

UNIVERSIDAD DE COSTA RICA
SISTEMA DE ESTUDIOS DE POSGRADO

DESARROLLO DE REPRESENTACIONES
VECTORIALES DE PALABRAS PARA ESPAÑOL
DE COSTA RICA

Trabajo final de investigación aplicada sometido a la
consideración de la Comisión del Programa de Estudios de
Posgrado en Computación e Informática para optar al grado y
título de Maestría Profesional en Computación e Informática

CRISTIAN ANGULO ARCE

Ciudad Universitaria Rodrigo Facio, Costa Rica

2019

Dedicatoria

A Lidieth Arce y Paola Guzmán, por su apoyo y motivación incondicional.

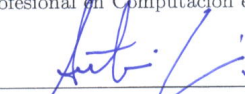
Agradecimientos

Quiero agradecer a todas las personas involucradas en mi formación en esta institución, a todos los profesores, familiares y amistades que constantemente estuvieron a mi lado. También al profesor Edgar Casasola por su tiempo, la guía y el apoyo en la creación de este proyecto. A Aurelio Sanabria y Gabriela Barrates, por su gran ayuda, apoyo y tiempo en la elaboración del documento.

A Gabriela Marín, Gabriela Barrates y Ricardo Villalón por la guía y apoyo brindado cuando estuve buscando y creando la propuesta.

¡Gracias!

“Este trabajo final de investigación aplicada fue aceptado por la Comisión del Programa de Estudios de Posgrado en Computación e Informática de la Universidad de Costa Rica, como requisito parcial para optar al grado y título de Maestría Profesional en Computación e Informática.”



Dr. Jorge Antonio Leoni de León
Representante del Decano
Sistema de Estudios de Posgrado



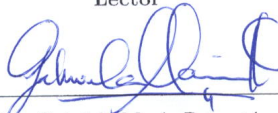
Dr. Edgar Casasola Murillo
Profesor Guía



Dra. Gabriela Barrantes Sliesarieva
Lectora



MSc. Aurelio Sanabria Rodríguez
Lector



Dra. Gabriela Marín Raventós
Directora del Posgrado en Computación e Informática



Cristian Angulo Arce
Sustentante

Índice general

Dedicatoria	ii
Agradecimientos	iii
Hoja de Aprobación	iii
Índice general	vii
Resumen	viii
Índice de cuadros	ix
Índice de figuras	xi
Licencia de publicación	xi
1 Introducción	1
1.1 Antecedentes	2
1.1.1 Construcción de <i>word embeddings</i>	2
1.1.2 Corpus en español	3
1.1.3 Contexto de una palabra	3
1.1.4 Clasificación de polaridad con <i>word embedding</i>	4
1.2 Planteamiento del problema	4
1.3 Objetivos	5
1.4 Justificación	5
1.5 Alcances y limitaciones	6
1.6 Descripción del resto del documento	6
2 Marco Teórico	8
2.1 Representaciones vectoriales de palabras	8
2.1.1 Construcción de representaciones vectoriales de palabras	11
2.1.2 Corpus	12
2.1.3 Ventanas de contexto	14

2.1.4	Word2Vec	15
2.2	Clasificación de la polaridad	17
2.2.1	Clasificadores	19
2.2.2	Conjunto de entrenamiento	22
2.2.3	Modelos de clasificación o predicción	23
2.2.4	Análisis de sentimiento	23
2.2.5	Modelo conceptual computacional para especificación de sistemas de análisis de sentimiento (SAM)	24
2.3	Evaluación de resultados de clasificación	26
2.3.1	Conjunto de pruebas	26
2.3.2	Métricas de evaluación	27
3	Metodología	30
3.1	Construcción de representaciones vectoriales de palabras	30
3.1.1	Corpus FBCR2013	32
3.1.2	Corpus Twitter	33
3.1.3	Preprocesamiento	34
3.1.4	Generación de las representaciones vectoriales con Word2Vec	36
3.2	Construcción de los modelos de clasificación	37
3.2.1	Conjunto de entrenamiento	38
3.2.2	Preprocesamiento	39
3.2.3	Enriquecimiento	39
3.2.4	Representación	39
3.2.5	Clasificación	40
3.3	Corpus InterTASS_CR	43
3.4	Evaluación	45
3.4.1	Conjunto de pruebas	46
3.4.2	Word embeddings SBW	46
3.5	Software utilizado para el desarrollo y experimentación	48
3.6	Hardware utilizado para la experimentación	48
4	Resultados	50
4.1	Representaciones vectoriales de las palabras	50
4.1.1	Tamaño del vocabulario de los embeddings	51

4.1.2	Duración de la creación de las representaciones vectoriales . . .	52
4.2	Modelos de clasificación	53
4.3	Evaluación de los embeddings utilizados en los modelos de clasificación	55
4.3.1	Evaluaciones de los modelos de CNN	55
4.3.2	Evaluaciones de los modelos SVM	56
4.3.3	Observaciones con respecto a la Exactitud	57
4.3.4	Observaciones con respecto a la macro puntuación F1	58
4.3.5	Observaciones generales	58
4.4	Herramienta Usure como resultado del proceso de desarrollo del proyecto	59
4.4.1	Descripción de la aplicación Usure	60
5	Conclusiones y trabajo futuro	63
5.1	Conclusiones	63
5.2	Trabajo futuro	64
	Bibliografía	65

Resumen

Las representaciones vectoriales de palabras, también conocidas como *word embeddings*, son modelados del lenguaje, donde la semántica de palabras o textos es transferida a vectores de números reales. Estos son utilizados en algunas aplicaciones de procesamiento del lenguaje natural o NLP (por sus siglas en inglés).

En este proyecto se crearon *word embeddings* con texto que incluye español de Costa Rica. Se utilizaron diferentes tamaños de ventana de contexto para su generación. Se aplicaron los *word embeddings* generados para resolver la tarea de análisis de sentimiento a partir de comentarios escritos en español costarricense. Se generaron modelos para dos tipos de clasificador: redes neuronales y máquinas de soporte vectorial.

Específicamente se evaluó la tarea de identificación de la polaridad de un texto. Se replicaron estas evaluaciones con un *word embeddings* preexistente que fue utilizado como línea base o referencia de comparación para todos los nuevos *word embeddings* generados en este trabajo. Las observaciones de estas evaluaciones muestran valores mayores o iguales en los resultados del análisis de sentimiento de comentarios de español de Costa Rica, al utilizar textos con variantes del español costarricense para crear los *embeddings*.

Palabras clave

análisis de sentimiento, español costarricense, representaciones vectoriales de palabras, ventanas de contexto

Índice de cuadros

2.1	Conjunto de entrenamiento para la creación del vector de la palabra rogando.	16
2.2	Matriz de confusión.	27
3.1	<i>Embeddings</i> o representaciones vectoriales de las palabras.	32
3.2	Estadísticas de la longitud de los comentarios para el corpus FBCR2013.	33
3.3	Estadísticas de la longitud de los comentarios para el corpus Twitter. .	34
3.4	Componentes de preprocesamiento utilizado en cada corpus.	35
3.5	Hiperparámetros para el algoritmo Word2Vec.	37
3.6	Hiperparametrización de la red neuronal convolucional.	41
3.7	Hiperparametrización de la máquina de soporte vectorial.	42
3.8	Estadísticas de la longitud de los comentarios para el corpus InterTASS_CR.	44
3.9	Distribución de las categorías para los subconjuntos de entrenamiento y pruebas del corpus InterTASS_CR.	45
3.10	Descripción del corpora utilizado en SBW.	47
3.11	Descripción de hiperparametros utilizados en Word2Vec para SBW. . .	47
3.12	Especificaciones del hardware	49
4.1	Duración de la creación de las representaciones vectoriales de las palabras por tamaño de ventana.	53

Índice de figuras

2.1	Ejemplo de word embeddings para tres palabras, 300 dimensiones. . . .	10
2.2	Vectores semánticos en un espacio de 3 dimensiones.	10
2.3	Vectores semánticos en un espacio de 3 dimensiones.	11
2.4	Componentes principales en la construcción de los word embeddings. .	12
2.5	Ventanas de contexto para cada palabra objetivo w	14
2.6	Componentes principales para la clasificación de polaridad.	19
2.7	Una red neuronal artificial simple, consta de una capa de entrada, una capa oculta y una capa de salida. Adaptación de [O’Shea y Nash, 2015].	20
2.8	Espacio bi-dimensional, e hiperplano separando dos clases.	22
2.9	Conjunto de entrenamiento.	22
2.10	Símbolo para representar un modelo de clasificación en esta documentación.	23
2.11	Diagrama general del modelo conceptual. Tomado de [Casasola Murillo, 2018].	25
2.12	Principales conceptos para la evaluación de los resultados al predecir datos.	26
3.1	Diagrama de descripción de la creación de las representaciones vectoriales de las palabras y sus variantes determinadas por el tamaño de ventana.	31
3.2	Diagrama de descripción de la creación de un modelo de clasificación. .	38
3.3	Diagrama de descripción de la evaluación.	46
4.1	Resultados de representaciones vectoriales de palabras.	51
4.2	Vocabulario para los embeddings creados a partir de los diferentes corpus.	52
4.3	Modelos de clasificación como resultado del entrenamiento en los clasi- ficadores con diferentes embeddings y el conjunto de entrenamiento de InterTASS_CR.	54

4.4	Exactitud y macro puntuación F1 para los modelos de clasificación construidos a partir de la CNN.	56
4.5	Exactitud y macro puntuación F1 para los modelos de clasificación construidos a partir de la SVM.	57
4.6	Exactitud para los modelos de clasificación construidos a partir del corpus de Twitter.	57
4.7	Macro puntuación F1 para los modelos de clasificación construidos a partir de la embeddings con ventana 2.	58
4.8	Vocabulario para los embeddings creados a partir de los diferentes corpus.	59
4.9	Paquetes principales de la aplicación Usure.	60
4.10	Componentes de un paquete principal de la aplicación Usure.	61



Autorización para digitalización y comunicación pública de Trabajos Finales de Graduación del Sistema de Estudios de Posgrado en el Repositorio Institucional de la Universidad de Costa Rica.

Yo, Cristian Angulo, con cédula de identidad 304360858, en mi condición de autor del TFG titulado Desarrollo de representaciones vectoriales de palabras para español de Costa Rica.

Autorizo a la Universidad de Costa Rica para digitalizar y hacer divulgación pública de forma gratuita de dicho TFG a través del Repositorio Institucional u otro medio electrónico, para ser puesto a disposición del público según lo que establezca el Sistema de Estudios de Posgrado. SI [X] NO * []

*En caso de la negativa favor indicar el tiempo de restricción: año (s).

Este Trabajo Final de Graduación será publicado en formato PDF, o en el formato que en el momento se establezca, de tal forma que el acceso al mismo sea libre, con el fin de permitir la consulta e impresión, pero no su modificación.

Manifiesto que mi Trabajo Final de Graduación fue debidamente subido al sistema digital Kerwá y su contenido corresponde al documento original que sirvió para la obtención de mi título, y que su información no infringe ni violenta ningún derecho a terceros. El TFG además cuenta con el visto bueno de mi Director (a) de Tesis o Tutor (a) y cumplió con lo establecido en la revisión del Formato por parte del Sistema de Estudios de Posgrado.

INFORMACIÓN DEL ESTUDIANTE:

Nombre Completo: Cristian Angulo

Número de Carné: B58010 Número de cédula: 304360858

Correo Electrónico: cristian.angulo@ucr.ac.cr

Fecha: 05-11-2019, Número de teléfono:

Nombre del Director (a) de Tesis o Tutor (a): Edgar Casasola Murillo

FIRMA ESTUDIANTE

Nota: El presente documento constituye una declaración jurada, cuyos alcances aseguran a la Universidad, que su contenido sea tomado como cierto. Su importancia radica en que permite abreviar procedimientos administrativos, y al mismo tiempo genera una responsabilidad legal para que quien declare contrario a la verdad de lo que manifiesta, puede como consecuencia, enfrentar un proceso penal por delito de perjurio, tipificado en el artículo 318 de nuestro Código Penal. Lo anterior implica que el estudiante se vea forzado a realizar su mayor esfuerzo para que no sólo incluya información veraz en la Licencia de Publicación, sino que también realice diligentemente la gestión de subir el documento correcto en la plataforma digital Kerwá.

Capítulo 1

Introducción

Una de las tendencias actuales dentro del área de procesamiento del lenguaje natural es el uso de representaciones vectoriales de palabras, también conocidas con el nombre de *word embeddings*. Por lo tanto, un *word embedding* es una representación de la semántica de una palabra mediante el uso de un vector de números reales [Jurafsky y Martin, 2018].

Desde la propuesta de una creación eficiente de vectores densos en el año 2013, por parte de Mikolov [Mikolov et al., 2013a], su uso se ha extendido a aplicaciones de procesamiento de lenguaje natural tales como: la clasificación de textos, sistemas de recomendación, extracción de conocimiento, detección de autor y plagio, la traducción entre lenguajes [Lample et al., 2018], la búsqueda y extracción de información [Mikolov et al., 2013a] y el análisis de sentimiento [Pang et al., 2008].

Los *word embeddings* son generados a partir de corpus lingüísticos. Un corpus lingüístico es un conjunto de textos de materiales escritos y/o hablados, debidamente recopilados para realizar ciertos análisis lingüísticos [Sierra Martínez, 2017]. Debido a que el desarrollo de este tipo de representaciones vectoriales es relativamente reciente, al momento de proponer esta investigación, no se habían construido ni evaluado *word embeddings* utilizando corpus provenientes del español costarricense.

El presente trabajo final de investigación aplicada construye representaciones vectoriales con base en textos de español de Costa Rica. Este es el principal aporte de esta investigación, ya que no existían *word embeddings* generados a partir de corpus costarricenses.

Para evaluar la calidad del corpus se decidió aplicarlo a una tarea propia del área del procesamiento del lenguaje natural. Se seleccionó el análisis de sentimiento [Pang et al., 2008], ya que se cuenta con un conjunto de datos de prueba con comentarios de español costarricense [Diaz-Galiano et al., 2018]. Además, se logró identificar la existencia de un *word embedding* conocido como SBW *Spanish Billion Words*, generado a partir de textos en español internacional [Cardellino, 2016].

La existencia de un *word embedding* en español genérico disponible y preconstruido nos permitió contar con una línea base para comparación.

1.1. Antecedentes

En esta sección se lleva a cabo una revisión de los antecedentes con la intención de identificar: los métodos utilizados para la creación de *word embeddings*, los corpus en español que se están utilizando para su creación, las diferentes técnicas utilizadas para delimitar el contexto de una palabra para la generación de los *word embeddings* y su uso para la tarea de identificación de polaridad en análisis de sentimiento.

1.1.1. Construcción de *word embeddings*

Las representaciones vectoriales de las palabras o *word embeddings* asocian cada palabra con un vector perteneciente a un espacio vectorial [Jurafsky y Martin, 2018]. Se basan en la teoría de la semántica distribucional [Harris, 1954], esta teoría establece que las palabras que ocurren y son usadas en el mismo contexto tienden a tener significados similares. Dicho de otra forma, establece que una palabra es caracterizada por las palabras que la acompañan [Firth, 1957].

La teoría descrita en el párrafo anterior sentó las bases del trabajo de Mikolov [Mikolov et al., 2013a]. Mikolov estudió modelos de predicción para representar palabras según la probabilidad de aparición en un contexto específico. Los contextos se obtienen utilizando las palabras ubicadas a la derecha y a la izquierda dentro de frases extraídas de una colección de documentos o corpus. La implementación más conocida se conoce como Word2Vec [Mikolov et al., 2013a].

El proceso de creación de representaciones vectoriales con Word2Vec ha sido descrito en el trabajo de [Grave et al., 2018], en el que se crean vectores de palabras para 157 lenguajes. En ese trabajo se describe de donde se obtuvieron los corpus, las fases para crear los modelos, la cantidad de palabras, hiperparametrización y herramientas utilizadas.

Aunque en este trabajo se decidió trabajar con Word2Vec, cabe mencionar que también existe otros métodos de creación de representaciones vectoriales como lo son: GloVe [Pennington et al., 2014], BERT [Devlin et al., 2018], FastText

[Bojanowski et al., 2016], entre otros. Estos métodos son variantes que incorporan desde aspectos morfológicos de las palabras como es el caso de FastText [Bojanowski et al., 2016], hasta relaciones de dependencia gramatical en oraciones [Devlin et al., 2018] o contextos globales [Pennington et al., 2014].

1.1.2. Corpus en español

La creación de *word embeddings* requiere de un corpus como insumo para la extracción de contextos. Actualmente existen corpus genéricos para español como es el caso de SBW o *Spanish Billion Corpus*. Además, existen corpus específicos para variantes del español como lo son: el corpus “FBCR2013” [Casasola Murillo y Leoni de León, 2016] y el corpus de “Twitter Costarricense” [Casasola Murillo y Marín Raventós, 2016] que contienen textos con variante del español costarricense.

Al llevar acabo esta parte de la revisión de antecedentes se pudo identificar la existencia de un modelo pre entrenado de *word embeddings* construido con Word2Vec a partir del corpus SBW [Cardellino, 2016]. El *word embedding* SBW fue generado a partir de texto en español internacional y no utiliza textos de Costa Rica.

Al finalizar esta etapa de revisión de antecedentes no fue posible encontrar *word embeddings* precalculados para español de Costa Rica. Sin embargo, la existencia de *embeddings* precalculados para SBW ofrece una oportunidad para ser utilizados como referente o línea base para comparar la calidad de las representaciones vectoriales que se crearan en este trabajo, al ser utilizadas para análisis de sentimiento de comentarios en español de Costa Rica.

1.1.3. Contexto de una palabra

La representación vectorial de una palabras es creada a partir del contexto en la cual se encuentra. Este contexto, compuesto por palabras, puede ser representado como una ventana corrediza (*sliding window*) que limita la cantidad de palabras ubicadas a su izquierda y a su derecha. La dimensión o cantidad de palabras de esta ventana tienen un impacto directo en el aspecto semántico capturado por el *word embedding*, y esto tiene un efecto sobre la calidad de los resultados al aplicarlos para la solución de alguna tarea particular [Lison y Kutuzov, 2017].

El tamaño de la ventana determina el significado de la distancia entre los términos. Ventanas grandes tienden a producir similitudes más tópicas (temáticas), mientras que las ventanas pequeñas tienden a producir similitudes más funcionales o sintácticas (agrupando verbos, sustantivos, adjetivos)[Goldberg, 2016]. Este tema será ampliado en el marco teórico de este trabajo.

1.1.4. Clasificación de polaridad con *word embedding*

El análisis de sentimiento fue de las primeras aplicaciones donde el uso de las representaciones vectoriales mejoraron la calidad de los resultados que tradicionalmente se obtenían con otros métodos [Le y Mikolov, 2014]. Mikolov propuso variantes de metodologías existentes para la creación de representaciones vectoriales y su uso en algunas tareas dentro del procesamiento natural del lenguaje, como clasificación de textos.

Debido al éxito de las representaciones vectoriales para llevar a cabo análisis de sentimiento en inglés, en el Taller de Análisis Semántico de la Sociedad Española para el Procesamiento del Lenguaje Natural (TASS), varios participantes han recurrido al uso de este tipo de representaciones generadas con texto en español para la clasificación de comentarios [Luque, 2019], [González et al., 2019], [Godino y D’Haro, 2019], [Montañés-Salas et al., 2019], [Altin et al., 2019], [Pastorini et al., 2019], [Garain y Mahata, 2019]. Algunos de estos participantes generaron sus propios *word embeddings*, utilizando el estado del arte, y otros utilizan *word embeddings* ya pre entrenados.

1.2. Planteamiento del problema

Tareas propias del procesamiento del lenguaje natural, como lo es el análisis de sentimiento, se ven afectadas por la variante del idioma o jerga propia del país o región donde se desarrollan [Brooke et al., 2009]. Hay palabras que puedan tener connotaciones positivas o negativas según el lugar en el que se utilicen [Casasola Murillo y Leoni de León, 2016]. Un caso particular corresponde a las frases idiomáticas y palabras que tienen significados diferentes entre países, o que solo existen en una región o contexto particular [Casasola Murillo y Leoni de León, 2016].

Al momento de llevar a cabo esta investigación no existía un modelo de represen-

tación vectorial de palabras construido a partir de español costarricense. Este recurso es necesario para la construcción de aplicaciones adaptadas al contexto nacional. Surge entonces el problema asociado a la construcción de representaciones vectoriales con textos de Costa Rica. Se hace necesario no solo identificar las fuentes de datos y el método para su desarrollo, sino también para su aplicación y evaluación de desempeño en una tarea particular.

Con el fin de producir un aporte para la solución del problema planteado se proponen los objetivos que se detallan a continuación.

1.3. Objetivos

Objetivo General

Desarrollar representaciones vectoriales de palabras a partir de dos corpus con español costarricense, evaluando su efecto sobre la clasificación de la polaridad de comentarios en español costarricense.

Objetivos Específicos

1. Seleccionar un método de construcción de representaciones vectoriales de palabras utilizando dos corpus de comentarios.
2. Construir variantes de representaciones vectoriales de palabras a partir de los corpus seleccionados con diferentes ventanas de contexto.
3. Construir modelos de clasificación a partir de dos clasificadores de comentarios utilizando las representaciones vectoriales creadas.
4. Comparar los resultados de los modelos de clasificación al clasificar comentarios de Costa Rica.

1.4. Justificación

Este acercamiento a las representaciones vectoriales de palabras en español costarricense es importante porque servirá de referencia para futuras investigaciones y

evaluación de aplicaciones. Este trabajo brinda una especificación de cómo desarrollar modelos de representación vectorial de las palabras para español de Costa Rica. El modelo generado sirve como punto de comparación para la evaluación de diferentes tareas de procesamiento de lenguaje natural en Costa Rica. Su disponibilidad como recurso lingüístico y computacional facilita la innovación en entornos productivos y académicos en el contexto costarricense.

1.5. Alcances y limitaciones

El presente proyecto se limitara a la generación de representaciones vectoriales utilizando Word2Vec, para dos corpus con los que ya se dispone: “FBCR2013” [Casasola Murillo y Leoni de León, 2016] y “Twitter” [Casasola Murillo y Marín Raventós, 2016].

Los resultados de la evaluación que se lleva a cabo en esta investigación solo son aplicables al conjunto de datos utilizados en la competencia TASS [Diaz-Galiano et al., 2018]. No se pretende generalizar los resultados, esto se debe a que no se cuentan con una muestra representativa de comentarios seleccionada aleatoriamente y bajo las condiciones que requiere un estudio experimental formal y que permita un tipo de generalización a este dominio. Se aclara entonces, que tanto la selección de los corpus, como los datos de prueba, fueron llevados a acabo siguiendo un criterio de oportunidad.

El análisis de sentimiento se llevó a cabo utilizando dos clasificadores de texto, que se seleccionaron por ser los más frecuentemente utilizados en los trabajos revisados en los antecedentes. Por consiguiente, se decidió utilizar una red neuronal convolucional y una máquina de soporte vectorial.

1.6. Descripción del resto del documento

En el capítulo 2 se explican los conceptos más relevantes para la comprensión de este trabajo. Se explica la idea principal de la semántica de vectores y uno de los métodos para crear vectores llamado Word2Vec. Después se explican dos tipos de clasificadores y los elementos para la evaluación.

En el capítulo 3 de metodología se explican los pasos para crear las representaciones vectoriales, los clasificadores para el análisis de sentimiento y la evaluación sobre la clasificación.

Los resultados son mostrados en el capítulo 4 y describen las representaciones vectoriales creadas, los modelos de clasificación y las observaciones de las evaluaciones realizadas en el análisis de sentimiento. En el capítulo final 5, concluye el proyecto y describe el trabajo futuro.

Capítulo 2

Marco Teórico

En este capítulo se presentan los conceptos teóricos necesarios para la comprensión del presente trabajo. Se explica qué son las representaciones vectoriales de las palabras o *word embeddings* fundamentadas en las teorías que le dieron origen. Se explica brevemente en que consiste la hipótesis distribucional [Harris, 1954] y la representación vectorial de las connotaciones de las palabras [Osgood et al., 1957]. Posteriormente, se presenta el método para crear representaciones vectoriales de las palabras propuesto por [Mikolov et al., 2013a]. Se explica particularmente la implementación conocida como Word2Vec y la importancia del tamaño de las ventanas para la definición del contexto. Luego, se explica la forma en que los clasificadores automáticos generan y utilizan modelos de clasificación. Para finalizar se da una explicación de las métricas conocidas como: exactitud y macro puntuación F1, que son comúnmente utilizadas para evaluar resultados de clasificación.

2.1. Representaciones vectoriales de palabras

La representación de palabras en un espacio semántico, comúnmente llamada en la literatura como *word embedding* consiste en una representación vectorial, que mediante un modelado algebraico, pretende representar el significado de una palabra. Se les llama así porque las palabras están empotradas en un espacio vectorial particular [Jurafsky y Martin, 2018]. La idea de llevar a cabo representaciones semánticas mediante el uso de vectores no es nueva. La semántica de vectores combina dos ideas previas: la hipótesis distribucional [Harris, 1954] y la representación vectorial de las connotaciones de una palabra [Osgood et al., 1957].

La primera idea a la que se hace mención es la **hipótesis distribucional** formulada en la década de los cincuenta por los lingüistas Martin Jooss, Zellig Harris y John R. Firth, en la cual se expone que las palabras que aparecen en contextos similares, tienden

a tener significados similares.

Es decir, que existe una relación entre la similitud en donde ocurren las palabras y el significado de las mismas [Harris, 1954]. Por ejemplo, los sinónimos “médico” y “doctor” tienen los mismos significados y pueden aparecer en los mismos contextos u oraciones. En este caso, si se tienen las oraciones: “En el hospital el doctor me operó y me sanó” y “El médico operó a mi mamá en el hospital”, se puede notar como las palabras “operó” y “hospital” se encuentran en ambas oraciones determinando una especie de contexto. A pesar de que las palabras “doctor” y “médico” no comparten una misma oración su contexto sugiere que se trata de términos correlacionados a nivel semántico.

La segunda idea, propuesta en 1957 por el psicólogo Charles E. Osgood, pretende representar **las connotaciones de las palabras** a través de tres dimensiones de afectividad. Las connotaciones son los significados de los aspectos de una palabra que están relacionados con las emociones y sentimientos del lector o escritor. Las tres dimensiones de Osgood son: la *valencia*, que representa lo agradable de un estímulo; la *excitación*, que es la intensidad de la emoción provocada por un estímulo; y la *dominancia*, que es el grado de control ejercido por un estímulo [Osgood et al., 1957]. Esas tres dimensiones se representan en una escala del 0 al 10.

Por ejemplo: la palabra “ruptura” en una connotación amorosa puede tener una valencia baja de 2.45, lo que significa que el estímulo no es agradable; una excitación de 5.65, es decir que la intensidad de la emoción es media; y una dominancia de 3.58, que connota un bajo control del estímulo. Por lo tanto, como resultado se obtiene un vector con los siguientes valores [2,45, 5,65, 3,58] para la palabra “ruptura” en una connotación amorosa [Jurafsky y Martin, 2018]. De esta manera la idea de Osgood se convirtió en una forma de representar el significado de una palabra mediante un punto en un espacio semántico.

A modo de ilustración, en la figura 2.1 se muestra un ejemplo de *word embedding* para cada una de las siguientes palabras: “Robo”, “Chorizo”, “Negocio”. El vector que representa cada una de estas palabras tiene un tamaño de 300. Al calcular la distancia entre los vectores se logra establecer una similitud en el espacio semántico. En la figura 2.2, se puede visualizar estos vectores en un plano de dos dimensiones, al aplicar reducción dimensional.

Palabra	<i>Word embedding</i>				
	1	2	3	...	300
robo →	0.01179157	0.2339416	0.0592651	...	0.4446244
negocio →	0.01189156	0.1637416	0.0572653	...	0.4646241
chorizo →	0.01272158	0.1737816	-0.0472353	...	0.6646241

Figura 2.1: Ejemplo de word embeddings para tres palabras, 300 dimensiones.

La figura 2.2 muestra la distribución de algunas palabras en el espacio vectorial de dos dimensiones. Cada punto de la imagen representa un vector de una palabra en un plano, donde palabras como “imagen”, “foto”, “vídeo” y “noticia” comparten una similitud tópica con respecto a periodismo. Las palabras “negocio”, “robo” y “chorizo” sugieren una connotación de hechos delictivos en el contexto costarricense.

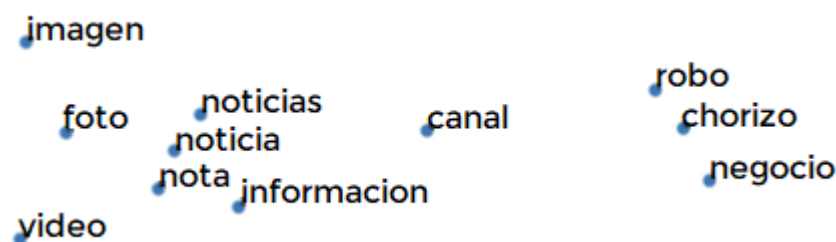


Figura 2.2: Vectores semánticos en un espacio de 3 dimensiones.

Se puede notar la utilidad semántica de los vectores al ser empleados en la resolución de analogías, como por ejemplo “¿Cuál es una palabra similar a mujer, de la misma forma que un hombre es similar a un rey?”. Esta analogía puede ser resuelta por medio de operaciones algebraicas aplicadas a los vectores de las palabras en un espacio semántico, como el que se observa en la figura 2.3¹, donde cada palabra de esta analogía es representada en un espacio de tres dimensiones. La solución de este problema viene dado por la ecuación $vector(“Rey”) - vector(“Hombre”) + vector(“Mujer”)$ cuyo

¹Adaptación tomada de <https://www.tensorflow.org/tutorials/representation/word2vec> el 15 de agosto del 2019

resultado es “Reina”. [Mikolov et al., 2013a].

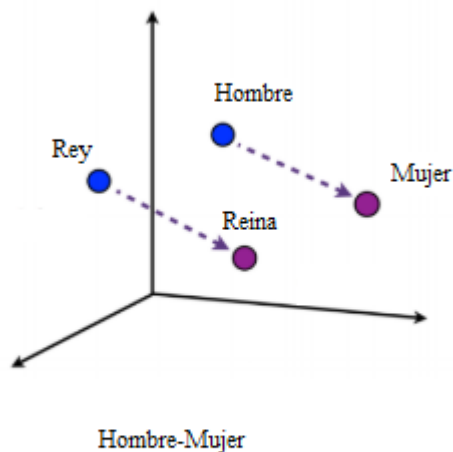


Figura 2.3: Vectores semánticos en un espacio de 3 dimensiones.

2.1.1. Construcción de representaciones vectoriales de palabras

Para la construcción de *word embeddings*, se deben considerar tres conceptos importantes: el corpus o textos que representan los insumos o materia prima de donde extrae la semántica, el algoritmo para crear los *word embeddings* y el contexto utilizado por ese algoritmo.

De acuerdo con [Baroni et al., 2014], los diferentes enfoques que son aprovechados por los algoritmos utilizados para la creación de *word embeddings*, se pueden dividir en dos categorías: los métodos basados en contar y los métodos predictivos.

Los **métodos que se basan en contar** realizan cálculos estadísticos, basándose en el hecho de cuantas veces aparece una palabra con otras palabras vecinas en un corpus o texto, plasmando así, datos estadísticos en vectores semánticos para cada palabra.

Los **métodos predictivos** tratan de predecir una palabra directamente a partir de sus palabras vecinas, y al mismo tiempo creando un vector de pesos que puede predecir

el contexto en el cual esta palabra se encuentra.

Los tipos de algoritmos comúnmente conocidos para crear representaciones vectoriales de las palabras son Tf-idf, GloVe, y Word2Vec [Jurafsky y Martin, 2018]. Este trabajo final de investigación se basó en el uso de Word2Vec.

El diagrama 2.4 muestra los principales conceptos utilizados para la creación de los *word embeddings* en este proyecto. Los componentes son: el corpus, el tamaño de ventana y el algoritmo Word2Vec, explicados a continuación.

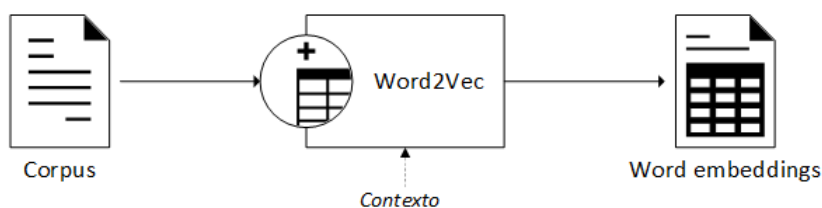


Figura 2.4: Componentes principales en la construcción de los word embeddings.

2.1.2. Corpus

Un corpus lingüístico consiste en un conjunto de textos de materiales escritos y/o hablados, debidamente recopilados para realizar ciertos análisis lingüísticos [Sierra Martínez, 2017].

Los corpus pueden ser constituidos por uno o varios libros, una revista, artículos periodísticos, textos científicos o literarios, mensajes de texto enviados entre dos personas por medio de computadoras, entre otros. Se puede hacer un compendio diacrónico o sincrónico de la lengua, de tal forma que un corpus puede componerse por la obra entera de un autor o todas sus obras, en efecto los corpus se conforman por textos.

Los corpus deben estar debidamente recopilados, no cualquier conjunto de textos puede ser un corpus. Para ilustrar mejor, una biblioteca sea cual sea el tipo o forma, sea digital o de material impreso, no constituye un corpus como tal, pero sí lo es la correcta selección de los documentos de esta biblioteca, con criterios bien delimitados y con una finalidad para un posterior análisis [Sierra Martínez, 2017].

Los corpus deben tener un objetivo, este puede ser con el fin de realizar un análisis cualitativo de alguna característica o fenómeno de la lengua, o con el objetivo de realizar análisis cuantitativos, como por ejemplo frecuencias de aparición de alguna característica.

Los corpus son valiosos en investigaciones lingüísticas, tanto teóricas como aplicadas, así mismo, su uso en las llamadas tecnologías del lenguaje es casi indispensable [Sierra Martínez, 2017]. Esta última área de aplicación es la de interés en este proyecto, pues los corpus son utilizados para crear *word embeddings*.

Los datos proporcionados en los textos de los corpus deben mostrar cómo funciona una lengua natural. A pesar de las diferencias que entre los estudios empíricos y los intuitivos, es necesario reconocer que los datos de un corpus reflejan la realidad y que la lengua escrita u oral pueda ser modelable a través de los corpus [Sierra Martínez, 2017].

Para la construcción de corpus se debe tener claro algunos aspectos como por ejemplo: el idioma, la localidad geográfica, los tópicos, el tipo de texto, las fuentes, el espacio temporal, entre otros que permitan delimitar su constitución para llevar a cabo un fin específico.

Los corpus anotados son aquellos a los que se les ha agregado algún tipo de información lingüística interpretativa. Ejemplo de esto, etiquetar un texto o palabra con la finalidad de clasificar o identificar dicho elemento. Este tipo de corpus anotados facilitan, mejoran o son indispensables en algún tipo de aplicación.

El corpus “Spanish Billions Words” (SBW) [Cardellino, 2016] sirve para ejemplificar un corpus. Las fuentes de los datos, los tópicos y tipos de textos son descritos brevemente a continuación.

SBW es un recurso que cuenta con 1500 millones de palabras. Consiste de un corpus no anotado del idioma español, compilado de diferentes corpus y recursos web. Los datos del corpus SBW provienen las siguientes fuentes:

1. La porción en español del corpus de SenSem.
2. La porción en español del corpus de Ancora.
3. *Tibidabo Treebank and IULA Spanish LSP Treebank Train and Test Partitions*.
4. La porción en español del los corpus del proyecto OPUS. Algunos recursos son libros, textos legislativos, comentarios de noticias y resoluciones de las naciones unidas.
5. La porción en español de algunas resoluciones del Europarl (*European Parliament*).

6. Recursos en español de Wikipedia, Wikisource y Wikibooks del año 2015.

2.1.3. Ventanas de contexto

El **contexto** y **ventana de contexto** son conceptos que se utilizan para comprender cómo funciona el algoritmo Word2Vec, a continuación su definición.

Dada una palabra objetivo w y su contexto C , C será el conjunto de palabras que se encuentran a la izquierda y a la derecha de w [Goldberg, 2016]. El contexto podría ser un documento entero, sin embargo, para efectos computacionales se utilizan contextos más pequeños. Este tipo de contexto es llamado ventana de contexto. El tamaño de esta ventana está determinado por un número c , que representa la distancia c máxima de palabras a la izquierda y la distancia c máxima de palabras a la derecha de w . Por lo tanto el tamaño total del contexto es $2 * c$.

Para ejemplificar estos conceptos nos basaremos en la siguiente oración:

“A Dios rogando y con el mazo dando”

De la oración anterior se puede tomar cualquier palabra como palabra objetivo w . Para una mejor comprensión, en la figura 2.5 se muestra cada palabra objetivo en negrita rodeada por su contexto, representado por las palabras en las celdas grises, tanto a la derecha como la izquierda.

Si se toma la palabra “rogando” (en el círculo) como palabra objetivo w , y un tamaño de ventana 2, el contexto estaría formado por las palabras “A”, “Dios”, “y” y “con”, con un tamaño de contexto de 4.

A	Dios	rogando	y	con	el	mazo	dando
A	Dios	rogando	y	con	el	mazo	dando
A	Dios	rogando	y	con	el	mazo	dando
A	Dios	rogando	y	con	el	mazo	dando
A	Dios	rogando	y	con	el	mazo	dando

Figura 2.5: Ventanas de contexto para cada palabra objetivo w .

Las ventanas de contexto se utilizan de tal manera que son deslizadas a través de un texto. Conforme la ventana se desliza de izquierda a derecha genera un conjunto de datos o grupo de palabras que es considerado el contexto de la palabra ubicada en el centro de la ventana. Estos datos son entonces utilizados para entrenar un modelo de predicción con el que se construyen los vectores.

El tamaño de las ventanas también tienen un efecto significativo en la similitud vectorial que se obtiene al calcular la distancia entre dos vectores. Ventanas grandes tienden a producir similitudes más tópicas, por ejemplo en inglés “*dog*”, “*bark*”, “*leash*” estarán agrupadas, lo mismo que “*walked*”, “*run*”, “*walking*”², mientras que las ventanas pequeñas tienden a producir similitudes más funcionales y sintácticas, por ejemplo: “*poodle*”, “*pitbull*”, “*rottweiler*”, que son razas de perro o incluso: “*walking*”, “*running*”, “*approaching*”³ [Goldberg, 2016].

Cuando se utilizan ventanas pequeñas los *word embeddings* que tienen una similitud alta indican que las palabras están relacionadas semánticamente, esto no necesariamente indica que las palabras sean sinónimos. Por ejemplo, las palabras “bueno” y “malo” aparecen en contextos similares pero en este caso no son sinónimos sino que son antónimos.

2.1.4. Word2Vec

Word2Vec es un método predictivo para representar las palabras como vectores **cortos** y **densos** (donde la mayoría de valores son distintos a cero). Los vectores densos trabajan mejor en tareas de procesamiento del lenguaje natural que los vectores dispersos (con la mayoría de valores iguales a cero) [Jurafsky y Martin, 2018].

Word2Vec fue propuesto por Tomas Mikolov y sus colegas en el 2013, en el artículo llamado “*Efficient estimation of word representations in vector space*” [Mikolov et al., 2013a]. El término “Word2Vec” es mencionado como una referencia a un paquete escrito en el lenguaje de programación C++, y representa la implementación de dos arquitecturas: CBOw y Skip-grama, explicadas brevemente más adelante.

Para la construcción de los *word embeddings*, se utiliza la ventana de contexto que cumple una función importante al desplazarse palabra por palabra, y captura el contexto de una palabra objetivo, que será computado para crear los *embeddings*.

²En español corresponde a los términos: perro, ladrido, correa

³Traducido al español corresponde a: caminando, corriendo o aproximando

Lo anterior se esclarecerá, tomando como ejemplo la oración “A Dios rogando y con el mazo dando” y la palabra “rogando” como objetivo w . Con la ventana se puede capturar o crear el conjunto de entrenamiento para el algoritmo Word2Vec, como se muestra en el cuadro 2.1, donde la primera columna representa la palabra objetivo, la segunda, representa la palabra propuesta como contexto, y la tercera, representa una etiqueta señalando si la palabra propuesta como contexto se encuentra en el contexto de la palabra objetivo.

Cuadro 2.1: Conjunto de entrenamiento para la creación del vector de la palabra rogando.

Palabra objetivo	Palabra contexto	¿Está en el contexto?
rogando	A	sí
rogando	Dios	sí
rogando	y	sí
rogando	con	sí
rogando	casa	no
rogando	hierva	no
rogando	cazar	no

Cuando se crea este conjunto de entrenamiento para cada palabra objetivo, aparte de agregar las palabras que están en el contexto, también se agregan palabras que no lo están, como se ve en el cuadro 2.1, con las palabras “casa”, “hierva”, y “cazar”, esto con la finalidad de evitar sobreentrenamiento. A este proceso de agregar palabras que no están en el contexto se le llama muestreo negativo.

La idea principal del método Word2Vec es que en lugar de contar la frecuencia con la que una palabra w aparece cerca de otras, se entrena un clasificador sobre una predicción binaria, basada en una palabra objetivo w y una palabra c , la cual es: ¿La palabra c aparece en el contexto de w ? Una vez entrenado el clasificador se tomarán los pesos de este, como los valores que representan el vector de la palabra w .

La ventaja de este tipo de método (Word2Vec) para crear vectores de palabras, radica en que no se requiere de etiquetado manual como se muestra en el cuadro 2.1, por lo tanto, para entrenar un clasificador se puede utilizar la palabra w y las

palabras que están a su alrededor como entrada, teniendo así salidas o etiquetas definidas de forma inferencial. De esta manera se realiza el aprendizaje no supervisado [Jurafsky y Martin, 2018].

Otra de las ventajas presentes en el modelo de Word2Vec es que hace uso de clasificadores de regresión logística, en lugar de una red neuronal compleja con capas ocultas, que requiere de algoritmos de entrenamiento más sofisticados y complejos, además de realizar predicción binaria en lugar de predicción de palabras [Mikolov et al., 2013a].

Mikolov y compañía proponen dos arquitecturas de Word2Vec: CBOW y Skip-grama, las cuales pretenden minimizar la complejidad computacional a la hora de calcular los vectores, y son descritos a continuación.

Continuous Bag-of-Words (CBOW)

El tipo de arquitectura CBOW trata de predecir la palabra objetivo basándose en las palabras que están en el contexto. Es computacionalmente más cara que la arquitectura Skip-grama y menos precisa en la similitud de las relaciones semánticas, pero precisa sintácticamente [Mikolov et al., 2013a]. Además es útil cuando la cantidad de datos de entrenamiento es pequeña.

Skip-grama

La arquitectura de Skip-grama predice el contexto basándose en la palabra objetivo. Se utiliza la palabra objetivo como entrada para un clasificador lineal-logarítmico y se predicen las palabras dentro de un rango a la izquierda y a la derecha de la palabra actual. Es semánticamente más precisa, pero sintácticamente menos precisa. En términos computacionales requiere menos procesamiento que el modelo CBOW. [Mikolov et al., 2013a].

2.2. Clasificación de la polaridad

La clasificación de texto tiene un importante papel en muchas aplicaciones de la actualidad. Por ejemplo: extracción de documentos, búsqueda web, clasificación de correo electrónico [Le y Mikolov, 2014], filtrado de información, análisis de sentimiento, sistemas de recomendación, gestión del conocimiento, resumen de documentos

[Kowsari et al., 2019], detección de lenguaje, detección de temas, son algunos de estas aplicaciones actuales. Para efectos de este documento nos enfocaremos en el análisis de sentimiento y la tarea de identificación de la polaridad de un fragmento de texto.

La clasificación de polaridad tiene la tarea de asignar, etiquetar o categorizar un texto o documento en específico. En el procesamiento natural del lenguaje los clasificadores de texto pueden analizar y asignar un conjunto de etiquetas o categorías predefinidas basadas en el contenido de los textos de forma automática. La definición formal de clasificación de texto según [Baeza-Yates y Ribeiro-Neto, 2008] es: dada una colección $D = \{d_1, d_2, \dots, d_n\}$ de documentos y un conjunto $C = \{c_1, c_2, \dots, c_L\}$ con L las clases de polaridades, un clasificador de texto es una función binaria dada por $F : D \times C \rightarrow \{0, 1\}$, esto es, una función que asigna el valor de 0 o 1 a cada par $[d_j, c_p]$, tal que $d_j \in D$ y $c_p \in C$. Si el valor asignado es 1, se dice que el documento d_j es un miembro de la clase c_p . Si el valor asignado es 0, se dice que el documento d_j no es un miembro de la clase c_p .

El proceso de clasificación de texto podría ser descompuesto en dos fases de [Kowsari et al., 2019]:

- Extracción de características o *features*: Se preprocesan los textos y se aplica la extracción de características, utilizando métodos como TF-IDF, TF, Word2Vec, GloVE entre otros.
- Técnicas de clasificación: Se selecciona el algoritmo de clasificación entre ellos: naïve bayes, máquinas de soporte vectorial, redes neuronales, entre otros.

El diagrama 2.6 muestra los principales conceptos, que se deben tener claros para la comprensión del proceso de clasificación en este proyecto. En las siguientes secciones se explican los tipos de clasificadores utilizados en este proyecto, se explica que es un conjunto de entrenamiento, y que es un modelo de clasificación. Adicional a estos conceptos se explica un tipo de clasificación de texto llamado análisis de sentimiento y un modelo para su especificación.

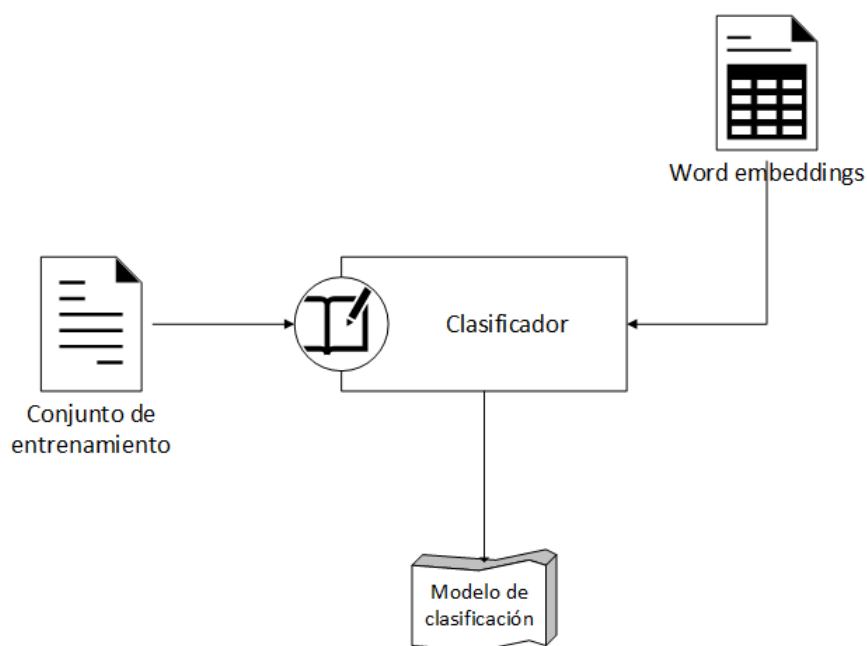


Figura 2.6: Componentes principales para la clasificación de polaridad.

2.2.1. Clasificadores

La clasificación de texto se puede llevar a cabo por medio aprendizaje la supervisado que utiliza datos etiquetados por humanos [Jurafsky y Martin, 2018]. Para efectos de este marco teórico se explican los dos algoritmos supervisados utilizados en este proyecto. A continuación, se presenta la idea general y fundamentos de las redes neuronales convolucionales y la máquina de soporte vectorial.

Redes neuronales convolucionales (CNN por sus siglas en inglés)

Se puede decir que las CNN son un tipo de Redes Neuronales Artificiales (ANN por sus siglas en Inglés). Las ANN son procesos computacionales inspirados en la función biológica del sistema nervioso. Las ANN están formadas por un gran cantidad nodos neuronales o unidades neuronales interconectadas, cuyo trabajo se entrelaza de una manera distribuida para colectivamente aprender de datos de entrada con la finalidad de optimizar la predicción de datos de salida [O’Shea y Nash, 2015].

La estructura básica de una ANN se puede observar en la figura 2.7, los datos son cargados usualmente en forma de un vector multidimensional en la capa de entrada,

que a la vez distribuye estos datos en la capa oculta (que pueden ser varias capas). Las capas ocultas realizan decisiones basadas en los resultados de las capas previas y ponderara cambios estocásticos en si mismas de forma tal que mejoran o perjudican el resultado final de las predicciones, esto se conoce como proceso de aprendizaje.

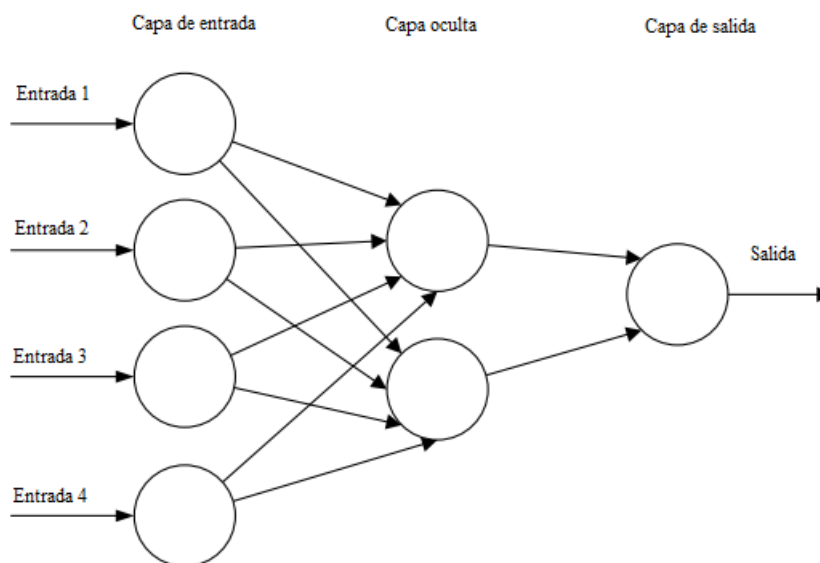


Figura 2.7: Una red neuronal artificial simple, consta de una capa de entrada, una capa oculta y una capa de salida. Adaptación de [O'Shea y Nash, 2015].

Las redes neuronales convolucionales son análogas las ANN tradicionales. Fueron diseñadas principalmente para reconocer patrones en imágenes, permitiendo codificar las características específicas en una arquitectura, esto permite a la red ser ideal en tareas para procesamiento de imágenes. Pero también han mostrado buenos resultados en otras áreas, por ejemplo en la clasificación de texto [Kim, 2014].

Las CNN están compuestas principalmente por tres tipos de capas, estas son: la capa convolucional, la capa de reducción (*pooling*), y la capa de clasificación (*fully-connected*). La **capa convolucional** determina la salida de las neuronas, las cuales están conectadas a regiones locales de las entradas y través de cálculos del producto escalar entre sus pesos y los valores de una región local de la entrada producen una convolución o salida. Al final del proceso de la capa se aplica una función de activación a cada uno de los valores de salida, los cuales serán datos de entrada de la siguiente capa. La **capa de reducción** calcula una reducción a través de la dimensionalidad

espacial de una entrada reduciendo el número de parámetros de salida. La **capa de clasificación** ejecuta la misma función que una ANN y predice las clases para cada valor de entrada a la CNN.

Máquinas de soporte vectorial (SVM por sus siglas en inglés)

Las máquinas de soporte vectorial constituyen un método de espacio vectorial para problemas de clasificación binaria. Es una de las técnicas utilizadas por algunos participantes en los talleres de InterTASS, para realizar análisis de sentimiento [Díaz-Galiano et al., 2018].

Dadas las representaciones vectoriales de textos, la idea principal de la SVM es buscar una superficie de decisión (un hiperplano) que pueda ser usada para separar de la mejor manera los elementos en dos clases c_a y c_b . El hiperplano que es aprendido a partir los datos de entrenamiento, divide el espacio en dos regiones de tal forma que los textos de la clase c_a están en una región y los textos de la clase c_b están en otra región. En un espacio bi-dimensional este hiperplano es una línea. En un espacio tri-dimensional este hiperplano es un plano. Una vez el hiperplano ha sido aprendido un nuevo texto t_j puede ser clasificado computando su posición relativa al hiperplano [Baeza-Yates y Ribeiro-Neto, 2008].

Para ilustrar esta separación considere un espacio bidimensional de ejemplo, cuyos datos de entrenamiento son separables linealmente, como se puede observar en la figura 2.8, la línea azul maximiza la distancia hacia los puntos más cercanos y constituye la mejor separación de hiperplano. En contraste con la línea roja la cual provee una peor separación, en este caso porque las distancia del hiperplano se encuentran más cerca de las clases separadas. En la figura de la derecha podemos observar las delimitaciones del hiperplano las cual están marcadas por las líneas de puntos, los vectores que delimitan el hiperplano son llamados vectores de soporte, las líneas que cruzan este espacio son candidatas a formar el hiperplano de decisión, y son paralelas a la delimitación del hiperplano. En este caso la línea W que divide el espacio en porciones iguales forma el mejor hiperplano y es llamada la decisión del hiperplano.

Las máquinas de soporte vectorial cuentan con un conjunto de hiperparámetros que deben ser ajustados dependiendo del problema. Dentro de estos hiperparámetros tenemos: el kernel que cambia el hiperplano a un modo no lineal, la constante C que cuando es baja permite un margen de error más grande, γ para hiperplanos no

lineales y cuando su valor es alto los datos de entrenamiento se ajustan con mayor rigidez (*sobre entrenamiento*), y el grado polinomial, entre otros.

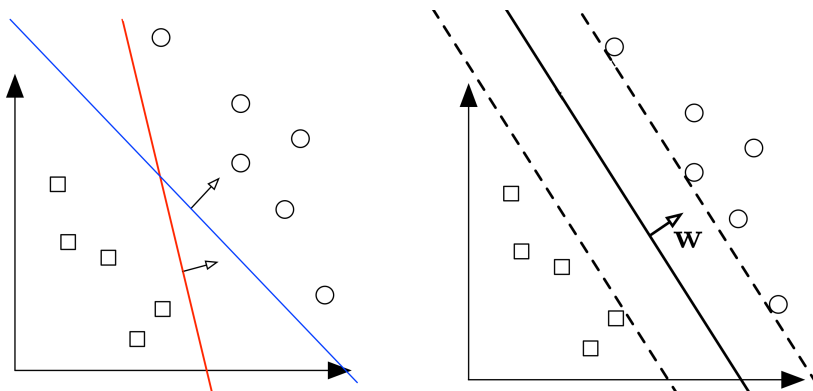


Figura 2.8: Espacio bi-dimensional, e hiperplano separando dos clases.

2.2.2. Conjunto de entrenamiento

Un conjunto de entrenamiento, es un conjunto de ejemplos utilizados para aprender [Raschka y Mirjalili, 2017]. En el caso de los algoritmos de clasificación de texto, cada dato de entrenamiento está compuesto por un texto y su debida polaridad o categoría. Como se puede observar en la figura 2.9, que cuenta con cinco comentarios y su respectiva polaridad, por ejemplo el comentario “la pura vida” se encuentra dentro de la categoría de los comentarios positivos. Estos comentarios y su debida polaridad se utilizan para entrenar los clasificadores y crear modelos de clasificación.

Comentario	Polaridad
la pura vida	Positivo
Que madre pasar la semana de cumpleaños así	Negativo
@Karlaram yo le llego mi negro, usted sabe que yo soy fiel a sus fiestas	Positivo
Que rica lluvia, pero los rayos me dan miedo	Neutro
Buenos días @MontserratCampo1 por DM le brindamos detalles para el retiro de las entradas	Ninguno

Figura 2.9: Conjunto de entrenamiento.

2.2.3. Modelos de clasificación o predicción

Un modelo de clasificación surge al entrenar un clasificador con un conjunto de entrenamiento, anterior a esto el clasificador tiene que estar previamente configurado con parámetros establecidos. Un modelo de clasificación es utilizado para predecir y en muchos casos puede seguir siendo entrenado [Raschka y Mirjalili, 2017].

Se puede ver a modelo de clasificación como una caja negra inteligente, la cual, recibe un dato de entrada y cuyo dato de salida es una etiqueta para el dato de entrada. La principal diferencia entre un clasificador y un modelo de clasificación es que el clasificador es una abstracción y el modelo de clasificación es una concreción o instancia del clasificador. La figura 2.10 representa un modelo de clasificación utilizado en esta documentación.

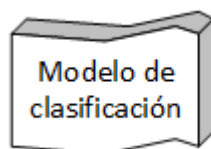


Figura 2.10: Símbolo para representar un modelo de clasificación en esta documentación.

2.2.4. Análisis de sentimiento

Es una una forma de clasificación de texto que se encarga de categorizar el sentimiento del escritor hacia un tema determinado. Permite la clasificación de la proyección del estado afectivo, intención comunicativa o evaluación que el interlocutor da a un determinado tema. Algunos ejemplos de actos valorativos sentimentales pueden ser la calificación o evaluación hacia una película, libro, actividad, producto, temas políticos o personales. El análisis de elementos subjetivos del sentimiento se apoya en el procesamiento del lenguaje natural, lingüística, y análisis de texto para extraer, identificar y clasificar textos.

La versión más simple de análisis de sentimiento es la que realiza la clasificación binaria. Es decir, predecir la polaridad negativa o positiva que denota un texto. Versiones más complejas se llevan a cabo, por ejemplo en las tareas de las ediciones de TASS, en las que se utilizan cuatro tipos de polaridad para los textos, estas son POSITIVO, NEGATIVO, NINGUNA y NEUTRO [Diaz-Galiano et al., 2018]. La etiqueta POSITIVO

representa una polaridad positiva o de compatibilidad, la NEGATIVA representa o denota rechazo o desacuerdo, la etiqueta NEUTRA presenta polaridad pero sin tendencia positiva o negativa, o con la misma tendencia a ambas, y NINGUNA que simplemente denota información sobre un hecho.

Algunos ejemplos de textos y su polaridad para el español de Costa Rica son:

Negativo: “Que madre pasar la semana de cumpleaños así”

Positivo: “@jnjiron @Karlaram yo le llevo mi negro, usted sabe que yo soy fiel a sus fiestas”

Neutro: “Que rica lluvia, pero los rayos me dan miedo”

Ninguno: “Buenos días @MontserratCampo1 por DM le brindamos detalles para el retiro de las entradas.”

Una de las metodologías para modelar y crear sistemas de análisis de sentimiento es el llamado “Modelo conceptual computacional para especificación de sistemas de análisis de sentimiento” el cual es utilizado en este proyecto y se explicara brevemente a continuación.

2.2.5. Modelo conceptual computacional para especificación de sistemas de análisis de sentimiento (SAM)

The Sentiment Analysis Model (SAM) es un modelo conceptual que permite especificar en términos computacionales las características implementadas en los sistemas de análisis de sentimiento [Casasola et al., 2019]. Se incluye su descripción en este trabajo ya que se utilizó de base para la descripción para la descripción de algunas secciones o pasos metodológicos. Por ejemplo, la descripción del proceso de preprocesamiento llevado a cabo.

La figura 2.11 muestra las relaciones existentes entre los elementos del modelo. En el lado izquierdo de la figura se puede apreciar el flujo de procesamiento típico de una aplicación de procesamiento de lenguaje natural. En la parte central de la figura se observa los diferentes componentes del modelo según su rol dentro del sistema, aquí se incluyen los componentes de preprocesamiento que modifican la estructura del

texto, los componentes utilizados para enriquecimiento que agregan información extra a cada texto de los comentarios, los componentes de representación que transforman los comentarios con fines computacionales, y finalmente los componentes para llevar a cabo la clasificación de los comentarios. A la derecha de la figura se muestran los recursos externos como diccionarios para identificar elementos o palabras y aplicar funciones de mutación sobre dichos elementos y herramientas computacionales o lexicones. Las flechas representan funciones que pueden ser de dos tipos: funciones de transformación o funciones de utilización de recursos externos.

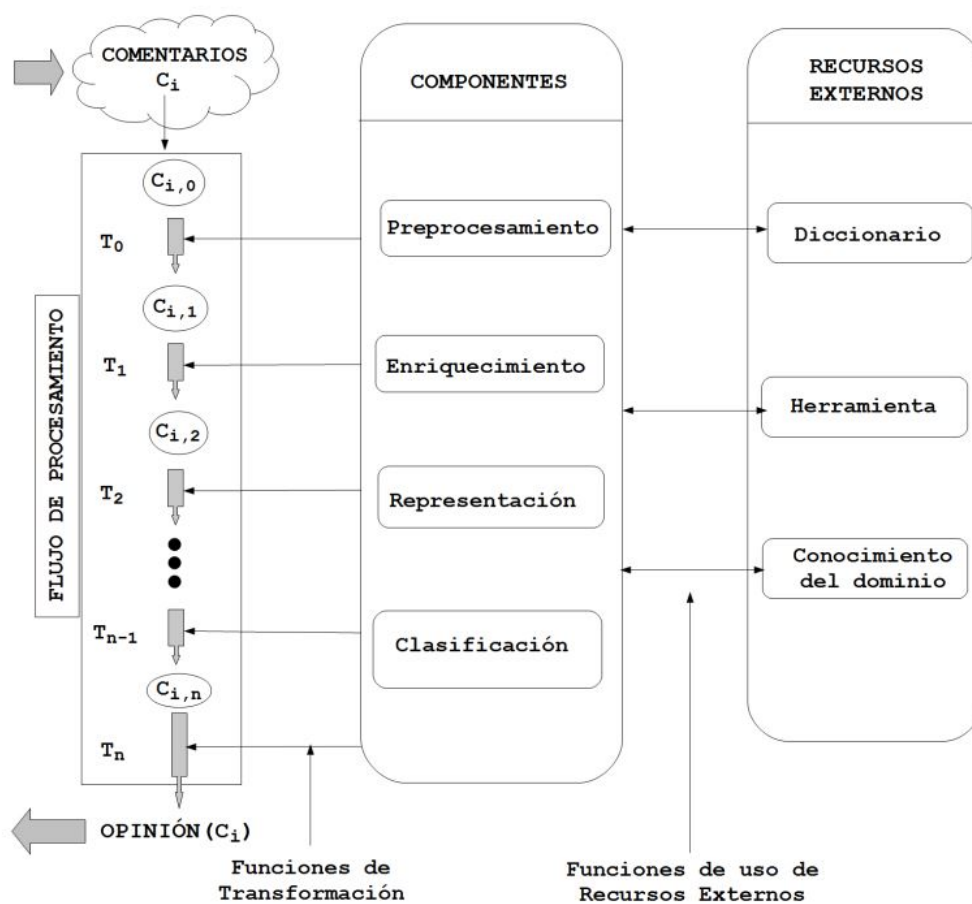


Figura 2.11: Diagrama general del modelo conceptual. Tomado de [Casasola Murillo, 2018].

El Modelo SAM será utilizado en el presente trabajo para la descripción de los

procesos asociados al procesamiento de lenguaje natural.

2.3. Evaluación de resultados de clasificación

La evaluación es el principal paso para validar la propuesta de un nuevo método de clasificación [Baeza-Yates y Ribeiro-Neto, 2008], es decir, cuantificar que tan bueno o que tan malo es un método para clasificar.

Las evaluaciones de los *word embeddings* para este proyecto se llevaron a cabo evaluando su uso en clasificadores de polaridad. La figura 2.12 muestra las principales definiciones que son explicadas en esta sección. Una vez obtenidos los modelos de clasificación, se procede a evaluar las predicciones realizadas por dichos modelos al utilizar un conjunto de pruebas. Las exactitud y macro puntuación F1 son las métricas utilizadas para evaluar las predicciones de un modelo de clasificación en este proyecto.

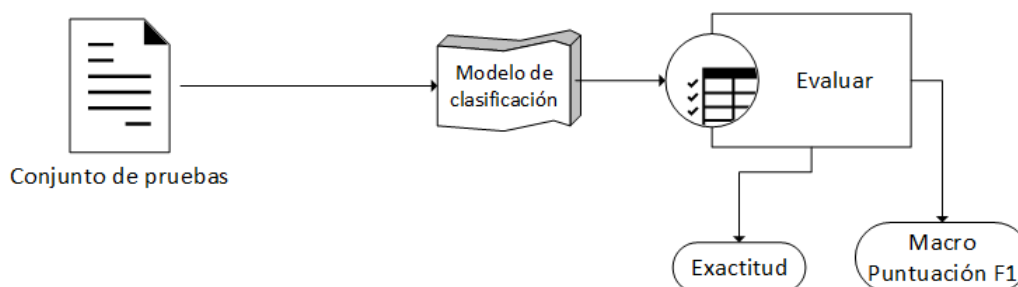


Figura 2.12: Principales conceptos para la evaluación de los resultados al predecir datos.

2.3.1. Conjunto de pruebas

En el caso de la clasificación de texto, cada elemento de un conjunto de pruebas está constituido por un texto, y su etiqueta o polaridad. A diferencia del conjunto de entrenamiento que es utilizado para entrenar los clasificadores, el conjunto de pruebas es utilizado para evaluar los modelos de clasificación.

El conjunto de pruebas tiene que ser independiente al conjunto de entrenamiento, esto es, que no tiene que existir intersección entre los elementos de cada conjunto. Ambos conjuntos deben seguir la mismas distribución probabilística.

2.3.2. Métricas de evaluación

En el caso de este proyecto, se requiere evaluar los *word embeddings* creados al ser utilizados en la clasificación de polaridad. Para evaluar los resultados de las predicciones de los los clasificadores se utilizará una matriz de confusión para describir las métricas expuestas en esta subsección. Por lo tanto, se explican las características del uso de una matriz de confusión para clasificación.

Matriz de confusión

La matriz de confusión es una tabla que a partir de los resultados de un clasificador n-ario, compara la cantidad de valores predichos contra la cantidad de valores actuales, correctos o reales de ciertos datos [Baeza-Yates y Ribeiro-Neto, 2008]. El cuadro 2.2 muestra una matriz de confusión para un clasificador n-ario con clases c_1, c_2, \dots, c_n . Exceptuando los valores del eje diagonal, los valores de las celdas del eje vertical (predicho) representan la cantidad de falsos positivos de una clase c_x , y los valores de las celdas del eje horizontal representan la cantidad de falsos negativos de una clase c_x .

Cuadro 2.2: Matriz de confusión.

Real/Predicho	c_1	c_2	..	c_n
c_1	V $c_{1,1}$	F $c_{1,2}$..	F $c_{1,n}$
c_2	F $c_{2,1}$	V $c_{2,2}$..	F $c_{2,n}$
..
c_n	F $c_{n,1}$	F $c_{n,2}$..	V $c_{n,n}$

Esta matriz se construye con la finalidad de obtener valores de varias métricas que permitirán evaluar los *embedigs* generados. A continuación se describen cada una de esas métricas.

Exactitud (*Accuracy*)

La exactitud es la métrica que representa la cantidad de comentarios de las clases que fueron clasificados correctamente con respecto al total de comentarios del conjunto.

Como se puede observar en la fórmula 2.1, n es igual a la cantidad de clases o categorías, Vc_i la cantidad de clasificaciones correctas para la categoría i , y Fc_i el total de clasificaciones mal predichas para la categoría i (falsos positivos o falsos negativos).

$$\frac{\sum_{i=1}^n Vc_i}{\sum_{i=1}^n Vc_i + Fc_i} \quad (2.1)$$

Precisión (*Precision*)

La precisión representa la fracción de instancias recuperadas que son relevantes, e indica que tan bueno es el clasificador identificando falsos positivos. Como se puede observar en la fórmula 2.2, c_x representa una categoría, Vc_x representa la cantidad de predicciones correctas para la categoría x , y FPc_x representa la cantidad de predicciones que fueron clasificadas como c_x pero que en realidad no lo son (falsos positivos), esto sería el eje vertical en nuestra matriz de confusión 2.2.

$$Pc_x = \frac{Vc_x}{Vc_x + FPc_x} \quad (2.2)$$

Cobertura o exhaustividad (*Recall*)

La cobertura representa la fracción de instancias relevantes que han sido recuperadas, o que tan bueno es el clasificador identificando falsos negativos. Como se puede apreciar en la fórmula 2.3, c_x representa una categoría, Vc_x representa la cantidad de predicciones correctas para la categoría x , y FNc_x representa la cantidad de predicciones que no fueron clasificadas como c_x pero que en realidad sí lo son (falsos negativos), esto sería el eje horizontal en nuestra matriz de confusión 2.2.

$$Rc_x = \frac{Vc_x}{Vc_x + FNc_x} \quad (2.3)$$

Si la precisión es mayor a la cobertura el modelo es mejor identificando valores correctos que identificando valores incorrectos y vice versa.

Puntuación F1

La puntuación F1 es una métrica que pretende ponderar en un único resultado la precisión y la cobertura de una clase. La fórmula 2.4 representa la puntuación F1, donde

Pc_x es la precisión para la clase x y Rc_x representa la cobertura para la clase x .

$$F1c_x = 2 \cdot \frac{Pc_x \cdot Rc_x}{Pc_x + Rc_x} \quad (2.4)$$

Promedios macro y micro

Cuando la clasificación no es binaria (más de dos categorías de clasificación) se requiere una unidad de medida de agregación para las métricas de precisión, cobertura y puntuación F1.

El promedio **micro** calcula las métricas globalmente contando el total de verdaderos positivos, falsos negativos y falsos positivos. El promedio **macro** calcula las métricas para cada categoría y seguidamente aplica el promedio a estas métricas.

Una vez claros los fundamentos teóricos utilizados para la creación de esta investigación, en el siguiente capítulo se define la metodología empleada para alcanzar cada uno de los objetivos.

Capítulo 3

Metodología

La metodología utilizada para lograr el objetivo principal y cada uno de los objetivos específicos en este proyecto está dividida en seis secciones descritas brevemente a continuación.

Primero, se describen los pasos llevados a cabo, y cada uno de los insumos o corpus utilizados para la creación de los *word embeddings*. Seguidamente, se realiza la descripción para la creación de los modelos de clasificación, con base en el modelo conceptual computacional para la especificación de sistemas de análisis de sentimiento (SAM).

En tercer lugar, se describe el corpus InterTASS_CR, que fue utilizado para entrenar, crear y probar los modelos de clasificación. Como cuarto punto, se detalla el proceso de evaluación de los modelos de clasificación que utilizaron los *word embeddings* creados en este proyecto. Después, se realiza la descripción del software utilizado, donde se exponen datos técnicos y de uso de esta herramienta. Por último, se realiza una breve descripción del hardware utilizado en este proyecto.

3.1. Construcción de representaciones vectoriales de palabras

En esta sección se describen los procesos de construcción de los *word embeddings* y sus variantes. De esta manera, se describe el proceso metodológico para el primer objetivo específico del proyecto, el cual es seleccionar un método de construcción, y el segundo objetivo relacionado con las variantes de estas representaciones vectoriales.

La figura 3.1 muestra los pasos necesarios utilizados para la creación de cada *word embedding* en este proyecto. Primero, se usó cada uno de corpus (descritos más adelante), siendo éstos la materia prima con los que se crearon los *word embeddings*. En el caso de este proyecto cada corpus estuvo compuesto por comentarios provenientes de redes sociales.

Seguidamente, cada uno de los comentarios fue preprocesado. El preprocesamiento incluyó la aplicación de funciones de mutación a cada uno de los comentarios del corpus. Las funciones fueron eliminación de signos de puntuación, cambio de mayúsculas por minúsculas, entre otras.

Un ejemplo de preprocesamiento es el siguiente, si tenemos el comentario “¡Es la pura vida!” el mismo será preprocesado o normalizado de tal forma que el resultado será “pura vida”; expresión a la que se le eliminaron palabras “ruido” y los signos de interrogación. Al igual que como se realizó con “¡Es la pura vida!”, este preprocesamiento se aplicó a cada uno de los comentarios o sentencias del corpus, teniendo como resultado un corpus normalizado, como se observa en la figura 3.1. La descripción del preprocesamiento es abordada detalladamente en la subsección 3.1.3.

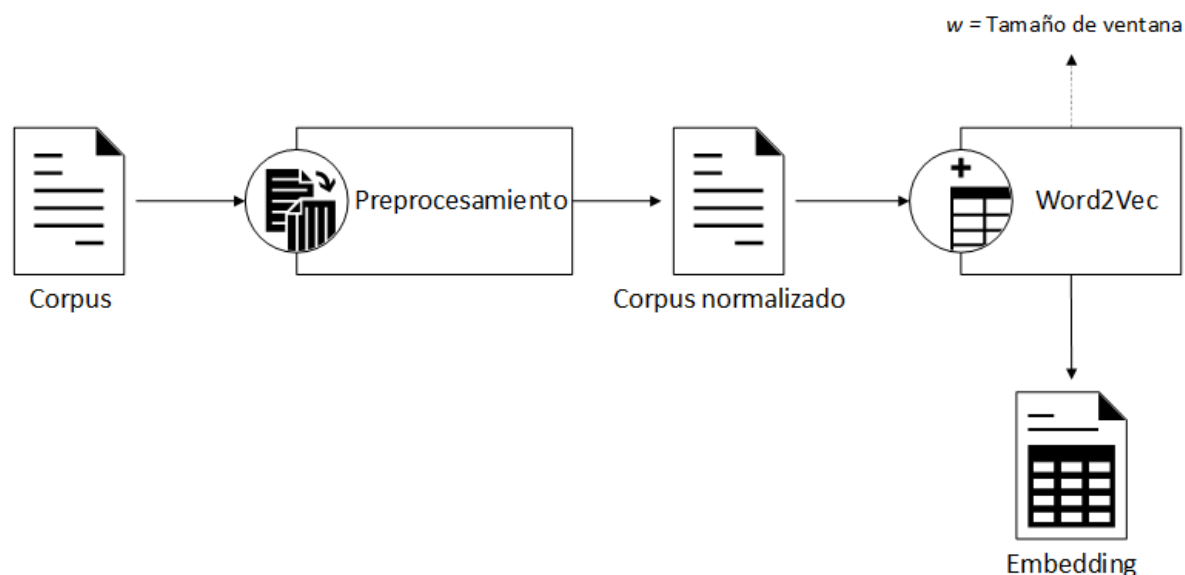


Figura 3.1: Diagrama de descripción de la creación de las representaciones vectoriales de las palabras y sus variantes determinadas por el tamaño de ventana.

Cada corpus normalizado fue utilizado para alimentar el algoritmo que crea las representaciones vectoriales de las palabras. Para las mismas se utilizó el método Word2Vec en su arquitectura skip-grama, utilizando como hiperparámetro el tamaño de ventana representado en la figura por la letra w . Se debe de tener claro que se generó un *word embedding* por cada variante de ventana. Sobre esto se profundizará en la subsección 3.1.4.

Los *word embeddings* o representaciones vectoriales de las palabras son diccionarios del tipo palabra-vector, como se puede observar en el cuadro 3.1; pero también representan modelos que pueden seguir siendo entrenados con más corpus. Continuando con el ejemplo anterior, la representación vectorial de la palabra “pura” sería el vector $[0.34, \dots, -0.56]$ y de “vida” el vector $[2.03, \dots, 0.90]$, ambos vectores de tamaño 300.

Cuadro 3.1: *Embeddings* o representaciones vectoriales de las palabras.

Índice	Palabra	Vector
...
876	pura	$[0.34, \dots, -0.56]$
...
56730	vida	$[2.03, \dots, 0.90]$

Teniendo claro cuál es el proceso de creación de los *word embeddings* es importante mencionar y especificar cuáles fueron los corpus utilizados para su creación, también detallar los procesos de “preprocesamiento” y “la creación de las representaciones con Word2Vec”. En las siguientes subsecciones se describen estas partes metodológicas con mayor detalle.

3.1.1. Corpus FBCR2013

La totalidad de los datos del corpus FBCR2013 fueron obtenidos en el ámbito de Costa Rica. Con este corpus se crearon varias de las representaciones vectoriales en el presente trabajo.

Es un corpus recopilado utilizando recolectores de información de la herramienta Sentímetro [Casasola Murillo y Leoni de León, 2016]. El corpus fue extraído de publicaciones y comentarios de los foros costarricenses más populares en Facebook. Todos los comentarios se refieren a Costa Rica y las publicaciones abarcan la totalidad del año 2013. Los tópicos más relevantes en ese año fueron la elecciones presidenciales y la participación de la selección nacional en el campeonato mundial de fútbol.

Este corpus fue utilizado en el análisis de la transferencia de la polaridad semántica de frases idiomáticas a comentarios de opinión

[Casasola Murillo y Leoni de León, 2016], en el cual las frases idiomáticas en una opinión de redes sociales están vinculadas con la polaridad del comentario.

El corpus está compuesto por 1,912,152 comentarios, con una media de 19.93 palabras por comentario, y una desviación estándar de 24.87 palabras de la media; como se puede observar en el cuadro 3.2. Además, podemos observar los percentiles (P_x) que representan la cantidad de palabras que se encuentran en un comentario, ubicado en un porcentaje de una lista en la que cantidad de palabras de los comentarios es ordenado de menor a mayor. Nótese que el comentario que tiene mayor tamaño cuenta con 1,198 palabras, mientras que el que menos tiene, cuenta con una palabra.

Cuadro 3.2: Estadísticas de la longitud de los comentarios para el corpus FBCR2013.

Estadística	Longitud
Comentarios	1,912,152
\bar{x}	19.93
σ	24.87
P_0	1
P_{25}	6
P_{50}	12
P_{75}	24
P_{100}	1,198

3.1.2. Corpus Twitter

El corpus Twitter está conformado por comentarios de Twitter y es más grande que el corpus FBCR2013. Fue recolectado durante el año 2013 utilizando el motor de búsqueda implementado con el software libre "Solr" de la fundación apache [Casasola Murillo y Marín Raventós, 2016]. Cuenta con comentarios de español americano (incluye Costa Rica) y de España.

Los datos estadísticos presentados en el cuadro 3.3 muestran la cantidad de más de 35 millones de tweets recolectados, con un tamaño promedio de 10 palabras por comentario y desviación estándar de 6 palabras; mucho menor que la desviación estándar de los textos presentes en el corpus FBCR2013. Lo que nos indica poca variabilidad

en el tamaño de los comentarios. El comentario más pequeño tiene una palabra y el comentario más grande tiene 70 palabras.

Cuadro 3.3: Estadísticas de la longitud de los comentarios para el corpus Twitter.

Estadística	Longitud
Comentarios	35,000,007
\bar{x}	10.02
σ	6.03
P_0	1
P_{25}	5
P_{50}	9
P_{75}	14
P_{100}	70

3.1.3. Preprocesamiento

El preprocesamiento es un conjunto de operaciones que se llevan a cabo sobre textos o corpus con el fin de descomponer, normalizar e identificar los diferentes tipos de términos. En este caso se utilizó para especificar la normalización de los corpus para la creación de los *word embeddings*.

Los textos se preprocesaron utilizando los componentes que se listan en el cuadro 3.4. Como se puede observar, a la mayoría de componentes se les aplicó la función de eliminación. Los caracteres codificados en HTML, y UTF-8 fueron sustituidos por su equivalente en ASCII. Los diacríticos fueron eliminados, por ejemplo, se transformó la palabra “yigüirro” a “yiguirro”. Los emoticones fueron sustituidos por palabras predefinidas “positiveemoticon” o “negativeemoticon”, dependiendo del diccionario en el cual se encuentre el emoticón. Los hashtag y menciones fueron eliminados. En el cuadro 3.4 se puede ver el resumen de todas las funciones aplicadas sobre los componentes.

Cuadro 3.4: Componentes de preprocesamiento utilizado en cada corpus.

Componente	Función de transformación
Diacríticos	Eliminación los diacríticos
Recodificación	Sustitución de caracteres en Html y UTF-8 por su equivalente en ASCII
Emotición	Sustitución por “positiveemoticon” o “negativeemoticon”
Hashtag	Eliminación
Mayúsculas	Sustitución todas las palabras a minúscula
Mención	Eliminación
Numeral	Eliminación
Puntuación	Eliminación
Stopwords	Eliminación
Url	Eliminación
Fechas	Eliminación
Elongaciones	Sustitución de dos letras o silabas repetidas continuamente por una sola vez

Diccionarios

Para identificar los emoticones que fueron sustituidos se utilizaron dos diccionarios, uno llamado “negative.txt” y otro “positive.txt”, los cuales son una compilación de las fuentes mencionadas en [Wolny, 2016]. Como su nombre lo describe, cada uno de ellos se utilizó para identificar emoticones negativos o positivos.

Para la identificación de las palabras que se eliminaron por no tener significado o ser vacías, se utilizó un diccionario de *stopwords* llamado “stopwords-es.txt” extraído de [Diaz, 2016].

Herramientas

Para dividir los comentarios en palabras o *tokens* se utilizó la herramienta “nltk.tokenize” del *framework* NLTK. También, se utilizó BeautifulSoup que es una herramienta para el manejo de archivos de lenguajes de marcado, y al mismo tiem-

po se utilizaron funciones de esta biblioteca para decodificar entidades que estaban codificadas en HTML o XML.

Para el procesamiento de operaciones que requerían manejo de expresiones regulares, se utilizó la biblioteca nativa de Python “re”.

3.1.4. Generación de las representaciones vectoriales con Word2Vec

Las representaciones vectoriales de las palabras se pueden crear con diferentes técnicas. Para efectos de este trabajo, se utilizó el modelo predictivo Word2Vec [Mikolov et al., 2013a]. Esta decisión se debió a que Word2Vec es la técnica más utilizada en los trabajos en español obtenidos durante la revisión de antecedentes.

El objetivo principal de esta investigación se basa en la creación de estas representaciones vectoriales en el contexto de Costa Rica. Los corpus utilizados son: FBCR2013, que contiene comentarios de español de Costa Rica, y el corpus Twitter, que presenta textos con variantes distintas del español incluidas las de Costa Rica.

Para crear los *embeddings* con Word2Vec se necesita ajustar o parametrizar el algoritmo, este tipo de parametrización es llamado hiperparametrización, descrito a continuación.

Hiperparámetros

Los hiperparámetros son variables de ajuste para la creación de los *embeddings*. En el caso de Word2Vec, los más importantes son: el tamaño del vector, el tamaño de la ventana, el tipo de arquitectura, el muestreo negativo y las épocas. En el cuadro 3.5, se puede observar el resumen de las configuraciones utilizadas en este proyecto para la creación de las representaciones vectoriales.

La elección del tamaño de vector, el tipo de arquitectura y las épocas se basaron en las configuraciones realizadas en los modelos de experimentación en [Mikolov et al., 2013a]. El tamaño del muestreo negativo se basó en la experimentación realizada en corpus pequeños y grandes en [Mikolov et al., 2013b] y también en las experimentaciones de [Lison y Kutuzov, 2017].

Para cada corpus se crearon dos versiones de *embeddings* con dos ventanas distintas. Los tamaños de ventana fueron de **dos** y de **cinco**, de esta forma la palabra objetivo

obtuvo un contexto de dos o cinco palabras a la izquierda y la derecha. Esto permitió abarcar la totalidad de la media por comentario de los corpus en estudio.

Cuadro 3.5: Hiperparámetros para el algoritmo Word2Vec.

Hiperparámetro	FBCR2013	Twitter
Tamaño del vector	300	300
Ventana	2, 5	2, 5
Arquitectura	skip-gram	skip-gram
Muestreo negativo	10	10
Epocas	5	5

Herramientas

Para la creación de estos modelos se utilizó el *framework* Gensim *framework* Gensim¹ que es utilizado para el modelado de tópicos relacionados al procesamiento del lenguaje natural y tareas de búsqueda y recuperación de información.

Este proyecto utilizó la clase “Word2Vec” para la creación de los modelos. La clase “Word2Vec” permite persistir los modelos completos para seguir entrenándolos o solo los vectores de las palabras.

3.2. Construcción de los modelos de clasificación

Para cumplir el tercer objetivo específico del presente proyecto relacionado con la construcción de los modelos de clasificación, se toma como referente la figura 3.2, que sigue el flujo descrito en el modelo SAM. Cada una de las cajas rectangulares representa un componente que aplica funciones de transformación a cada comentario del conjunto de entrenamiento.

¹ Gensim es una biblioteca para el procesamiento del lenguaje natural. <https://radimrehurek.com/gensim>

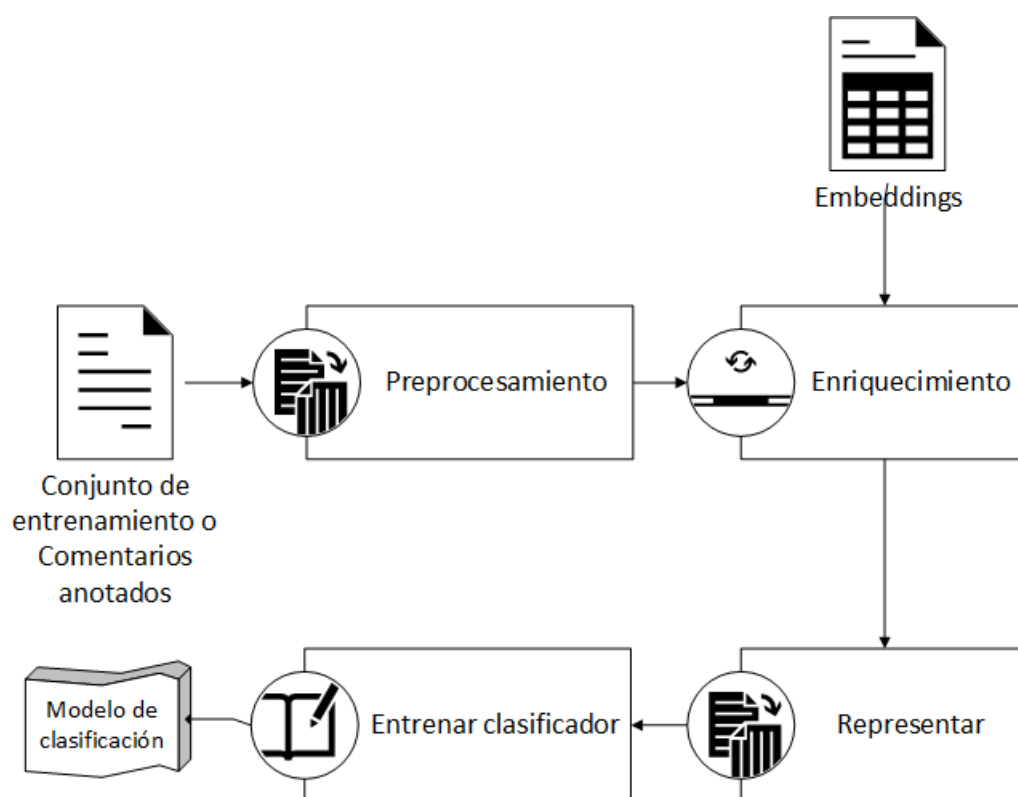


Figura 3.2: Diagrama de descripción de la creación de un modelo de clasificación.

Como lo muestra la figura, el primer paso para construir los modelos de clasificación es obtener un conjunto de entrenamiento que son comentarios anotados, etiquetados o con polaridad definida. Estos comentarios sirven para entrenar el clasificador y generar los modelos de clasificación, pero antes los mismos son preprocesados o normalizados y enriquecidos con los vectores de cada palabra; para seguidamente ser representados para entrenar el clasificador y finalmente obtener el modelo de clasificación.

En las siguientes subsecciones se explica con más detalle cada uno de los pasos o componentes mencionados anteriormente, así como los recursos utilizados, siguiendo el orden lógico de la figura 3.2.

3.2.1. Conjunto de entrenamiento

El conjunto de entrenamiento es un subconjunto del corpus etiquetado Inter-TASS_CR, el cual es descrito en la sección 3.3. Posee 1166 comentarios cada uno eti-

quetado con un nivel de polaridad distribuidos de la siguiente manera: 338 positivos (P), 149 neutrales (NEU), 456 negativos (N) y 223 como ninguno (N). Cada uno de estos comentarios fue normalizado para ser enriquecido y presentado para entrenar el clasificador.

3.2.2. Preprocesamiento

El preprocesamiento de los comentarios del conjunto de entrenamiento fue el mismo que se le aplicó a los corpus para la construcción de *word embeddings*. Este proceso fue descrito en la subsección 3.1.3 llamada “Preprocesamiento”.

3.2.3. Enriquecimiento

De acuerdo al modelo SAM, el enriquecimiento del texto se lleva a cabo cuando se obtiene información que no está presente en forma explícita en el texto del comentario y se agrega a la vista lógica existente [Casasola et al., 2019]. En este caso, con los comentarios para el análisis de sentimiento del corpus InterTASS_CR, se procedió a sustituir cada palabra por un vector o “embedding” (en función de peso), a partir de un recurso externo, siendo este una de las representaciones vectoriales creadas a partir de uno de los corpus.

3.2.4. Representación

Se describe aquí la forma en que se llevaron a cabo cambios en la estructura de representación del comentario, para prepararlo para el proceso de análisis o clasificación. Se aclara que esta descripción se lleva a cabo según se indica en especificación del modelo [Casasola et al., 2019].

En el caso de este proyecto se utilizaron vectores de números reales y se realizaron cambios a los vectores previamente procesados en el enriquecimiento, para ser presentados a los clasificadores. Como anteriormente se mencionó, se utilizó un subconjunto del corpus InterTASS_CR para entrenar el clasificador, y se representó cada comentario previamente enriquecido como una concatenación o promedio de los vectores de cada una de las palabras.

Como parte del proceso de representación, la cantidad de palabras de cada uno de los comentarios con los cuales se predijo y entrenó los clasificadores fue de 20. Los comentarios que contaban con menos palabras, fueron rellenados con palabras hasta completar las 20. En este proyecto estas palabras de relleno o palabras vacías fueron representadas por vectores nulos de dimensión 300 de izquierda a derecha. Los comentarios que excedieron las 20 palabras fueron truncados.

Estas representaciones fueron utilizadas por dos clasificadores. Para el clasificador de la red neuronal convolucional, se realizó una concatenación de todas las representaciones vectoriales de las palabras de cada comentario obteniendo así un único vector de tamaño 6000, esto pues cada comentario tuvo una longitud de 20 palabras y el peso de cada palabra fue un vector de tamaño 300.

La presentación para la máquina de soporte vectorial fue un promedio de todos los vectores de cada palabra de cada comentario de tamaño 20 (con relleno), de esta forma se obtuvo un único vector de tamaño 300.

La siguiente subsección detalla el proceso de clasificación una vez representados los datos de entrada para los mismos.

3.2.5. Clasificación

Para cumplir con el objetivo específico número tres, que es construir los modelos de clasificación, es importante la creación de los clasificadores, que son entrenados con los diferentes *word embeddings*, para generar un modelo de clasificación. Estos modelos son utilizados para predecir o clasificar los comentarios que son utilizados en las evaluaciones.

La clasificación de los comentarios es una de las partes más importantes en este proyecto, pues es por medio de esta que se obtuvieron los resultados para la evaluación de las representaciones vectoriales; la cual se realizó por medio de dos clasificadores: una red neuronal convolucional (CNN) y una máquina de soporte vectorial (SVM).

Cada modelo de clasificación es identificado por una tripleta, que es el corpus con el que se creó la representación vectorial, el tamaño de ventana y el clasificador que generó el modelo de clasificación. Ejemplo de esto es el modelo de clasificación creado con la red neuronal convolucional utilizando la representación vectorial creada con el corpus FBCR2013 y con ventana 2 teniendo la tripleta “fb 2 cnn”

A continuación se especifican los clasificadores que fueron utilizados en el presente proyecto.

Red Neuronal Convolutiva (CNN)

La red utilizada en este proyecto se creó basándose en “la red neuronal convolutiva para clasificación de sentencias” [Kim, 2014]. Para esto se utilizó la arquitectura con vectores estáticos propuesta por [Kim, 2014]

A continuación se describe la configuración de la versión utilizada en esta propuesta, la cual se puede ver en el cuadro 3.6.

Cuadro 3.6: Hiperparametrización de la red neuronal convolutiva.

Hiperparametro	Valor
Capas convolucionales	3
Filtros	100
Tamaño de kernels	2,3,4
Pasos de deslizamiento	1
Activación	ReLU
Neuronas de ANN	4
Función de pérdida	Cross-entropy
Optimizador	Adam
Tamaño del batch	32
Epocas	5

La capa de entrada tuvo una dimensión de 20 x 300, la cual representó las 20 palabras de cada comentario, con una representación vectorial de tamaño 300. De esta forma se tuvo una capa de entrada de 6000 características.

Seguidamente se obtuvieron tres capas convolucionales paralelas de una dimensión cada una contó con 100 filtros, una función de activación de tipo ReLu y deslizamientos de un paso. El tamaño de kernel podría ser visto como un n-grama, para la primera capa este es de 3, para la segunda 4 y la última 5. Cada una de estas capas recibió la capa de entrada en paralelo.

Se tuvieron tres capas de reducción conectadas recibiendo los datos de salida de las capas convolucionales. El tamaño del *pool* de las capas fue de 18, 17 y 16 (*max pool size*) respectivamente.

A cada una de las salidas de las capas de reducción se le aplicó una función de concatenación en paralelo, creando así un vector de tamaño 300.

Por último, se tuvo la capa de clasificación, compuesta por cuatro unidades neuronales y *softmax* como la función de activación.

Para evitar sobre entrenamiento se aplicó una técnica conocida como *dropout* con un valor de un 50%.

El optimizador del modelo fue ADAM y la función de error fue *categorical cross entropy*. Se aplicaron 5 épocas de entrenamiento con lotes de tamaño 32.

La salida de esta red neuronal fue un vector disperso de longitud 4, por ejemplo [0,0,1,0]. El índice donde se encuentra el uno indica la clase predicha por la red.

Máquina de soporte vectorial

La máquina de soporte vectorial que se utilizó en este proyecto recibió como entrada un vector de tamaño 300 que representa el comentario.

La configuración de la máquina de soporte vectorial se puede visualizar en cuadro 3.7. Se utilizó un kernel del tipo *Radial Basis Function* (RBF) basado en [Jadav y Vaghela, 2016], el parámetro gamma en 0.178 y el parámetro C en 8.

Cuadro 3.7: Hiperparametrización de la máquina de soporte vectorial.

Hiperparametro	Valor
Kernel	RBF
Gamma	0.178
C	8
Función de decisión	Uno contra todos

Herramientas

La creación de los clasificadores de texto se llevó a cabo con una combinación de herramientas descritas a continuación.

Para la creación de la red neuronal convolucional se utilizó Keras² con Tensorflow³ como *backend*. Para la implementación de la máquina de soporte vectorial utilizó la clase SVC de la biblioteca scikit-learn⁴. Por su parte, las representaciones vectoriales utilizaron estructuras de datos del *framework* NumPy⁵ para la vectorización.

3.3. Corpus InterTASS_CR

El corpus InterTASS_CR representa un subconjunto del corpus de español costarricense utilizado en la competencia de TASS 2019 [Diaz-Galiano et al., 2018]. Es un conjunto de comentarios extraídos de la red social de Twitter y fue creado en el 2018. Cada uno de sus comentarios o *tweets* fue etiquetado con un nivel de polaridad el cual puede ser positivo (P), neutral (NEU), negativo (N) y sin polaridad (NONE). Cada tweet fue etiquetado por tres anotadores y el consenso fue de 2048 comentarios. Para el resto de tweets fueron necesarios dos anotadores más para llegar a un acuerdo. Este corpus etiquetado fue utilizado para entrenar los clasificadores y realizar la evaluación de las representaciones vectoriales generadas con los corpus de Twitter y FBCR2013.

El corpus cuenta con 2333 comentarios con un tamaño promedio de 14 palabras por comentario y una desviación estándar de 5.6, como se puede ver en la cuadro 3.8. El comentario más pequeño cuenta con 4 palabras y el más grande con 33.

² Keras es un API de alto nivel para uso de redes neuronales. <https://keras.io>.

³ Tensorflow es una biblioteca para el aprendizaje automático. <https://tensorflow.org>.

⁴ scikit-learn es una herramienta para el análisis de datos . <https://scikit-learn.org>.

⁵ NumPy es usado como un contenedor multi dimensional eficiente. <https://numpy.org>.

Cuadro 3.8: Estadísticas de la longitud de los comentarios para el corpus InterTASS_CR.

Estadística	Longitud
Cantidad	2333
\bar{x}	14.04
σ	5.69
P_0	4
P_{25}	9
P_{50}	13
P_{75}	18
P_{100}	33

Es importante mencionar que para efectos del proyecto se decidió dividir el corpus en dos subconjuntos complementarios de manera balanceada o con la misma distribución probabilística, de acuerdo a las categorías (P, N, NEU, NONE). Para ello se creó el corpus de **entrenamiento** y el corpus de **pruebas**, asignando un 50% (basado en [Diaz-Galiano et al., 2018]) de los comentarios a cada uno, como se puede ver en el cuadro 3.9. Nótese que el porcentaje proporcional de cada categoría en los subconjuntos es el mismo que el del conjunto original.

La función de cada uno de estos subconjuntos es de suma importancia, pues con el subconjunto de entrenamiento, un clasificador, y un *embedding*, se puede crear un modelo de clasificación utilizado para predecir. Asimismo, Teniendo este modelo de clasificación, se puede proceder a evaluar su desempeño utilizando el subconjunto de pruebas.

Cuadro 3.9: Distribución de las categorías para los subconjuntos de entrenamiento y pruebas del corpus InterTASS_CR.

Categoría	Entrenamiento	Pruebas	Total
N	456	456	912
NEU	149	148	297
NONE	223	224	447
P	338	339	677
Total	1166	1167	2333

3.4. Evaluación

La evaluación se llevó a cabo con la finalidad de medir el efecto de las representaciones vectoriales de las palabras sobre la clasificación de polaridad de los comentarios costarricenses, y con ella, se abordó el objetivo específico número tres, que es comparar los resultados de los modelos de clasificación.

Para lograr una mejor comprensión del proceso de evaluación se creó la figura 3.3, en la cual se utiliza el conjunto de pruebas de InterTASS_CR que es utilizado por un modelo de clasificación para predecir la polaridad de los comentarios que se encuentran en dicho conjunto.

Como se muestra en la figura, cuando se tienen todos los resultados se procede a evaluar las predicciones comparándolas con la polaridad real de cada comentario del conjunto de pruebas. Para obtener las métricas de evaluación de las polaridades o categorías se crea una matriz de confusión con la que se calcula la exactitud, precisión, cobertura y puntuación F1 para cada categoría, para finalmente calcular la macro puntuación F1. Se debe tener claro que cada evaluación es llevada a cabo para cada modelo de clasificación.

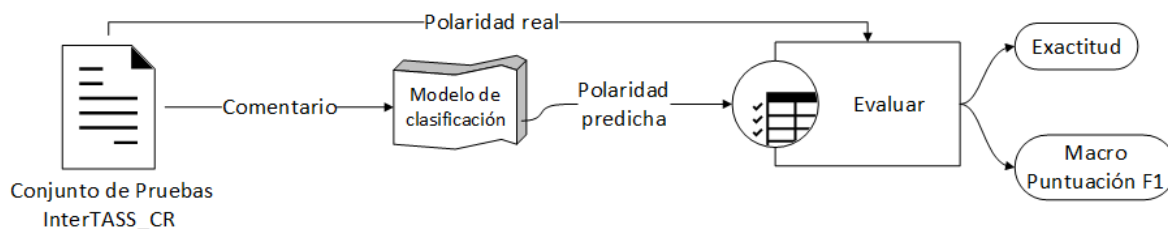


Figura 3.3: Diagrama de descripción de la evaluación.

También como dato importante en la evaluación se generaron modelos de clasificación de línea base para comparar con los modelos creados utilizando los *word embeddings* generados en este proyecto. Estos modelos de clasificación de línea base fueron creados utilizando un *word embedding* obtenido de una fuente pública, en el caso de este proyecto se utilizó el *word embedding* SBW descrito con detalle en la subsección 3.4.2.

Una vez obtenidas las dos métricas (exactitud y macro puntuación F1) para cada modelo de clasificación, se procedió a comparar los resultados de cada modelo que utilizó las representaciones costarricenses, con las representaciones de la línea base SBW, separando estas comparaciones por clasificador. Así se obtuvieron los resultados de las comparaciones y se realizaron observaciones sobre los mismos.

3.4.1. Conjunto de pruebas

El conjunto de pruebas es un subconjunto del corpus etiquetado InterTASS_CR, el cual es descrito en la sección 3.3. Posee 1167 comentarios cada uno etiquetado con un nivel de polaridad distribuidos de la siguiente manera: 339 positivos (P), 148 neutrales (NEU), 456 negativos (N) y 224 como ninguno (N). Cada uno de estos comentarios fue normalizado para ser enriquecido y presentado para probar cada modelo de clasificación.

3.4.2. Word embeddings SBW

SBW (*Spanish Billion Word Corpus*) es una representación vectorial de palabras creada a partir de varios corpus no anotados, que son compilados de diferentes recursos de la web. Fueron creados utilizando Word2Vec y la biblioteca Gensim en Python [Cardellino, 2016].

Los datos fueron preprocesados para obtener las porciones en español, se reemplazaron todos los caracteres no alfanuméricos con espacios en blanco, los números se sustituyeron con la palabra "DIGITO", y los espacios múltiples continuos se les dejó un espacio

Esta representación fue seleccionada pues dentro de lo investigado era la que tenía la mayor cantidad referencias web para representaciones en español, además las herramientas utilizadas para su creación y la dimensionalidad de los vectores son las mismas con la que se crearon las representaciones en este proyecto.

Cuadro 3.10: Descripción del corpora utilizado en SBW.

Elemento	Original	Preprocesado
# Palabras	1420,665,810	771,508,817
# Sentencias	46,925,295	-
# Palabras únicas	3,817,833	1,000,653

La cantidad de vectores de palabras creados fueron más de 1,000,000, como se puede observar en el cuadro 3.10. Cabe destacar que se realizó un preprocesamiento de los datos antes de crear las representaciones vectoriales, es por esto que el cuadro 3.10 posee dos columnas, una con la cantidad de datos originales y una después de preprocesar. Se puede observar como la cantidad de palabras disminuye después de descartar palabras poco frecuentes o muy frecuentes.

Cuadro 3.11: Descripción de hiperparámetros utilizados en Word2Vec para SBW.

Hiperparámetro	Valor
Dimensión	300
Ventana	5
Frecuencia mínima	5
Eliminar los primeros más comunes	273
Muestreo negativo	20

Como se puede observar en el cuadro 3.11 la dimensionalidad de estos vectores es de 300, la cual es compatible con los vectores creados en este proyecto, además se

puede observar la frecuencia mínima de las palabras que fueron eliminadas y el tamaño de ventana 5, los valores de los demás hiperparámetros se dejan con la configuración por defecto de la herramienta Gensim.

3.5. Software utilizado para el desarrollo y experimentación

Para el desarrollo de este proyecto se utilizó un software que permite crear escenarios de experimentación en Python. Este cuenta con tres aplicaciones independientes que se encargan de crear los insumos necesarios para realizar el análisis de sentimiento. Las tareas llevadas a cabo por cada una de estas aplicaciones fueron el preprocesamiento, la creación de representaciones vectoriales y la clasificación.

La aplicación de preprocesamiento recibió como insumo los corpus descritos anteriormente y se encargó de normalizarlos aplicando las funciones descritas en la etapa de preprocesamiento. Una vez llevada a cabo esta tarea, se persistieron los datos normalizados.

La aplicación de representaciones vectoriales, se encargó de crear y persistir los *word embeddings* necesarios para la experimentación con corpus preprocesados.

Por su parte, la clasificación fue llevada a cabo por una aplicación independiente que se puede definir como un laboratorio para realizar experimentos con los *word embeddings* creados en la aplicación de representaciones vectoriales. Esta aplicación se encargó de enriquecer y representar los comentarios para el análisis de sentimiento.

3.6. Hardware utilizado para la experimentación

Otro elemento importante de la metodología a mencionar es el laboratorio computacional en el cual se realizaron los experimentos. El cuadro 3.12 muestra las características técnicas del hardware en donde se programaron y se ejecutaron los experimentos.

Las características físicas mostradas en el cuadro 3.12 fueron suficientes en la ejecución de todas las tareas programadas para este proyecto.

Cuadro 3.12: Especificaciones del hardware .

Componente	Descripción
Sistema Operativo	Linux Ubuntu 18 LTS
RAM	15.8 GB
SSD	100 GB
Cores	4 (8 virtuales)
Velocidad base	1.99 Ghz

Este capítulo describió la metodología utilizada en el proyecto. La concreción o instanciación de la misma brindó los resultados que serán descritos y analizados en el siguiente capítulo.

Capítulo 4

Resultados

Este capítulo describe los resultados obtenidos en el proyecto, los mismos fueron las representaciones vectoriales de las palabras, los modelos de clasificación, las evaluaciones, y la aplicación con la que se realizó este proyecto.

4.1. Representaciones vectoriales de las palabras

Uno de los resultados más importantes en este proyecto fue el desarrollo de las representaciones vectoriales de las palabras generadas a partir de textos de Costa Rica, las cuales representaron un insumo importante para el sistema de análisis de sentimiento en el presente trabajo. En total fueron creados cuatro conjuntos *embeddings* como se muestra en la figura 4.1, a partir de dos corpus y con dos variantes de ventana de tamaño 2 y 5.

La figura 4.1 muestra el proceso y resultados de construcción de los *embeddings* de tal forma que se tuvieron dos corpus (descritos anteriormente), estos fueron el corpus “FBCR2013” una variante costarricense y “Twitter” una variante mixta. A cada uno de los corpus se les aplicó un preprocesamiento para normalizar los datos, y seguidamente a estos corpus normalizados se les aplicó el algoritmo Word2Vec para crear las representaciones vectoriales. Word2Vec creó una representación vectorial por cada variante de ventana, en este caso de tamaño 2 y 5.

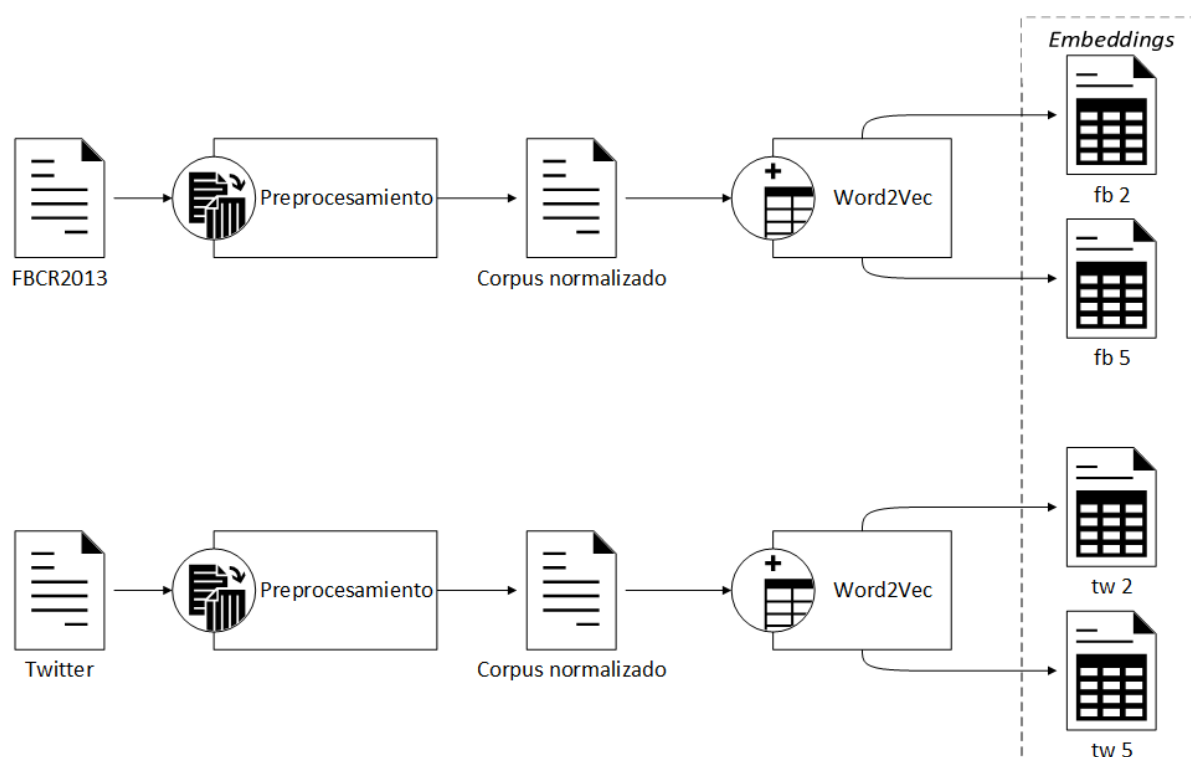


Figura 4.1: Resultados de representaciones vectoriales de palabras.

Los nombres de los *embeddings* presentes en la figura representan una dupla del tipo “corpus - tamaño de ventana”. En este caso la abreviatura “fb” representa el corpus “FBCR2018” y “tw” al corpus “Twitter”, el segundo integrante de la tupla representa el tamaño de ventana.

De esta forma se completó el objetivo específico número uno, que se refiere al método de construcción de las representaciones, y el objetivo número dos, que establece la construcción de variantes de representaciones vectoriales de palabras con diferentes ventanas de contexto.

Como datos adicionales, en las siguientes subsecciones se especifica el tamaño del vocabulario y la duración de creación de los *embeddings*.

4.1.1. Tamaño del vocabulario de los embeddings

Es importante describir la cantidad de palabras únicas como resultado de la creación de las representaciones vectoriales de las palabras creadas en este proyecto, esto

pues cada palabra de los comentarios a clasificar será sustituida por el vector de la palabra presente en los *embeddings*. Cuantas más palabras del comentario existan en el vocabulario de los *embeddings*, mejores serán los resultados.

La figura 4.2 muestra el tamaño del vocabulario para los *embeddings* creados a partir de los corpus de “FBCR2018” (azul) y “Twitter” (verde).

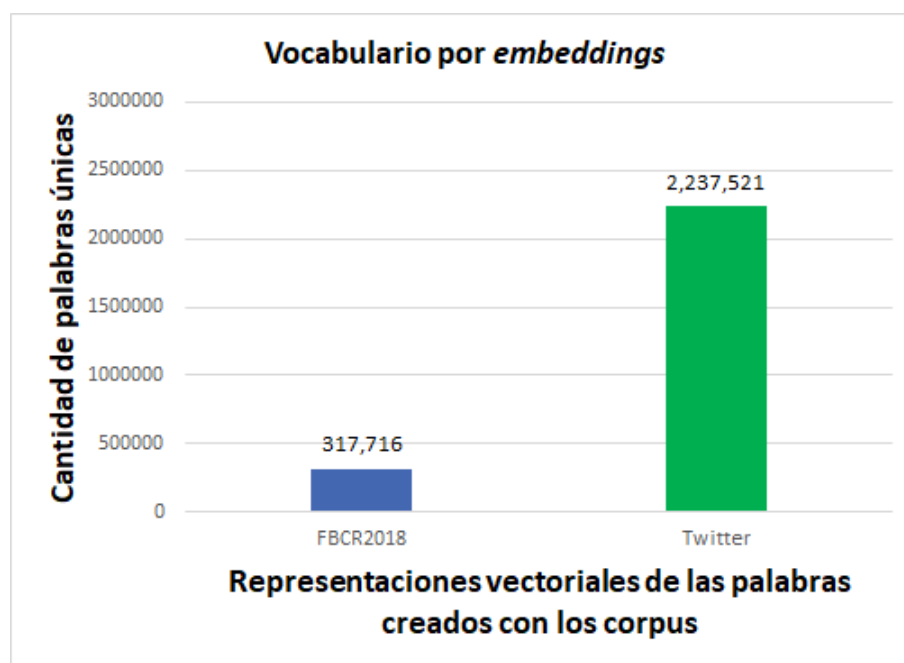


Figura 4.2: Vocabulario para los embeddings creados a partir de los diferentes corpus.

Se puede apreciar como el tamaño de vocabulario de los *embeddings* creados con “Twitter” es casi 7 veces el tamaño del los que fueron creados con el corpus “FBCR2018”.

4.1.2. Duración de la creación de las representaciones vectoriales

El cuadro 4.1 muestra los *word embeddings* generados y la duración de procesamiento para cada uno de los corpus. Cabe resaltar que la hiperparametrización para Word2Vec con cada uno de estos *word embeddings* fue la misma, con excepción del tamaño de la ventana.

Cuadro 4.1: Duración de la creación de las representaciones vectoriales de las palabras por tamaño de ventana.

Ventana	FBCR2013	Twitter
02	00:04:39	00:46:58
05	00:08:01	00:53:58

La duración para crear la totalidad de estos *word embeddings* con Word2Vec fue de aproximadamente dos horas, pero previo a esto se realizaron muchas otras pruebas y optimizaciones.

4.2. Modelos de clasificación

Cada modelo de clasificación de este proyecto fue creado a partir de un clasificador (CNN o SVM), el conjunto de entrenamiento de InterTASS_CR y un *embedding* en alguna de sus variantes. Los cuales fueron los encargados de predecir la polaridad de los comentarios en el sistema de análisis de sentimiento.

La figura 4.3 muestra los modelos creados por los clasificadores en este proyecto. Se crearon cinco modelos por clasificador al entrenar cada uno con los *word embeddings* presentes en el recuadro ubicado a la izquierda titulado “Embeddings”. Nótese que se agregó el *word embedding* predefinido “sbw 5” en la figura, ya que se utilizó como línea base para la comparación de los resultados de evaluación.

Como resultado, para cada clasificador a partir del corpus de “FBCR2013” se obtuvieron dos modelos creados con ventana 2 y 5, y del corpus “Twitter” se crearon otros dos modelos con ventana 2 y 5. Al mismo tiempo se tiene el modelo creado a partir del *word embedding* línea base SBW con ventana 5.

En el encuadre “Modelos de clasificación” se tienen los modelos creados, cada uno de ellos representado por una triplete de la forma “corpus - tamaño de ventana - clasificador”.

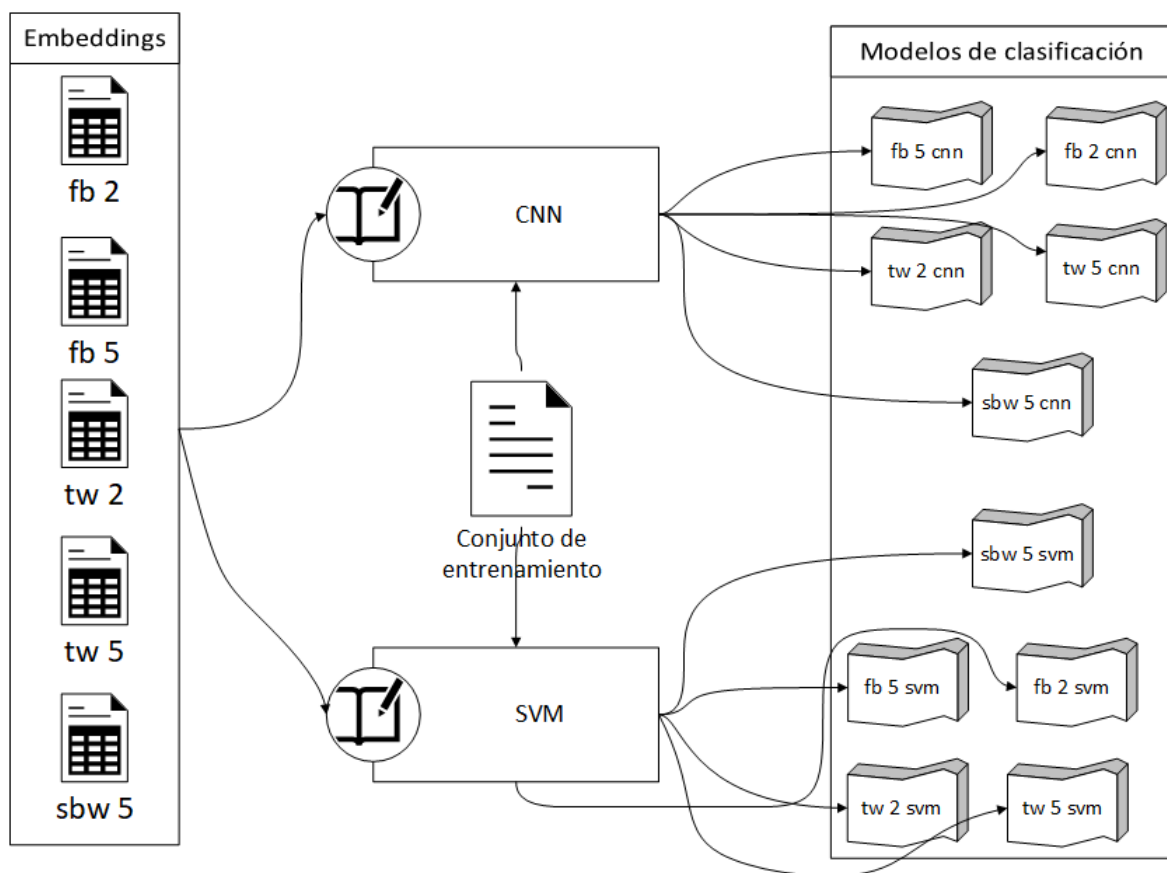


Figura 4.3: Modelos de clasificación como resultado del entrenamiento en los clasificadores con diferentes embeddings y el conjunto de entrenamiento de InterTASS_CR.

De esta forma, se muestra como se llegó a cumplir con el objetivo específico número tres. Cuyo propósito fue la construcción de los modelos de clasificación a partir de dos clasificadores de comentarios utilizando las representaciones vectoriales creadas.

En la siguiente sección se muestran los resultados de la evaluación del uso de los *word embeddings* en los modelos de clasificación para el análisis de sentimiento.

4.3. Evaluación de los embeddings utilizados en los modelos de clasificación

En esta sección se muestran las métricas de exactitud y macro puntuación F1 al evaluar cada modelo de clasificación con el conjunto de pruebas del corpus anotado InterTASS_CR, que como se menciona en la sección 3.3, cuenta con 1167 comentarios etiquetados con su respectiva polaridad. También se realizan observaciones sobre los resultados de dichas métricas, teniendo como línea base los modelos creados con el *embeddings* SBW.

Esta sección completa el objetivo general al evaluar el efecto de los *word embeddings* sobre la clasificación de la polaridad de comentarios en español costarricense y el objetivo específico número cuatro al comparar los resultados de los modelos de clasificación.

Las observaciones se realizan en cuatro subsecciones. Las primeras dos están enfocadas en los clasificadores, mientras que las últimas dos están enfocadas a las métricas: exactitud y macro puntuación F1.

Se debe aclarar que en todos los gráficos mostrados en las siguientes subsecciones, las barras de color gris representan los modelos de clasificación que utilizan el *word embedding* SBW, las barras de color verde, los que fueron creados con los *word embeddings* de “Twitter”, mientras que las barras azules, representan los que fueron creados con los de “FBCR2013”.

4.3.1. Evaluaciones de los modelos de CNN

Los gráficos de la figura 4.4 muestran las métricas de exactitud y macro puntuación F1, al evaluar todos los modelos de clasificación creados con la red neuronal convolucional.

En la exactitud se puede observar como todos los modelos de clasificación creados con los *word embeddings* de Twitter tienen una diferencia positiva de un 0.03 con respecto a la línea base y en el caso de los de FBCR2013, no hay diferencia con el *embedding* con ventana de tamaño 2. Todos los modelos de clasificación parecen tener una diferencia positiva en la macro puntuación F1, entre un 0.03 y un 0.05, con respecto a la línea base.

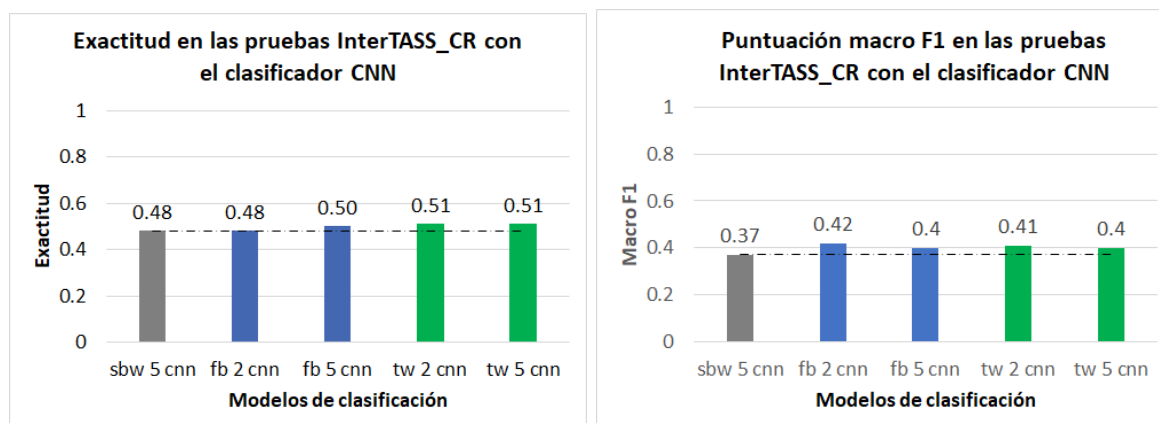


Figura 4.4: Exactitud y macro puntuación F1 para los modelos de clasificación contruidos a partir de la CNN.

4.3.2. Evaluaciones de los modelos SVM

Los gráficos de la figura 4.5 muestran las métricas de exactitud y macro puntuación F1 al evaluar todos los modelos de clasificación creados con la máquina de soporte vectorial.

En la exactitud se puede observar como todos los modelos creados con los *word embeddings* de Twitter tienen una diferencia positiva entre un 0.04 a un 0.05 con respecto a la línea base, y con los de FBCR2013 no hay ninguna diferencia con el *word embedding* con ventana de tamaño 5.

Para la macro puntuación F1 todos los modelos tienen una diferencia positiva entre un 0.01 y un 0.07 con respecto a la línea base.

A diferencia de los modelos creados con la CNN, los que fueron creados con SVM muestran valores más altos en ambas métricas.

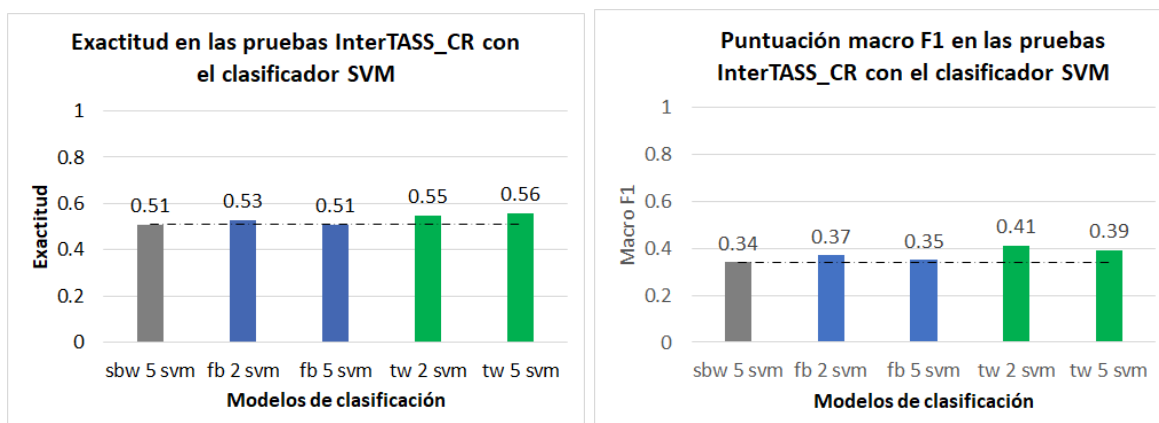


Figura 4.5: Exactitud y macro puntuación F1 para los modelos de clasificación construidos a partir de la SVM.

4.3.3. Observaciones con respecto a la Exactitud

Todas las variantes de *word embeddings* creadas con el corpus de Twitter muestran mayores valores para la exactitud con ambos clasificadores con respecto a la línea base y a los *word embeddings* de FBCR2013. Como muestran los gráficos de la figura 4.6 con la CNN la exactitud es superior en un 0.03, y con la SVM entre un 0.04 y un 0.05.

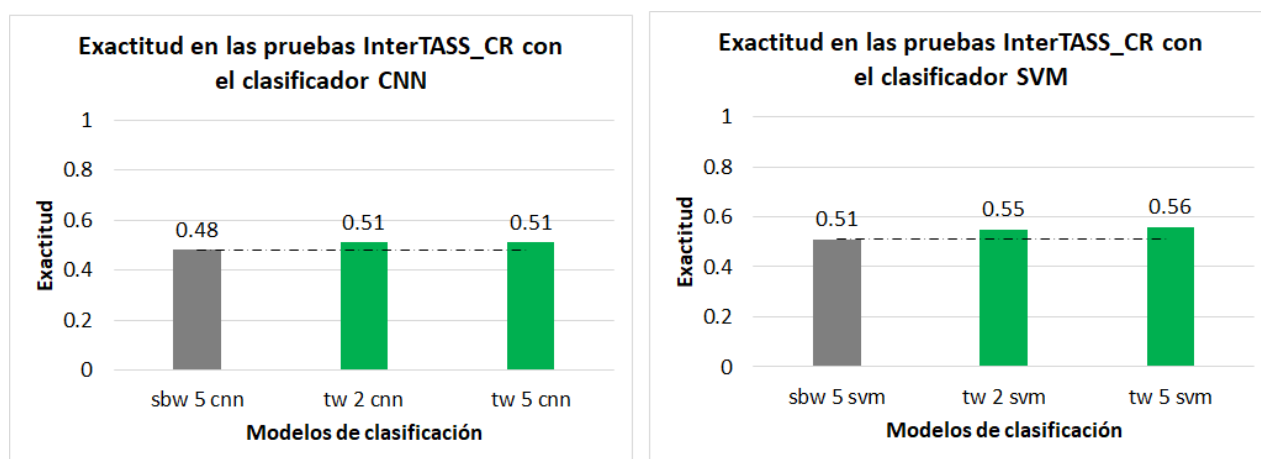


Figura 4.6: Exactitud para los modelos de clasificación construidos a partir del corpus de Twitter.

4.3.4. Observaciones con respecto a la macro puntuación F1

Todas las representaciones vectoriales de las palabras creadas con el corpus de Twitter o FBCR2013 con tamaño de ventana **2** mostraron valores superiores en la evaluación de la métrica F1.

Los gráficos de la figura 4.7 muestran los resultados de la métrica F1, y se observa que el valor f1 usando CNN está entre en un 0.04 a un 0.05 por encima de la línea base, al igual que los de la SVM que muestran valores superiores entre un 0.03 y un 0.07.

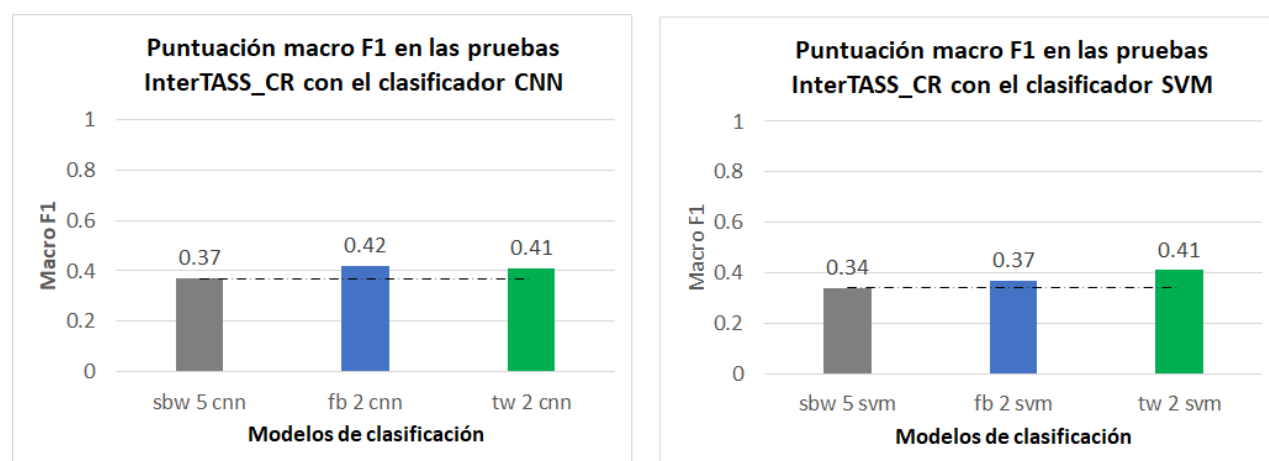


Figura 4.7: Macro puntuación F1 para los modelos de clasificación construidos a partir de la embeddings con ventana 2.

4.3.5. Observaciones generales

Como se mostró anteriormente los resultados en las métricas de exactitud y macro puntuación F1 de los modelos de clasificación creados con los *word embeddings*, que a su vez fueron creados con los corpus que contienen textos en español de Costa Rica son mayores al analizar el sentimiento de comentarios de español de Costa Rica.

Esta diferencia positiva con respecto a la línea base se pueden deber a la variante del español utilizada para la construcción de los *word embeddings*, en este caso es español de Costa Rica, también al tamaño del vocabulario y la estructura de los corpus para crear los *embeddings*.

Como se muestra en el gráfico 4.8, los *word embeddings* de Twitter (con variante del español mixta) presentan la mayor cantidad de palabras únicas superando casi por el

doble al vocabulario de SBW, el mismo tiene mayor valor en las métricas en el análisis de sentimiento con respecto a la línea base, mientras que el *embedding* de FBCR2013 iguala o supera en todas las métricas a la línea base, no cuenta con tanto vocabulario, pero para su creación solo se utilizaron textos de la variante del español de Costa Rica.

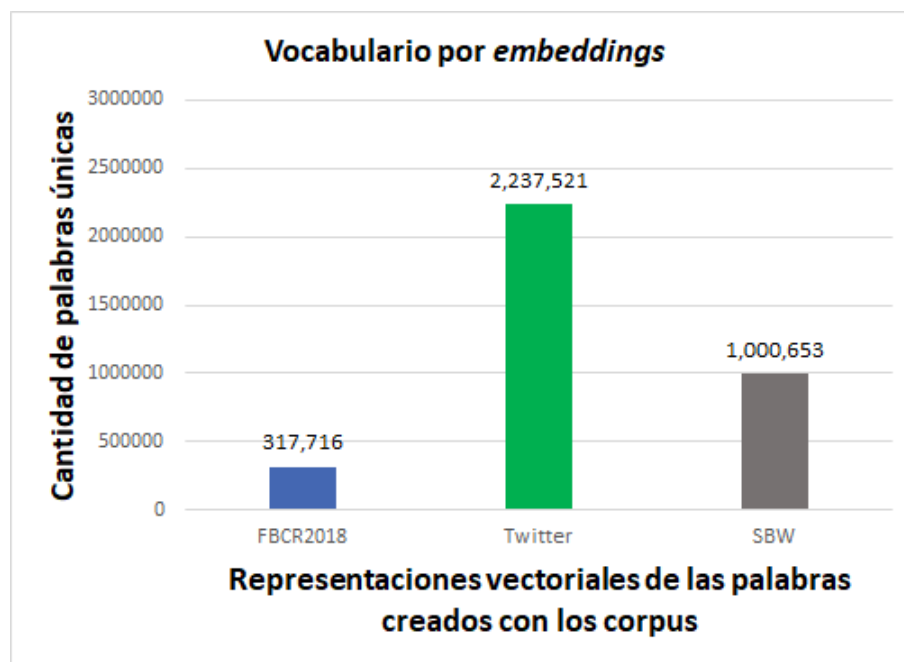


Figura 4.8: Vocabulario para los embeddings creados a partir de los diferentes corpus.

Si bien la mayoría de las diferencias entre los resultados de las evaluaciones de la línea base y los demás modelos es poca, por ejemplo entre un 0.01 y un 0.07, éstas pueden ser cruciales para obtener mejores resultados al competir en alguna de las tareas de los talleres de TASS, en donde los equipos pueden ganar por diferencias de un 0.01 [Diaz-Galiano et al., 2018].

4.4. Herramienta Usure como resultado del proceso de desarrollo del proyecto

Como producto o resultado colateral producido por este trabajo se encuentra la herramienta que fue creada a la que se denominó Usure. A la hora de crear este proyecto

se logró construir una herramienta de automatización para la creación de representaciones vectoriales. La herramienta no solo permite la creación de las representaciones si no que también permite la experimentación y evaluación de ellas en diferentes tipos de clasificadores.

Esta herramienta brinda una facilidad para el desarrollo de la experimentación, permitiendo ahorrar tiempo y recursos. Se espera que pueda servir y agregar valor en trabajos futuros de estudiantes o cualquier otra persona interesada en el tema.

La aplicación para el desarrollo del presente proyecto resultó contar con 1,882 líneas de código y fue codificada en 2 meses y dos semanas (al 21 de Agosto del 2019), utilizando el entorno de desarrollo Visual Code.

El nombre clave del proyecto o herramienta es “Usure” (Úsure) y como dato curioso, representa la casa cósmica de los aborígenes Bribri de Costa Rica.

Se encuentra versionada en un repositorio de dominio público, en la siguiente dirección: <https://github.com/coraxcr/usure>.

4.4.1. Descripción de la aplicación Usure

La aplicación Usure se encuentra dividida en tres paquetes (Python) principales, pero la conceptualización lógica que se le da a estos paquetes para efectos del proyecto es de aplicación. Las tres aplicaciones son *preprocessing*, *wordvectors* y *classification*, pueden ser observadas en la figura 4.9. Estas aplicaciones cuentan con responsabilidades específicas que se mencionan a continuación.

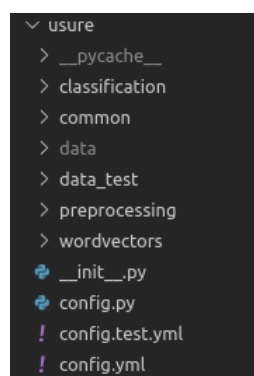


Figura 4.9: Paquetes principales de la aplicación Usure.

La aplicación de preprocesamiento (*preprocessing*) se encarga de transformar los cor-

pus existentes. Esta aplica funciones de transformación a cada uno de los comentarios de un corpus. Además, almacena en forma persistente el resultado de estas transformaciones en archivos con extensión “.usu”.

La aplicación *wordvectors*, recibe como insumo los archivos “.usu”, los cuales representan el corpora preprocesado. Se encarga de crear las representaciones vectoriales de las palabras y persistirlas, por medio del *framework* Gensim¹, que a su vez es el encargado de generar estas representaciones. Cabe destacar que la arquitectura permite desacoplar el *framework* de vectorización de forma fácil. En este caso se utilizó Gensim pero este puede ser sustituido por otro, como por ejemplo fastText.

La aplicación *classification*, es la encargada de generar y evaluar los modelos, teniendo como insumo cualquier corpus (en este proyecto el corpus InterTASS_CR) y las representaciones vectoriales generadas por la aplicación *wordvectors*.

En la figura 4.10, podemos ver como está organizado el paquete *classification*, que sirve como punto de referencia para describir la estructura interna de las tres aplicaciones. Los principales paquetes que se pueden encontrar en estas aplicaciones son: el *core*, donde reside la lógica del negocio, *infrastructure*, que representa los accesos a recursos externos o ajenos a la lógica del negocio, como por ejemplo la persistencia; y por último se tiene *ui* que representa el paquete de interfaz de usuario. Utilizándose aquí Jupyter, una aplicación HTML para visualización y ejecución de código.

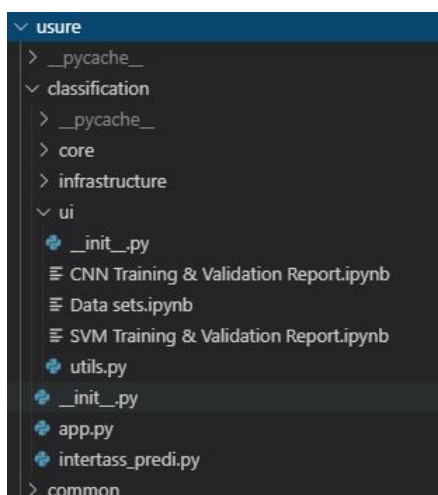


Figura 4.10: Componentes de un paquete principal de la aplicación Usure.

¹ Gensim es una biblioteca para el procesamiento del lenguaje natural. <https://radimrehurek.com/gensim>

Los resultados presentes en este capítulo demuestran la importancia del corpus utilizado en la creación de las representaciones vectoriales y su impacto en el análisis de sentimiento para comentarios de español de Costa Rica. El siguiente capítulo concluye el desarrollo del presente proyecto y se muestra una visión del trabajo futuro.

Capítulo 5

Conclusiones y trabajo futuro

Al finalizar el presente trabajo final de investigación aplicada, se exponen una serie de conclusiones y recomendaciones para trabajos futuros, que se citan a continuación en dos secciones.

5.1. Conclusiones

En el presente proyecto se seleccionó un método de construcción de representaciones vectoriales utilizando la herramienta “Usure”, con los corpus FBCR2013 y Twitter. Con esta herramienta y corpus se construyeron cuatro variantes de *word embeddings* con ventanas de tamaño 2 y 5.

Se crearon diez modelos de clasificación a partir de los *word embeddings* creados y el preexistente que se utilizó como línea base SBW. Con esos embeddings se generaron modelos con ambos clasificadores (CNN y SVM) utilizando el conjunto de entrenamiento de comentarios costarricenses InterTASS_CR.

Con los modelos de clasificación creados se evaluaron los resultados (métricas Macro F1 y Exactitud) obtenidos al clasificar los comentarios del conjunto de pruebas del corpus InterTASS_CR. En las evaluaciones del experimento se observó que la variante del español del corpus para generar los *word embeddings* parece ser importante, pues como se muestra en la sección 4.3, los modelos de clasificación construidos con las representaciones vectoriales de Costa Rica muestran resultados superiores o iguales en todas las métricas a los modelos que fueron entrenados con el *word embedding* base (SBW) y en ningún caso fueron inferiores.

Asimismo, se observó que el balance entre la variante del español, la estructura de los textos y el tamaño del vocabulario de los *word embeddings* parece tener un impacto en los resultados al evaluar la clasificación, pues los que fueron creados con el corpus “Twitter” (con variante del español mixta y que supera por vocabulario al SBW) tienen

mayor exactitud. Además el tamaño de ventana que se observa permite lograr mayor macro puntuación F1 fue de tamaño 2.

5.2. Trabajo futuro

Las representaciones vectoriales creadas en este proyecto fueron utilizadas para crear modelos de clasificación en una red neuronal convolucional y una máquina de soporte vectorial. Para un trabajo futuro se podría considerar su uso en otro tipo de clasificadores.

También para un trabajo futuro, sería importante considerar la estructura de los corpus para generar los *word embeddings* de la línea base. Esto es, usar como línea base *embeddings* que hayan sido creados a partir corpus de comentarios de redes sociales (a la fecha no había ninguno de forma pública).

En este trabajo se utilizaron las representaciones vectoriales de las palabras en sistemas de análisis de sentimiento para español de Costa Rica, sin embargo las mismas podrían ser utilizadas en otras tareas del procesamiento del lenguaje natural como sistemas de extracción de información, resumen de documentos, entre otros.

Bibliografía

- [Altin et al., 2019] Altin, L. S. M., Bravo, A., y Saggion, H. (2019). Lastus/taln at tass 2019: Sentiment analysis for spanish language variants with neural networks. Proceedings of TASS.
- [Baeza-Yates y Ribeiro-Neto, 2008] Baeza-Yates, R. y Ribeiro-Neto, B. (2008). Modern Information Retrieval: The Concepts and Technology Behind Search. Addison-Wesley Publishing Company, USA, 2nd edition.
- [Baroni et al., 2014] Baroni, M., Dinu, G., y Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. En Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 238–247, Baltimore, Maryland. Association for Computational Linguistics.
- [Bojanowski et al., 2016] Bojanowski, P., Grave, E., Joulin, A., y Mikolov, T. (2016). Enriching word vectors with subword information. CoRR, abs/1607.04606.
- [Brooke et al., 2009] Brooke, J., Tofiloski, M., y Taboada, M. (2009). Cross-linguistic sentiment analysis: From english to spanish. En Proceedings of the 7th International Conference on Recent Advances in Natural Language Processing, Borovets, Bulgaria, pp. 50–54.
- [Cardellino, 2016] Cardellino, C. (2016). Spanish Billion Words Corpus and Embeddings.
- [Casasola et al., 2019] Casasola, E., Pimentel, A., Sierra, G., Cámara, E. M., y Marín, G. (2019). Análisis comparativo de las características computacionales en los sistemas modernos de análisis de sentimiento para el español. Procesamiento del Lenguaje Natural, 62:69–76.
- [Casasola Murillo, 2018] Casasola Murillo, E. (2018). Desarrollo de un modelo computacional para la especificación de sistemas de análisis de sentimiento con comentarios de redes sociales en español. PhD thesis, Escuela de Ciencias de la Computación e Informática.
- [Casasola Murillo y Leoni de León, 2016] Casasola Murillo, E. y Leoni de León, J. A. (2016). Transferencia de la polaridad semántica de frases idiomáticas a comentarios de opinión. Káñina, 40(3):65–76.
- [Casasola Murillo y Marín Raventós, 2016] Casasola Murillo, E. y Marín Raventós, G. (2016). Evaluación de modelos de representación del texto con vectores de dimensión reducida para análisis de sentimiento. En TASS@ SEPLN, pp. 23–28.
- [Devlin et al., 2018] Devlin, J., Chang, M., Lee, K., y Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.
- [Diaz, 2016] Diaz, G. (2016). Stopwords spanish (es). [Online; accessed 1-June-2019].
- [Diaz-Galiano et al., 2018] Diaz-Galiano, M., García-Vega, M., Casasola, E., Chiruzzo, L., Garcia-Cumbrera, M., Martínez Cámara, E., Moctezuma, D., Montejó Ráez, A., Sobrevilla Cabezedo, M. A., Tellez, E., Graff, M., y Miranda, S. (2018). Overview of tass 2019: One more further for the global spanish sentiment analysis corpus. Lang. Resour. Eval., 2421(2):645–672.
- [Firth, 1957] Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. 1952-59:1–32.
- [Garain y Mahata, 2019] Garain, A. y Mahata, S. K. (2019). Sentiment analysis at sepln (tass)-2019: Sentiment analysis at tweet level using deep learning. arXiv preprint arXiv:1908.00321.
- [Godino y D'Haro, 2019] Godino, I. G. y D'Haro, L. F. (2019). Gth-upm at tass 2019: Sentiment analysis of tweets for spanish variants. Proceedings of TASS.

- [Goldberg, 2016] Goldberg, Y. (2016). A primer on neural network models for natural language processing. J. Artif. Int. Res., 57(1):345–420.
- [González et al., 2019] González, J.-Á., Hurtado, L.-F., y Pla, F. (2019). Elirf-upv at tass 2019: Transformer encoders for twitter sentiment analysis in spanish.
- [Grave et al., 2018] Grave, E., Bojanowski, P., Gupta, P., Joulin, A., y Mikolov, T. (2018). Learning word vectors for 157 languages. En Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018), Miyazaki, Japan. European Languages Resources Association (ELRA).
- [Harris, 1954] Harris, Z. S. (1954). Distributional structure. word, 10(2-3):146–162.
- [Jadav y Vaghela, 2016] Jadav, B. M. y Vaghela, V. B. (2016). Sentiment analysis using support vector machine based on feature selection and semantic analysis. International Journal of Computer Applications, 146(13).
- [Jurafsky y Martin, 2018] Jurafsky, D. y Martin, J. H. (2018). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.
- [Kim, 2014] Kim, Y. (2014). Convolutional neural networks for sentence classification. En Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751, Doha, Qatar. Association for Computational Linguistics.
- [Kowsari et al., 2019] Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., y Brown, D. E. (2019). Text classification algorithms: A survey. CoRR, abs/1904.08067.
- [Lample et al., 2018] Lample, G., Ott, M., Conneau, A., Denoyer, L., y Ranzato, M. (2018). Phrase-based & neural unsupervised machine translation. En Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 5039–5049, Brussels, Belgium. Association for Computational Linguistics.
- [Le y Mikolov, 2014] Le, Q. y Mikolov, T. (2014). Distributed representations of sentences and documents. En Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML'14, pp. II–1188–II–1196. JMLR.org.
- [Lison y Kutuzov, 2017] Lison, P. y Kutuzov, A. (2017). Redefining context windows for word embedding models: An experimental study. En Proceedings of the 21st Nordic Conference on Computational Linguistics, pp. 284–288, Gothenburg, Sweden. Association for Computational Linguistics.
- [Luque, 2019] Luque, F. M. (2019). Atalaya at tass 2019: Data augmentation and robust embeddings for sentiment analysis. Proceedings of TASS.
- [Mikolov et al., 2013a] Mikolov, T., Chen, K., Corrado, G., y Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- [Mikolov et al., 2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., y Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. En Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, pp. 3111–3119, USA. Curran Associates Inc.
- [Montañés-Salas et al., 2019] Montañés-Salas, R. M., del Hoyo-Alonso, R., y Aznar-Gimeno, R. (2019). From recurrency to attention in opinion analysis comparing rnn vs transformer models. Proceedings of TASS.

- [Osgood et al., 1957] Osgood, C. E., Suci, G. J., y Tannenbaum, P. H. (1957). The measurement of meaning. Number 47. University of Illinois press.
- [O’Shea y Nash, 2015] O’Shea, K. y Nash, R. (2015). An introduction to convolutional neural networks. CoRR, abs/1511.08458.
- [Pang et al., 2008] Pang, B., Lee, L., et al. (2008). Opinion mining and sentiment analysis. Foundations and Trends® in Information Retrieval, 2(1-2):1-135.
- [Pastorini et al., 2019] Pastorini, M., Pereira, M., Zeballos, N., Chiruzzo, L., Rosá, A., y Etcheverry, M. (2019). Retuyt-inco at tass 2019: Sentiment analysis in spanish tweets. Proceedings of TASS.
- [Pennington et al., 2014] Pennington, J., Socher, R., y Manning, C. (2014). Glove: Global vectors for word representation. En Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543.
- [Raschka y Mirjalili, 2017] Raschka, S. y Mirjalili, V. (2017). Python machine learning. Packt Publishing Ltd.
- [Sierra Martínez, 2017] Sierra Martínez, G. E. (2017). Introducción a los Corpus Lingüísticos. 1 edition.
- [Wolny, 2016] Wolny, W. (2016). Sentiment analysis of twitter data using emoticons and emoji ideograms.