PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

ESCUELA DE POSGRADO

REPRESENTACIÓN VECTORIAL DE RELACIÓN DE HIPONIMIA E HIPERONIMIA EN ESPAÑOL

Tesis para optar el grado de Magíster en Informática con mención en Ciencias de la Computación que presenta:

JOSE VICENTE UTIA DEZA

ASESOR:

MG. FELIX ARTURO ONCEVAY MARCOS

LIMA – PERÚ

Agosto de 2019

Resumen

Actualmente, gracias a Internet y a la Web se dispone de información casi ilimitada, la cual está representada a nivel de textos en su mayoría. Así, dado que acceder a estos textos en su mayoría es de libre acceso, nace el interés por su manipulación de una manera automatizada para poder extraer información que se considere relevante. El presente trabajo de investigación se ubica dentro de la detección automática de relaciones léxicas entre palabras, que son relaciones que se establecen entre los significados de las palabras tal como se consigna en el diccionario. En particular, se centra en la detección de relaciones de hiponimia e hiperonimia, debido a que éstas son relaciones de palabras en las que una de ellas engloba el significado de otra o viceversa, lo cual podría considerarse como categorización de palabras. Básicamente, el método propuesto se basa en la manipulación de una representación vectorial de palabras denominado *Word Embeddings,* para resaltar especialmente áquellas que tengan relación jerárquica, proceso que se realiza a partir de textos no estructurados.

Tradicionalmente, los *Word Embeddings* son utilizados para tareas de analogía, es decir, para detectar relaciones de sinonimia, por lo que se considera un poco más complejo utilizar estos vectores para la detección de relaciones jerárquicas (hiperonimia e hiponimia), por consecuencia se proponen métodos adicionales para que, en conjunto con los *Word Embeddings*, se puedan obtener resultados eficientes al momento de detectar las relaciones entre distintos pares de palabras.

Abstract

Currently, thanks to the Internet and Web, almost unlimited information is available, which is mostly represented at text level. Thus, given that access to these texts is mostly freely available, interest in their manipulation is born in an automated way to extract information that is considered relevant. The present research work is located within the automatic detection of lexical relations between words, which are relations that are established between the meanings of words as it is stated in the dictionary. In particular, it focuses on the detection of hyponymy and hyperonymy relationships, because these are word relationships in which one of them encompasses the meaning of another or vice versa, which could be considered as categorization of words. Basically, the proposed method is based on the manipulation of Word Embeddings to highlight especially words that have a hierarchical relationship, a process that is carried out from unstructured texts.

Traditionally, Word Embeddings are used for analogy tasks, that is, to detect synonymy relationships, so it is considered a bit more complex to use these vectors for the hierarchical relationships (hyperonimia and hyponymy) detection, therefore, additional methods are proposed, so in conjunction with the Word Embeddings, efficient results can be obtained when detecting the relationships between different pairs of words.

Agradecimientos

Gracias a mi asesor Magíster Felix Arturo Oncevay Marcos. A los miembros del jurado Hugo Alatrista Salas y José Antonio Pow-Sang Portillo.

Quiero agradecer a todos mis familiares y amigos los cuales fueron una gran ayuda y apoyo en el proceso de preparación de este trabajo, desde la etapa de la elección del tema, durante la investigación y recolección de la información, hasta el proceso de desarrollo y redacción de la tesis. Asimismo, un enorme agradecimiento a las personas que tuvieron el tiempo y la paciencia para revisarla.

Muchas gracias a todos por su tiempo y dedicación.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Generalidades

1.1. Introducción

La representación semántica es un lenguaje abstracto (formal) en el que los significados pueden ser representados. Diversas opiniones difieren sobre si la representación semántica es suficiente o necesaria, sobre su forma y sobre cómo se relaciona con las representaciones sintácticas. Teorías representativas del significado afirman que una representación semántica mental es necesaria para explicar el hecho de que las personas capten los significados. Por otra parte, teorías denotacionales de significado afirman que éste solo puede explicarse en términos de denotaciones en el mundo. La representación semántica puede tomar la forma de una estructura de características semánticas o fórmulas de un sistema lógico.

Cuando se habla de esta representación, hay cuatro preguntas fundamentales que se debe hacer: ¿Cómo se relacionan los significados de las palabras con las estructuras conceptuales? ¿Cómo se representa el significado de cada palabra? ¿Cómo se relacionan los significados de diferentes palabras entre sí? ¿Pueden los mismos principios de organización mantenerse en diferentes dominios de contenido? Este trabajo analiza y representa estos significados mediante *Word Embeddings*.

Los *Word Embeddings* , tales como Word2Vec (Mikolov *et al.*, 2013a; Mikolov *et al.*, 2013b; Pennington *et al.*, 2014) o GloVe (Pennington *et al.*,2014), son muy utilizados para tareas respecto a Procesamiento de Lenguaje Natural, porque se consideran representaciones útiles de palabras y, a menudo, conducen a un mejor rendimiento en las diversas tareas realizadas. Esta técnica de representación de palabras es utilizada para reducir la distancia entre palabras con contextos similares. Esto significa que palabras que son similares entre sí deberían tener la misma o una similar aproximación entre ellas. Un

ejemplo se da cuando x es rey, y_*p* es mujer,*y x_p* es hombre. La palabra buscada y es reina, pues hombre es a rey como mujer es a reina.

Inicialmente se ha podido observar que los *Word Embeddings* codifican relaciones léxicas simples como un singular-plural o país-capital dentro de su espacio vectorial, y si bien este campo es cubierto mayoritariamente por éstos, aún no se puede decir al 100% que pueda abarcar todas las relaciones semánticas de determinado lenguaje (Nayak *et al.*,2015). Este trabajo se enfoca en una relación en específico: la hiperonimia.

La hiperonimia es un tipo de relación semántica que se da entre una palabra de carácter más general y otra más específica, o sea un hiperónimo designa aquel término general que pudiese ser usado para englobar un significado mayor al que se refiere un término más particular. Así, la hiperonimia es importante en tareas de Procesamiento de Lenguaje Natural, sin embargo esta relación no ha sido muy explorada a través de la utilización de *Word Embeddings*, siendo que este tipo de relación es muy importante para tareas como *Question Answering* (Ferrucci *et al.*,2010), *Natural Language Inference* (MacCartney *et al.*,2009) y *Coreference Resolution* (Ng *et al.*,2002). Un ejemplo de uso de estos *Word Embeddings* orientados a hiperonimia e hiponimia sería para predecir la categoría o categorías de algún contenido textual como un nuevo artículo de Wikipedia o algún evento público creado en Facebook.

Por lo tanto, introducimos una metodología para generar *Word Embeddings* para el idioma español que esté más orientada a la hiperonimia en vez de la sinonimia, los cuales serán explicados más a detalle en puntos posteriores.

1.2. Definición del problema

En los últimos tiempos, las principales técnicas de Procesamiento de Lenguaje Natural (NLP) han estado basadas en el uso de modelos lineales como Regresión Logística o *Support Vector Machine* (SVM), los cuáles están

entrenados sobre vectores de características de muy alta dimensionalidad pero muy dispersos. Recientemente, este campo de estudio ha cambiado su tendencia hacia el uso de redes neuronales y específicamente al uso de *Word Embeddings*, lo cual no es más que una asignación de símbolos discretos a vectores continuos en un espacio dimensional comúnmente bajo. En términos simples, se podría decir que la distancia entre estos vectores es la representación de la distancia entre palabras, lo que permite hacer una generalización de cómo las palabras se comportan entre sí. Además, la red neuronal aprende a combinar los vectores de palabras de una forma que permite hacer distintos tipos de predicciones, una de las cuales es poder detectar si existe una relación de hiperónimo – hipónimo entre un par de palabras, siendo este tipo de predicción una de las que necesitan una optimización para poder mejorar los resultados enfocados hacia el idioma español. Así, a partir de esto, se podría utilizar estos resultados para tareas de clasificación de texto poniendo como ejemplo categorizar de una manera automática y más precisa un nuevo artículo de Wikipedia en base a hiperónimos representados en *Word Embeddings*.

1.3. Objetivo General

Mejorar el *accuracy* de una red neuronal recurrente para la predicción de categorías utilizando *Word Embeddings* previamente modificados en base a técnicas de reemplazos y orientados a relaciones de hiperonimia e hiponimia.

1.4. Objetivos Específicos

- OE1: Generar un corpus de pares hiperónimos – hipónimos.
- OE2: Generar *Word Embeddings* desde texto crudo de la Web.
- OE3: Generar *Word Embeddings* desde texto ajustado y enfocado a relaciones de hiperonimia e hiponimia.
- OE4: Ajustar *Word Embeddings* en base a manipulación de distancias para palabras con relaciones de hiperonimia e hiponimia.
- OE5: Implementar modelos de red neuronal recurrente utilizando *Word Embeddings* previamente generados.

- OE6: Comparar los resultados de *accuracy* generados por los modelos en una prueba extrínseca de clasificación de textos.

1.5. Resultados Esperados

- RE1: Un corpus con más de 30 mil pares de hiperónimos e hipónimos del idioma español. Este resultado permitirá completar el OE1
- RE2: Un corpus de más de 1 millón de artículos en el idioma español. Este resultado permitirá completar el OE2.
- RE3: Una metodología de generación de *Word Embeddings* desde la perspectiva de hipónimos – hiperónimos manipulando el corpus que será utilizado. Este resultado permitirá completar el OE3.
- RE4: Un espacio de características, constituidos por los *Word Embeddings* enfocados hacia la hiponimia – hiperonimia. Este resultado permitirá completar el OE4.
- RE5: Una red neuronal recurrente *Long Short Term Memory* (LSTM) con parámetros previamente definidos. Este resultado permitirá completar el OE5.
- RE6: Un reporte comparativo entre los resultados generados a partir de los modelos que utilizan *Word Embeddings* sin modificar y modificados. Este resultado permitirá completar el OE6.

1.6. Fuente de Datos

- Para el presente trabajo se utilizó un archivo XML proporcionado por Wikipedia que contiene todos sus artículos publicados hasta la fecha y el corpus del WordNet (Miller *et al.*,1990) tanto del idioma ingles y el español. A continuación se detalla cada uno de ellas:

- Corpus de Wikipedia para el idioma español: Wikipedia ofrece archivos de descarga en distintos idiomas, siendo uno de ellos el español, con el contenido de todos sus artículos (wikis) publicados en la web en solo un archivo consolidado en formato XML. Para el idioma español, actualmente

Wikipedia tiene publicado más de 1,210,000 artículos. Estos contenidos son actualizados semanalmente.

- WordNet: Es una base de datos léxica originalmente en inglés, la cual agrupa palabras en conjuntos de sinónimos llamados *synsets* (synonym set), brindando definiciones cortas y generales así como almacenando las relaciones semánticas entre los distintos conjuntos de sinónimos.

- WordNet en Español: Es una extensión al WordNet obtenida utilizando traducción y alineamiento automático a partir de la versión 3.0. de WordNet en inglés (Fernandez *et al.*,2008).

1.7. Algoritmos y herramientas

Para el cumplimiento del objetivo general del presente trabajo se implementaron y se aplicaron diversos algoritmos y técnicas que se encuentran descritos en la literatura. La implementación fue realizada en el lenguaje Python. Entre los algoritmos utilizados se pueden mencionar los siguientes:

- Algoritmo para procesamiento de corpus de Wikipedia usando las funciones propias del lenguaje Python.

- Algoritmo para generación de pares de hiperónimos – hipónimos en base al WordNet para el idioma español en base a los códigos de cada palabra.

- Algoritmo para generación de *Word Embeddings* en base a textos no estructurados usando la librería Gensim y Word2Vec.

1.8. Justificación

La presente investigación se enfocará en implementar técnicas para mejorar la predicción de categorías de determinado contenido de texto haciendo uso de una manera original de predicción que es a través de *Word Embeddings* y redes neuronales. Así, el presente trabajo permitiría mostrar

como ciertos tipos de *Word Embeddings* se adaptan mejor para tareas de clasificación de texto y demostrar que éstos no necesariamente solo son útiles para relaciones de similaridad como es la sinonimia, sino también para relaciones jerárquicas como es el caso de la hiperonimia e hiponimia.

1.9. Límites del proyecto

En el proyecto se realiza la investigación e implementación de una técnica para detectar con mayor precisión la relación de hiperonimia e hiponimia entre un par de palabras mediante *Word Embeddings*. Para poder realizar esta detección, es necesario tener primero un corpus con múltiples palabras que ya tengan definidas estas relaciones y para esto se utiliza el *WordNet* (original en inglés), el cual se ha transformado a un corpus en español, siendo que el corpus generado para el español es más pequeño debido a que diversos trabajos de traducción no contemplan todas las palabras, por lo que al momento de mapear las palabras, no todas son incluidas.

1.10. Organización del documento

El presente documento está dividido en capítulos. El primer capítulo contiene la motivación, la definición del problema, el objetivo general, objetivos específicos, resultados esperados, fuentes de datos, algoritmos y herramientas, justificación y límites del proyecto. El segundo capítulo del proyecto contiene la base teórica. El tercer capítulo contiene la revisión del estado del arte y el cuarto capítulo presenta la metodología del trabajo y los resultados. El quinto capítulo presenta las conclusiones y los trabajos futuros.

Capítulo 2

Marco Conceptual y Metodológico

2.1. Conceptos Generales

Una de los puntos importantes que ha sido importante para el éxito del Deep Learning ha sido la habilidad de las redes neuronales multi-capas de extraer representaciones distribuidas de los datos para así simplificar el proceso de entrenamiento y aprendizaje. Estas representaciones permiten entender la estructura estadística y los factores que expresan la variación de los datos, abstraer la información relevante de éstos y aumentar la eficiencia del proceso de entrenamiento. Una de estas representaciones es Word2Vec, el cual es una representación vectorial cuyo objetivo es generar una función que pueda convertir cada palabra en un vector, de tal manera que la cercanía entre vectores sea similar a la cercanía semántica de las palabras.

2.2. Deep Learning

El Deep Learning es una técnica de aprendizaje automático que enseña a las computadoras a realizar lo que resulta natural para las personas: aprender mediante ejemplos (LeCun *et al.*,2015) .

Con esta técnica, un modelo informático tiene la capacidad de realizar tareas de clasificación directamente a partir de un conjunto de imágenes, datos textuales o hasta sonidos. Los modelos de Deep Learning son tan eficientes que pueden llegar a obtener tal grado de precisión que, en su mayoría de ocasiones últimamente, son mucho mejores que el propio rendimiento humano. Los modelos se entrenan utilizando grandes volúmenes de datos y redes neuronales que contienen muchas capas.

2.3. Redes neuronales

Las redes neuronales son modelos matemáticos que intentan imitar el funcionamiento del sistema nervioso humano, principalmente compuestos por un conjunto de unidades llamadas neuronas conectados entre sí. El primer modelo de red neuronal fue dado a conocer por McCulloch y Pitts (1943). Éste era un modelo binario donde cada neurona tenía un umbral prefijado, y sirvió de base para los modelos posteriores (Escobar *et al.*,2014). Estas redes permiten obtener un modelo cuyo fin es predecir cuál es el valor de salida, dados unos valores de entrada.

2.3.1. Estructura

Una red neuronal se compone principalmente por:

- neuronas de entradas
- neuronas ocultas
- neuronas de salidas
- interconexiones entre las neuronas.

La figura 1 muestra la estructura de una red neuronal artificial mientras la figura 2 muestra distintas regiones de decisión.

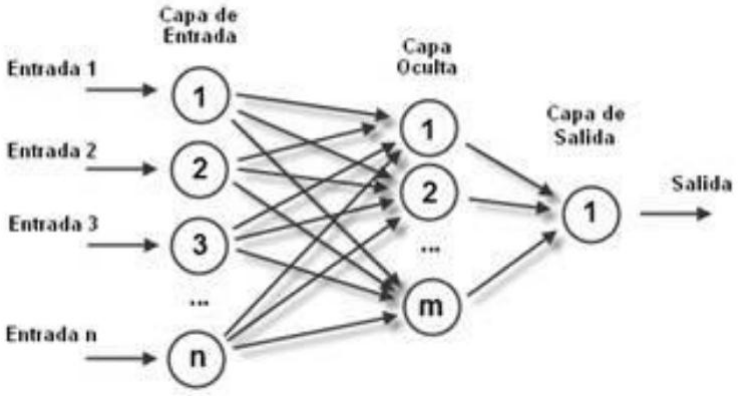


Figura 1. Estructura de una red neuronal

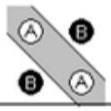
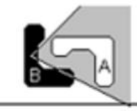
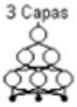
Estructura	Regiones de Desición	Problema de la XOR	Clases con Regiones Mezcladas	Formas de Regiones más Generales
1 Capa 	Medio Plano Limitado por un Hiperplano			
2 Capas 	Regiones Cerradas o Convexas			
3 Capas 	Complejidad Arbitraria Limitada por el Número de Neuronas			

Figura 2. Regiones de decisión

2.4. Redes Neuronales Recurrentes

Una red neuronal recurrente (RNN) es una clase de red neuronal artificial donde las conexiones entre nodos generan un gráfico dirigido a lo largo de una secuencia. Esto le permite exhibir un comportamiento dinámico temporal para una secuencia de tiempo. A diferencia de las redes neuronales tradicionales, los RNN pueden usar su estado interno (memoria) para procesar secuencias de entradas. Esto los hace aplicables a tareas como el reconocimiento de escritura a mano no segmentado o el reconocimiento de voz.

Las redes LSTM (Long – Short Term Memory) son un tipo especial de redes recurrentes que se distinguen en que la información puede persistir mediante la introducción de ciclos repetidos en el diagrama de la red, con el fin de recordar estados previos y así hacer uso de esta información para decidir cuál será el siguiente. Así, las LSTM tienen la capacidad de aprender dependencias largas, a diferencia de las redes recurrentes estándar las cuales se enfocan más en el corto plazo.

2.5. Representación semántica

La semántica es el campo que abarca el estudio del significado de los signos lingüísticos así como de sus combinaciones. Así, cuando se habla de relaciones semánticas entre las palabras, esto se refiere a las relaciones de significado que hay entre ellas (Garrido *et al*,2010). Hay distintos tipos de

relaciones semánticas entre las palabras, entre las cuales podemos mencionar la hiperonimia e hiponimia.

2.6. Hiperonimia e hiponimia

La hiperonimia es la relación semántica que vincula a una palabra con otras de significado más específico por las que puede ser sustituida. Por ejemplo, el significado de *animal* es más general que el de *perro, gato, caballo,* etc. Estos términos que son más específicos se los denomina hipónimos. Así, la relación entre estas 2 palabras (hiperónimo e hipónimo) se podría considerar como de inclusión en la que la primera es el término general y la segunda es el término especifico.

2.7. Word2Vec

Uno de los modelos más usados de representaciones distribuidas de palabras es *word2vec* (Mikolov *et al.*, 2013a; Mikolov *et al.*, 2013b; Pennington *et al.*, 2014). El modelo se basa en redes neuronales de varias capas y tiene dos posibles arquitecturas: Skip-gram y CBOW.

La diferencia entre ambas arquitecturas es que la primera predice las palabras contexto a partir de la palabra central. La segunda es lo inverso a la primera, el modelo predice la palabra central a partir de una ventana de palabras contexto (la predicción es independiente del orden de las palabras contexto).

2.8. WordNet

Es una gran base de datos léxica del inglés (Miller *et al.*,1993). Sustantivos, verbos, adjetivos y adverbios se agrupan en conjuntos de sinónimos (*synsets* cognitivas), cada uno expresando un concepto distinto. Los *synsets* están vinculados entre sí por medio de las relaciones conceptuales, semánticas y léxicos.

La red resultante de las palabras y los conceptos relacionados de manera significativa se puede navegar con el navegador. WordNet es también

libre y públicamente disponible para descarga. La estructura de la WordNet hace que sea una herramienta útil para la lingüística computacional y procesamiento del lenguaje natural.

WordNet superficialmente se parece a un diccionario de sinónimos, ya que los grupos de palabras juntas sobre la base de sus significados. Sin embargo, hay algunas diferencias importantes. En primer lugar, WordNet articula no sólo la palabra formas de cadenas de letras, pero los sentidos específicos de las palabras. Como resultado, las palabras que se encuentran en estrecha proximidad entre sí en la red son semánticamente desambiguadas. En segundo lugar, las etiquetas de WordNet las relaciones semánticas entre las palabras, mientras que las agrupaciones de palabras en un diccionario de sinónimos no sigue ningún patrón explícito que no sea el sentido de similitud.

Capítulo 3

Estado del Arte

3.1. Introducción

En este capítulo se presenta el estado de arte de la aplicación de *Word Embeddings* para detección de pares hiperónimos - hipónimos, una revisión bibliográfica de los métodos y las técnicas aplicadas.

Durante el proceso de revisión bibliográfica se analizaron los artículos y libros obtenidos de las bases de datos como SCOPUS, IEEEXplore y Scholar Google. Se buscaron los trabajos relacionados con el tema del trabajo presente. Para tal propósito se formularon preguntas específicas para que el resultado de la búsqueda fuera más detallado y enfocado en el tema estudiado. Además se incluyeron los criterios de inclusión y exclusión para disminuir la cantidad de la bibliografía.

3.2. Extracción de hiperónimos, procesamiento de corpus y generación de Word Embeddings

Revisando los trabajos de extracción de hiperónimos, se observó que los trabajos predominantes respecto a este tema son en el idioma ingles (Rion Snow *et al.*,2005; Morin *et al.*,2004), mientras que son limitados aquellos orientados a otros idiomas incluido el español (Camacho-Collados *et al.*, 2018). La principal forma de extracción de hiperónimos es a través de patrones predefinidos tales como: la <hipónimo> es una <hiperónimo> (Dorantes, M & Pimentel *et al.*, 2018), incluyendo valores de confianza para cada patrón debido a que pueden haber falsos positivos en ciertos patrones (Ortega *et al.*,2011). Es por eso, que dicho trabajo categoriza en base a experiencias anteriores, que aquellos patrones con valores de confianza mayores a 0.60 serán aquellos que tengan mucha mayor probabilidad de ser pares hiperónimos – hipónimos.

Otra manera de extracción de hiperónimos es a través del WordNet (Miller *et al.*,1990) mediante librerías propias (Dias *et al.*,2008). Igualmente predomina el idioma Ingles debido a que el WordNet fue construido en base a este idioma, por lo que se tuvo que emplear una versión de WordNet construida para el español en base al original (Fernandez *et al.*,2008) adaptando las implementaciones de las librerías en ingles previamente mencionados para el español.

Respecto a la obtención de un corpus para poder trabajar en base a *Word Embeddings*, se analizaron distintos trabajos que contengan corpus con millones de palabras, siendo algunos de estos enfocados a temas como biomedicina (Chiu *et al.*,2016) o redes sociales (Zeng *et al.*, 2017) incluidos los tweets y data en general (Li *et al.*, 2017) siendo el corpus más variado y formal el de Wikipedia (Chi-Yen *et al.*, 2017) debido a que éste corpus contiene millones de artículos descritos de una manera muy detallada.

Para la generación de los *Word Embeddings* se analizaron diversas implementaciones como Word2Vec (Mikolov *et al.*, 2013a; Mikolov *et al.*, 2013b; Pennington et al., 2014), FastText (Athiwaratkun et al,2018), Glove (Pennington *et al.*,2014), LSA entre otros siendo en la mayoría de artículos Word2Vec mejor calificado respecto a los otros en base a la forma en que éste representa las palabras sobre todo para corpus demasiado grandes (Marwa *et al.*,2017).

Para la detección de relaciones de hiperonimia e hiponimia existen diversos trabajos (la mayoría enfocados al idioma Ingles) orientados hacia Word Embeddings que utilizan modelos de redes neuronales (Nayak *et al.*,2015) incluidas las redes neuronales convolucionales (Mohammed & Maharjan *et al.*,2016) el cual trabaja con *Word Embeddings* generados a partir del corpus de Wikipedia más los feeds de noticias de Google (*Google News*) el cual obtiene como medida de calidad un 79% de F-score para la detección de relaciones de hiperonimia. Igualmente existe otro trabajo que se aproxima a este porcentaje mediante Projection Learning y el uso de clusters (Ruiji *et al.*,2014), el cual es de 73.74%.

En resumen, podemos señalar que hay algunos trabajos sobre *Word Embeddings* enfocados hacia la relación de hiperonimia – hiponimia en el idioma Ingles, mientras que para el idioma Español no se ha logrado encontrar trabajos en los que se usen *Word Embeddings* para la detección de dicha relación, mas solo existen trabajos que llegan solamente hasta la extracción de pares hiperónimos – hipónimos para este idioma. Se analizaron las investigaciones desde 2004 hasta el 2018. En los distintos artículos revisados se observó tanto el uso de *Word Embeddings* generados a partir de Word2Vec como de Glove. Asimismo, se vio que para la detección de la relación de hiperonimia entre un par de palabras, se basaron tanto en un corpus extraído desde WordNet así como corpus construidos manualmente.

Capítulo 4

Desarrollo y Resultados

4.1. Dump de Wikipedia en Español

Wikipedia ofrece a través de Wikimedia una completa copia de todos sus artículos en diferentes idiomas en formato XML y otros tipos de formato los cuales son actualizados periódicamente. Para el idioma español, se observó que a la fecha cuenta con un poco más de 1,210,000 artículos incluidos en la copia el cual pesa aproximadamente 12 GB, por lo que no es recomendable abrir directamente este archivo con algún editor de texto ya que es muy grande. En esta copia, cada artículo inicia con el tag <page> y cierra con el tag </page> y dentro de esta estructura se puede encontrar diversa información como <title> el cual se refiere al título del artículo, <revisión> el cual a su vez muestra información sobre las modificaciones realizadas y uno de los más importantes tag <text> el cual es el que almacena toda la información en texto del artículo en mención.

Así, toda la información de cada artículo se encuentra en un solo archivo publicado por Wikipedia. El dump que se está realizando para el presente trabajo se puede descargar de la siguiente URL: https://dumps.wikimedia.org/eswiki/latest/eswiki-latest-pages-articles.xml.bz2

4.2. Generación de corpus de pares hiperónimos – hipónimos en base a patrones

La primera aproximación para crear un propio corpus de pares hiperónimos – hipónimos es la extracción de estos desde el mismo corpus de Wikipedia en base a patrones lexicales. En el cuadro 1 se puede apreciar un listado completo de patrones a utilizar para la construcción del corpus.

N°	Patrón
1	el <hipónimo> es el único <hiperónimo>

2	el uso de la <hipónimo> como <hiperónimo>
3	el <hipónimo> es uno de los <hiperónimo> más
4	de la <hipónimo> como <hiperónimo> de
5	de las <hipónimo> como <hiperónimo>
6	las <hipónimo> son una <hiperónimo>
7	el <hipónimo> es un <hiperónimo> que
8	el <hipónimo> es el <hiperónimo> que
9	de <hiperónimo> como <hipónimo> y
10	la <hipónimo> es un <hiperónimo>
11	la <hipónimo> una <hiperónimo>
12	las <hipónimo> son <hiperónimo> que
13	el <hipónimo> es un <hiperónimo> de
14	la <hipónimo> es la <hiperónimo>
15	la <hipónimo> es una <hiperónimo> que
16	la <hipónimo> como una <hiperónimo>
17	que la <hipónimo> es una <hiperónimo>
18	el <hipónimo> es una <hiperónimo>
19	la <hipónimo> es el <hiperónimo> de
20	de <hipónimo> y otras <hiperónimo>
21	del <hipónimo> como <hiperónimo>
22	el <hipónimo> es la <hiperónimo>
23	<hiperónimo> de <hipónimo> de
24	de <hipónimo> y <hiperónimo>
25	<hiperónimo> de <hipónimo> y
26	de <hipónimo> o <hiperónimo>
27	los <hipónimo> son <hiperónimo>
28	de <hipónimo> como <hiperónimo> de
29	el <hipónimo> y las <hiperónimo>
30	de los <hipónimo> y <hiperónimo>
31	de los <hipónimo> y los <hiperónimo>
32	la <hipónimo> es el único <hiperónimo> natural
33	<hiperónimo> de la actividad <hipónimo> y el deporte

34	la anorexia y la <hipónimo> son <hiperónimo>
35	de <hipónimo> y otros <hiperónimo>
36	el <hipónimo> es el <hiperónimo> de mayor longevidad
37	los <hipónimo> y otros <hiperónimo>
38	facultad de <hiperónimo> de la actividad <hipónimo> y
39	la <hipónimo> y otros <hiperónimo>
40	las <hipónimo> marinas son <hiperónimo>
41	el <hipónimo> es el <hiperónimo> interno más
42	licenciado en <hiperónimo> de la actividad <hipónimo> y del deporte
43	el <hipónimo> es el <hiperónimo> más grande del cuerpo

Cuadro 1. Lista general de patrones de pares hiperónimos – hipónimos

La utilización de estos patrones en Wikipedia generó 43,000 pares de hiperónimos e hipónimos, de los cuales en base a una pequeña muestra de 1000 pares se verificó que habían más de 60% de pares que en realidad no correspondían a la relación hiperónimo – hipónimo. Por ejemplo para el patrón n°8 se tiene la siguiente sentencia:

El **rock** es el **mismo** que

Esto definitivamente no es una relación de hiperonimia y existen muchos casos que se encuentran en los patrones anteriores y que no cumplen con la relación. Es por eso que en base a valores de confianza ya definidos en otros trabajos, la lista es reducida a la mostrada en el Cuadro 2.

N°	**Patrón**
1	de <hipónimo> y otras <hiperónimo>
2	de <hipónimo> y otros <hiperónimo>
3	el <hipónimo> es el único <hiperónimo>
4	el <hipónimo> es un <hiperónimo> que
5	los <hipónimo> y otros <hiperónimo>
6	el <hipónimo> es uno de los <hiperónimo> más

8	que la <hipónimo> es una <hiperónimo>
9	la <hipónimo> y otros <hiperónimo>
10	la <hipónimo> es una <hiperónimo> que
11	el <hipónimo> es un <hiperónimo> de

Cuadro 2. Lista de patrones de pares hiperónimos – hipónimos con mayor precisión

En base a solo estos patrones, solo se consiguieron 7 000 pares de hiperónimos e hipónimos siendo 2500 pares únicos, de los cuales se validó con una muestra de 200 pares que más del 70% cumplen con la relación. En base a esto se optó por otra manera de conseguir un corpus más grande y que además, este correctamente validado.

4.3. Generación de corpus de pares hiperónimos – hipónimos en base a WordNet

Para la construcción de un corpus de pares hiperónimos – hipónimos más grande que el anterior se hizo uso del WordNet, el cual es un diccionario léxico que además también incluye relación jerárquica de palabras y el cual su versión original esta en inglés. Para obtener este listado de pares en español, es necesario bajar una extensión del WordNet para el idioma en cuestión y mediante código Python , buscar mediante el ID de cada palabra la respectiva traducción y almacenarla. Finalmente para elaborar el nuevo corpus de hipónimos e hiperónimos, WordNet incluye una función "hypernim" el cual al ingresarle como input determinada palabra, ésta devolverá las palabras que sean hipónimos/hiperónimos. Todo este trabajo se realizará para cada palabra encontrada en WordNet y de esta manera podremos generar nuestro corpus con los pares necesarios para la posterior evaluación. Al realizar todo este proceso, se generó un corpus de 40 900 pares de palabras, siendo una muestra la que se observa en el Cuadro 3.

```
1822 amo alguno
1823 amo alma
1824 amo empresario
1825 amo gobernador
1826 amo humano
1827 amo individuo
1828 amo mortal
1829 amo persona
1830 amo ser_humano
1831 amonestación castigo
1832 amoniuria síntoma
1833 amor afección
1834 amor afecto
1835 amor amante
1836 amor atracción_secual
1837 amor calentura
1838 amor concupiscencia
1839 amor cuidado
1840 amor deseo_sexual
1841 amor emoción
1842 amor eros
```

Cuadro 3. Muestra de pares hipónimo - hiperónimo

Como se ve en la imagen anterior, se puede ver que una palabra puede tener múltiples hiperónimos. Según el listado, el número total de hipónimos distintos es de 17 676 palabras, siendo los que tienen mayor hiperónimos los indicados en el Cuadro 4.

Hipónimo	**Numero de Hiperónimos**
Dirección	35
parte	34
Estudio	31
entrada	30
Golpe	28
Familia	26
inclinación	26
Señal	26
aficionado	25
Tipo	25

Cuadro 4. Lista de hipónimos con mayor cantidad de hiperónimos

4.4. Preprocesamiento del dump de Wikipedia

Luego de obtenido el corpus, es necesario hacer un procesamiento para remover todas las etiquetas embebidas en el XML ya que nuestro algoritmo de procesamiento de *Word Embeddings* necesita texto puro. Es por eso que se ha realizado un procesamiento del dump mediante librerías ya creadas en el lenguaje Python. Para realizar esto, fue necesario descargar el módulo gensim ya que este trae la librería WikiCorpus el cual contiene una función llamada get_texts que es la cual quitara todos los tags innecesarios y dejará solamente la información textual relevante de cada artículo. Realizar todo este procesamiento puede tomar aproximadamente una hora para el procesamiento del dump de Wikipedia en español.

4.5. Vectorización

Luego de haber procesado el dump de Wikipedia, el siguiente paso fue convertir todo este texto en vectores con la idea de obtener relaciones entre palabras. En los objetivos de este trabajo se propone 2 maneras para encontrar relaciones de hiperonimia – hiponimia entre palabras. La primera es mediante la generación de los *Word Embeddings* orientados a relaciones de hiperonimia e hiponimia, lo cual significa que habría que hacer un trabajo previo en el corpus obtenido para que los *Word Embeddings* generados a partir de éste tenga una representación distinta a la que tuviese si es que se generara con el corpus puro de Wikipedia. La otra alternativa seria modificar directamente los *word embeddings* generados con el corpus original mediante técnicas de *Fine Tuning* (Vulic *et al.*,2017) . A continuación se presenta el detalle de ambos métodos utilizados

4.6. Generación de Word Embeddings orientados a relaciones de hiperonimia e hiponimia.

Para la generación de *Word Embeddings* orientados a las relaciones indicadas, es necesario que desde el corpus de Wikipedia el texto ya esté orientado a dichas relaciones. Para esto, se realizaron varias pruebas en base al corpus de hiperónimos – hipónimos construidos desde WordNet. Se realizaron

varias formas de manipulación al corpus de Wikipedia, entre las cuales destacan:

4.6.1. Reemplazo directo del hipónimo por sus respectivo hiperónimo.

Dado un hipónimo "a" y sus hiperónimos "b","c","d","e" y "f", entonces si en un artículo de Wikipedia aparece la palabra "a", ésta será reemplazada por toda una cadena "**a** b c d e f g".

4.6.2. Reemplazo directo del hipónimo por sus respectivo hiperónimo con formato definido.

Dado un hipónimo "a" y sus hiperónimos "b","c","d","e","f","g","h" e "i", entonces si en un artículo de Wikipedia aparece la palabra "a", ésta será reemplazada por toda una cadena "b c **a** d e f g **a** h i". Esta manera de reemplazo se ideó teniendo en cuenta como trabaja Word2Vec y el parámetro Window Size.

4.6.3. Duplicidad de artículo en el que se encuentre un hipónimo por sus hiperónimos.

Dado un hipónimo "a" y sus hiperónimos "b","c","d","e" y "f", entonces si en un artículo de Wikipedia aparece la palabra "a", se agregarán 5 nuevas entradas al corpus de Wikipedia en el que en el primero, "a" será reemplazado por "b" y en el último de ellos "a" será reemplazado por "f". Esta manera de la manipulación del corpus es demasiado pesada debido a que tenemos una gran cantidad de corpus de pares y a partir de éstos, se generarán millones de nuevos artículos. Es por eso, que se realizaron formas para reducir esta carga. Una de ellas fue utilizar como muestra 50 hipónimos con sus respectivos hiperónimos, siendo alguno de éstos hipónimos los que mayor número de hiperónimos presentan. A partir de esta muestra se realizó el reemplazo. La otra forma fue limitar el número de reemplazos por cada hipónimo, es decir, si un hipónimo aparece en más de 100 artículos, solo los primeros 100 artículos serán duplicados y el hipónimo será reemplazado por sus respectivos hiperónimos. El resto de artículos en los que aparezcan estos hipónimos no serán duplicados.

Para la generación de los distintos *Word Embeddings* se tuvieron en cuenta los siguientes parámetros:

- ✓ **Número de dimensiones:** 300
- ✓ **Window Size:** 05. Al tratar de utilizar un *window size* menor, los resultados no fueron los esperados. Esto asumimos que debe ser debido a que el corpus utilizado es muy grande y es necesario tener un *window size* no tan pequeño.
- ✓ **MinCount:** Se considera un mínimo de 05 apariciones de la palabra en todo el corpus.
- ✓ **Negative Sampling:** Se realizarán solo 05 cambios cuando sea necesario actualizar los pesos de palabras "negativas" respecto a una palabra dada. Utilizamos este valor ya que es recomendable que para datasets grandes el valor esté entre 2 - 5.
- ✓ **Learning Rate:** 0.025
- ✓ **Learning Rate mínimo:** 0.0001
- ✓ **Batches:** 10,000
- ✓ **Epochs (épocas):** 05
- ✓ **Número de *hidden layers*:** 01
- ✓ **Tamaño del dataset:** 1214259 artículos

Para poder decir que en el *Word Embedding* una palabra es hiperonima de otra, utilizaremos la distancia entre ellas como forma de detección de la relación siendo aquellas que tengan menor distancia las que tengan mayor probabilidad de que cumplan la relación.

Para 4.6.1. y 4.6.2 los resultados no fueron los esperados debido a que no todos los hiperónimos de un hipónimo se encontraban cerca entre sí. Pongamos de ejemplo el hipónimo "perro" y sus hiperónimos ya definidos en el corpus: "antipático", "cánido", "desagradable", "animal", "mamífero, "carnívoro" y cuadrúpedo"

Los resultados de una de las palabras seleccionadas ("perro") utilizando el *Word Embedding* sin manipulación se observan en el Cuadro 5.

```
[('gato', 0.7427746653556824),
 ('perrito', 0.6987060904502869),
 ('gatito', 0.6677889823913574),
 ('caniche', 0.6553350687026978),
 ('sabueso', 0.6537164449691772),
 ('faldero', 0.6313320398330688),
 ('conejo', 0.6240070462226868),
 ('lebrel', 0.6234978437423706),
 ('cachorro', 0.622416615486145),
 ('mapache', 0.6200912594795227)]
```

Cuadro 5. Resultados utilizando Word Embedding original

Como se ve, ninguna palabra de los resultados se encuentra en el corpus de pares definidos para "perro", es por eso que necesitamos hacer manipulación al corpus.

Con las formas definidas en 4.6.1 Y 4.6.2 se obtuvieron similares resultados tal como se aprecia en el Cuadro 6. Las palabras obtenidas son las mismas pero las distancias varían mínimamente e igualmente se nota la ausencia de algunos hiperónimos definidos tales como "animal" o "cánido".

```
[('cuadrúpedo', 0.6358938217163086),
 ('mamífero', 0.6164237260818481),
 ('carnívoro', 0.5128576755523682),
 ('carnívoros', 0.509699821472168),
 ('antipático', 0.5066691040992737),
 ('desagradable', 0.4704347550868988),
 ('sabuesos', 0.4083105921745 3003),
 ('gatos', 0.4062526524066925),
 ('gato', 0.4033600687980652),
 ('lebreles', 0.401793211698 5321)]
```

Cuadro 6. Resultados utilizando Word Embedding generado en bases a reemplazos

Esta misma tendencia se observó al analizar otros hipónimos. Con la forma propuesta en 4.6.3 ya se empezaron a ver mejores resultados a las formas anteriores, tal como se ve en el Cuadro 7.

```
[('antipático', 0.9912312030792236),
 ('cánido', 0.9851634502410889),
 ('desagradable', 0.9677984714508057),
 ('cuadrúpedo', 0.9605344533920288),
 ('carnívoro', 0.9600476622581482),
 ('mamífero', 0.9590555429458618),
```

```
('animal', 0.826754093170166),
('perrito', 0.7160046100616455),
('gato', 0.668850839138031),
('cachorro', 0.6467772722244263)]
```

Cuadro 7. Resultados utilizando Word Embedding generado en bases a duplicidad

Como se ve, todas las palabras se encuentran en el listado de palabras con menor distancia. Cabe decir que estos resultados son realizando todos los reemplazos, sin limitaciones. Igualmente probamos limitando el número de reemplazos por hipónimo. Se puso una limitación tanto de 1000 como de 10000 reemplazos máximos y los resultados no fueron tan buenos como al realizar el reemplazo total.

Siendo que la forma 4.6.3 y de la forma de reemplazo total es la que mejor resultados ofrece, el siguiente paso es realizar pruebas de clasificación de texto utilizando el *Word Embedding* generado. En este caso, como ejemplo en el resultado utilizaremos el artículo de "perro" de Wikipedia el cual puede ser encontrado en https://es.wikipedia.org/wiki/Canis_lupus_familiaris. La idea es que el resultado que nos pueda devolver el *Word Embedding* se aproxime a la realidad (hiperónimos del animal perro) o a las categorías definidas propiamente por Wikipedia. Wikipedia define las siguientes categorías para esta página. En el Cuadro 8 se puede observar la forma en que Wikipedia devuelve las categorías para el artículo relacionado a "perro".

```
Categories
Categoría:Animales descritos en 1758 (id: ??, ns: 14)
Categoría:Animales domesticados (id: ??, ns: 14)
Categoría:Cánidos (Canidae) no amenazados (id: ??, ns: 14)
Categoría:Perros (id: ??, ns: 14)
```

Cuadro 8. Resultados de categorías definidas para perro

Para obtener los hiperónimos que se podrían aplicar a este artículo, lo que se hace es obtener un vector resultante de todo el contenido del artículo y este vector será evaluado contra nuestro Word Embedding generado utilizando la función similar_by_vector de Word2Vec. Los resultados obtenidos luego de aplicar dicha lógica se visualizan en el Cuadro 9.

```
[('perros', 0.6560007333755493),
 ('animales', 0.6353904008865356),
 ('roedores', 0.5916348695755005),
 ('mamíferos', 0.5899962186813354),
 ('cánidos', 0.5685439109802246),
 ('carnívoros', 0.5682492852210999),
 ('herbívoros', 0.5642678141593933),
 ('rumiantes', 0.5632807016372681),
 ('animal', 0.5519993901252747),
 ('pelaje', 0.5423393845558167)]
```

Cuadro 9. Resultados de categorías utilizando Word Embedding generado en base a reemplazos

Como se ve, las categorías definidas por Wikipedia "Animales", Cánidos" y "Perros" han sido incluidos en los resultados devueltos por nuestro Word Embedding. Igualmente otras categorías no definidas por Wikipedia como "mamíferos" o "carnívoros" se están mostrando en los resultados debido a que éstas palabras están incluidas en nuestro corpus de pares hiperónimos – hipónimos. En el Cuadro 10 se puede apreciar que utilizando el *Word Embedding* original no todas las categorías son devueltas:

```
[('roedores', 0.5849004983901978),
 ('animales', 0.5761560201644897),
 ('rumiantes', 0.5748881101608276),
 ('mamíferos', 0.5688313245773315),
 ('herbívoros', 0.5685784220695496),
 ('carnívoros', 0.5661754608154297),
 ('menos', 0.5458819270133972),
 ('équidos', 0.5457005500793457),
 ('individuos', 0.5450974702835083),
 ('cánidos', 0.5443596243858337)]
```

Cuadro 10. Resultados de categorías utilizando Word Embedding original

Tomando como referencia las etiquetas definidas por Wikipedia y con una revisión de 50 artículos, obtuvimos aproximadamente un 70% de precisión en nuestros resultados utilizando el *Word Embedding* manipulado.

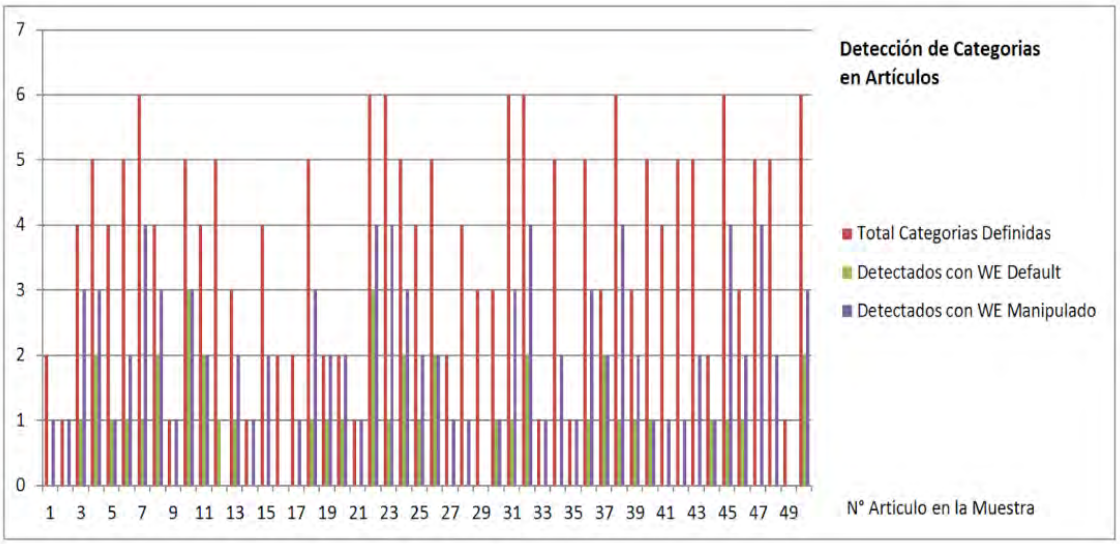


Figura 3. Resultados de detección de categorías de Wikipedia

4.7. Modificación de Word Embeddings pre-procesados para orientarlos a relaciones de hiperonimia e hiponimia.

Otra alternativa propuesta para la detección de *Word Embeddings* es la modificación de las posiciones de los vectores de ciertas palabras que se hayan generado a partir del corpus original de Wikipedia. Para realizar esto, es necesario obtener el cálculo de las distancias entre pares de hiperónimos – hipónimos en el espacio vectorial generado, y para esto primero es necesario tener definido un listado de pares de hiperónimos – hipónimos ya definidos. Para esto utilizaremos el corpus de WordNet generado anteriormente.

Así, para cada par de palabras de este corpus se le calculo la distancia en base al *Word Embedding* original. La idea fue encontrar algún patrón numérico en estas distancias, por ejemplo que estos pares se encuentren mayormente en el rango de 0.45 y 0.55 de distancia y a partir de eso solo trabajar con aquellos pares que cumplan el patrón, pero el resultado al analizar todo los pares fue que había demasiada dispersión, es decir, habían pares que se encontraban en la región con el mínimo valor, el valor medio y el máximo valor.

Por ejemplo, el siguiente grafico muestra la distancia de 1000 pares de palabras elegidas al azar, siendo que si el valor de la distancia entre un par de palabras se aproxima más a 1, éstas tendrían una similaridad más próxima.

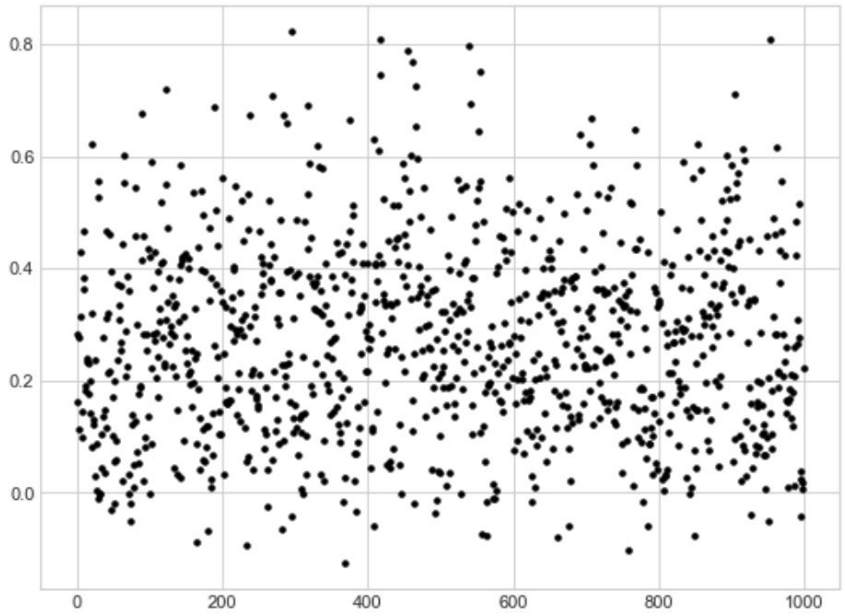


Figura 4. Dispersión de distancias de 1000 pares aleatorios de hiperónimos - hipónimos

Siendo que los resultados iniciales no ayudan a detectar algún patrón en la distancia de pares hiperónimos – hipónimos, es necesario hacer una evaluación de las distancias de cada par de palabras no solo a nivel de todas las dimensiones definidas del *Word Embedding*, sino a nivel de "n-1" dimensiones o menos. Así, es necesario hacer un trabajo de fuerza bruta con combinaciones de distintas dimensiones para verificar cual o cuales dimensiones se adecuan mejor a la relación jerárquica que se está tratando de optimizar.

Como el *Word Embedding* se ha generado con 300 dimensiones, se realizaron pruebas entre pares de hiperónimos e hipónimos utilizando todas las combinaciones posibles tanto de 299 y 298 dimensiones, con la idea de visualizar si excluyendo 1 o más dimensiones ayudaban a mejorar los resultados a nivel de distancias y para también ver si alguna combinación de dimensiones predominaba sobre otra. Luego de analizados los resultados tanto para 299 y 298 dimensiones, se observaron distancias muy similares para cada

combinación. En el Cuadro 11 se muestran las distancias obtenidas para el par de palabras "agua"-"liquido".

Distancia global (300 dimensiones):
0.5324103294621951

Distancias tomando 299 dimensiones:

Combinación#	Distancia
234	0.5419720283412621
32	0.5405958964172276
53	0.5396337902130239
109	0.5390345820940234
177	0.5383895290401947
95	0.5381702442286695
267	0.5376882176592565
217	0.537468539219114
113	0.5368904798276923
152	0.5364479893619349
257	0.5364262537435356
284	0.5363879831780085
9	0.5363620010848927
4	0.5363159359609702
94	0.5363149473991379
227	0.5360709549866652
281	0.5359842710121203
291	0.5357947737494951
157	0.5357771218453251
190	0.5355059668720482

Distancias tomando 298 dimensiones:

27293	0.5502730438174667
27314	0.5492872926141932
27370	0.5486801605646211
27438	0.5480434050957994
1410	0.5479009329897236
27356	0.5478070287289704
35745	0.5473248999456612
5918	0.547295346169155
27478	0.5470949818584901
15608	0.5466516757241157
27374	0.5465105999732843
4497	0.5464212623086502
5939	0.5463226897182201
27413	0.5461265951788649
33130	0.5460397502696955
40420	0.5460005492743277
27270	0.545975253471318
27265	0.5459548737789393
35543	0.5459441814643151
27355	0.5459318814432695

Cuadro 11. Resultados de distancias utilizando 299 y 298 dimensiones

Por temas de performance no se realizaron evaluaciones de 297 dimensiones a menos debido a que las combinaciones posibles son demasiadas. Igualmente, en base a los resultados tanto de 299 y 298 dimensiones, vemos que no hay mucha diferencia entre dichas distancias, por lo que se concluye descartar este método de manipulación de *Word Embedding*.

4.8. Implementación de Red Neuronal Recurrente para predicción de categoría

Teniendo en cuenta que el *Word Embedding* que mejor resultado nos dio es aquel generado en el punto 4.6.3 mediante duplicidad del artículo por cada hiperónimo que se encuentra para un hipónimo dado, se procedió a utilizar este *Word Embedding* como un *input* a una red neuronal recurrente que se implementó como último paso para la predicción de hiperónimos. Para la construcción de la red neuronal se utilizó *Keras*. Cabe indicar que la prueba extrínseca de predicción que se realizó se enfocó como un problema de clasificación binaria.

El primer paso fue crear tanto el *dataset* de entrenamiento como el *dataset* de prueba. Como los *Word Embedding* generados fueron en base a todos los artículos de Wikipedia, para ambos *dataset* se consideró artículos externos, siendo que las fuentes utilizadas para construir el *dataset* de entrenamiento no fueron las mismas para el *dataset* de prueba, esto con el fin de aumentar la variedad textual en los artículos a analizar.

Para el *dataset* de entrenamiento se extrajeron diversos contenidos respecto a determinadas palabras y cada uno de estos contenidos son asociados a un posible hiperónimo. Finalmente se le asigna un valor 0 en caso se entienda que la palabra asociada al contenido no es un hiperónimo o un valor 1 en caso sí lo sea. La estructura del *dataset* de entrenamiento se puede apreciar en el Cuadro 12 y tiene la siguiente estructura:

- **Id:** Id del registro
- **Artículo:** Contenido de una palabra selecciona a analizar.
- **Categoría:** Palabra asociada al contenido del artículo.

- **Es Categoría:** Valor que indica si la palabra indicada en la columna "Categoria" es en realidad un hiperónimo relacionado al contenido del artículo.

id	articulo	categoria	es categoria
1	El perro es un animal mamífero y cuadrúpedo que fue domesticado hace unos 10000 años y que actualmente convive con el hombre como una mascota Su nombre científico es Canis lupus familiaris	roedores	0
2	El perro es un animal mamífero y cuadrúpedo que fue domesticado hace unos 10000 años y que actualmente convive con el hombre como una mascota Su nombre científico es Canis lupus familiaris	animales	1
...			
323	Definimos edificio a la Construcción de grandes dimensiones fabricada con piedras ladrillos y materiales resistentes que está destinada a servir de vivienda o de espacio para el desarrollo de una actividad humana	vivienda	0
324	Definimos edificio a la Construcción de grandes dimensiones fabricada con piedras ladrillos y materiales resistentes que está destinada a servir de vivienda o de espacio para el desarrollo de una actividad humana	urbanismo	1

Cuadro 12. Artículos utilizados para data de entrenamiento

Tal como se puede apreciar en el cuadro anterior, las 2 primeras entradas corresponden a un artículo sobre "perro", siendo que la primera tiene un valor 0 en la columna "es categoría" debido a que no pertenece a esta jerarquía ("roedores"), en cambio para la palabra "animales" tiene un valor 1 ya que sí pertenece. Así, construido el *dataset* de entrenamiento, se obtuvieron más de 500 registros que serán separados posteriormente en datos de entrenamiento y validación para la generación del modelo a utilizar para la futura predicción en los datos de prueba.

El segundo paso fue definir un *dataset* de prueba. La estructura es casi similar al *dataset* de entrenamiento, lo que cambió en este caso fue la extracción del contenido de distintas palabras, los cuales como se indicó anteriormente, fueron extraídos de otras fuentes no utilizadas previamente. Asimismo, se incluyeron contenidos de palabras no incluidas en los datos de entrenamiento. Por ejemplo, en el *dataset* de entrenamiento se incluyó contenido de la palabra "perro" y en el *dataset* de prueba se incluyó contenido tanto para la palabra "perro" como para "gato", esto con el fin de analizar si el modelo también distingue esta palabra la cual tiene un contenido parecido y

ver si acierta o no en algunas categorías que para ese palabra no se entrenaron.

id	artículo	categoría
1	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	roedores
2	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	animales
...		
52	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	roedores
53	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	animales
...		
706	La vivienda es una edificación cuya principal función es ofrecer refugio y habitación a las personas protegiéndolas de las inclemencias climáticas y de otras amenazas.	construcción
707	La vivienda es una edificación cuya principal función es ofrecer refugio y habitación a las personas protegiéndolas de las inclemencias climáticas y de otras amenazas.	edificación

Cuadro 13 Artículos utilizado para data de prueba

En el cuadro 13 se puede apreciar que la estructura del *dataset* de prueba es similar al *dataset* de entrenamiento, la única diferencia es el contenido en la columna "artículo". Como se puede apreciar tanto para los datos de entrenamiento y de prueba se le asignan diversas categorías, siendo éstas palabras aquellas que se encuentran con menor distancia en los *Word Embedding* generados, por ejemplo para el contenido relacionado a la palabra "perro", se utilizaron aquellas palabras más cercanas a ésta (Ver cuadro 8 y 9) como categorías. Se tomó un promedio entre 10 a 15 categorías por contenido a analizar. También se puede apreciar que no necesariamente todas las palabras más cercanas son categorías del contenido de la palabra a analizar. Por ejemplo, "animal" si es una categoría de la palabra "perro", pero "roedor" no lo es. Así, en la data de entrenamiento se representa con 0 a la relación entre el contenido de la palabra "perro" y la categoría "roedores", y con 1 la relación con "animales".

Finalmente, definido tanto los datos de entrenamiento y prueba, se procedió a construir el modelo utilizando una red neuronal recurrente LSTM. La idea fue comparar 2 modelos, el primero que utilice como input el Word *Embedding original* y el segundo que utilice como input el *Word Embedding* modificado en base a duplicidad. Ambos modelos se entrenaron con el mismo *dataset* de entrenamiento y ambos predijeron si la categoría definida por cada artículo en el dataset de prueba correspondía o no. El objetivo final es ver quien tiene mayor precisión. Ambos modelos construidos utilizaron los mismos parámetros, tales como el número de células (se hicieron varias pruebas entre 175 y 275 células), la activación (relu) , optimizador Adam o precisión como métrica). El modelo generado con el *Word Embedding* modificado tuvo una precisión un poco mayor que el modelo que utiliza el *Word Embedding* general luego de realizar el proceso de entrenamiento en el cual se dividió en 90% como data de entrenamiento y 10% de validación con batches de 100 en 100 y utilizando desde 5 hasta 150 epocas.

La mayor precisión que se obtuvo para el modelo con el *Word Embedding* original fue de 87%.

```
early_stopping =EarlyStopping(monitor='val_loss', patience=3)
bst_model_path = STAMP + '.h5'
model_checkpoint = ModelCheckpoint(bst_model_path, save_best_only=True, save_weights_only=True)

hist = model.fit([data_1_train, data_2_train], labels_train,
                 validation_data=([data_1_val, data_2_val], labels_val, weight_val),
                 epochs=150, batch_size=100, shuffle=True,
                 class_weight=class_weight, callbacks=[early_stopping, model_checkpoint])

model.load_weights(bst_model_path)
bst_val_score = min(hist.history['val_loss'])
```

```
WARNING:tensorflow:From C:\Users\UTIAJO4\AppData\Local\Continuum\anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.
py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 900 samples, validate on 102 samples
Epoch 1/150
900/900 [==============================] - 14s 16ms/step - loss: 0.8386 - acc: 0.5211 - val_loss: 0.7406 - val_acc: 0.6078
Epoch 2/150
900/900 [==============================] - 13s 14ms/step - loss: 0.6769 - acc: 0.6333 - val_loss: 0.7404 - val_acc: 0.6078
Epoch 3/150
900/900 [==============================] - 15s 17ms/step - loss: 0.5511 - acc: 0.7400 - val_loss: 0.6837 - val_acc: 0.6078
Epoch 4/150
900/900 [==============================] - 15s 17ms/step - loss: 0.4608 - acc: 0.7856 - val_loss: 0.7296 - val_acc: 0.5882
Epoch 5/150
900/900 [==============================] - 15s 17ms/step - loss: 0.3813 - acc: 0.8478 - val_loss: 0.7369 - val_acc: 0.6471
Epoch 6/150
900/900 [==============================] - 15s 17ms/step - loss: 0.3272 - acc: 0.8789 - val_loss: 0.7231 - val_acc: 0.6667
```

Figura 5. Resultados al entrenar el modelo con el Word Embedding original

Mientras que para el modelo con el *Word Embedding* modificado se obtuvo 93% luego de varios procesamientos.

```
Instructions for updating:
Use tf.cast instead.
Train on 900 samples, validate on 102 samples
Epoch 1/150
900/900 [==============================] - 17s 19ms/step - loss: 0.8782 - acc: 0.5533 - val_loss: 0.5458 - val_acc: 0.7451
Epoch 2/150
900/900 [==============================] - 16s 18ms/step - loss: 0.6347 - acc: 0.6689 - val_loss: 0.4700 - val_acc: 0.7745
Epoch 3/150
900/900 [==============================] - 17s 19ms/step - loss: 0.5034 - acc: 0.7622 - val_loss: 0.3875 - val_acc: 0.8627
Epoch 4/150
900/900 [==============================] - 19s 21ms/step - loss: 0.4309 - acc: 0.8167 - val_loss: 0.3749 - val_acc: 0.8529
Epoch 5/150
900/900 [==============================] - 19s 21ms/step - loss: 0.3423 - acc: 0.8556 - val_loss: 0.3169 - val_acc: 0.8824
Epoch 6/150
900/900 [==============================] - 19s 21ms/step - loss: 0.3119 - acc: 0.8767 - val_loss: 0.2599 - val_acc: 0.9020
Epoch 7/150
900/900 [==============================] - 20s 23ms/step - loss: 0.2804 - acc: 0.8922 - val_loss: 0.2331 - val_acc: 0.9020
Epoch 8/150
900/900 [==============================] - 20s 22ms/step - loss: 0.2362 - acc: 0.9033 - val_loss: 0.2302 - val_acc: 0.9020
Epoch 9/150
900/900 [==============================] - 20s 22ms/step - loss: 0.2168 - acc: 0.9178 - val_loss: 0.2706 - val_acc: 0.9020
Epoch 10/150
900/900 [==============================] - 21s 24ms/step - loss: 0.1996 - acc: 0.9333 - val_loss: 0.2400 - val_acc: 0.9020
Epoch 11/150
900/900 [==============================] - 21s 23ms/step - loss: 0.1893 - acc: 0.9300 - val_loss: 0.2732 - val_acc: 0.9020
```

Figura 6. Resultados al entrenar el modelo con el Word Embedding modificado

Finalmente, luego de generados los modelos, se procedió a realizar la predicción (mediante model.predict) de las categorías utilizando el *dataset* de prueba, siendo que el modelo generado con el *Word Embedding* modificado acierta mas tanto cuando una categoría en realidad no es una categoría (se acerca más al valor 0) así como cuando una categoría sí lo es (se acerca más al valor 1). También se pudo apreciar que para las palabras nuevas (como "gato" que no fue incluido en el entrenamiento) también tiene un mucho mejor nivel de precisión siendo que en algunos casos acierta cuando una categoría si corresponde mientras que el modelo con el *Word Embedding* original considera lo contrario. Se distinguió también mínimos falsos positivos predecidos siendo esto para ambos modelos. En el Cuadro 14 se reporta una muestra de los resultados obtenidos de forma comparativa de un total de más de 900 entradas analizadas.

test_id	artículo	categoría	v. real	original	ajustado
1	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	roedores	0	0.110043	0.01267
2	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	animales	1	0.556793	0.895941

3	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	herbívoros	0	0.077567	0.002958
4	Pese a que todos los perros actuales tienen un antepasado común hoy en día se conocen alrededor de 800 razas distintas con tamaños y fisonomías muy diferentes y originadas a partir de la selección artificial por parte de los seres humanos	rumiantes	0	0.138494	0.004529
...					
52	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	roedores	0	0.271476	0.018592
53	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	animales	1	0.733544	0.92304
54	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	herbívoros	0	0.155239	0.005445
55	El gato es uno de los animales más conocidos y distribuidos alrededor de todo el mundo este por lo general se asocia como una mascota	rumiantes	0	0.283409	0.007423
...					
78	El oso es significado de valentía paz resurrección poder benevolencia soberanía maternidad paciencia e introspección. Aunque el oso es omnívoro el oso prefiere una dieta sencilla con aperitivos dulces como bayas. Además al oso le gusta tumbarse en los lugares soleados en su tiempo libre.	roedores	0	0.127467	0.014687
79	El oso es significado de valentía paz resurrección poder benevolencia soberanía maternidad paciencia e introspección. Aunque el oso es omnívoro el oso prefiere una dieta sencilla con aperitivos dulces como bayas. Además al oso le gusta tumbarse en los lugares soleados en su tiempo libre.	animales	1	0.596851	0.918646
80	El oso es significado de valentía paz resurrección poder benevolencia soberanía maternidad paciencia e introspección. Aunque el oso es omnívoro el oso prefiere una dieta sencilla con aperitivos dulces como bayas. Además al oso le gusta tumbarse en los lugares soleados en su tiempo libre.	herbívoros	0	0.105998	0.004676
81	El oso es significado de valentía paz resurrección poder benevolencia soberanía maternidad paciencia e introspección. Aunque el oso es omnívoro el oso prefiere una dieta sencilla con aperitivos dulces como bayas. Además al oso le gusta tumbarse en los lugares soleados en su tiempo libre.	rumiantes	0	0.156285	0.00671
...					
749	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	usuario	0	0.105295	0.064544
750	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	uso	0	0.259547	0.020774

751	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	software	0	0.261169	0.053699
752	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	interfaz	0	0.346508	0.010078
753	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	hardware	1	0.587016	0.968927
754	Si no tienes ni idea de qué es un gadget ni para qué sirven checa este video: 1. La palabra gadget es una palabra tecnológica que se refiere a dispositivos que tienen un propósito y una función específica y práctica para la vida cotidiana de quien los usa.	dispositivo	1	0.902623	0.957701
...					
799	La gripe es causada por un virus de la influenza. La mayoría de las personas contraen la gripe cuando inhalan gotitas provenientes de la tos o los estornudos de alguien que tenga gripe	síntoma	0	0.398372	0.084014
800	La gripe es causada por un virus de la influenza. La mayoría de las personas contraen la gripe cuando inhalan gotitas provenientes de la tos o los estornudos de alguien que tenga gripe	bacteriemia	1	0.357736	0.273295

Cuadro 14. Muestra comparativa de resultados a partir del modelo LSTM

4.9. Análisis de resultados

A continuación se procedió a comparar los resultados obtenidos tanto con el modelo que utiliza el *Word Embedding* original y el modelo con el *Word Embedding* ajustado (*Word Embedding* modificado en base a duplicidad), considerando la pertenencia a la clase Hiperónimo.

En la figura 9, se presenta el diagrama de cajas considerando los resultados de ambos modelos: en el caso de aquellos datos que no son hiperónimos, se aprecia que ambos modelos existe un asimetría positiva (el sesgo de los datos se encuentran en la izquierda) que nos da indicios que la carga de datos se acerca al valor 0, sin embargo visualmente se aprecia que el valor mediano del modelo con en *Word Embedding* ajustado se acerca más al valor 0 en

comparación del modelo con el *Word Embedding* original, el cual nos da indicios que el modelo con el *Word Embedding* ajustado es mejor que el modelo con el *Word Embedding* original.

Por otro lado, también se analizó para los datos que son hiperónimos, encontrándose que en dichos casos la asimetría cambió en comparación de los datos no hiperónimos y que ahora es una asimetría negativa (sesgo de datos en el lado derecho) con concentración de datos que se acercan a 1. Adicionalmente se aprecia que el valor mediado del modelo con el *Word Embedding* ajustado se acerca más a dicho valor frente el valor mediano del modelo con el *Word Embedding* original.

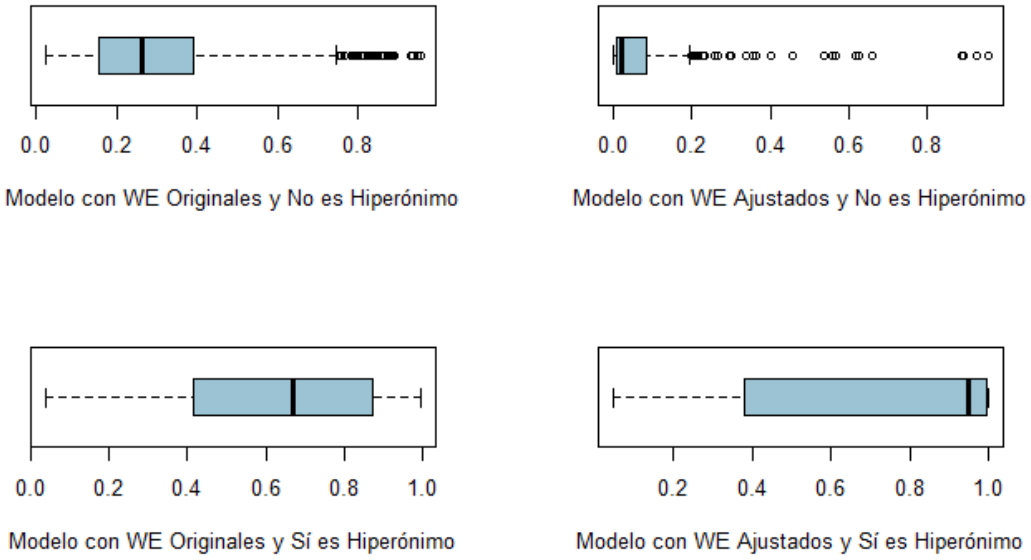


Figura 7. Diagrama de cajas con las comparaciones de los modelos con WE Original y Modificado, considerando los hiperónimos. Elaboración: Software R

En la figura 10, se presenta las curvas ROC, donde se evidencia que para el modelo con el *Word Embedding* original el valor es 0.752 y para el modelo con el *Word Embedding* ajustado el valor es 0.851, dado que para determinar el mejor modelo tiene que ser aquel que se acerca al valor de 1, se evidencia que el modelo con el *Word Embedding* ajustado tiene mejor predicción que el Modelo con el *Word Embedding* original.

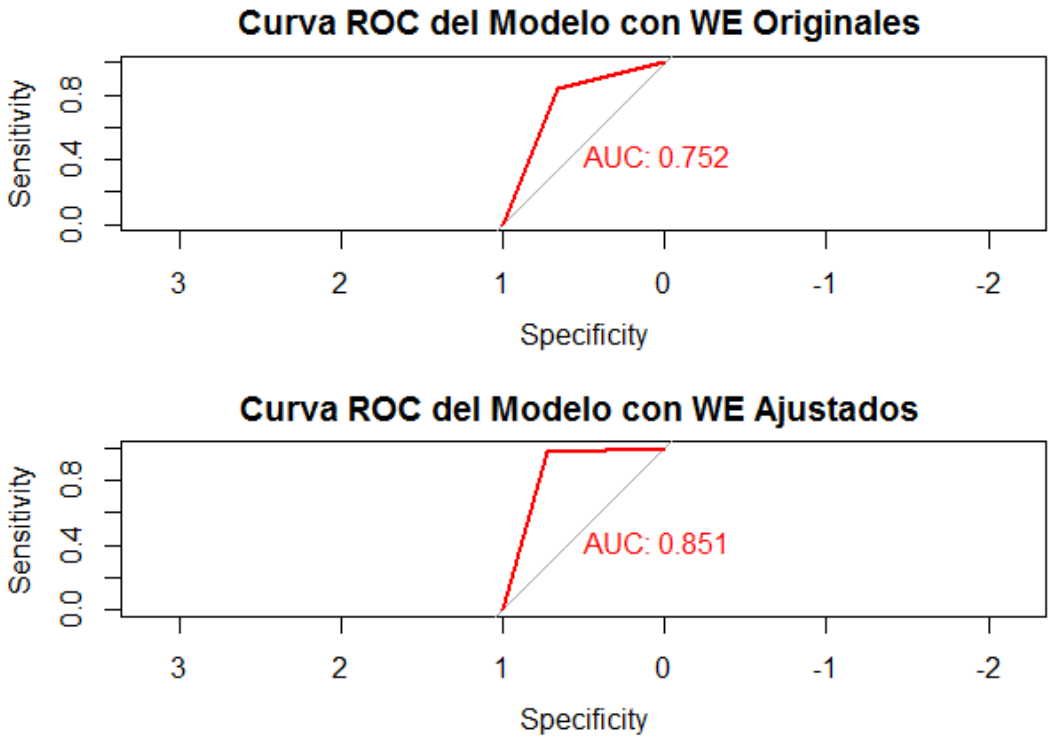


Figura 8. Comparación de las Curvas ROC de los Modelos con WE Originales y Ajustados. Elaboración: Software R

Capítulo 5

Conclusiones y Trabajos Futuros

5.1. Conclusiones

En este trabajo recopilamos un corpus de pares hiperónimos – hipónimos ya validados y un gran corpus de texto como lo es Wikipedia para la generación de *Word Embeddings* orientados a la hiperonimia. El objeto del presente trabajo fue evaluar la detección de categorías dado como entrada un contenido de texto que puede ser un artículo, blog u otros. Para ello nos concentramos primero en definir bien qué datos utilizar y como utilizarlos. Para esto definimos distintas maneras de manipulación del corpus de Wikipedia original, siendo el que mejor resultados obtuvo aquel en el que duplicamos los artículos en caso un hipónimo existente en nuestro corpus se encuentre en el texto del artículo en cuestión.

Cabe indicar que también se realizaron pruebas para el objetivo de manipulación directa del *Word Embedding* original analizando previamente los patrones de distancia de cada par de hipónimo – hiperónimo tanto en su totalidad de dimensión como descartando 1 o más dimensiones del total para verificar si alguna predominaba sobre otra. Al visualizar la gran dispersión resultante, se dejó este objetivo de lado y se enfocó más en el objetivo de manipulación del corpus previamente a la generación del nuevo Word Embedding.

Para la evaluación de la precisión de nuestros resultados, se evaluó nuestro *Word Embedding* generado en base a múltiples duplicidades contra el contenido de texto de determinados artículos y comparándolo con el *Word Embedding* original. Para esto se crearon 2 modelos de redes neuronales recurrentes (LSTM), en el que cada uno trabajaba con su respectivo *Word Embedding*. Luego de entrenados ambos modelos y posteriormente utilizados para la predicción de resultados, se concluyó que la precisión obtenida

utilizando el *Word Embedding* ajustado es mejor que la del Word Embedding original.

5.2. Trabajos futuros y recomendaciones.

Como trabajo futuro se propone optimizar el modelo utilizado en la red neuronal recurrente para clasificación de texto para poder aumentar el valor de precisión obtenido, ya sea ajustando los parámetros del modelo o ajustando aún más el *Word Embedding* ajustado que será el input para la capa *Embedding* de la red.

Asimismo se recomienda proponer nuevas técnicas de reemplazo que permitan mejorar la performance del algoritmo debido a que es necesario contar con muchos recursos de sistema para poder manipular y generar el *Word Embedding* tal como se está realizando actualmente.

Bibliografía

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In HLT-NAACL, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. EMNLP, 2014.

Neha Nayak, Neha Nayak. 2015. In Learning Hyperonyms over Word Embeddings. Student technical report.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building Watson: An overview of the DeepQA project. AI magazine, 31(3):59–79, 2010.

Bill MacCartney. Natural language inference. PhD thesis, Stanford University, 2009.
Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In ACL, Stroudsburg, PA, USA, 2002.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1993. Introduction to WordNet: An On-line Lexical Database

Ivan Vulic, Nikola Mrksic, Roi Reichart, Diarmuid O'Seaghdha, Steve Young, and Anna Korhonen. 2017. Morph-fitting: Fine-tuning word vector spaces with simple language-specific rules

Rion Snow, Daniel Jurafsky, Andrew Y. Ng . Learning syntactic patterns for automatic hypernym discovery. 2005.

Morin, Emmanuel & Jacquemin, Christian. (2004). Automatic Acquisition and Expansion of Hypernym Links. Computers and the Humanities. 38. 363-396. 10.1007/s10579-004-1926-2.

Jose Camacho-Collados, Claudio Delli Bovi, Luis Espinosa-Anke, Sergio Oramas, Tommaso Pasini, Enrico Santus, Vered Shwartz, Roberto Navigli, and Horacio Saggion.

2018. SemEval-2018 Task 9: Hypernym Discovery. In Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), New Orleans, LA, United States. Association for Computational Linguistics

Dorantes, M & Pimentel, Alejandro & Sierra, Gerardo & Bel-Enguix, Gemma & Molina, Claudio. (2018). Extracción automática de definiciones analíticas y relaciones semánticas de hiponimia-hiperonimia con un sistema basado en patrones lingüísticos. Linguamática. 9. 10.21814/lm.9.2.257.

Ortega, R., C. Aguilar, L. Villaseñor, M. Montes and G. Sierra. 2011. Hacia la identificación de relaciones de hiponimia/hiperonimia en Internet. Revista Signos 44(75): 68-84

LeCun, Yann & Bengio, Y & Hinton, Geoffrey. (2015). Deep Learning. Nature. 521. 436-44. 10.1038/nature14539.

Dias, G., Raycho M. and Guillaume C. (2008) Mapping General-Specific Noun Relationships to WordNet Hypernym/Hyponym Relations. Springer-Verlag Berlin Heidelberg. pp. 198-212

Escobar, R. (2014). Redes neuronales, procesos cognoscitivos y análisis de la conducta. Revista Internacional de

Fernández, Ana & Vázquez, Gloria & Fellbaum, Christiane. (2008). The Spanish Version of WordNet 3.0. 10.1515/9783110211818.3.175.

Billy Chiu, Gamal Crichton, Sampo Pyysalo, and Anna Korhonen. 2016. How to train good word embeddings for biomedical NLP. In Proceedings of BioNLP'16

Ziqian Zeng, Yichun Yin, Yangqiu Song, and Ming Zhang. Socialized word embeddings. In IJCAI, pages 3915–3921, 2017.

Li, Q., Shah, S., Fang, R., Liu, X., Nourbakhsh, A. Data Sets: Word Embeddings Learned from Tweets and General Data. 2017. [114] Globerson , A., Chechik, G.,

Chi-Yen Chen, Wei-Yun Ma, "Embedding Wikipedia Title Based on Its Wikipedia Text and Categories," *International Conference on Asian Language Processing*, December 2017.

Athiwaratkun, B.; Wilson, A. G.; and Anandkumar, A. 2018. Probabilistic fasttext for multi-sense word embeddings. arXiv preprint arXiv:1806.02901.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2017. Recent trends in deep learning based natural language processing. arXiv preprint arXiv:1708.02709

M. Marwa, A. Chaibi, and H. Ghezala, "Comparative Study of word embeddings methods in topic segmentation", International Conference on Knowledge Based and Intelligent Information and Engineering Systems, pp. 340-349, 2017.

Attia, Mohammed & Maharjan, Suraj & Samih, Younes & Kallmeyer, Laura & Solorio, Thamar. (2016). Detecting Semantic Relations via Word Embeddings.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1199–1209, Baltimore, Maryland.

R. Bamler and S. Mandt. Dynamic word embeddings. In ICML, 2017.

N. Campoy Garrido, "Relaciones semánticas entre las palabras: hiponimia, sinonimia, polisemia, homonimia y antonimia. los cambios de sentido", Contribuciones a las Ciencias Sociales. 2010. [En línea] disponible en www.eumed.net/rev/cccss/08/ncg.htm.

Koerhsen Will. "Using a Recurrent Neural Network to Write Patent Abstracts". 2018. [En linea] disponible en https://towardsdatascience.com/recurrent-neural-networks-by-example-in-python-ffd204f99470