https://doi.org/10.37815/rte.v33n2.844
Artículos originales

Comparando técnicas de minería de datos en un centro de emergencias

Comparing data mining techniques in an emergency center

Bayron Gutierrez[1], Brandon Llivisaca[1], Andrés Patiño[1] https://orcid.org/0000-0001-9504-6498, Marcos Orellana[1] https://orcid.org/0000-0002-3671-9362, Priscila Cedillo[1, 2] https://orcid.org/0000-0002-6787-0655

[1]*Universidad del Azuay*, Cuenca, Ecuador
bgutierrez@es.uazuay.edu.ec, brando@es.uazuay.edu.ec, andpatino@uazuay.edu.ec, marore@uazuay.edu.ec, icedillo@uazuay.edu.ec

[2]*Universidad de Cuenca*, Cuenca, Ecuador
priscila.cedillo@ucuenca.edu.ec

Enviado: 2021/07/10
Aceptado: 2021/09/28
Publicado: 2021/11/30

Resumen

El procesamiento del lenguaje natural es un campo dentro de la inteligencia artificial que estudia cómo modelar computacionalmente el lenguaje humano. La representación de palabras a través de vectores, conocida como Word embeddings, se populariza en los últimos años a través de técnicas como Doc2Vec o Word2Vec. El presente estudio evalúa el uso de Doc2Vec en un conjunto de conversaciones recopiladas por el centro de emergencia ECU911, perteneciente al cantón Cuenca de la provincia del Azuay durante el año 2020, con el fin de clasificar los incidentes para que el operador pueda tomar la mejor decisión, en cuanto a las acciones a realizar cuando se presente una emergencia. Además, se compara Doc2Vec con la técnica Word2Vec para verificar su nivel de desempeño tanto en precisión como en tiempo. A base de las pruebas realizadas se concluye que Doc2Vec tiene un desempeño sólido al utilizar modelos entrenados con gran corpus, superando a Word2Vec en este aspecto.

Palabras clave: minería de datos, centro de emergencias, Word2Vec, Doc2Vec, PLN.

Sumario: Introducción, Trabajos relacionados, Materiales y Métodos, Resultados y Discusión y Conclusiones.

Como citar: Gutierrez, B., Llivisaca, B., Patiño, A., Orellana, M. & Cedillo, P. (2021). Comparando técnicas de minería de datos en un centro de emergencias. *Revista Tecnológica - Espol, 33(2), 159-171.* http://www.rte.espol.edu.ec/index.php/tecnologica/article/view/844

Abstract

Natural language processing is a field within artificial intelligence that studies how to computationally model human language. The representation of words through vectors, known as Word embedding, has become popular in recent years through techniques such as Doc2Vec or Word2Vec. This study evaluates the use of Doc2Vec in a set of conversations collected by ECU911 emergency center. The purpose was to classify incidents, consequently the operator is able to make the best decision regarding the actions to be taken when an emergency occurs. The data were recorded during 2020, in the emergency center located in Cuenca, Ecuador. In addition, Doc2Vec was compared with the Word2Vec technique to verify its performance level both in terms of accuracy and time. Based on the tests performed, it was concluded that Doc2Vec has a solid performance when using trained models with large corpus, outperforming Word2Vec.

Keywords: data mining, emergency center, Word2Vec, Doc2Vec, PLN.

Introducción

En la actualidad existe una amplia gama de algoritmos que mejoran el comportamiento de muchas tareas de Procesamiento de Lenguaje Natural (PLN). También se encuentran varios trabajos relacionados con la identificación de nombres e idiomas, traducción automática, entre otros. Las técnicas de Word Embeddings (WE) codifican los significados de las palabras en espacios vectoriales de baja dimensionalidad, los mismos que se vuelven muy populares en la investigación del PLN (Senel et al., 2018). En este contexto, estas técnicas demuestran ser un mecanismo que facilita la tarea de calcular similitudes entre palabras (Guti & Keith, 2019); además, actúan mediante modelos matemáticos que codifican relaciones de palabras dentro de un espacio vectorial. Estas relaciones se crean mediante un proceso de formación no supervisado, es decir, basado en información de coocurrencia entre palabras en un corpus grande (Heimerl & Gleicher, 2018). Las relaciones codificadas incluyen propiedades semánticas y sintácticas de las palabras (Heimerl & Gleicher, 2018).

El interés por WE despierta gracias al algoritmo Word2Vec, que proporciona una forma eficiente de aplicar WE a partir de grandes corpus basados en el contexto de las palabras y el muestreo negativo (Gomez-Perez et al., 2020). Como consecuencia, Le y Mikolov propusieron Doc2Vec como una extensión de Word2Vec, aplicada a nivel de documento (Lau & Baldwin, 2016).

En términos generales, se puede definir a Doc2Vec como un algoritmo de PLN que incorpora palabras y oraciones, y se basa en representaciones de datos distribuidos y simbólicos denominados modelos de lenguaje de red neuronal (Kim et al., 2020). Además, puede capturar la relación semántica entre documentos de una gran colección de textos de forma eficaz. Este algoritmo aprende la semántica y composicionalidad de los elementos lingüísticos mediante el uso de una arquitectura de aprendizaje profundo. Esta arquitectura neuronal es simple, reduce significativamente el esfuerzo humano y comprime toda la información contextual y estructural en un vector numérico unidimensional (Nath Nandi et al., 2018).

Doc2Vec tiene la ventaja de permitir el análisis rápido de grandes cantidades de datos expresando palabras en el modelo de espacio vectorial y, al mismo tiempo, considerando el contexto (basado en la concurrencia de palabras) durante el aprendizaje (Kim et al., 2020). Sin embargo, dado que Doc2Vec asigna el mismo peso a todas las palabras de un documento, su eficacia se limita a encontrar un vector de palabras general que no expresa bien determinados temas (Kim et al., 2020).

En la actualidad, el Sistema Integrado de Seguridad ECU911 es el encargado de la atención y despacho de emergencias dentro del territorio ecuatoriano (Gobierno de la República del Ecuador, 2019). Los operadores que reciben las llamadas al número 911 utilizan los sistemas de despacho asistido por computadora (CAD) para priorizar y registrar las llamadas de incidentes, identificar el estado del mismo, así como establecer la ubicación de los socorristas en el campo y despachar eficazmente al personal de respuesta (Security & Directorate, 2011). A pesar de ser un sistema compatible con el análisis de incidentes, la clasificación de una llamada en diferentes niveles de emergencia no es automática y depende de varios factores relacionados con el operador, entre los que se encuentran su experiencia, capacitación y normativa interna de la plataforma ECU 911. Debido a la necesidad de mejorar el sistema de clasificación de incidentes, este artículo propone un proceso de clasificación automatizado basado en la técnica Doc2Vec, el mismo que se aplica para el análisis de textos originados por Sistema Integrado de Seguridad ECU911.

Finalmente, el documento está organizado de la siguiente manera. En la Sección I se presenta la introducción, en la Sección II, se describen los trabajos relacionados con el preprocesamiento de texto y se discuten los enfoques presentados por otros autores. Los materiales y métodos para implementar el sistema propuesto y los componentes básicos se analizan en la Sección III. La sección IV exponen los resultados y la discusión. Finalmente, se incluyen las conclusiones.

Trabajos relacionados

Los sistemas de despacho asistido por computadora (CAD) juegan un papel crítico en la respuesta a emergencias y la gestión de las unidades de rescate, sin embargo, aún cuentan con tareas que no están automatizadas como el ingreso de nombres, registro de direcciones y categorización de llamadas (Zhang et al., 2018). Estos procesos se pueden ver afectados por retrasos en la velocidad de escritura de los operadores, ruido de fondo e información incompleta o ambigua proporcionada por el llamante (Blomberg et al., 2019). Con base en estos antecedentes, se analizaron trabajos que involucran el uso de técnicas de PLN en el proceso de atención de llamadas de emergencia. Estas técnicas se utilizan en tareas de lectura y comprensión de audio y texto en grandes cantidades, detectando similitudes y diferencias (Nakata, 2017). Su propósito es el de clasificar y extraer información de forma automática, optimizando el tiempo de atención.

En el estudio de Ganguly y Ghosh (2018), se encuentra un nuevo enfoque de entrenamiento de vectores de palabras, que utiliza los tweets para mejorar las asociaciones entre ellas. En dicho estudio se utiliza un conjunto de datos relacionados a desastres, FIRE-2016. Su objetivo es demostrar que la representación de documentos con suma de vectores de palabras transformadas produce grupos más efectivos que otras técnicas como BOW o la suma de representaciones Word2Vec no transformadas. La efectividad de la clusterización mejora hasta un 14%.

Por otro lado, en la investigación de Dai (2017), se encuentra un método de agrupación basado en Word Embeddings para la clasificación relacionada con la salud, utilizando las redes sociales como fuente de datos. El modelo de WE que se emplea en esta investigación es Word2Vec. Se evalúa el desempeño en términos de precisión y recuperación, y se determina que el umbral más alto consigue una mayor precisión, mientras que el umbral más bajo se tiene una taza de recuerdo mejor. El algoritmo obtiene simulaciones que demuestran un buen rendimiento y una precisión que alcanza un 87,1%.

Por otra parte, en la investigación de Shao (2019), se utiliza las funciones de Word2Vec y Doc2Vec para un conjunto de tareas de clasificación de textos clínicos. Además, comprueba BOW-1, 2-gram con Word2Vec y se deduce que, en el conjunto más grande de combinación de las seis modalidades es BOW-1, 2-gram que logra un mejor desempeño, mientras que en los conjuntos más pequeños de modalidades individuales, Word2Vec presenta un mejor desempeño en cinco de los seis casos.

En el estudio de Gautam y Basava (2017), se busca la identificación automática y clasificación de ayudas de emergencia en comunidades macro de redes sociales. Se utiliza un modelo que se basa en la formación de vectores de incrustación con la ayuda de preprocesamiento textual y estadístico, aplicando el modelo en un conjunto de datos del terremoto de Nepal, logrando un 6,81% de precisión promedio en 5.250.000 vectores de incrustación.

Por otro lado, en el estudio de Balcerek (2017), se analiza un nuevo enfoque de clasificación de conversaciones telefónicas de emergencia mediante una red neuronal artificial. En dicho estudio se emplea la herramienta Neural Network Toolbox, la misma que detecta casos específicos de emergencias. Esta técnica requiere valores numéricos para sus procesos. En el caso de datos nominales o no numéricos, por ejemplo (nombre de ciudades), se trabaja mediante la asignación de un número único a cada entrada. Finalmente, se presenta un modelo de red neuronal artificial basado en la recolección de llamadas grabadas para identificar la clase a la que pertenecen.

En el estudio de Blomberg (2019), se utiliza aprendizaje automático como herramienta de apoyo para reconocer un paro cardíaco en llamadas de emergencias. En dicho estudio se implementa un marco de aprendizaje automático para identificar un paro cardiorrespiratorio extrahospitalario (OHCA – Por sus siglas en inglés) a partir de grabaciones sin editar de las llamadas de emergencia a un centro de despacho médico. El objetivo del estudio es probar si un marco de aprendizaje automático único puede mejorar la tasa de reconocimiento de OHCA en comparación con la que se obtiene por parte de despachadores capacitados.

En la investigacion de Balcerek (2017), se emplea la técnica de WE para representar la historia o anamnesis del paciente. En dicho estudio, utilizan los registros de las historias clínicas de 268,989 pacientes aplicando modelos Word2Vec y BERT. El objetivo de esta investigación es implementar un sistema de recomendación diagnóstica y clasificación de la anamnesis de los pacientes con cáncer. Con Word2Vec obtienen buenos resultados del modelo en textos donde la sintaxis está compuesta principalmente por las siglas de las especialidades.

En conclusión, existen varios métodos en el procesamiento de lenguaje natural para clasificar documentos, los mismos que demuestran resultados positivos en cuanto a la predicción de resultados clínicos o de otros tipos. El presente estudio aborda la implementación de PLN con la técnica de Doc2Vec, la misma que es una extensión de Word2Vec, cuya funcionalidad se enmarca en la captura de las relaciones entre palabras y que, a diferencia de los otros métodos, puede comparar documentos u oraciones completas.

Materiales y Métodos

En este estudio se propone la utilización y comparación de dos métodos de PLN: Word2Vec y Doc2Vec. En Word2Vec se usaron redes neuronales para la predicción de palabras y se consideraron dos técnicas: continuous bag of words (CBOW) y Skip-gram (SG) (Truşcă, 2019). En el caso de CBOW se predijo la palabra actual según el contexto, en tanto que en el modelo Skip-gram se usó una palabra actual para predecir las palabras que la rodean

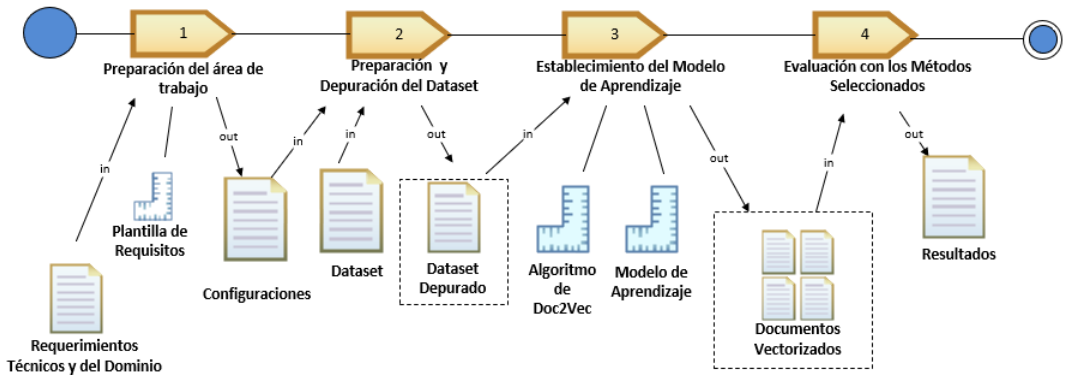
(Mikolov, Chen, et al., 2013). Doc2Vec se propuso como una extensión de Word2Vec que pudo ser igualmente aplicado a un párrafo o documento, es decir, fue independiente de la granularidad. Se buscó maximizar la probabilidad logarítmica media a partir de palabras de entrenamiento w_1, w_2 hasta w_t. Se consideró c como el tamaño del contexto de entrenamiento (Mikolov, Sutskever, et al., 2013).

$$1/T \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log p\left(w_{t+j} \mid \omega_t\right) \quad (1)$$

Para representar el método utilizado en esta implementación se utilizó Software Process Engineering Metamodel (SPEM), el cual es un metamodelo diseñado para describir procesos y sus componentes siguiendo un enfoque de modelado orientado a objetos con base en UML (Bendraou et al., 2008). El diagrama esquemático del enfoque propuesto es representado en la Figura 1. Este proceso se dividió en cuatro tareas: i) preparación del área de trabajo, ii) preparación y depuración del conjunto de datos, iii) establecimiento del modelo de aprendizaje y, iv) evaluación con los métodos seleccionados (i.e., Word to Vec, Doc to Vec). Las entradas se muestran a la izquierda de los procesos y las salidas a la derecha a manera de artefactos.

Figura 1

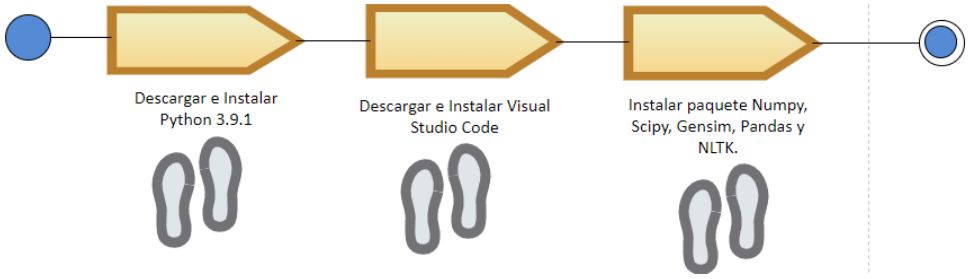
Proceso comparativo Doc2Vec – Word2Vec



Preparación del área de trabajo: La actividad representada en la Figura 2 muestra los pasos realizados para la preparación del área de trabajo. Se ha utilizado como entorno de desarrollo Visual Studio Code y como lenguaje de programación Python. Este lenguaje es ampliamente utilizado en el campo de la Inteligencia Artificial, por contar con varias librerías para su uso.

Figura 2

Preparación del área de trabajo



Se instaló Numpy, que es una biblioteca que da soporte a la creación de vectores y matrices multidimensionales de gran tamaño, junto con una gran colección de funciones matemáticas de alto nivel para operar con ellas (McKinney, 2013). Otro componente adicionado fue el paquete Pandas, el mismo que permite la lectura de archivos con formato delimitado por comas (CSV – por sus siglas en inglés). Se agregó la biblioteca de rutinas numéricas SciPy que proporciona bloques de construcción fundamentales para modelar y resolver problemas científicos, e incluyó estructuras de datos especializadas, como matrices dispersas y árboles k-dimensionales (Virtanen et al., 2020). Adicionalmente, se utilizó la biblioteca Gensim que permite: 1) la indexación de documentos digitales y búsqueda de similitudes; y 2) algoritmos escalables, rápidos y eficientes en memoria para la descomposición de valores singulares y la asignación de Dirichlet latente (Rehurek & Sojka, 2011). Finalmente, se instaló NLTK, la cual es una biblioteca para procesamiento de lenguaje natural y análisis de texto.

Preparación y Depuración del DataSet: Las tareas de preparación y depuración del dataset se ilustran en la Figura 3.

Figura 3

Preparación y Depuración del Dataset

Una vez configurada el área de trabajo, el dataset fue analizado. Los campos se definieron y se seleccionaron aquellos que intervienen en la clasificación, luego se depuró el dataset eliminando ruido y caracteres especiales como: (), [], {}, >, _, =, emoticones, entre otros. También se eliminaron las palabras que no tenían mayor relevancia dentro del contexto de estudio (como pronombres o artículos) sin que aquello signifique la pérdida de la similitud semántica del documento.

Establecimiento del Modelo de Aprendizaje: En esta actividad se incluyen los pasos a realizar para el definir el Modelo de Aprendizaje, con el cual se ejecuta la clasificación de una frase o documento (Figura 4).

Figura 4

Establecimiento del Modelo de Aprendizaje

Una vez preparado y depurado el dataset, se realizó la tokenización; este proceso dividió cadenas de texto más largas en piezas más pequeñas o tokens (Mayo, 2018). Un token es parte de un todo, por lo que una palabra es un token en una oración y una oración puede significar un token en un párrafo. Se ejecutó este proceso por medio del método word_tokenize(), el cual invoca a la instrucción tokenize para separar las palabras mediante espacios o puntuaciones; luego se utilizó el comando lower() para transformarlas en minúsculas

(Figura 5). Se consideró que la complejidad de la tokenización varió según la necesidad de la aplicación de PLN.

Figura 5

Ejemplo de Tokenización en Python

```
def tokenizacion(line):
    for d in line:
        tokenized_doc.append(word_tokenize(d.lower()))
```

Una vez tokenizado el documento, se aplicó el método TaggedDocument, el cual recorrió las listas tokenizadas, y las etiquetó empezando desde el valor cero (Figura 6).

Figura 6

Ejemplo de Etiquetación en Python

```
tagged_data = [TaggedDocument(d, [i]) for i, d in enumerate(tokenized_doc)]
```

Posteriormente se entrenó el Modelo de Aprendizaje, para lo cual se empleó la función Doc2Vec. Una vez que el modelo se entrenó, fue utilizado para la implementación, (Figura 7). Para empezar el entrenamiento se utilizó el método Doc2Vec y se establecieron siete parámetros:

Figura 7

Ejemplo de Doc2Vec en Python

```
model = Doc2Vec(tagged_data, vector_size=20, window=2, min_count=2, workers=4, epochs = 100, dm = 1)
model.save("ModeloEntranado.model")
```

- tagged_data: documento etiquetado
- vector_size: dimensión de neuronas de la capa oculta y del vector de salida
- window: máxima distancia entre la palabra actual y la pronosticada dentro de una frase
- min_count: ignora palabras que aparecen menos que el valor configurado
- workers: número de threads para entrenar el modelo
- epochs: número de iteraciones sobre el corpus
- dm: define el algoritmo de training. Si dm = 1 significa 'memoria distribuida' (PV-DM) y dm = 0 significa 'bolsa distribuida de palabras' (PV-DBOW). El modelo de memoria distribuida conserva el orden de las palabras en un documento, mientras que la bolsa de palabras distribuida solo utiliza el enfoque de la bolsa de palabras, que no conserva ningún orden de palabras.

Finalmente, para clasificar una frase, se la comparó con los modelos de aprendizaje creados previamente, con lo que se catalogó de acuerdo con el mayor nivel de similitud (Figura 8).

Figura 8

Ejemplo de Clasificación en Python

```python
def verificarSimilitud(color,frase):
    # Cargamos el modelo entrenado
    model = Doc2Vec.load(f'Modelos/MODELO_{color}.model')
    # imprimimos el modelo de vocabulario
    # print(model.wv.vocab)
    #print("\n",frase,"\n")
    test_doc = word_tokenize(frase.lower())
    print("\t",model.docvecs.most_similar(positive=[model.infer_vector(test_doc)],topn=1),"\n"
)
```

Resultados y Discusión

Evaluación con los Métodos

El experimento realiza utilizando la metodología propuesta por (Basili et al., 1986), que tiene los siguientes pasos: i) definición del alcance, ii) planificación del experimento, iii) operación, iv) análisis e interpretación de resultados y v) reporte de resultados. Estos pasos se detallan a continuación:

Definición del alcance

El alcance del experimento es la investigación del rendimiento de los métodos Word2Vec y Doc2Vec, basándose en un conjunto de conversaciones de un centro de emergencia, con el fin de determinar cuál de los dos métodos es el más eficiente al momento de buscar similitudes.

Planificación del experimento

Para realizar la evaluación de los métodos Doc2Vec y Word2Vec, se utiliza un programa que evalúa el porcentaje de similitud y el tiempo de ejecución de cada método utilizando los modelos entrenados anteriormente. La investigación busca responder: i) ¿Qué método es más eficiente en cuánto a similitud? y ii) ¿Qué método es más eficiente en cuánto a tiempo?

Operación

Las conversaciones de un centro de emergencias utilizadas en esta investigación son proporcionadas por Laboratorio de Investigación y Desarrollo en Informática – LIDI de la Universidad del Azuay, las cuales son preparadas para PLN. El conjunto de datos consta de 1051 conversaciones de emergencias de los operadores del centro de comando y control ECU 911, las cuales se pueden dividir en cuatro tipos de alertas según su criticidad en: i) verde, ii) amarilla, iii) naranja y iv) roja. Se clasifican las conversaciones que corresponden a cada tipo de alerta (Tabla 1).

Tabla 1

Clasificación de las conversaciones del ECU911

TIPO DE ALERTA	NÚMERO DE CONVERSACIONES
Verde	97
Amarilla	76
Naranja	359
Roja	519
Total	1051

Análisis e interpretación de resultados

Una vez clasificadas las conversaciones se entrenan distintos esquemas, tanto Doc2Vec como Word2Vec, con el propósito de crear modelos de acuerdo con el tipo de emergencia. En la Figura 9 se muestran clusters con los modelos entrenados empleando el método Doc2Vec en donde cada punto representa la vectorización de cada una de las conversaciones, agrupadas por la similitud que tienen entre sí. Se forma un modelo para cada tipo de emergencia.

En cambio, en la Figura 10se muestran clusters con los modelos entrenados empleando el método Word2Vec en donde cada punto representa la vectorización de cada una de las palabras de las conversaciones, agrupadas por la similitud que tienen con otras. De igual manera, se forma un modelo para cada tipo de emergencia.

Figura 9

Modelos entrenados con Doc2Vec

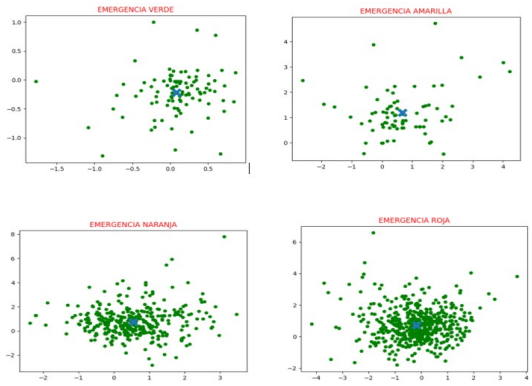
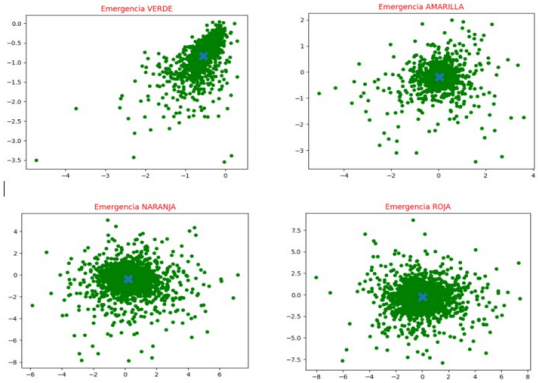


Figura 10

Modelos entrenados con Word2Vec



La Tabla 2 muestra las frases utilizadas para evaluar los modelos creados de Doc2Vec y Word2Vec. Cada frase fue depurada y cada palabra convertida en minúsculas.

Tabla 2

Frases para las pruebas en los modelos de Doc2Vec y Word2Vec

	FRASES A CLASIFICAR
A	amigo llamé hace ratito cuenca cerca camioneta toyota señor está manejando …
B	buenas tardes hágame favor disculpe quien tengo hablar pasa aquí sector monay …
C	aquí azogues buenas noches aquí azogues número para llamar policía pasa baile arriba local …
D	buenas buenos días hablo policía verá pasa barrio totoracocha estoy aquí casa señor llegaron …

En la Tabla 3 y Tabla 4 se detallan los resultados correspondientes en cuanto a similitud, utilizando los modelos de Doc2Vec y Word2Vec, empleando las frases de la Tabla 2. A continuación se realiza un análisis para predecir, mediante la frase ingresada, a qué tipo de emergencia pertenece.

Tabla 3

Porcentaje de Similitud empleando los modelos Doc2Vec

FRASE	MODELO	% DE SIMILITUD
A	**Verde**	**0,866**
	Amarilla	**0,721**
	Naranja	**0,627**
	Roja	**0,614**
B	Verde	**0,874**
	Amarilla	**0,924**
	Naranja	**0,618**
	Roja	**0,607**
C	Verde	**0,796**
	Amarilla	**0,685**
	Naranja	**0,962**
	Roja	**0,611**
D	Verde	**0,867**
	Amarilla	**0,867**
	Naranja	**0,762**
	Roja	**0,981**

Tabla 4

Porcentaje de Similitud empleando los modelos Word2Vec

FRASE	MODELO	% DE SIMILITUD
A	**Verde**	**-0,092**
	Amarilla	**0**
	Naranja	**0**
	Roja	**0**
B	Verde	**0**
	Amarilla	**0,163**
	Naranja	**0**
	Roja	**0**
C	Verde	**0**
	Amarilla	**0**
	Naranja	**0,498**
	Roja	**0**
D	Verde	**0**
	Amarilla	**0**
	Naranja	**0**
	Roja	**-0,072**

En la Tabla 5 y Tabla 6 se detallan los resultados correspondientes al tiempo de ejecución, utilizando los modelos de Doc2Vec y Word2Vec y empleando las frases de la Tabla 2. Se realiza un análisis para ver el tiempo de ejecución de cada modelo al ingresar una frase.

Tabla 5

Tiempo de ejecución empleando los modelos Doc2Vec

FRASE	TIEMPO DE EJECUCIÓN (S)
A	0,173
B	0,161
C	0,114
D	0,105

Tabla 6

Tiempo de ejecución empleando los modelos Word2Vec

FRASE	TIEMPO DE EJECUCIÓN (S)
A	1,478
B	1,382
C	1,430
D	1.110

Reporte de resultados

A base del análisis efectuado, se obtienen los siguientes resultados:

a) El método Doc2Vec resulta más eficaz cuando se analiza la similitud que tiene una frase entre sus modelos entrenados. Esto se debe a que un vector representa una frase del documento utilizado para el entrenamiento, y por lo tanto existe una mayor similitud de resultados en un menor tiempo de ejecución, lo que facilita una mejor clasificación de la frase ingresada ya sea en verde, amarilla, naranja o roja.

b) El método Word2Vec no resulta eficaz, dado que busca palabra por palabra en el modelo entrenado. Si el modelo tiene este tipo de búsqueda en que cada vector representa una sola palabra, se dificulta la búsqueda de una frase entera en vectores de palabras y da como resultado un mayor tiempo de ejecución.

Con estos resultados se concluye que el método más eficiente en cuanto a similitud de datos es Doc2Vec. De igual forma, se establece que el método más eficiente en cuanto a tiempo es Doc2Vec, debido a la forma de representar vectorialmente cada texto.

Conclusiones

En este estudio se ha explorado la aplicación de los modelos de aprendizaje automático sin supervisión Word2Vec y Doc2Vec en el Lenguaje de Programación Python. Se implementan modelos enfocados a recomendar una clasificación de los incidentes registrados por el Servicio Integrado de Seguridad ECU911. Se determina que Doc2Vec tiene mejor rendimiento para buscar similitudes en un corpus extenso, en tanto que Word2Vec es capaz de distinguir entre subconjuntos con diferentes proporciones de términos (positivo y negativo).

Para la evaluación se emplea un dataset de 1051 conversaciones de emergencias, sobre las que se aplican procesos de limpieza y depuración, para luego realizar la implementación del modelo y ejecutar las pruebas correspondientes. Como resultado se deduce que a nivel de rendimiento Doc2Vec es mejor al momento de comparar un texto y obtener su similitud, tanto en precisión como en tiempo. En cambio, Word2Vec realiza la vectorización por palabras, lo

que produce una pérdida en la semántica de la oración. En trabajos futuros se espera aplicar diferentes técnicas de PLN partiendo desde el análisis del audio de las grabaciones para luego procesar el texto asociado.

Agradecimientos

Los autores desean agradecer al Vicerrectorado de Investigaciones de la Universidad del Azuay por el apoyo financiero y académico, así como a todo el personal de la escuela de Ingeniería de Sistemas y Telemática, y el Laboratorio de Investigación y Desarrollo en Informática (LIDI).

Referencias

Balcerek, J., Pawlowski, P., & Dabrowski, A. (2017). Classification of emergency phone conversations with artificial neural network. *Signal Processing - Algorithms, Architectures, Arrangements, and Applications Conference Proceedings, SPA*, *2017-Septe*, 343–348. https://doi.org/10.23919/SPA.2017.8166890

Basili, V. R., Selby, R. W., & Hutchens, D. H. (1986). Experimentation in Software Engineering. In *IEEE Transactions on Software Engineering: Vol. SE-12* (Issue 7). https://doi.org/10.1109/TSE.1986.6312975

Bendraou, R., Combemale, B., Cregut, X., & Gervais, M.-P. (2008). *Definition of an Executable SPEM 2.0*. 390–397. https://doi.org/10.1109/aspec.2007.60

Blomberg, S. N., Folke, F., Ersbøll, A. K., Christensen, H. C., Torp-Pedersen, C., Sayre, M. R., Counts, C. R., & Lippert, F. K. (2019). Machine learning as a supportive tool to recognize cardiac arrest in emergency calls. *Resuscitation*, *138*(October 2018), 322–329. https://doi.org/10.1016/j.resuscitation.2019.01.015

Dai, X., Bikdash, M., & Meyer, B. (2017). From social media to public health surveillance: Word embedding based clustering method for twitter classification. *Conference Proceedings - IEEE SOUTHEASTCON*, *Table I*. https://doi.org/10.1109/SECON.2017.7925400

Gobierno de la República del Ecuador. (2019). *Servicio Integrado de Seguridad ECU911*. https://www.ecu911.gob.ec/

Gomez-Perez, J. M., Denaux, R., & Garcia-Silva, A. (2020). A Practical Guide to Hybrid Natural Language Processing. In *A Practical Guide to Hybrid Natural Language Processing*. Springer International Publishing. https://doi.org/10.1007/978-3-030-44830-1

Guti, L., & Keith, B. (2019). *A Systematic Literature Review on Word Embeddings* (Issue April 2020). Springer International Publishing. https://doi.org/10.1007/978-3-030-01171-0

Heimerl, F., & Gleicher, M. (2018). Interactive Analysis of Word Vector Embeddings. *Computer Graphics Forum*, *37*(3), 253–265. https://doi.org/10.1111/cgf.13417

Kim, S., Park, I., & Yoon, B. (2020). Sao2vec: Development of an algorithm for embedding the subject-action-object (SAO) structure using Doc2Vec. *PLoS ONE*, *15*(2), 1–26. https://doi.org/10.1371/journal.pone.0227930

Lau, J. H., & Baldwin, T. (2016). *An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation*. 78–86. https://doi.org/10.18653/v1/w16-1609

Mayo, M. (2018). *Preprocesamiento de datos de texto: un tutorial en Python*. https://medium.com/datos-y-ciencia/preprocesamiento-de-datos-de-texto-un-tutorial-en-python-5db5620f1767

McKinney, W. (2013). Python for data analysis. In J. S. and M. Blanchette (Ed.), *Journal of Chemical Information and Modeling* (Melanie Ya, Vol. 53, Issue 9). O'Reilly Media, Inc.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector

space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 1–12.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations ofwords and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 1–9.

Nakata, T. (2017). Text-mining on incident reports to find knowledge on industrial safety. *Proceedings - Annual Reliability and Maintainability Symposium*. https://doi.org/10.1109/RAM.2017.7889795

Nath Nandi, R., Arefin Zaman, M. M., Al Muntasir, T., Hosain Sumit, S., Sourov, T., & Jamil-Ur Rahman, M. (2018). Bangla News Recommendation Using doc2vec. *2018 International Conference on Bangla Speech and Language Processing, ICBSLP 2018*, 1–5. https://doi.org/10.1109/ICBSLP.2018.8554679

Rehurek, R., & Sojka, P. (2011). *Gensim — Statistical Semantics in Python* (Vol. 6611, Issue May 2010).

Security, H., & Directorate, T. (2011). *Computer Aided Dispatch Systems Computer-aided*. *September*.

Senel, L. K., Utlu, I., Yucesoy, V., Koc, A., & Cukur, T. (2018). Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio Speech and Language Processing*, *26*(10), 1769–1779. https://doi.org/10.1109/TASLP.2018.2837384

Shao, Y., Taylor, S., Marshall, N., Morioka, C., & Zeng-Treitler, Q. (2019). Clinical Text Classification with Word Embedding Features vs. Bag-of-Words Features. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 2874–2878. https://doi.org/10.1109/BigData.2018.8622345

Truşcă, M. M. (2019). Efficiency of SVM classifier with Word2Vec and Doc2Vec models. *Proceedings of the International Conference on Applied Statistics*, *1*(1), 496–503. https://doi.org/10.2478/icas-2019-0043

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Vázquez-Baeza, Y. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Zhang, J., Zhang, M., Ren, F., Yin, W., Prior, A., Villella, C., & Chan, C. Y. (2018). Enable automated emergency responses through an agent-based computer-aided dispatch system. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, *3*, 1844–1846.