texto puede resultar de particular interés observar gráficamente cuales son las palabras con mayor frecuencia dentro de toda la información recopilada. La función *count()* del paquete collections, nos permite contar la frecuencia de las palabras en el texto tokenizado. Esto facilita proceder a realizar algunos gráficos, en este caso podemos observar una nube de palabras, creada con los paquetes *matplotlib y* *wordcloud:*



Figura 1: Nube de palabras de los 50 términos con mayor frecuencia.

También resulta de particular interés observar la frecuencia de pares de palabras en el texto de interés, tanto para token de una palabra (unigrama) como de dos (bigrama), a continuación se muestra el gráfico de frecuencias para pares de palabras por término consultado (bigrama).

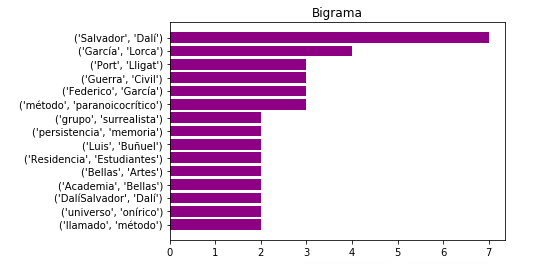


Figura 2: Frecuencias de pares de palabras por término o tema consultado

Otra manera de visualizar los bigramas es a través de un gráfico de redes, el cual nos permite interpretar la relación lógica directa entre los diferentes pares de palabras más relevantes del texto:

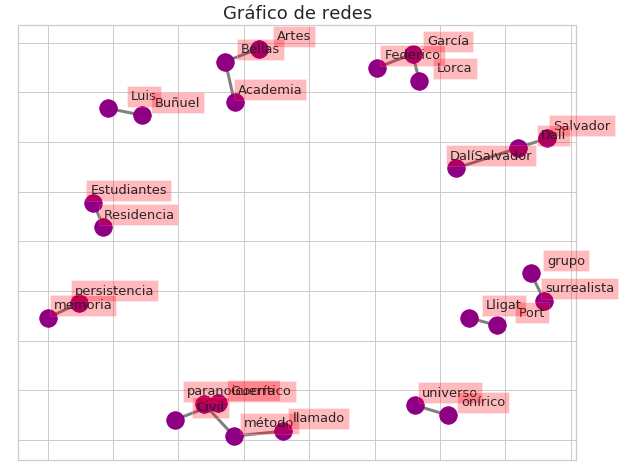


Figura 3: Gráfico de redes

Los tipos de gráficos mostrados ofrecen grandes posibilidades para la representación de datos y pueden ser utilizados en múltiples situaciones, incluso para representar los resultados obtenidos por métodos de análisis más complicados.

1. **Aplicación:**

Esencialmente hacemos uso de *kivy*, cuya librería es de acceso abierto para desarrollo rápido de aplicaciones, que hace uso de interfaces innovativas como las aplicaciones multitouch, haciéndola más amigable y fácil de usar para el usuario. Además, funciona en diferentes plataformas, como lo son Windows, Linux, OS X, Raspberry Pi, Android y iOS.

Los pasos clave dentro de la aplicación son nombrar diferentes clases, las cuales contienen diversas funciones dentro de ellas. Estas reciben como parámetro una *URL* o directamente un texto al que el usuario le interese analizar, luego se puede elegir crear un resumen extractivo o abstractivo y distintos tipos gráficos como nubes de palabras, bigramas, redes. Siempre en orden lógico, las clases nos permiten dar una secuencia de lo que queremos que haga la aplicación. Es así como *kivy* nos permite integrar diferentes funcionalidades interactivas con el usuario.

**Conclusiones**

La minería de texto reduce recursos y tiempo empleados para analizar textos extensos, facilitando el acceso a conocimiento por medio de métodos automáticos, extrayendo patrones e ideas generales para la generación de conocimiento basado en análisis textual, es por esto que esta metodología representa una de las mejores opciones para para extraer analizar este tipo de dato no estructurado que permitirá a la organizaciones mejorar la toma de decisiones.

**Referencias**

[1] Comprehensive Guide to Text Summarization using Deep Learning in Python tomado de:[https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summ](https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summarization-using-deep-learning-python/)

[arization-using-deep-learning-python/](https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summarization-using-deep-learning-python/)

[2]Analyze Co-occurrence and Networks of Words Using Twitter Data and Tweepy in Python

<https://www.earthdatascience.org/courses/earth-analytics-python/using-apis-natural-language-processing-twitter/calculate-tweet-word-bigrams-networks-in-python/>