UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA
ESCUELA DE POSGRADO
UNIDAD DE POSGRADO DE LA FACULTAD DE INGENIERÍA
DE PRODUCCIÓN Y SERVICIOS

CLASIFICACIÓN DE SENTIMIENTOS USANDO MODELOS PROBABILÍSTICOS, DEEP LEARNING Y WORD EMBEDDINGS PARA TEXTOS CORTOS EN ESPAÑOL

Tesis presentada por el bachiller:

DISRAELI FAUSTO ARI MAMANI

Para optar el Grado Académico de Maestro en Ciencias: **Informática**, con mención en **Tecnologías de Información**.

DR. JOSÉ EDUARDO OCHOA LUNA.
ASESOR

AREQUIPA-PERÚ

2019

Agradecimentos

En primer lugar deseo agradecer a Dios por haberme guiado a lo largo de estos años de estudio y darme la gracia de tener a mis seres queridos brindándome su apoyo y también permitirme conocer personas en el camino, que hicieron posible este trabajo.

También agradezco de forma muy especial a mi asesor Dr. José Eduardo Ochoa Luna por su apoyo y guía en la elaboración de esta tesis, su conocimientos, orientación y paciencia ha sido fundamentales para mi formación como investigador.

También deseo agradecer a mi familia por su apoyo incondicional, que contribuyó en alcanzar mis objetivos.

Durante el desarrollo de este trabajo se presentaron muchos inconvenientes que pudieron llegar a ser motivo para rendirse, momentos en los que se esperada y anhelada mejores resultados y estos no parecían llegar. Pero en esos momentos resaltó dentro de mi vida esa persona que siempre me ha acompañado, mi compañera, mi amiga, mi consejera y mi complemento hacia la felicidad; mi novia Lesli. Le agradezco de la manera más sincera, e infinitamente por su ayuda, y por su incontable apoyo en mi vida.

Deseo agradecer a la Universidad Nacional de San Agustín por brindarme la oportunidad de ser parte del Programa de Maestría. Agradezco al personal administrativo de la Universidad por la atención y sobretodo el apoyo brindado.

RESUMEN

CLASIFICACIÓN DE SENTIMIENTOS USANDO MODELOS PROBABILÍSTICOS, DEEP LEARNING Y WORD EMBEDDINGS PARA TEXTOS CORTOS EN ESPAÑOL

Disraeli Fausto Ari Mamani

Arequipa-Perú

Asesor: José Eduardo Ochoa Luna.

La clasificación de sentimientos se refiere al dictamen o juicio que se le atribuye a un texto. Esta tarea es algo que las personas realizan naturalmente al leer un contenido. Por el crecimiento de las redes sociales y la cantidad de datos que se generan diariamente en la web, podemos encontrar opiniones sobre noticias, productos, personas, etc. Con tanta información, la capacidad humana para clasificarla se convierte en una tarea muy difícil de realizar. Por eso se han propuesto diversos tipos de clasificadores automáticos que realizan esta tarea.

En esta investigación se desarrolló diversas técnicas que permiten el análisis de textos cortos, desde clasificadores probabilísticos, que mostraron un rápido desempeño y buenos resultados, hasta clasificadores basados en aprendizaje profundo, con los que se consiguió resultados comparables al estado del arte. Además, se utilizaron características no supervisadas basadas en *word embeddings* para modelar el contexto. Los algoritmos resultantes permiten clasificar textos cortos en el idioma español.

Palabras clave

Aprendizaje automático, análisis de opiniones, campos condicionales aleatorios, aprendizaje profundo, Representaciones de palabras

ABSTRACT

CLASIFICACIÓN DE SENTIMIENTOS USANDO MODELOS PROBABILÍSTICOS, DEEP LEARNING Y WORD EMBEDDINGS PARA TEXTOS CORTOS EN ESPAÑOL

Disraeli Fausto Ari Mamani

Arequipa-Perú

Advisor: José Eduardo Ochoa Luna.

Sentiment analysis refers to the opinion or judgment attributed to a text. This task is something that people naturally do when reading a content. For the growth of social networks and the amount of data that is generated daily in the web, we can find opinions on news, products, people, etc. With so much information, the human capacity to classify it becomes a very difficult task to perform. That is why a large number of automated classifiers have been proposed to perform this task.

In this research, several techniques were developed that allow the analysis of short texts, from probabilistic classifiers, which showed fast performance and good results, to classifiers based on deep learning, with which allowed us to obtain comparable state-of-the-art results. In addition, unsupervised features such as it word embeddings were used to provide context. The classification is mainly focused on short texts for the Spanish language.

Keywords

Machine Learning, Sentiment Analisys, Conditional Random Fields, Deep Learning, Word Embeddings.

Lista de Figuras

Lista de Tablas

Lista de Abreviaturas y Siglas

NLP	Natural Language Processing
SA	Sentiment Analisys
ABSA	Aspect Based Sentiment Analisys
CRF	Conditional Random Field
HMM	Hidden Markov Model
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSA	Latent Semantic Análisis
SVM	Support Vector Machines
GloVe	Global Vectors for Word Representation
GPU	Unidad de procesamiento gráfico
NPMI	Normalized Pointwise Mutual Information
PGM	Probabilistic Graphical Models

Índice general

Capítulo 1

Introducción

En los últimos años el uso de las redes sociales se ha incrementado, despertando cada vez más el interés de los investigadores por el estudio del análisis de opiniones en redes sociales y micro *b*logging, proponiendo técnicas y modelos para determinar la polaridad de textos [68].

Los foros y redes sociales han logrado que muchos usuarios expresen sus opiniones sobre una diversidad de temas. Esta gran cantidad de opiniones puede ser de mucha utilidad para vendedores, fabricantes, políticos, empresas, etc, que encuentran en estas redes la manera de conocer información valiosa y la necesidad de monitorear estos comentarios. Esto ha contribuido a que la minería de opiniones o el análisis de sentimientos tengan un papel importante en los últimos años [20].

La naturaleza de las redes sociales, de crecimiento constante, hace necesario analizar y procesar grandes cantidades de texto. La mayor parte de investigaciones relacionadas al análisis de sentimientos en redes sociales han sido creadas para el idioma inglés [68]. Para el idioma español, estas tareas no poseen un alto desempeño hasta el momento, por lo cual su aplicación ha sido limitada [59].

La tarea de analizar textos es muy complicada, incluso para anotadores humanos, pues existen diferencias relacionadas a la subjetividad cuando se califica un texto. La interpretación personal de una persona es muy distinta a la de otra, ya sea por factores culturales, experiencias, etc. La tarea se complica más si trabajamos con

textos cortos y si estos poseen errores en escritura o significado como ocurre en las redes sociales, donde los usuarios colocan sus comentarios en Facebook [1] y Twitter [2]. ¿Qué significado darle a una palabra mal escrita y que no posee significado semántico?, ¿Qué significado darle al uso de vulgarismos en las opiniones?, estas son solo algunas de las preguntas y retos que afronta el análisis de sentimientos y como esta tarea se complica según el dominio donde es aplicado.

Tradicionalmente el problema de análisis de sentimientos ha sido abordado por dos enfoques principales. El primer enfoque se basa en técnicas de aprendizaje computacional [68] que consisten en entrenar un clasificador teniendo un algoritmo de aprendizaje supervisado, haciendo uso de una colección de textos anotados. Cada texto es representado habitualmente por un vector de palabras (*B*ag of Words), n-gramas o *s*kip-grams [33], esto combinado con otras características para poder modelar la estructura de un texto. Este enfoque usa manejo de intensificadores, negación, ironía, y otras técnicas dependiendo del idioma usado. Los clasificadores más populares están basados en SVM [79] (*S*upport Vector Machines), *N*aive Bayes [29] y otros. Recientemente se está haciendo uso de técnicas más avanzadas como LSA (*L*atent Semantic Análisis) y de *Deep Learning*[8].

El enfoque basado en aprendizaje computacional en algunos casos se comporta como una caja negra. Depende mucho de las características modeladas, la cantidad de datos etiquetados con los que fue entrenado y la capacidad de los algoritmos para modelar los datos. A pesar de estas dependencias, este enfoque ha obtenido el mejor desempeño en esta tarea [85].

El segundo enfoque está basado en aproximaciones semánticas [89] que hace uso de diccionarios de términos con polaridad. En este enfoque se procesa el texto dividiéndolo en palabras y eliminando características que no poseen significados, para posteriormente ser normalizadas y lematizadas [24]. Después, se comprueba la aparición de palabras del diccionario de términos en el texto para determinar su clasificación. El enfoque basado en aproximaciones semánticas [89] es el más simple

[1] https://www.facebook.com/

[2] https://twitter.com/?lang=es

y basa su exactitud en la construcción de un *l*exicon (diccionarios de términos), lo más detallado posible y que con muchos términos podría ayudar a que tengamos un modelo más preciso; sin embargo construir un *l*exicon, requiere mucho tiempo y trabajo. Este *l*exicon debería ser creado dependiendo del dominio, por lo que para cada nuevo dominio se tendría que crear un *l*exicon distinto, lo que lo hace poco adaptable.

El presente trabajo, usa el enfoque de aprendizaje computacional para realizar el análisis de sentimientos en textos cortos. En particular se usan dos abordajes: Modelos Gráficos Probabilísticos que han mostrado buenos resultados en la tarea de clasificación como POS[64] y NER[43], el segundo abordaje usa *Deep Learning* con *W*ord Embeddings como características, que son usadas en diversas tareas del Procesamiento de Lenguaje Natural (PLN)[18], por ser representaciones útiles que abstraen mucha información textual y que posteriormente conducen a un mejor rendimiento. Asimismo, se hace un estudio comparativo a fin de encontrar el clasificador óptimo.

1.1 Motivación y contexto

Realizar análisis de sentimientos para el idioma Inglés es una tarea que se viene desarrollando hace años y que posee diversos modelos y herramientas que permiten decidir sobre la orientación positiva / negativa de textos [36, 34]. Sin embargo, para el idioma español no existe una gran cantidad de modelos desarrollados, incluso los conjuntos de datos disponibles para experimentar son escasos, lo que dificulta la construcción y entrenamiento de modelos.

El uso de redes sociales se ha convertido en una fuente extensa de información para diferentes temas, por esta razón las estrategias para extraer información se han perfeccionado en los últimos años. Esto motiva a encontrar algoritmos para procesar estas opiniones que se encuentran en la Web, ya que la tarea manual de etiquetarlos por sentimiento sería inviable, dado el flujo gigantesco de datos.

Una de las redes sociales más usadas hoy en día es Twitter [3], que posee millones de cuentas registradas, en más de 190 países en el mundo y que genera un tráfico y creación de datos inmensos a cada hora. Por ello es necesario diseñar modelos que entiendan el lenguaje humano y puedan dar una clasificación satisfactoria a esta gran cantidad de datos que se genera diariamente. Su clasificación automática, permitiría conocer el punto de vista de los usuarios respecto a algún producto, servicio, persona u otro tema en especial a nivel comercial, político, etc.

La tarea de clasificar polaridades de texto, análisis de sentimientos, por el momento no llega a una exactitud cercana al 100%. Este resultado se debe al contexto, idioma, tipo de lenguaje, entre muchas otras características. Ni los seres humanos somos capaces de realizar esta tarea a un 100% . Esto se puede comprobar haciendo que distintas personas clasifiquen textos y los resultados no serían los mismos, por lo cual llegar a una exactitud real es aún una tarea muy complicada. Además de esto debemos agregar la complejidad de cada idioma, que puede poseer frases complicadas, conocimiento implícito, ironías y otras alteraciones semánticas que son reflejadas en el texto.

1.2 Planteamiento del problema

La tarea de análisis de sentimientos, implica determinar el sentimiento que expresa un texto respecto a un tema, esta clasificación puede recibir las etiquetas de positivo, negativo o neutro. La principal característica de los textos a clasificar son las palabras que poseen. Las pocas entradas encontradas en textos cortos dificulta encontrar las características óptimas para la clasificación de los mismos. Además, esta escasez ocasiona que el texto pueda tener poco contexto y resulte difícil de clasificar incluso manualmente.

Por otro lado, las investigaciones sobre análisis de sentimientos se han desarrollado en gran parte para el idioma inglés. Esto es una limitante importante cuando se aborda otro idioma como el español, ya que no existen la misma variedad de *Data-*

[3] https://twitter.com/?lang=es

sets para realizar el entrenamiento de modelos. Cabe resaltar que las características que se extraen varían de un idioma a otro.

Asimismo, los *Datasets* existentes poseen pocos datos y clases desbalanceadas. Esto en el contexto de aprendizaje computacional, crea problemas de *Overfitting* que es el efecto de sobre entrenar un algoritmo y la Alta Varianza que se produce por la desproporción de clases en los *Datasets*.

1.3 Objetivos

1.3.1 Objetivo general

Proponer un clasificador entre técnicas basadas en modelos gráficos probabilísticos y *Deep Learning*, que permita clasificar la polaridad de los sentimientos en textos cortos, para el idioma español, con resultados comparables al estado del arte.

1.3.2 Objetivos específicos

- Proponer y seleccionar las características principales a ser observadas en el idioma español.
- Experimentar con características basadas en Word Embeddings para mejorar la clasificación. [42]
- Comparar el desempeño de clasificadores basados en Conditional Random Fields y *Deep Learning* para clasificar las opiniones, usando conjuntos de datos estándares
- Realizar casos de estudios con conjuntos de datos propios.

1.4 Contribuciones

La principal contribución de esta tesis consiste en proponer un clasificador capaz de detectar la polaridad de *Tweets* en español mediante el análisis comparativo de

diversos clasificadores. Asimismo, obtener resultados comparables con el estado del arte.

Esta búsqueda por el mejor clasificador ha llevado a la publicación de dos artículos en conferencias internacionales que tienen *proceedings* indexados en *Scopus*:

- *"Word Embeddings and Deep Learning for Spanish Twitter Sentiment Analysis"* en SIMBig 2018 [4].
- *"Deep Neural Network Approaches for Spanish Sentiment Analysis of Short Texts"* en IBERAMIA 2018 [86].

1.5 Organización de la tesis

La tesis está organizada como sigue:

- **El capítulo 2**, introduce los términos básicos y conceptos necesarios para entender el presente trabajo.
- **El capítulo 3**, describe los principales trabajos de investigación relacionados al área de análisis de sentimientos, en particular para el idioma español.
- **El capítulo 4**, describe el uso de *Conditional Random Fields* (CRF) como clasificador de textos cortos. También describe el uso de *Word Embeddings* y sus modificaciones como características adicionales en el clasificador. Esta constituye la primera propuesta para clasificar Textos cortos en español.
- **El capítulo 5**, describe el uso de *Recurrent Neuronal Network* (RNN), *Convolutional Neuronal Network* (CNN) y la combinación de CNN-RNN como clasificadores de textos cortos. Esta constituye la segunda propuesta para clasificar Textos cortos en español.
- **El capítulo 6**, describe los resultados de aplicar los clasificadores y la interpretación de los resultados obtenidos.

- **El capítulo 7**, describe las conclusiones finales al haber realizado la clasificación de textos cortos con los enfoques propuestos.

Capítulo 2

Conceptos previos

Para lograr el objetivo principal, es necesario el desarrollo de definiciones y conceptos. En este capítulo se exponen estos conceptos previos como el procesamiento de lenguaje natural (NLP), el preprocesamiento, modelos gráficos probabilísticos, *Deep Learning*, *w*ord embeddings entre otros.

2.1 *Word Embeddings*

Los *Word Embeddings* son un conjunto de técnicas para el modelado y aprendizaje del lenguaje natural, donde las palabras de un vocabulario son asignadas a vectores de números reales de longitud n, de tal manera que se puede colocar palabras en el espacio. Esto implica una transformación de las palabras a un modelo matemático en el espacio con varias dimensiones.

Los métodos para generar la representación de palabras incluyen redes neuronales, modelos probabilísticos y la representación explícita en términos del contexto en el que aparecen las palabras [50].

Se ha demostrado que el uso de *Word Embeddings* como representación de palabras y frases, aumenta el rendimiento en tareas de procesamiento del lenguaje natural, análisis de sentimientos [82] y análisis sintáctico [83].

En la Figura 2.1 se muestra la representación de algunas palabras en el espacio, y

se puede ver como las palabras con significado semántico similar, son representados como puntos cercanos.

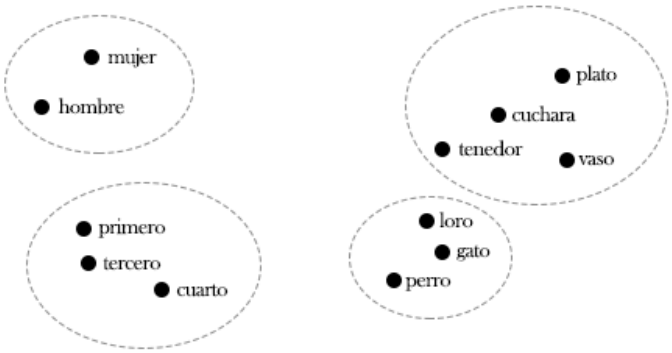

Figura 2.1: Palabras organizadas en campos vectoriales

El aprendizaje de estos vectores se realiza mediante el uso de textos sin etiquetar, mediante la predicción de contextos y modelos no supervisados [28]. Los principales trabajos en *w*ord embeddings y usados en el siguiente trabajo son los siguientes:

2.1.1 Word2vec [34]

Propone una implementación basada en bolsa de palabras (*B*ag of Words) y *s*kip-grams para calcular las representaciones de palabras en vectores, toma un texto como entrada y produce vectores de palabras, para la construcción de *s*kip-grams se toma como entrada la palabra actual w y se predice la distribución de probabilidad del contexto de palabras (anterior y posterior). [28]

El modelo skip-gram usa una red neuronal de tres capas, la entrada es la representación dispersa de la palabra w, luego se proyecta la palabra a la siguiente capa para predecir el contexto de la palabra, y así de esta manera se aplica este mismo modelo para cada palabra del texto, a fin de descubrir contextos.

Dada una secuencia de entrenamiento w_1, w_2, w_3, ... , w_T; cada w_i representa una palabra en el texto, luego se realiza una maximización, usando la función *Softmax*.

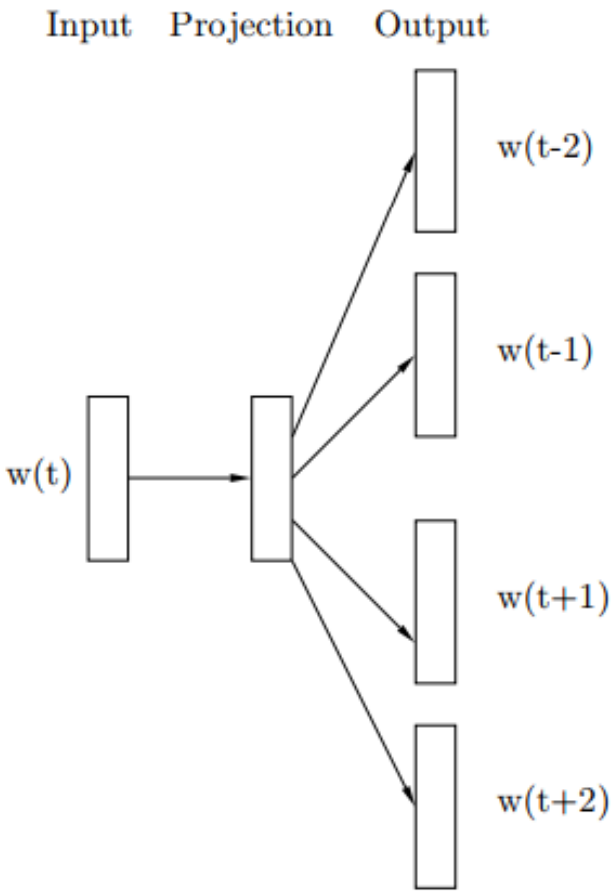

Figura 2.2: Modelo *Skip-gram*[60]

$$p(w_s|w_e) = \frac{\exp(e'^{T}_{w_s} v_{w_e})}{\sum_{w=1}^{W} \exp(e'^{T}_{w} v_{w_e})} \tag{2.1}$$

Como resultado del cálculo de *W*ord Embeddings se obtiene un vector por palabra en el vocabulario con el que fue entrenado.

2.1.2 GloVe: Global Vectors for Word Representation [42]

Es una algoritmo de aprendizaje no supervisado para obtener vectores para palabras. Usa para este proceso estadísticas agregadas de co-ocurrencias de palabra-palabra en un conjunto de entrenamiento.

Se tabulan en una matriz las frecuencias con las que las palabras co-ocurren entre si en un conjunto de entrenamiento. El algoritmo recorre una sola vez esta matriz

y aún así puede poseer un alto costo computacional, pero es un costo inicial único. Las herramientas proporcionadas por este modelo automatizan la recopilación y preparación de estadísticas de co-ocurrencia para la entrada del modelo.

GloVe es un modelo *l*og-bilinear (lineal en cada variable cuando las demás variables son fijas) con un objetivo ponderado de mínimos cuadrados [42]. En la Figura 2.3 representa las probabilidades reales de co-ocurrencia de un corpus de 6 millones de palabras. Como se comprueba en el gráfico, el hielo ocurre más frecuentemente con el sólido que con el gas, mientras que el vapor ocurre más frecuentemente con el gas que con el sólido. Ambas palabras coinciden con el agua ya que es una de sus propiedades compartidas, y ambas no coinciden con la moda que es una palabra con poca frecuencia para el contexto de las palabras.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figura 2.3: Descripción general del modelo [42]

Características semánticas de los vectores:

- **Nearest neighbors:** Usa la distancia euclidiana o la similitud del coseno entre dos vectores, es la manera de medir la similitud lingüística y semántica de las palabras correspondientes. En ciertas ocasiones esta distancia relaciona palabras con poco sentido de relación pero que son relevantes. por ejemplo:

  1. Frog
  2. Frogs
  3. Toad
  4. Litoria
  5. Leptodactylidae
  6. Rana
  7. Lizard

8. Eleutherodactylus

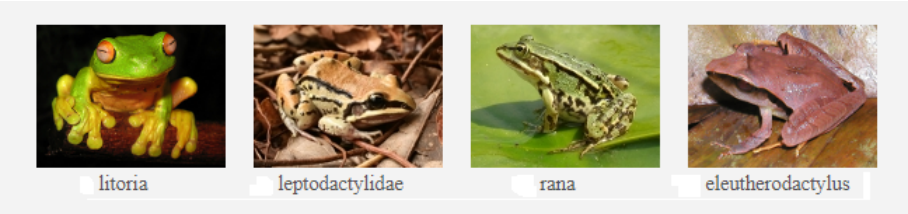

Figura 2.4: Uso de *N*earest neighbors para determinar el significado de vectores según agrupamiento [42]

- **Linear substructures:** Las medidas con *N*earest neighbors producen un escalar que indica la relación entre dos palabras, pero dos palabras casi siempre exhiben relaciones complejas, por ejemplo: Hombre y mujer pueden ser considerados similares por *N*earest neighbors, sin embargo estas palabras con opuestas. Para esta tarea es necesario asociar más de un par de palabras. Glove está diseñado para capturar diferencias vectoriales al yuxtaponer dos palabras opuestas en cuadrantes vectoriales distintos.

2.1.3 FasText[61]

FasText es una biblioteca de representaciones de palabras. Consta de modelos y vectores de palabras para el idioma inglés y de otros 157 idiomas, entre ellos el idioma español. Es el producto de un aprendizaje eficiente de representaciones de palabras y clasificación de oraciones.

En su sitio oficial disponemos de vectores de palabras pre-entrenados, algunos de ejemplos de estos vectores son:

- **wiki-news-300d-1M.vec.zip:** Vectores de 1 millón de palabras entrenados de Wikipedia 2017, corpus de la base de web UMBC y conjunto de datos de noticias statmt.org (16B tokens).
- **wiki-news-300d-1M-subword.vec.zip:** Vectores de 1 millón de palabras entrenados con palabras claves de Wikipedia 2017, conjunto de datos de la base de datos web UMBC y statmt.org (16B tokens).

- **crawl-300d-2M.vec.zip:** Vectores de 2 millones de palabras entrenados en Common Crawl (600B tokens).

La primera linea en los archivos contiene la cantidad de palabras del vocabulario por el tamaño de los vectores (300 dimensiones). El formato esta compuesto por el nombre, seguido de su vector y separados por un espacio, Además están ordenados por frecuenta descendente.

2.1.4 Generación de los Word Embeddings

Para generar los embeddings se uso un corpus de wikipedia[91] disponible en ***https://dumps.wikimedia.org/eswiki/latest/*** con un peso de 2710338407 bytes y generado el 17 de diciembre del 2016.

Este corpus es un xml, que fue convertido a texto plano con un script de Python ***process_wiki.py***, generando un archivo ***wiki.es.text*** de 3.6 Gigabytes, dicho archivo posteriormente generó los vectores de palabras, haciendo uso de los algoritmos de Word2vect[34] y de Glove [42]. Los vectores de FastText se obtuvieron pre entrenados de su sitio oficial[26]. Los detalles de los vectores por algoritmos, son detallados en la siguiente lista:

- **Glove** contiene 894442 vectores de 300 dimensiones.
- **Word2vect** contiene 1000653 vectores de 300 dimensiones.
- **FastText** contiene 985667 vectores de 300 dimensiones

2.2 Preprocesamiento

2.2.1 Procesamiento de Lenguaje Natural

Es un campo de la inteligencia artificial y lingüística que estudia el lenguaje humano y lo intenta modelar para hacer interpretaciones propias. Investiga mecanismos para la comunicación entre personas y máquinas mediante el uso del len-

guaje natural. En la actualidad existe una revolución de algoritmos de aprendizaje computacional para el procesamiento de lenguaje natural. Las principales tareas desarrolladas por estos algoritmos[73] son:

- **Machine Translation [92]:** Traducción de un lenguaje a otro.
- **Sentiment Analysis [21]:** Determinar el sentimiento de un texto.
- **Semantic Role Labeling [93]:** Asignar roles semánticos.
- **Summarization [2]:** Resumir textos.
- **Part-of-Speech Tagging [1] (POS):** Asignar etiquetas gramaticales.
- **Aspect Based Sentiment Analisys [55] (ABSA):** Extraer aspectos o temas tratados en un texto y su sentimiento.
- **Named Entity Recognition [77] (NER):** Reconocimiento de entidades en un texto.

2.2.1.1 Análisis de sentimientos

El análisis de sentimientos, o minería de opiniones se refiere al uso de técnicas del procesamiento del lenguaje natural para identificar información subjetiva de un texto, es decir extraer su connotación positiva o negativa. En términos de aprendizaje computacional, es una tarea de clasificación de textos para determinar si un texto es positivo, negativo o neutro.

El análisis de sentimientos, intenta determinar la actitud que posee un texto con respecto a un tema y lenguaje como se ilustra en la Figura 2.5.

En la Tabla 2.1 se presenta la clasificación de sentimientos, donde las etiquetas para esta clasificación son positiva, negativa y neutra, simbolizadas por P, N y NEU respectivamente.

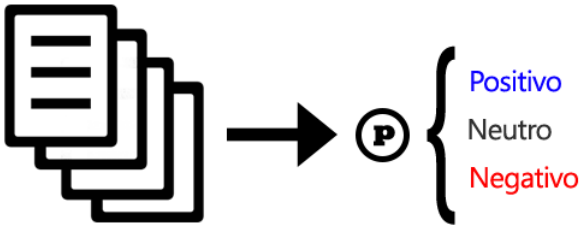

Figura 2.5: Interpretación de texto [37]: Un texto puede interpretarse por el sentimiento que expresa, calificándolo como positivo, negativo o neutro.

2.2.1.2 Twitter

Es una red social basada en *m*icroblogging que permite mandar mensajes de texto con un tamaño máximo de 140 caracteres, que reciben el nombre especial de *T*weet. Mediante estos textos se puede:

- Publicar textos cortos de infinidad de temas: Todo tipo de información es publicable en esta red social siempre que cumpla con las condiciones y normas de Twitter.
- Tener conversaciones públicas: Tener conversaciones y seguir otras conversaciones.
- Seguir temas de interés: Seguir temas y compartirlos con solo usar el carácter reservado. *h*ashtag.
- Seguir a personas: formar una red de personas y seguirlas.

2.2.1.2.1 Características

Twitter posee características únicas que lo diferencian de las demás redes sociales, estas características son:

- *Hashtags:* Se usan para referirse a un tema.
- Símbolo @: Se refiere a una entidad o usuario.

Clasificación de textos	
Texto	Polaridad
• Ser madre es ser líder • Ser madre es ser amorosa • Ser madre te permite hacer sacrificios por los demás con más naturalidad	P P P
• A veces mimar demasiado a los hijos • Dar una mala crianza • Temor al momento de dar a luz	N N N
• No es algo negativo, pero creo que todas tenemos derecho a elegir sin miedo al "qué dirán" • Es consustancial a ser mujer aunque en algunos casos, muy raro, algunas no sienten lo mismo • Es un "rol" que cualquiera puede cumplir, tanto hombre como mujer	NEU NEU NEU

Tabla 2.1: Textos clasificados por sentimiento

- *Emoticons:* Reflejan sentimientos directos.
- Palabras reservadas: RT que sirve para publicar temas de otras personas o páginas.
- *Unicode* que representa símbolos.

2.2.1.2.2 Problemas

Esta red social genera una colosal cantidad de datos por hora, es una fuente de datos casi inagotable para la tarea de procesamiento de lenguaje natural, pero estos textos poseen algunos problemas definidos por algunas investigaciones [45]. Esto en resumen indica que los datos válidos o que reflejan algún significado válido, es de menos del 40% de los existentes. El resto de datos posee palabras sin sentido, *s*pam, mensajes repetidos, auto promoción y texto que no valdría la pena clasificar. Esto se puede comprobar en la Figura 2.6 donde la información sin sentido, *spam* y

las controversias representan la mayor parte del total de información. Por lo que la elección de los datos que se deben usar para entrenar el modelo y clasificar, podría ser uno de los puntos a tener en cuenta en la tarea de análisis de sentimientos en esta red social.

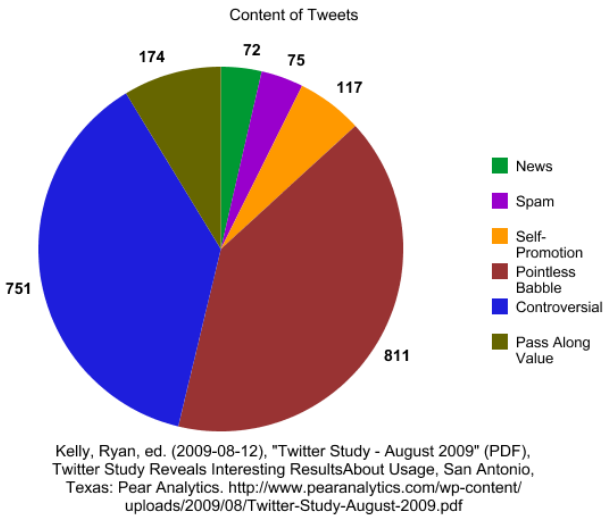

Figura 2.6: Proporción de Tweets por contenido

2.2.2 Preprocesamiento

El texto antes de entrar a un clasificador debe ser tratado, esto se refiere a hacer cambios de palabras por *t*okens significativos. Eliminar algunas palabras o símbolos. Tratar problemas relacionados al idioma y muchas otras técnicas de preprocesamiento que se han realizado en el estado del arte. Algunas de estas técnicas son las siguientes:

- Eliminación de características que poco significado aportan, como son: los links, correos electrónicos, signos de puntuación, conectores y palabras o letras que no aportan un significado semántico.
- En Twitter en específico, sustituir las referencias de Hashtags y convertirlas a palabras sin # o eliminarlas. Esto también ocurre en las referencias a usuarios, es decir eliminar el carácter @ y transfórmalo a palabras sin este símbolo.

- *E*longated words que se refiere a buscar palabras con letras repetidas y eliminar estas letras Ej. goooool − > gol[67].

2.3 Clasificadores usados

2.3.1 Clasificadores basados en Modelos gráficos probabilísticos

Los modelos gráficos probabilísticos hacen uso de la teoría de grafos y la teoría de probabilidades, permiten modelar la distribución de probabilidad conjunta de grandes colecciones de variables aleatorias.

Se ha aplicado con éxito en el diagnóstico médico, procesamiento de lenguaje natural, *traffic analisys*, *message decoding*, *computational vision*, *speech recognition*, *Segmentation of images*, etc[87].

2.3.1.1 Teoría de la probabilidad

La teoría de probabilidad proporciona un modelo matemático, para la interpretación de fenómenos aleatorios[12]. Esta teoría es la manera de cuantificar la incertidumbre y darle propiedades. Utilizamos las probabilidades para describir el resultado de un experimento incierto con un espacio de muestra E. La probabilidad condicional permite interpretar el experimento con información parcial. Es decir, la probabilidad de que ocurra un evento A, dado que ya hemos observado el evento B. P(A) se llama probabilidad a priori de A y P(A|B) se llama la probabilidad a posteriori de A dado B[70]. Entonces la probabilidad condicional es definida en la Ecuación 2.2.

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2.2}$$

2.3.1.2 Teorema de Bayes

El teorema de Bayes define la probabilidad de un evento A, dado que un evento B ha ocurrido posteriormente [88], como se define en la Ecuación 2.3.

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{[P(A) \cdot P(B|A)] + [P(\bar{A})] \cdot P(B|\bar{A})} \tag{2.3}$$

- **Hallazgo:** Determinación del valor de una variable, a partir de un dato (una observación, una medida, etc.).
- **Evidencia:** Conjunto de todos los hallazgos disponibles en un determinado momento o situación.
- **Probabilidad a priori:** Probabilidad de una variable cuando no existen hallazgos.
- **Probabilidad a posteriori:** Probabilidad de una variable dada la evidencia h: $P(x|h)$.

2.3.1.3 Naive Bayes

La regla de Bayes aplicada a la clasificación de texto, asume que todas las características son independientes de la clase. Para un documento d y una clase c se define la regla de Bayes en la Ecuación 2.4.

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \tag{2.4}$$

El clasificador *Naive Bayes* usa con conjunto de documentos d pertenecientes a un conjunto de clases $C = \{c_1, c_2, \ldots, c_j\}$ para poder predecir si c $\in$ C. Usa una representación simple del documento como Bolsa de palabras, que se refiere a un diccionario de palabras distintas y la representación de frecuencias como se puede ver el Tabla 2.2.

Palabra	**Contador**
Grandioso	2
Adorable	2
Recomendado	1
Chistoso	1
Divertido	1
...	...

Tabla 2.2: Representación de frecuencias

2.3.1.4 Conditional Random Fields

Los *c*onditional Random Fields (CRF) también llamados campos aleatorios de Markov. [84], son un método probabilístico para etiquetar datos estructurados. Hacen uso de las Redes de Markov para su representación. Las redes de Markov [48] permiten definir los *c*onditional Random Fields. Una Red de Markov es un modelo gráfico probabilístico que tiene una estructura de grafo no dirigido G=(V,E), donde los nodos son las variables. Las conexiones entre los nodos representan la interacción probabilística entre las variables. Las redes de Markov, son modelos generativos, denotan la distribución de probabilidad conjunta de un conjunto de variables. Por otro lado, los *c*onditional Random Fields son un tipo de red de Markov discriminativa, pues modelan directamente la probabilidad de la secuencia correcta de etiquetas condicionada por las observaciones, $P(E|X)$.

CRF es una modelo gráfico probabilístico definido similarmente como un conjunto de factores φ_1 (D_1), ..., φ_m (D_m) pero con variación de que la distribución de probabilidades resolverá la distribución condicional P(Y|X) donde Y y X son variables aleatorias, entonces la distribución de probabilidad de CRF se define en las Ecuaciones 2.5, 2.6 y 2.7 de la siguiente forma:

$$P(Y|X) = \frac{1}{Z(X)}\bar{P}(Y,X) \tag{2.5}$$

$$\bar{P}(Y,X) = \prod_{i=1}^{m} \phi_i(D_i) \tag{2.6}$$

$$Z(X) = \sum_{Y} \bar{P}(Y,X) \tag{2.7}$$

El modelo de CRF puede ser representado como en la Figura 2.7, donde los datos a etiquetar están representados por los nodos S_1, S_2 hasta S_n y las características por los nodos C_1, C_2 hasta C_n que pueden ser más de una, en este caso solo se aprecia una. También se puede observar las dependencias entre las variables S entre ellas y las variables C con las variables S.

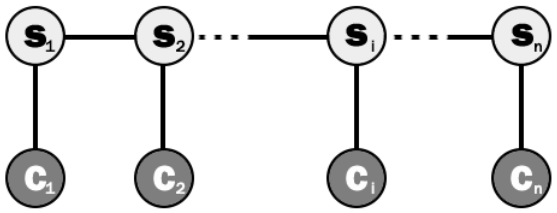

Figura 2.7: Modelo CRF lineal para etiquetar secuencias

2.3.2 Clasificadores basados en Deep Learning

La inteligencia artificial en sus inicios, abordó problemas difíciles para los seres humanos pero relativamente sencillos para las computadoras, ya que eran problemas que se podían resolver con matemáticas formales. Su verdadero desafió consiste en resolver aquellos problemas que no se pueden describir formalmente, problemas que el ser humano resuelve de manera intuitiva, como por ejemplo reconocer caras o palabras [40]. Por ello se requiere aprender conceptos complicados y construirlos de los mas simples, apilados uno sobre otro, por ello el término de profundo. Al añadir mas capas y unidades dentro de una capa, se puede representar funciones de complejidad creciente.

Existe muchas arquitecturas y variantes de *Deep Learning*, algunas de estas son Convolutional Neural Network y Recurrent Neural Network. Los algoritmos de Deep

Learning han sido aplicados en visión computacional[16], procesamiento de lenguaje natural[57] y reconocimiento de señales[3] obteniendo excelentes resultados[46] [41].

2.3.2.1 Redes Neuronales

Las redes neuronales están basadas en funcionamiento y estructura a una neurona del tejido nervioso. Trata de simular como funciona una neurona en el cerebro, dando respuesta a estímulos. La estructura de una Red Neuronal se puede ver en la Figura 2.8.

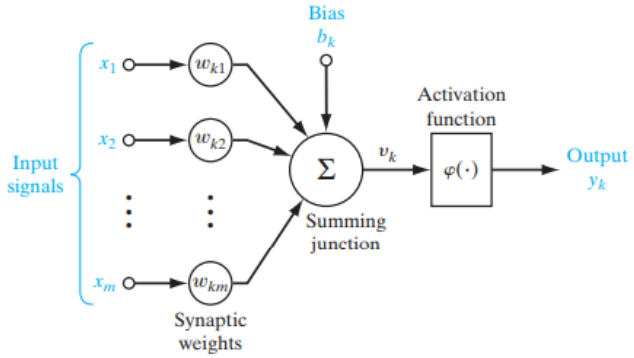

Figura 2.8: Modelo de una neurona no lineal[30]

El rango normalizado de salida de una neurona, se describe en un intervalo cerrado entre [0,1] o [-1,1]. También incluye un sesgo (b_k), que aumenta o disminuye la entrada de la función de activación. Matemáticamente se puede describir una neurona artificial con las Ecuaciones 2.8 y 2.9. Donde $x_1, x_2, \ldots, x_m$ son las entradas; $w_{k1}, w_{k2}, \ldots, w_{km}$ son los pesos de la neurona k; u_k corresponde a la salida del combinador lineal de las señales; b_k representa el sesgo; φ es la función de activación y y_k es la salida de la neurona [30].

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2.8}$$

$$y_k = \varphi(u_k + b_k) \tag{2.9}$$

El sesgo b_k aplica una transformación a la salida u_k del combinador lineal dado

por la Ecuación 2.10.

$$v_k = u_k + b_k \tag{2.10}$$

Las redes neuronales artificiales basan su comportamiento al modelo de aprendizaje propuesto por Donald O. Hebb [31]. La estructura de una red neuronal consta de un receptor, sumador, activador y salida.

- **capa receptora:** La capa receptora es la encargada de capturar las señales, entradas o estímulos. Estas entradas van acompañadas de un peso ω_i.
- **capa de Sumador:** Se encarga de realizar la suma ponderada de las entradas.
- **capa de activación:** La función de activación, se encarga de definir un umbral de aprendizaje.
- **capa de Salida:** Es la interpretación de la salida con respecto a la anterior.

2.3.2.2 Convolutional Neural Network (CNN)[47]

Son una de las técnicas mas usadas en *Deep Learning* y que mayormente se aplica en imágenes. Fueron planteadas en los años 90, pero en ese entonces no existía la arquitectura computacional necesaria para su ejecución. En cada convolución se detectan elementos de mayor complejidad como se puede ver en la Figura 2.9, donde vemos que en la primera capa se detectan los bordes, en la segunda capa se detectan elementos más complejos como partes de un rostro, y en la tercera capa se detectan rostros. Los principales problemas en los que se emplean las CNN son:

- Generación automática de color.
- Generación de píxeles.
- Descripción de contenido.
- Estimación de pose en tiempo real.

- análisis de comportamiento en tiempo real.
- Traducción de imágenes.
- Carros autónomos, Tesla / Google.
- Reconocimiento de voz.
- Reconocimiento de sonidos.

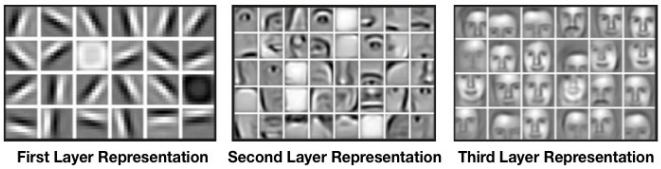

Figura 2.9: Complejidad de elementos detectables por capa, imagen extraída de StackExchange [35]

2.3.2.2.1 Convolución

La convolución consiste en pasar una matriz de $f \times f$ sobre otra de mayor tamaño $n \times n$, La matriz $f \times f$ recorre toda la matriz $n \times n$ y la reduce y convierte en otra. Por ejemplo en la detección de bordes, luego de la convolución se puede obtener los bordes.

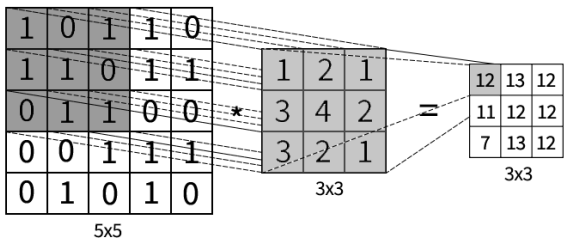

Figura 2.10: Complejidad de elementos detectables por capa, imagen extraída de *stackexchange*[35]

En la Figura 2.10 se representa una matriz de 5×5, a esta se aplica un filtro de 3×3, se multiplican los valores de esta matriz con la original y se suman para obtener la convolución, este filtro de desliza sobre toda la matriz y genera la matriz de convolución.

En ese sentido, una CNN está compuesta de varias capas de convoluciones con funciones de activación (*ReLu* o *tanh*) aplicadas a los resultados. Durante el entrenamiento, los filtros son calculados en función a la tarea a realizar.

Normalmente las CNNs son aplicadas sobre imágenes, para poder aplicarlo al texto, en lugar de píxeles de imagen, tendremos oraciones o palabras también representadas en una matriz. Ahora vamos a desarrollar algunas de las características principales de la CNN.

2.3.2.2.2 Padding

El *padding* se refiere a añadir información adicional a la matriz de entrada, Esta hiper parámetro se usa para preservar la mayor cantidad de información, también se puede lograr con esto que a pesar de los saltos (*Strides*), que veremos mas adelante, las dimensiones permanezcan inmutables y se retenga la información sin pérdidas. En la Figura 2.11 se puede ver un *padding* de una unidad rellenada con ceros y esto ocasiona que la matriz de entrada aumente de tamaño, con esto se incrementa la matriz de salida, las dimensiones son definidas por la Ecuación 2.11.

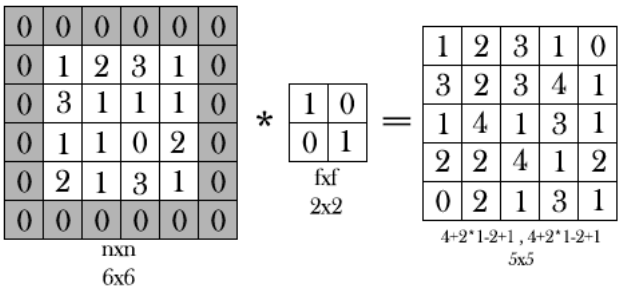

Figura 2.11: Ejemplo de *Stride* de 2 unidades.

$$n + 2p - f + 1, n + 2p - f + 1 \tag{2.11}$$

2.3.2.2.3 Strides

Se refiere al tamaño de salto o zancada que se debe desplazar el filtro, un tamaño de *Stride* muy grande hace que el filtro se aplique menos veces y esto ocasiona que la salidas mas pequeñas, también puede ocasionar que se comporte como una red neuronal parecida a un árbol. En la Figura 2.11 se define un tamaño de *Stride* de 2, lo que hace que el filtro recorra la matriz de entrada dando saltos de 2, tanto para el eje X y el eje Y. lo que crea una salida de 2x2. Las dimensiones de salida están definidas por la Ecuación 2.12.

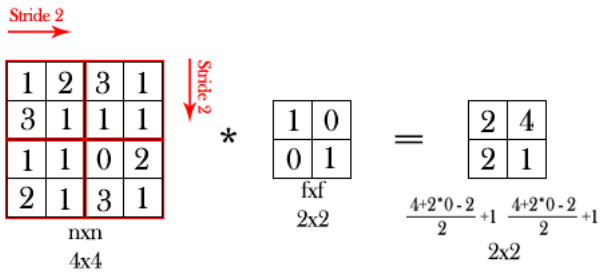

Figura 2.12: Ejemplo de *Stride de 2 unidades.*

$$\left\lfloor \frac{n+2p-f}{s}+1 \right\rfloor, \left\lfloor \frac{n+2p-f}{s}+1 \right\rfloor \tag{2.12}$$

2.3.2.2.4 ReLU (Rectified Linear Units)

Se refiere a la capa de activación que está después de la capa de convolución. las capas ReLU funcionan mejor que tangente hiperbólica o función sigmoidea debido a que son capaces de entrenar mucho más rápido debido a la eficiencia computacional. Esta función también ayuda a que las capas inferiores entrenen a mayor velocidad lo que previene el gradiente de fuga. La capa ReLU [32] aplica la Ecuación 2.13 que se aplica a todos los valores de entrada cambiando las activaciones negativas a 0.

$$f(x) = max(0, x) \tag{2.13}$$

2.3.2.2.5 Pooling Layers

Estas son capas de agrupación, la forma más usada es la agrupación de máximos como en la Figura 2.13. Esto crea una matriz de salida de tamaño fijo por lo que podemos usar entradas y filtros de tamaño variable y siempre se obtendrá las mismas salidas para el clasificador. También reduce la dimensionalidad pero manteniendo la información más importante. La capa de *Pooling* también reduce la cantidad de parámetros o ponderaciones a un 75% con esto se reduce el coste del cálculo, y también controla el *overfitting*[81], que resulta cuando un modelo no es capaz de generalizar su respuesta en los conjuntos de prueba.

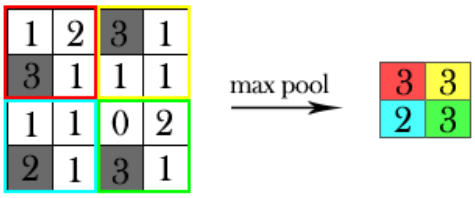

Figura 2.13: *Max pooling* con *stride* de 2.

2.3.2.3 Recurrent Neural Network (RNN)[63]

En una Red Neuronal Recurrente (RNN) las conexiones entre nodos forman un grafo dirigido secuencial. Esto le permite modelar la temporalidad y lo hace aplicable en tareas cono reconocimientos de voz, traducción de texto, etc.

La idea de Redes Neuronales recurrentes es agregar una conexión a cada red neuronal que haga referencia a un estado oculto anterior. Cada estado se calcula con la Ecuación 2.14.

$$h_i = \begin{cases} \tanh(W_{xh}X_t + W_{hh}h_{t-1} + b_h) & \text{si } t \geq 1 \\ 0 & \text{otro caso} \end{cases} \tag{2.14}$$

Como se puede ver en la Ecuación 2.14 se añade un estado $W_{hh}h_{t-1}$ conectándose al paso de tiempo t, esto podemos verlo gráficamente en la Figura 2.14.

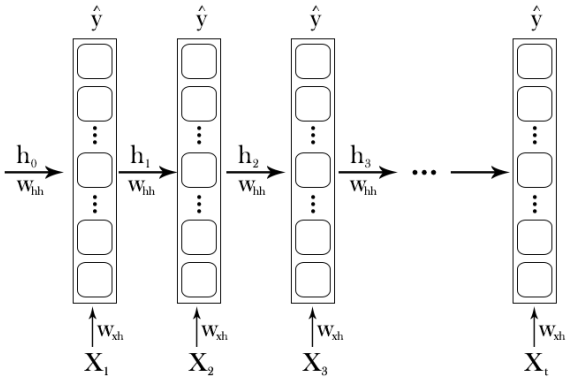

Figura 2.14: Recurrent Neural Network

2.4 Consideraciones finales

En este capítulo, se han desarrollado los conceptos necesarios para comprender este trabajo, como procesamiento de lenguaje natural, análisis de sentimientos, la red social fuente de datos (*Twitter*), las técnicas de preprocesamiento y los modelos que hacen de uso de *Deep Learning*, *Conditional Random Fields* y *Word Embeddings*.

En el siguiente capítulo, se va a revisar los trabajos propuestos para solucionar el problema de análisis de sentimientos en textos cortos para el idioma español.

Capítulo 3

Trabajos relacionados

Las investigaciones realizadas para idioma español existen en menor cantidad a las realizadas para el idioma inglés. Sin embargo se desarrollaron varias propuestas, aquí se muestran los trabajos realizados usando el idioma español.

Las primeras investigaciones en español hacían uso de léxicos para clasificar la polaridad, en Martinez-Camara [58] integraron integraron estos léxicos, además de usar *Support Vector Machines*, *Naïve Bayes* y regresión logística Bayesiana. Los resultados mejoraron al combinar distintos léxicos con el uso de meta clasificadores. En esa investigación se hizo uso del conjunto de datos muchoCine [74].

En relación a textos cortos en español, la mayor parte de investigaciones en este idioma se han venido desarrollando en el *TASS: Workshop on Semantic Analysis at SEPLN*[38] que propone un conjunto de datos de Tweets en español clasificados en 4 polaridades, positivo, negativo, neutro y sin ninguna clasificación.

Las investigaciones más relevantes en el análisis de sentimientos para el idioma español son listadas a continuación y ordenadas por la técnica empleada:

3.1 *Support Vector Machines*

Las investigaciones que hicieron uso de *Support Vector Machines* SVM, se listan a continuación:

- *BittenPotato* [9] combina SVM junto con otros 3 clasificadores para realizar esta tarea como son *Adabost*, *Ramdom Forest* y regresión lineal. El preprocesamiento para la tarea se basa en remover algunas características que no aportan significado semántico como *URLs*, correos electrónicos, signos de puntuación, emoticones, espacios entre palabras y palabras repetidas, estas técnicas son en parte, aplicadas en nuestra propuesta. Convirtiendo al final estas características en vectores [71] antes de llegar a los clasificadores.

- En *Sentiment Analysis for Twitter* [66] combina SVM con generación de *Q-gram*. Para esta tarea el procesamiento uso *Part of speech tagging* (POS), Manejo de negación, Corrección de errores (vocales o letras repetidas 'ruidoooooo'), un conjuntos de 512 emoticones (positivos y negativo), @user y enlaces son eliminadas, Usa además *Freeling Tool for Spanish Language* [67].

- En *ELiRF-UPV* en TASS 2015: Análisis de Sentimientos en Twitter [51] realiza una adaptación de *Tweetmotif* [11] que es un descubridor de temas del inglés al español, combinando esto también con *Freeling* [67]. Cada vector es representado con un vector que contiene coeficientes de *tf-idf*. este trabajo obtuvo el mejor resultado para el *TASS 2016*.

- En *Aspect Based Sentiment Analysis of Spanish Tweets* [65] se enfoca en la tarea de ABSA, para ello realiza la extracción de características (*N-grams*, mayúsculas a minúsculas, POS, frecuencias, *Hashtags*, signos de puntuación, letras repetidas, manejo de negación y polaridad general.

- *Spanish Twitter Dataset Classification Combining Wide-Coverage Lexical Resources and Text Features* [7] desarrolla una clasificación basada en el *lexicon* español de *Lingmotif* [62], basados en experimentos con combinaciones de conjuntos de funciones y algoritmos de aprendizaje computacional basados en regresión logística y SVM.

3.2 *Deep Learning*

Las investigaciones que hicieron uso de técnicas y procedimientos basados en *Deep Learning* se listan a continuación:

Uno de los primeros enfoques que usaron *Deep Learning* para el análisis de sentimientos en español, fue propuesto en el taller del TASS en SEPLN en el 2015, con el nombre de [90]. Los autores presentaron una arquitectura que estaba compuesta por una capa RNN (células LSTM), una capa densa y una función sigmoidea como salida. El rendimiento sobre el conjunto de datos general fue pobre, 0.60 en términos de precisión (el mejor resultado fue 0.69 en TASS 2015).

El primer enfoque de la red neuronal convolucional (CNN) para el análisis del sentimiento español fue descrito por Segura-Bedmar et al [80]. Su modelo estaba compuesto por una sola capa convolucional, seguida de una capa de acumulación máxima y un clasificador de *Softmax* como capa final. Las representaciones de palabras se usaron de tres maneras: una aprendida desde cero y dos modelos *word2vec* pre-entrenados. En términos de precisión obtuvieron 0.64, lo que distaba mucho del mejor resultado (0.72 fue el mejor resultado en TASS 2016).

Otro enfoque de CNN para el análisis de sentimientos en español fue presentado por Paredes et al [69]. Siguiendo un procedimiento de tokenización y normalización, para después usar el modelo de representación de palabras *Word2vec*. Este modelo estaba compuesto por una capa convolucional 2D, una agrupación máxima y una capa final de *Softmax*. Se informó una medida F de 0.887 sobre un corpus no público de Twitter de 10000 tweets.

La mayoría de los enfoques de *Deep Learning* para el análisis del sentimiento español se presentaron en TASS 2017. A continuación se listan las principales investigaciones en el TASS 2017 que hicieron uso de *Deep Learning* :

- Análisis de Sentimiento de Tweets en español utilizando SVM y CNN [6] presenta clasificadores basados en SVM y Redes neuronales convolucionales (CNN), al igual que parte de nuestra propuesta, combina ambos clasificadores

para optimizar sus resultados. Además agrega a su enfoque el uso de representación de palabras (*word Embeddings*) [18]. Su preprocesamiento se basa eliminar Url's y menciones de usuario (@usuario), referir usuarios de Twitter a un token único, eliminación de caracteres y palabras repetidas, sustitución de interjecciones y como parte final todo el texto de entrada es convertido a minúscula. Esta técnica de sustitución de palabras clave, también fue aplicada en la etapa de preprocesamiento en nuestra propuesta.

- Análisis de Sentimientos en Twitter basado en *Deep Learning* [52] este trabajo se centra en explotar aproximaciones basadas en *Deep Learning*. Usa una adaptación de *Tweetmotif* [11]. Agrupa además fechas, signos de puntuación, números, direcciones web, *hashtaging* y menciones de usuario. Experimenta en diferentes topologías de redes neuronales, como el *Multilayer Perceptron* (MLP), redes neuronales recurrentes *Long Short Term Memory* (LSTM) y stacks de redes neuronales convolucionales (CNN).

- *FastText* como alternativa a la utilización de *Deep Learning* en conjuntos de datos pequeños [76], usa Bolsa de palabras (BoW) y Joulin [75] como alternativa del uso de *Deep Learning* para corpus pequeños donde no da resultados adecuados. Para esto usa *FasText* que es una librería sobre C++ para el aprendizaje eficiente de representación de palabras y clasificación de textos. Otorgando una adaptación más rápida y sin necesidad de ejecutar sobre GPU.

3.3 Consideraciones finales

En este capítulo se ha listado las principales investigaciones para la clasificación de texto, Muchas de estas investigaciones tiene en común el preprocesamiento y el uso del mismo conjunto de datos. Se ha listado las distintas configuraciones que han sido creadas usando *Support Vector Machine* y *Deep Learning*, que son las investigaciones mas relevantes y con mejores resultados.

En el próximo capítulo se desarrolla la clasificación de texto usando PGM con *Word Embeddings* para la clasificación de sentimientos en textos cortos.

Capítulo 4

Clasificación de sentimientos usando modelos probabilísticos y Word Embeddings para textos cortos en español

En este capítulo se desarrolla la primera propuesta, que consiste en usar Modelos Gráficos Probabilísticos como clasificador, combinado con *Word Embeddings* como características, para determinar la polaridad de textos cortos en español.

El clasificador que hace uso de modelos probabilísticos usa como entradas los textos cortos en español, estos textos pasan por un procedimiento de preprocesamiento que elimina las características que no aportan significado al clasificador. Este clasificador hace uso de 2 tipos de características: la primera está conformada por el texto, *Part of Speech* , *Chunking*, lematizacion, etc. La segunda que hace uso de *word Embeddings*, un aprendizaje no supervisado que representa las palabras en campos vectoriales. Estos pasos están representados en la Figura 4.1.

A continuación, se va a detallar cada paso que se siguió para la clasificación usando CRF, obviando el paso de texto de entrada, que se refiere al texto propiamente dicho.

Figura 4.1: Pasos para la clasificación usando CRF

4.1 Preprocesamiento

Para hacer uso de los Modelos Gráficos Probabilísticos en la clasificación de textos, se requiere hacer uso de algunas técnicas adicionales de preprocesamiento descritos en la Sección 2.2.2, y que solo son necesarias al hacer uso de esta propuesta. A continuación se listan las técnicas de preprocesamiento usadas en este clasificador:

- Se hace uso de un *Tokenizador* especial que estructura los *Tweets*. El *NLTK TweetTokenizer* [25] divide las oraciones en tokens, para después poder descartar los tokens que no aportan un significado al clasificador. Un ejemplo del funcionamiento podemos verlo en la Tabla 4.1.

- El siguiente paso consiste en eliminar URLs, links, extraer las características de Twitter y trasformarlas a solo palabras, es decir, eliminar los tokens # y @ de los usuarios y Hashtags. Asimismo, elimina símbolos *Utf-8*, como los conocidos emoticones que forman parte de los textos en las redes sociales y es sustituido por palabras clave. Además se elimina las palabras o símbolos repetidos en un texto, usando expresiones regulares, con esto se transforman palabras como “goooooool” en “gol”. Un ejemplo del funcionamiento podemos verlo en la Tabla 4.2.

- Las repeticiones de signos de puntuación también se eliminan, por ejemplo !!!! es transformado a solo !.

Entrada	Salida
No es que ahora no sea feliz, pero antes lo era más	No es que ahora no sea feliz , pero antes lo era más
@mariamira67 que nos regales con capturas de imagen de tus actividades ya no es sorpresa. Lo sorprendente es que tienes inivitado!	@mariamira67 que nos regales con capturas de imagen de tus actividades ya no es sorpresa . Lo sorprendente es que tienes inivitado !
Nunca he tenido esta sensación, ¿me la harías sentir? Es muy importante para mí Blas @BlasAuryn #followspree	Nunca he tenido esta sensación , ¿ me la harías sentir ? Es muy importante para mí Blas @BlasAuryn #followspree

Tabla 4.1: Ejemplo de TweetTokenizer [25]

- Parte del etiquetado de voz se ha utilizado para identificar adjetivos, verbos y palabras que denotan sentimientos.
- Palabras de baja y alta frecuencia son eliminados en algunos clasificadores.

4.2 Características del texto

Las características propias del texto, que son usadas como características para la clasificación, son la obtención de *POS tagging*, *Chunk Annotation*, e indicadores de sintaxis, que se detallan a continuación:

- Se Extraen características de los textos para obtener la categoría gramatical léxica (POS), y que esta sea una característica más para el clasificador.
- Uso de *T*ext Chunking que consiste en dividir el texto en palabras correlacionadas sintácticamente, y que esta sea otra característica para el clasificador. Para esta técnica es necesario usar estilos de anotación, que es la forma en la que se etiqueta cada elemento. Los estilos de anotación son los siguientes:
  - **Anotación IO [78]:** La Ecuación 4.1 muestra la anotación dentro (In-

Entrada	Salida
Ya tengo uno , pero ese es más cheto @aweamasome .	ya tengo uno , pero ese es más cheto aweamasome .
@DovakhiinMudo 9 por lo buenico que eres ☺	DovakhiinMudo número por lo buenico que eres feliz
#feliz septiembre .. es bonito retarse .. es increíble lo mucho que puedes aprender .. medirse con el obstáculo .. eres la joyita de la corona	feliz septiembre . es bonito retarse . es increíble lo mucho que puedes aprender . medirse con el obstáculo . eres la joyita de la corona

Tabla 4.2: Ejemplo de Extracción de características [25]

side) y fueras (Outside).

$$w_i = \begin{cases} I & \text{si } w_i \text{ es entidad,} \\ O & \text{si } w_i \text{ no es entidad} \end{cases} \tag{4.1}$$

– **Anotación BIO [19]:** La Ecuación 4.2 muestra la BIO que añade el inicio de la entidad (Beginning), agregado a la anotación anterior.

$$w_i = \begin{cases} B & \text{si } w_i \text{ es una entidad y la inicia,} \\ I & \text{si } w_i \text{ es una continuación de entidad, va después de B,} \\ O & \text{si } w_i \text{ no es entidad} \end{cases} \tag{4.2}$$

– **Anotación BILOU [49]:** La Ecuación 4.3 muestra la notacion BILOU que añade la Anotación de fin (Last), y cuando solo se posee una entidad (Unique), agregado a la anotación anterior.

$$w_i = \begin{cases} B & \text{si } w_i \text{ es una entidad y la inicia,} \\ I & \text{si } w_i \text{ es una continuación de entidad, va después de B,} \\ L & \text{si } w_i \text{ es una entidad y la finaliza,} \\ O & \text{si } w_i \text{ no es entidad,} \\ U & \text{si } w_i \text{ es una entidad de único elemento} \end{cases} \tag{4.3}$$

- Bi-grams, Tri-grams y n-grams, que agrupa el texto en conjunto de palabras, estas agrupaciones parten las oraciones y agregan más características a los

textos, un ejemplo de como funcionan los n-grams, lo podemos ver en la Figura 4.2.

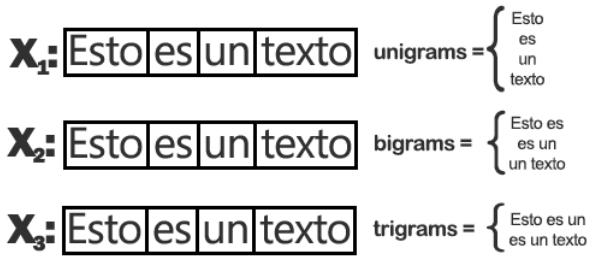

Figura 4.2: Ejemplo de n-grams y sus formaciones

- Indicador binario que indica si la palabra ingresada es un dígito, símbolo, combinación de dígito y símbolo, mayúscula y minúscula.
- El uso de prefijos y sufijos de una palabra, es decir las raíces de palabras que son muy usadas en el idioma español.

4.3 Características no supervisadas

Las palabras del texto ya pre-procesadas son utilizadas en la clasificación de texto. Las características básicas extraídas del texto son transformadas a representaciones de palabras o *Word Embeddings*, que consisten en la transformación de una palabra en un espacio matemático vectorial con distintas dimensiones.

- *Word embeddings*
  - Cada palabra es un punto en el espacio, donde es representado por un vector de un número de dimensiones (generalmente 300).
  - Se construyen de forma no supervisada, a partir de un corpus.
  - Por ejemplo "Hola"podría ser representado como [0.4, -0.11, 0.55, 0.3 . . . 0.1, 0.02]

Se han utilizado dos tipos de representaciones distribuidas de *Word embeddings* para la clasificación de texto, Word2Vec [34] y Glove [42]. Para la obtención de los vectores de Glove y Word2Vec, se uso el siguiente procedimiento de construcción:

- Se extrajo un archivo de wikipedia [91] en formato XML que es convertido a texto puro con un script de python $process_wiki.py$ que genera un archivo $wiki.es.text$, de donde se generan los vectores.
- Se generaron vectores de 50 y 300 dimensiones del mismo corpus que consta de mas 3.6 Gb de artículos propios de Wikipedia [91].
- Tanto Glove [42] como Word2vec [34] tienen bibliotecas que nos permiten definir representaciones de palabras. Las dimensiones elegidas en este trabajo son de 50 y 300 dimensiones.
- La representación de Tang [17] se puede usar para insertar polaridad a los vectores de palabras, para este proceso es necesario tener vectores ya entrenados y texto etiquetado según su polaridad, es un aprendizaje semi-supervisado.

4.3.1 Binarizacion, Clustering y Prototipos de word embeddings

Con la finalidad de reducir los vectores y hacerlos discretos, pero sin que esta acción reste los rasgos más resaltantes de los *word embeddings*, se realiza la binarización y clusterización de los vectores de palabras.

La Binarización se realiza calculando los umbrales superior e inferior a través del vocabulario. El umbral es calculado en cada dimensión, con los valores positivos (C_{i+}) y los valores negativos (C_{i-}). La Ecuación 4.4, representa la Binarización y la Tabla 4.3 el resultado.

Estas operaciones sobre los vectores, se usaron como características en el clasificador basado en *Conditional Random Fields*.

$$\phi(C_{ij}) = \begin{cases} U_+ & \text{si } C_{ij} \geq promedio(C_{i+}), \\ B_- & \text{si } C_{ij} \leq promedio(C_{i-}) \\ O. & \end{cases} \tag{4.4}$$

Dimensión	Valor	Binarizado
1	-0.008255	0
2	0.145529	U+
3	0.010853	0
⋮	⋮	⋮
298	0.050766	U+
299	-0.066613	B-
300	0.073499	U+

Tabla 4.3: Embeddings binarizados

La Clusterización hace uso de *K-means*, se toma para ello diferentes valores de grupos que contienen información de diferente granularidad. En la Tabla 4.4, se muestran 5 granularidades, y los grupos a los que pertenece.

Granularidad	K
500	299
1000	516
1500	1428
2000	1934
2500	2453

Tabla 4.4: Clustering de embeddings

Los prototipos, se refieren a agregar palabras próximas en el espacio como características, están basadas en los vectores de palabras y requieren de un conjunto de datos etiquetado para cada clase, calculan las palabras mas frecuentes o cercanas a cada clase y que tengan una representación vectorial similar, para crear esta relación palabra clase se hace uso de *Normalized Pointwise Mutual Information (NPMI)* [10].

Para esta tarea es necesario crear un corpus anotado del conjuntos de entrenamiento. Después, realizar un cálculo a partir de las etiquetas l y las palabras w. usando la Ecuación 4.5.

$$\lambda_n(l,w) = \frac{\lambda_n(l,w)}{-\ln p(l,w)}, \lambda(l,w) = \ln \frac{p(l,w)}{p(l)p(w)} \tag{4.5}$$

Luego de obtener los prototipos de las clases (Positivo, Negativo, Neutro y Ninguno), se reagrupa por palabras cada clase, y se buscan las palabras de una clase mas próximas. Para ello se usa la similaridad del coseno. Los prototipos que sobrepasan el umbral de 0.8 son elegidos como características de cada clase. Además, es necesario impedir un máximo de palabras prototipos por entrada, para no poseer demasiados prototipos asociados a una palabra.

4.4 Clasificador *Conditional Random Fields*

Modelar las probabilidades conjuntas tienen desventajas debido a la complejidad computacional. Por esa razón se elije un clasificador que modela probabilidad condicional ya que también esta demostrado que es útil en las tareas de clasificación de texto.

Este clasificador probabilístico modela la probabilidad condicional de una clase variable, sentimiento, dado un conjunto de variables de entrada, p(Y|X). En este sentido, este clasificador evita modelar la probabilidad conjunta de p(Y,X) de las características de entrada X y la clase salida Y.

El formato de entrada al clasificador probabilístico fue:

- La palabra es una entrada para la clasificador.
- Las dos palabras anteriores y dos posteriores son parte de la entrada para el clasificador.
- Características binarias que denotan si una palabra está compuesta solo por dígitos, símbolos, dígitos + símbolos, mayúsculas, minúsculas.

- Se usa sufijos y prefijos de las palabras.
- Fragmentación: pequeñas piezas de información como sustantivo frases verbales.
- Uso de *Part of Speech* como característica.
- Uso de *Chunking* con estilo de anotación BIO [19].
- Para realizar las dos tareas anteriores se uso CLiPS para el español, que es un módulo que contiene un etiquetador de voz rápida para español (identifica sustantivos, adjetivos, verbos, etc.) y herramientas para la conjugación de verbos en español, la singularización y la pluralización de sustantivos. [72]

4.5 Consideraciones Finales

En este capítulo se describió el procedimiento propuesto para realizar la clasificación de texto en el idioma español usando Modelos Probabilísticos, específicamente *Conditional Random Fields*. Además, se indicó las características empleadas (supervisadas y no supervisadas). También vale la pena indicar que el corpus para entrenamiento de los vectores de palabras fue generado y que los vectores usados no son un recurso externo. Las características obtenidas para **POS** y **CHUNKING** de obtuvieron de una adaptación de *Clips*[72] para el español.

En el siguiente capítulo de presentará el desarrollo de técnicas de *Deep Learning* como alternativa de clasificación de sentimientos en textos en español.

Capítulo 5

Clasificación de sentimientos usando Deep Learning y Word Embeddings para textos cortos en español

El *Deep Learning* es una de las áreas de investigación más usadas en el campo de la inteligencia artificial, Esto se debe a sus excelentes resultados obtenidos en sus distintos campos como: Procesamiento de Lenguaje Natural, Visión Computacional, etc. El *Deep Learning* hace uso de estructuras de redes neuronales de sucesivas capas. Su mismo nombre hace referencia al uso de muchas capas que se usan al modelar un problema y que aprenden automáticamente a medida que el modelo es entrenado con datos de entrada.

El clasificador que hace uso de *Deep Learning* también usa como entradas los textos cortos en español, como en los clasificadores probabilísticos, estos textos deben pasar por un procedimiento de preprocesamiento que elimina las características que no aportan significado, luego se extraen las características no supervisadas que son los *word embeddings* de cada palabra para luego usar estas características en el clasificador. Estos pasos están representados en la Figura 5.1.

A continuación, se va a detallar cada paso que se siguió para la clasificación usando Deep Learning y su prueba con distintos clasificadores hasta llegar a la

Figura 5.1: Pasos para la clasificación usando *Deep Learning*

configuración óptima, obviando el paso de texto de entrada, que se refiere al texto propiamente dicho.

5.1 Preprocesamiento

El preprocesamiento es muy similar al aplicado a los clasificadores probabilísticos, las mismas técnicas aplicadas en el capítulo anterior, fueron aplicadas a las entradas de los clasificadores que usan *Deep Learning* a excepción de hacer uso de *Part-Of-Speech* y *Chunking*, por no ser necesario en las entradas.

Todas las técnicas de preprocesamiento se presentaron en el Capítulo 2 en la Sección 2.2.2.

5.1.1 Data Augmentation

Data Augmentation[44] consiste en aumentar los datos de entrenamiento cuando estos son escasos y así reducir la varianza. En el corpus usado para hacer comparaciones, Tabla 5.1, se observa que las proporciones de clases no son similares, esto ocasiona que el clasificador no aprenda sobre clases que tienen muy pocos datos para entrenar.

Como se puede ver en la Tabla 5.1, la desproporción de clases es una de los impedimentos para obtener una buena clasificación, por ello esta técnica fue aplicada al

Clase	Train	Dev	Test
P	318	156	642
NEU	133	69	216
N	418	219	767
NONE	139	62	274
Total	1,008	506	1,899

Tabla 5.1: Desproporción para las clases NEU y NONE

clasificador que combina *Convolutional Neural Network* y *Recurrent Neural Network* a fin de obtener mejores resultados.

El procedimiento usado para aumentar datos de entrenamiento, consistió en los siguientes pasos:

- Se aplicó a las clases NEU y NONE.
- Se usó un diccionario de palabras con significado similar a los del entrenamiento. y se hizo sustituciones para crear nuevos datos.
- Se quitaron palabras que no aportan significado y se usaron las palabras restantes para crear N-grams como nuevas entradas.

5.2 Características no supervisadas

Las palabras de entrada son usadas para encontrar una representación de palabras asociada, se puede encontrar una representación distinta según el método usado para conseguirlo, como *Glove* y *Word2vect*. Esta variedad de representaciones son usadas como canales en las redes neuronales. En este tipo de clasificador se usa una representación adicional a las 2 anteriores, que se explicará en las características no supervisadas.

Se han utilizado tres tipos de representaciones de palabras (*Word Embeddings*) para la clasificación de texto, estos son: Word2Vec [34], Glove [42] y FasText [61]. Los vectores de FasText, se obtuvieron de su repositorio [26], donde existen varios

vectores y modelos para distintos idiomas.

5.3 Clasificadores *Deep Learning*

Para la clasificación de textos, se ha utilizado CNN (*Convolutional Neuronal Network*) y RNN (*Recurrent Neuronal Network*) por separado y combinadas, a continuación se detallan los clasificadores usados.

5.3.1 Clasificador: Convolutional Neuronal Network

Son Redes Neuronales similares a las ordinarias, ya que poseen pesos y sesgos. Las entradas a estas redes generalmente son matrices, por ejemplo imágenes, eso hace que la arquitectura cambie para reducir la cantidad de parámetros y ganar eficiencia. Las capas de una Red Neuronal son las siguientes:

- **Capa convolucional:** Realiza la convolución es decir, recibe una entrada y aplica un filtro sobre la entrada, de esta forma se reduce el número de parámetros y se obtiene características de la entrada.
- **Capa de reducción o Pooling:** Reductor de la cantidad de parámetros.
- **capa Fully Connected:** Es la encargada de hacer la clasificación.

La arquitectura de una *Convolutional Neuronal Network* aplicada al PLN, puede observarse en la Figura 5.2. En esta Figura se puede ver que la entrada a la CNN, corresponde a una matriz conformada por las palabras de la oración y sus representaciones de palabras, se aplicaron 3 canales, uno por cada *word embedding*. Luego de realizar las convoluciones con filtros de tamaño 2,3 y 5, se obtienen características que pasan a través de una capa de *max-pooling*, los vectores resultantes son concatenados y con una capa de *Softmax*, la oración es clasificada.

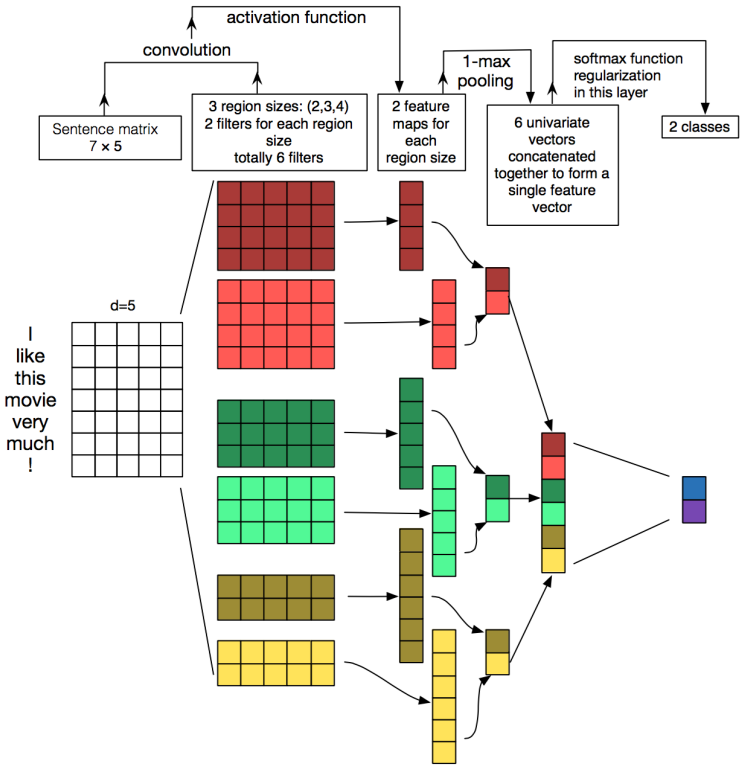

Figura 5.2: Convolutional Neuronal Network para procesamiento de lenguaje natural, imagen extraída de WildMl Blog [39]

5.3.2 Clasificador: Recurrent Neuronal Network

Una red neuronal recurrente (RNN) es un tipo de red neuronal artificial, forma un grafo dirigido conectando sus nodos. Es aplicada para modelar comportamiento temporal.Una RNN es una secuencia de redes neuronales enlazadas entre sí. Cada nodo pasa un mensaje a su sucesor. Como características se usa las representaciones de palabras de *Glove*[42] como única característica.

El tipo de *cell* usado fue LSTM (Long Short Term Memory)[53] que son capaces de aprender dependencias a largo plazo, con mas de una capa de red neuronal en el modulo de repetición, lo que le permite tener una memoria a largo plazo.

La arquitectura de una *Recurrent Neuronal Network* aplicada al PLN, puede observarse en la Figura 5.3. En esta Figura se puede ver que la entrada a la RNN corresponde a las palabras representadas por los *word embedding* de *Glove*[42]. Cada

representación de palabra es conectada a la siguiente con el uso de *cells* de tipo LSTM, que finalmente pasa por la capa de *Softmax* que clasifica la oración.

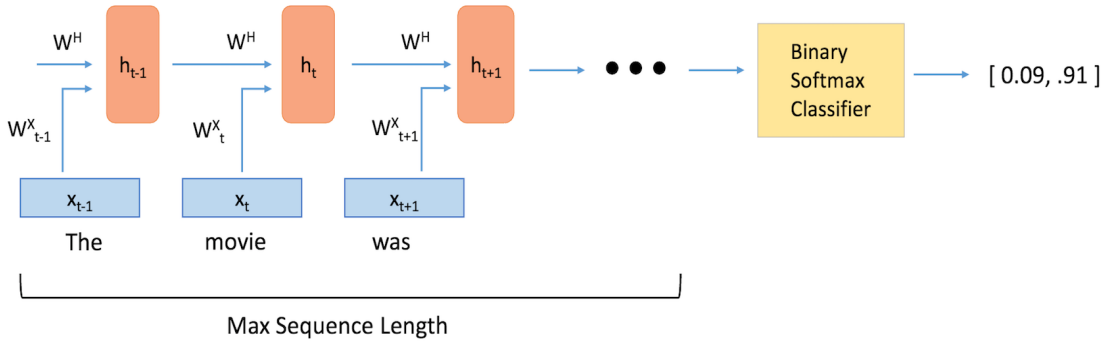

Figura 5.3: Long Short Term Memory Units para procesamiento de lenguaje natural[15]

5.3.3 Clasificador: Convolutional Neuronal Network - Recurrent Neuronal Network

Este clasificador combina los dos anteriores, hace uso de una *Convolutional Neuronal Network* y esta entrega nuevas entradas a una *Convolutional Neuronal Network* que se encarga de hacer la clasificación. Este procedimiento se puede ver en la Figura 5.4.

La entrada a la CNN *Convolutional Neuronal Network* esta conformada por 3 canales (3D CNN), cada canal corresponde a un *Word Embedding* distinto y que es usado al mismo tiempo, los *Word Embedding* usados fueron *Glove*, *Word2vec* y *FastText*, aplicadas en ese mismo orden. La última capa de la CNN es conectada secuencialmente a la RNN, específicamente luego de la capa de *max pooling*, La capa final de la *Recurrent Neuronal Network* es la encargada de clasificar el texto según su polaridad.

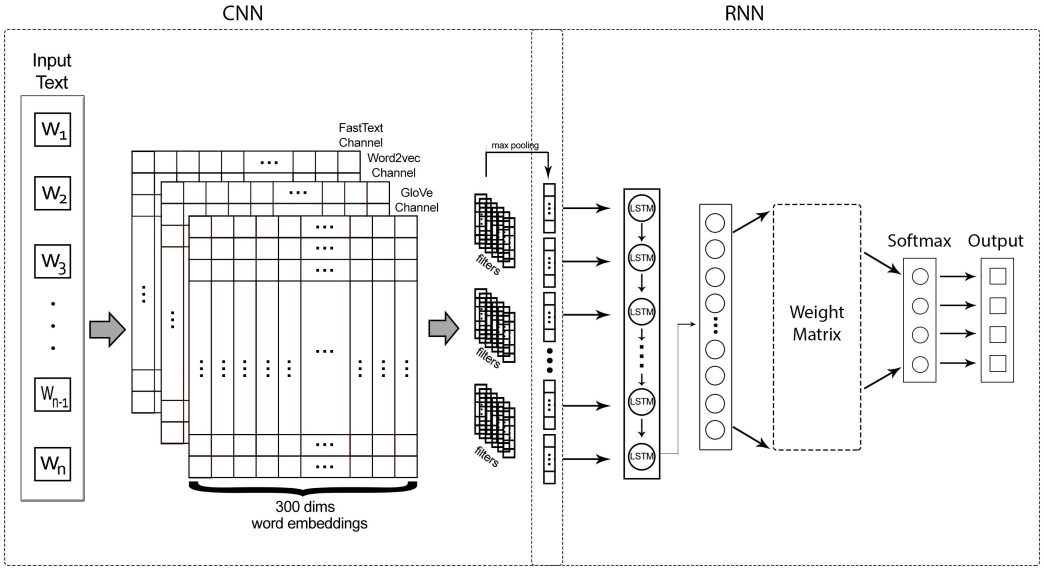

Figura 5.4: Clasificador CNN-RNN

5.4 Consideraciones Finales

En este capítulo se describió el procedimiento realizado para la clasificación de texto en el idioma español usando *Deep Learning* mostrando todos los procedimientos realizados y las características empleadas (supervisadas y no supervisadas).

En el siguiente capítulo de presentará los conjuntos de datos en los que se realizaron las pruebas, forma de evaluación y resultados de los modelos propuestos.

Capítulo 6

Resultados

En este capítulo se presenta los resultados obtenidos, explicando los conjuntos de datos, con y sin anotaciones, además de los detalles de la implementación de los clasificadores. Los resultados son evaluados sobre el conjunto de datos del TASS 2017[56] como datos *benchmarks* y con otros conjuntos de datos propios como casos de estudio.

6.1 Conjunto de datos

Aquí se describen 2 secciones, la primera es una sección de datos *benchmark* donde se listan 2 conjuntos de entrenamiento que han sido usados por otras investigaciones, en particular en el *Workshop SEPLN - TASS 2017* y mediante el cual se puede comparar el resultado obtenido con otras investigaciones. En la segunda sección se presentan conjuntos de entrenamiento de casos de estudio, es decir, conjuntos de datos que fueron recolectados de las principales redes sociales. Estos dos casos de estudios se organizan en dos conjuntos de entrenamiento, el primero está conformado por datos recolectados de *Google Play*[1] correspondiente a temas de aplicaciones móviles. El segundo corresponde a la unión de varios temas de datos recolectados de *Facebook*[2] y foros generales.

[1]https://play.google.com/store

[2]https://developers.facebook.com/

6.1.1 Datos *benchmark*

Para realizar la tarea de análisis de sentimientos, se ha usado el conjunto de datos del TASS 2017 [56] que en esa edición nos permitía el uso de 2 conjuntos de entrenamiento para la tarea de análisis de sentimientos. Presentaba las etiquetas de P, N, NEU y NONE; positivo, negativo, neutro y sin sentimiento respectivamente.

Cabe resaltar que este conjunto de datos muestra pequeñas variaciones desde el 2012, donde entonces se mostraba *Tweets* etiquetados en 6 polaridades (P+, P, NEU, N, N+, NONE), este corpus es el único disponible en español para poder hacer comparaciones entre propuestas.

El primer conjunto de entrenamiento de esta edición, se denominó **InterTASS2017** y para su construcción se tomó en cuenta los siguientes requisitos:

- Contenido de Tweets en español de España.
- Poseer una longitud mínima de 4 palabras en los Tweets.
- Poseer al menos un adjetivo en cada Tweet.

Los detalles en proporción de este conjunto de datos, se puede ver en la Tabla 6.1 donde se puede apreciar que la cantidad total es de 1,008 Tweets para el entrenamiento, 506 Tweets para el conjunto de desarrollo y 1,899 Tweets como conjunto de prueba. En la Tabla 6.2, se puede ver el mismo conjunto de datos, pero distribuido por clases.

Corpus	Tweets
Training	1,008
Developement	506
Test	1,899
Total	3,413

Tabla 6.1: Distribución del Corpus InterTASS 2017

El segundo conjunto de entrenamiento de denomina **General TASS**, y sus detalles de proporciones se puede ver en la Tabla 6.3, que tiene como totales un conjunto

Clase	Train	Dev	Test
P	318	156	642
NEU	133	69	216
N	418	219	767
NONE	139	62	274
Total	1,008	506	1,899

Tabla 6.2: Distribución del Corpus InterTASS 2017 por clase

de entrenamiento de 7220, y dos conjuntos de pruebas: uno de 60708 entradas y otro de 1000 entradas. En la Tabla 6.4, se puede ver el mismo conjunto de datos, pero distribuido por clases.

Corpus	Tweets
Training	7,220
Test	60,708
Test1k	1,000
Total	68,928

Tabla 6.3: Distribución del Corpus General TASS 2017

Clase	Train	Test	Test1k
P	2,932	22,233	507
NEU	474	1,305	63
N	2,332	15,844	307
NONE	1,482	21,416	123
Total	7,220	60,708	1,000

Tabla 6.4: Distribución del Corpus General TASS 2017 por clase

6.1.2 Casos de Estudio

Por la poca cantidad de conjuntos de datos disponibles, se realizaron pruebas sobre conjuntos de datos propios. Uno de los conjuntos de datos se recolectó de *GooglePlay*, que consta de comentarios de aplicaciones en la tienda y que se clasifica

según su *rating* en estrellas para determinar su polaridad. En la Tabla 6.5 se puede ver la distribución de este conjunto de datos, y en la Tabla 6.6 se puede ver su distribución por clases.

Corpus	Comentarios GooglePlay
Training	16,200
Test	1,800

Tabla 6.5: Distribución de Comentarios de GooglePlay

Clase	Train	Test
P	8,100	900
N	8,100	900
Total	16,200	1,800

Tabla 6.6: Distribución de Comentarios de GooglePlay por clase

Otro de los conjuntos de datos adicionales, consistió en recolectar documentos de *Facebook* y foros sobre diversos temas, y unirlos en un solo conjunto de datos. En las Tablas 6.7 y 6.8 se muestran las distribuciones por temas y por clase de este otro conjunto de datos de estudio.

Clase	Pos	Neg	Neu
FacebookGeneral	2,348	2,477	1,078
Temas sobre Mujer	1,452	1,305	217
Temas sobre Bancos	363	650	608
Temas sobre Elecciones	533	522	253
Total	4,696	4,954	2,156

Tabla 6.7: Distribución de Comentarios de Facebook y foros

Entonces, haciendo un resumen sobre los datos disponibles para hacer pruebas; en la Tabla 6.9 se muestran todos los conjuntos de datos *benchmark* y casos de estudio, en los que se hizo las comparaciones para encontrar el clasificador óptimo de sentimientos en esta tesis.

Clase	Train	Test
P	4,227	469
N	4,459	495
NEU	1,941	215
Total	10,627	1,179

Tabla 6.8: Distribución de Comentarios de Facebook y foros por clase

Conjunto de datos	Total de elementos
1. General TASS 2017	68,928
2. InterTASS 2017	3,413
3. ES-TASS 2018	8,614
4. GooglePlay	18,000
4. Facebook y Foros	11,806

Tabla 6.9: Distribución de Comentarios de Facebook y foros por clase

6.2 Experimentos

En este sección explicamos los clasificadores usados y algunos puntos importantes en la ejecución de los algoritmos usados.

6.2.1 Métricas de evaluación

Para realizar la tarea de evaluación se ha hecho uso de *Accuracy* (exactitud) (Ec. 6.1), *Recall* (Ec. 6.2) y F1 (Ec. 6.3). Para obtener estos valores es necesario tener en cuenta 4 valores tp (*true positive*), fp (*false positive*), fn (*false negative*), tn (*true negative*) que hacen referencia a los acumuladores del resultado de clasificación.

	Realidad	
Clasificador	Correcto	Incorrecto
Seleccionado	tp	fp
No Seleccionado	fn	tn

Tabla 6.10: Matriz de contingencia[54]

$$Accuracy = \frac{tp}{tp + fp} \tag{6.1}$$

$$Recall = \frac{tp}{tp + fn} \tag{6.2}$$

$$F1 = \frac{2PR}{P + R} \tag{6.3}$$

El resultado de la clasificación se obtiene de cada entrada de texto respecto a las clases o etiquetas, es decir que por cada entrada a clasificar se le asigna una clase y se le contabiliza según la tabla de contingencia 6.10.

6.2.2 *Cross Validation*

Para obtener los mejores resultados en el análisis de datos, se aplica la técnica llamada validación cruzada o *cross validation* con esto se garantiza que los resultados son independientes de partición entre datos de entrenamiento y datos de prueba. Como se puede ver en la Figura 6.1, se hace la partición del conjunto de datos donde los segmentos de entrenamiento y prueba varían en cada experimento. Con esta técnica además se encontraron los mejores parámetros para los clasificadores.

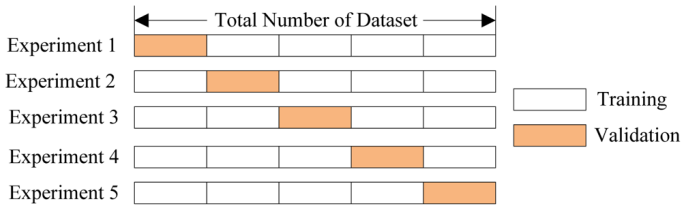

Figura 6.1: El procedimiento de validación cruzada[14].

6.2.3 *Conditional Random Fields*

La ejecución de este algoritmo se realizó en el clúster del Centro de Alto Rendimiento Computacional de la Amazonia Peruana, denominado **Supercomputador MANATI**[13]. Los experimentos requerían memoria adicional a la un computador

normal, especialmente en el corpus más grande. Cada palabra era cargada con sus respectivas representaciones de palabras, y características adicionales, además se cargan las 2 palabras antecesoras y sucesoras. Tomando en cuenta que esto era realizado por cada palabra, en una sola oración podían formarme entradas muy grandes al clasificador lo que generaba problemas de memoria.

Se uso python 2.7[22] para la ejecución de todos los *scripts* incluidos los de pre-procesamiento, ejecución del clasificador e interpretación de resultados. Se usó adicionalmente *CRFsuite*[64] como una implementación del clasificador.

Para el procedimiento de pre-procesamiento, se uso ParseTree de clips[72], para conseguir el árbol de características POS, CHUNK que identifica sustantivos, adjetivos, verbos, etc en una oración. Esta parte es importante resaltar, ya que es una de las características principales que hace uso el clasificador.

En la siguiente lista se muestra las características usadas por el clasificador Conditional Random Fields[84]:

- Un **Flag**, que determina si la palabra es un dígito, símbolo, combinación dígito+símbolo, mayúscula, letras, etc.
- 4 prefijos de la palabra.
- 4 sufijos de la palabra.
- 2 dígitos de la palabra.
- 4 dígitos de la palabra.
- Alfanuméricos.
- Dígitos y símbolos.
- Letras mayúsculas.
- Palabra con letra inicial mayúscula.
- Palabra con letras minúsculas.

- Si la palabra contienen una sola mayúscula.
- Si la palabra contienen una sola minúscula.
- Si la palabra contienen un dígito.
- Si la palabra contienen un símbolo.
- Si la palabra contienen un letra.
- palabra.
- **POS** de la palabra *Part-of-speech*.
- **CHUNK** de la palabra, sustantivos o segmentos verbales (también conocidos como frases nominales y frases verbales).
  - B - Inicio de un segmento.
  - I - Continuación de un segmento.
  - E - Final de un segmento.
  - NP - Segmento Nominal.
  - VP - Segmento Verbal.
- Características de los Word Embbedings o Word Representations.
  - de - word embeddings Glove
  - desa - word embeddings Glove + Tang
  - we - word embeddings Word2Vec
  - wesa - word embeddings Word2Vec + Tang
  - ce - cluster enbeddings
  - bi - brown embeddings
  - proto - proto embeddings

El modelo separa las características por conjunto(Tweet) y etiqueta según su polaridad (N,NEU,POS,NONE), estas etiquetas se asignan por palabras, cada palabra

tiene una de las características listada anteriormente, luego se combinan con las 2 palabras antecesores y predecesores con sus respectivas características. Todo ese procedimiento se convierte en la entrada para el clasificador.

En la Tabla 6.11 se puede apreciar los resultados de aplicar este clasificador al general TASS (Tabla 6.3), donde ***proc*** se refiere al uso de *tokenizeTwitter* que separa las oraciones por tokens de palabras, *tokenizerStanford* que complementa el tokenizador anterior, *preprocessTwitter* para eliminar algunas características propias de *Twitter* innecesarias para el clasificador y *PatternEs* para poder obtener el *Part-of-Speech* y el *Chunking*.

Corpus	Exactitud
General TASS 2017 (Test1k) usando word2vec y GloveTang	66,90 %
General TASS 2017 (Test1k) usando Glove-300dims	67,20 %
General TASS 2017 (Test1k) usando Glove-300dims Random	68,40 %
General TASS 2017 (Test1k) usando Proc + E. Glove50	66.7 %
General TASS 2017 (Test1k) usando Proc + E. Glove50Tang	65.8 %
General TASS 2017 (Test1k) usando Proc + E. word2vect50	66.2 %
General TASS 2017 (Test1k) usando Proc + E. word2vect50Tang	66.4 %
General TASS 2017 (Test1k) usando Proc + E. word2vec + E. GloveTang	66.9 %
General TASS 2017 (Test1k) usando Proc + E. Glove Random 300	**67.9** %
General TASS 2017 (Test1k) usando Proc + E. Glove Random 300	**68.4** %

Tabla 6.11: Resultados con CRF del Corpus General TASS 2017 (Tabla 6.3) aplicado al corpus de test de 1000 entradas (test1k)

Como se puede ver en la Tabla 6.11, la mejor configuración encontrada para el clasificador *Conditional Random Fields* y que da el mejor resultado es la que hace uso de representaciones de palabras aleatorias. Para ello es necesario hacer mas de una prueba y escoger la mejor alternativa. La segunda mejor configuración se da al usar solo representaciones de palabras de *Glove*. Para los siguientes experimentos solo se realizó pruebas con estas dos configuraciones.

En la Tabla 6.12 se detallan los resultados de aplicar el clasificador en los datos *benchmark* (Tablas 6.1 y 6.3).

Corpus	Exactitud
General TASS 2017 (Test1k)	68,40
General TASS 2017 (Test)	69,45
InterTASS 2017	55.87

Tabla 6.12: Resultados con CRF con datos *Benchmark*

Corpus	Exactitud
GooglePlay	87,48
Facebook y foros	84.29

Tabla 6.13: Resultados con CRF en casos de estudio

En la Tabla 6.13 se detallan los resultados de aplicar el clasificador en los casos de estudio (Tablas 6.5 y 6.7).

6.2.4 Convolutional Neuronal Network

Para el uso de este clasificador y los siguientes se hizo uso de python3[23], además se uso *TensorFlow*[5] para una implementación más escalable de los algoritmos de *Deep Learning*.

La *Convolutional Neuronal Network* CNN usa una combinación de 3 canales con filtros de 3,4 y 5, lo cual denota el uso de tres representaciones distintas paras las palabras.

Corpus	Exactitud	recall	F1
InterTASS 2017	55,52	66,36	62,99
General TASS 2017 (test1k)	69,90	0.72	0.77

Tabla 6.14: Resultados de CNN en datos *Benchmark*

En la Tabla 6.14 se representan los resultados de aplicar el clasificador en los conjuntos de datos *Benchmark* (Tablas 6.1 y 6.3).

En la Tabla 6.15 se representan los resultados de aplicar el clasificador en el conjunto de datos de estudio (Tablas 6.5 y 6.7).

Corpus	Exactitud	recall	F1
GooglePlay	91,16	90,22	81,08
Facebook	82,72	76,95	83,08

Tabla 6.15: Resultados de CNN en datos de estudio

6.2.5 Recurrent Neuronal Network

la *Recurrent Neuronal Network* RNN hace uso de un *cell lstm* y emplea un único conjunto de representaciones de palabra, el mejor desempeño se encontró en los embeddings generados por el algoritmo de *Glove*.

Corpus	Exactitud	recall	F1
InterTASS 2017	49,72	55.87	52.89
General TASS 2017 (test1k)	60,52	56,25	72,00

Tabla 6.16: Resultados de RNN en datos *Benchmark*

En la Tabla 6.16 se representan los resultados de aplicar el clasificador en los conjuntos de datos *Benchmark* (Tablas 6.1 y 6.3).

Corpus	Exactitud	recall	F1
GooglePlay	88.62	85,31	87,6
Facebook	81,68	89,06	84,21

Tabla 6.17: Resultados con RNN en datos de estudio

En la Tabla 6.17 se representan los resultados de aplicar el clasificador en los conjuntos de datos de casos de estudio (Tablas 6.5 y 6.7).

6.2.6 Recurrent Neuronal Network - Convolutional Neuronal Network

El clasificador RNN-CNN, usa una combinación de CNN de 3 canales conformados por una representación de palabras distinta por capa y filtros de 3,4 y 5. Luego se concatena con una RNN LSTM, La configuración de la red es detallada en la siguiente lista:

- Tamaño de batch de 32.

- Probabilidad drop-out de 0.5
- Dimensiones de *Word Embeddings* de 300.
- Evaluar cada 100.
- Tamaño de filtro de 3,4 y 5.
- Unidades ocultas de 300.
- L2 regularization lambda de 0.0
- Max pooling (discretización) de 4.
- Número épocas de 10.
- Número de filtros: 300 .

Al finalizar las pruebas preliminares, se logró superar por poco a los anteriores algoritmos, sin embargo no se lograba aún compararse al estado del arte. Luego se aplicó *Data Augmentation*[27], adaptado al procesamiento de lenguaje natural. Para esto se crearon nuevas entradas al clasificador basadas en POS y n-grams a todas las clases, con ello se trata de evitar la alta varianza de las clases (NEU y NON), neutros y sin clasificación. Al agregar esta técnica se logró llegar a un resultado muy cercano a las mejores investigaciones, cabe resaltar que esta técnica solo fue aplicada al conjunto de datos del TASS [56] por su alta varianza de clases. En los conjuntos de datos de estudio, no fue necesario por los buenos resultados obtenidos.

Corpus	Exactitud	recall	F1
InterTASS 2017	60,09	55,17	48,20

Tabla 6.18: Resultados con CNN-RNN en datos *Benchmark*

En la Tabla 6.18 se representan los resultados de aplicar el clasificador en los conjuntos de datos *Benchmark* (Tablas 6.1 y 6.3).

En la Tabla 6.19 se representan los resultados de aplicar el clasificador en los conjuntos de datos de casos de estudio (Tablas 6.5 y 6.7).

Corpus	Exactitud	recall	F1
GooglePlay	94,73	93,33	90,32
Facebook	84,21	83,33	80,0

Tabla 6.19: Resultados con CNN-RNN en casos de estudio

6.3 Comparaciones

El mejor desempeño en la tarea de clasificación, se obtuvo con el algoritmo final, es decir con el clasificador que combina CNN y una RNN en la clasificación. Para hacer la comparación se usó los resultados del TASS 2017.

Grupo	Macro-P	Macro-R	Macro-F1	Exactitud
ELiRF-UPV	0.507	0.471	0.489	0.612
UNSA-UCSP-DaJo	**0.472**	**0.496**	**0.482**	**0.609**
jacerong	0.479	0.440	0.459	0.608
ELiRF-UPV	0.497	0.490	0.493	0.607
RETUYT	0.490	0.453	0.471	0.596
tecnolengua	0.455	0.427	0.441	0.595
ITAINNOVA	0.450	0.423	0.436	0.576
SINAI	0.464	0.423	0.442	0.575
GSI	0.376	0.400	0.387	0.562
LexFAR	0.433	0.431	0.432	0.541
INGEOTEC	0.410	0.397	0.403	0.515
OEG	0.383	0.370	0.377	0.505

Tabla 6.20: Comparación InterTASS 2017

En la Tabla 6.20 se muestra la comparativa con el TASS 2017, cuyos datos ya fueron publicados y comprobados por el Workshop. Como se puede ver los resultados obtenidos son comparables con el estado del arte.

En la Tabla 6.21 se muestran los resultados de usar *Deep Learning* en la tarea de clasificación de sentimientos, como se puede ver los resultados están por debajo de los mejores resultados obtenidos con datos del TASS 2015, 2016 y 2017.

Trabajos con *Deep Learning*	**Exac. (Mejor)**
RNN (Vilares et al [90],TASS 2015)	0.60 (0.69)
CNN (Segura-Bedmar et al [80],TASS 2016)	0.64 (0.72)
CNN + SVM (Rosa et al [76], TASS 2017)	0.596 (0.609)
RNN +TFIDF(Garcia-Vega et al [69], TASS 2017)	0.44 (0.609)
LSTM + Lexicon (Araque et al [65], TASS 2017)	0.562 (0.609)
CNN+LSTM (Hurtado et al [52], TASS 2017)	0.436 (0.609)

Tabla 6.21: Trabajos con *Deep Learning*

6.4 Consideraciones finales

En los conjuntos de datos de casos de estudio se usaron 2 polaridades (positivo y negativo) para el conjunto de Google Play (Tabla 6.5) y 3 polaridades para el conjunto de datos de Facebook y foros (Tabla 6.7) encontrando el mejor desempeño en el clasificador de CNN-RNN.

Sobre los conjuntos de datos *benchmark*, el principal conjunto de comparación y estudio, es el conjunto del *interTASS* (Tabla 6.1), donde se encontraron alentadores resultados en la clasificación de textos cortos en español.

Finalmente, en este capítulo se ha presentado la evaluación de los clasificadores en los distintos conjuntos de datos empleados, detalles de implementación y finalmente los resultados que se alcanzan con la propuesta.

Capítulo 7

Conclusiones y Trabajos Futuros

Las principales conclusiones a las que se ha llegado en este trabajo son:

- En este trabajo se ha evaluado la clasificación de sentimientos de textos cortos en español, dando énfasis en el uso de clasificadores probabilísticos y de *Deep Learning*. Logrando encontrar un clasificador que brinda resultados prometedores en esta tarea.

- El clasificador basado en *Conditional Random Fields* CRF, mostró ser eficiente cuando tenemos pocos datos de entrenamiento y las etiquetas a clasificar son pocas. En los datos de casos de estudio CRF, mostró un desempeño considerablemente bueno. Por lo que el contexto ideal para usar este clasificador seria en casos donde no tenemos muchos datos y las etiquetas a clasificar son menores a 3.

- El clasificador que hace uso de *Deep Learning* dio un resultado comparable al estado del arte, y es la única propuesta que mostró un desempeño alto comparándolo con enfoques similares.

- Se hizo uso de *Word Representations* en español, como característica la clasificación de textos cortos en español. Esta propuesta entrena sus propios vectores de palabras y hace uso de ellos como característica principal. Los

experimentos muestran resultados prometedores en la clasificación de sentimientos comparados con investigaciones similares[56].

- *Data Augmentation* Añadido al clasificador, muestra un aumento significativo de exactitud, esto aplicado a corpus con alta varianza , es decir con muy alta desproporción de clases.

- El clasificador que hace uso de *Deep Learning* combinando CNN y RNN mostró el mejor desempeño en comparación con otras propuestas, tomando como conjuntos de datos para comparar el Dataset del TASS 2017 (Tabla 6.18). Este clasificador también obtiene resultados óptimos para los datos de casos de estudio.

7.1 Limitaciones

En este trabajo se busca clasificar textos en español cortos, una de las limitantes a este trabajo fue que no existen muchos datos *Benchmark* en los cuales experimentar y comparar, por ello se propusieron conjuntos de datos adicionales para poder hacer pruebas y comparaciones. El presente trabajo solo se enfoca a la clasificación de sentimientos, dejando de lado la detección de aspectos por polaridad.

7.2 Trabajos Futuros

En el futuro, se espera poder experimentar en un mayor numero de conjuntos de datos y hace poder construir un mejor clasificador que pueda incluso ser independiente de idioma.

En este trabajo solo se hace la clasificación de sentimientos, queda como trabajo futuro la detección de características que influyen mucho en el sentimiento que se le asigna a un texto.

Referencias

[1] Part-of-speech tagging. *Oxford Handbooks Online* (2012).

[2] Appendix 1: Information retrieval, nlp and automatic text summarization. *Automatic Text Summarization* (2014).

[3] Pattern recognition and signal analysis in medical imaging. *Pattern Recognition and Signal Analysis in Medical Imaging* (2014).

[4] 5th International Conference on Information Management, e Data, B. Simbig 2018.

[5] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., e Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[6] Aiala Rosá, Luis Chiruzzo, M. E. S. C. Retuyt en tass 2017: Análisis de sentimiento de tweets en español utilizando svm y cnn. *TASS 2017* (2017).

[7] Antonio Moreno-Ortiz, C. P. H. Tecnolengua lingmotif at tass 2017: Spanish twitter dataset classification combining wide-coverage lexical resources and text features. *TASS 2017* (2017).

[8] Bader-W., B., e Chew, K.-P. Multilingual sentiment analysis using latent semantic indexing and machine learning. *2011 IEEE 11th International Conference on Data Mining Workshops* (2011).

[9] Borda, I. M., e Saladich, J. C. Bittenpotato: Tweet sentiment analysis by combining multiple classifiers. *TASS 2015, septiembre 2015, pp 71-74* (2015).

[10] Bouma, G. Normalized (pointwise) mutual information in collocation extraction. *Department Linguistik, Universit¨at Potsdam* (2009).

[11] Brendan O'Connor, Michel Krieger, D. A. Tweetmotif: Exploratory search and topic summarization for twitter. *http://tweetmotif.com/about* (2010).

[12] Cifuentes, J. F. S. Introducción a la teoria de probabilidad. *Universidad Nacional de Colombia, sede Manizales*.

[13] Computer), S. M. H. H. P. Centro de alto rendimiento computacional de la amazonía peruana.

[14] DanB. Cross-validation.

[15] Deshpande, A. Perform sentiment analysis with lstms, using tensorflow. *oreilly* (2017).

[16] Dieter Jaeger, R. J. Artificial vision. *Encyclopedia of Computational Neuroscience* (2015).

[17] Duyu Tang, Furu Wei, B. Q. N. Y. T. L.-M. Z. Ieee transactions on knowledge and data engineering. *Journal of Machine Learning Research* (2015).

[18] Duyu Tang, Furu Wei, N. Y. M. Z. T. L.-B. Q. Learning sentiment-specific word embedding for twitter sentiment classification.

[19] Erik F, T. K. S. Introduction to the conll-2002 shared task: Language-independent named entity recognition. *Proceedings of the sixth conference on Natural Language Learning, Taipei* (2002).

[20] Fersini, E. Sentiment analysis in social networks. *Sentiment Analysis in Social Networks* (2017).

[21] Fersini, E. Sentiment analysis in social networks. *Sentiment Analysis in Social Networks* (2017).

[22] Foundation, P. S. Python language reference. version 2.7.

[23] Foundation, P. S. Python language reference. version 3.5.

[24] Garrigós, H. *Lematización y análisis lingüístico mediante técnicas de inteligencia artificial*. Universidad de Murcia, Servicio de Publicaciones, 2002.

[25] Gauthier, J. Spanish faq for stanford corenlp, parser, pos tagger, and ner. *https://nlp.stanford.edu/software/spanish-faq.shtml* (2015).

[26] Grave, E., Bojanowski, P., Gupta, P., Joulin, A., e Mikolov, T. crawl-vectors.

[27] Gudivada, J. D.-X. L.-V. Augmentation and evaluation of training data for deep learning. *IEEE International Conference on Big Data (Big Data)* (2017).

[28] Guo, J., C. W.-W. H., e Liu, T. Revisiting embedding features for simple semi-supervised learning. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 110–120, Doha, Qatar. Association for Computational Linguistics* (2014).

[29] Hand, D. J., Y.-K. Idiot's bayes—not so stupid after all? *International statistical review* (2001).

[30] Haykin, S. *Neural Networks and Learning Machines*, 3 ed. Pearson Education, Inc., 2009.

[31] Hebb, D. O. The organization of behavior: A neuropsychological theory. *Psychology Press* (2005).

[32] Hinton, G. D.-T. S.-G. Select improving deep neural networks for lvcsr using rectified linear units and dropout. *IEEE International Conference on Acoustics, Speech and Signal Processing* (2013).

[33] Hirst, G. *Neural Network Methods in Natural Language Processing (Synthesis Lectures on Human Language Technologies)*. Morgan&Claypool publishers, 2017.

[34] https://code.google.com/archive/p/word2vec/. word2vec.

[35] (https://stats.stackexchange.com/users/204007/john doe), J. D. Kernel sizes for multiple convolutional layer neural networks. Cross Validated. URL:https://stats.stackexchange.com/q/340285 (version: 2018-04-13).

[36] https://textblob.readthedocs.io/en/dev/. Textblob: Simplified text processing.

[37] https://www.lexalytics.com/technology/sentiment. lexalytics.

[38] http://www.sepln.org/workshops/tass/2017/. sociedad española para el procesamiento de lenguaje natural.

[39] (http://www.wildml.com, D. B. Understanding convolutional neural networks for nlp.

[40] Ian Goodfellow, Y. B., e Courville, A. Deep learning.

[41] Ii, T. Deep learning and other example problems. *Introduction to Deep Learning Using R* (2017).

[42] Jeffrey Pennington, Richard Socher, C. D. M. Glove: Global vectors for word representation.

[43] Jenny Copara, Jose Ochoa, C. T. G. G. Conditional random fields for spanish named entity recognition using unsupervised features. *Iberania2016* (2016).

[44] Junhua Ding, XinChuan Li, V. N. G. Augmentation and evaluation of training data for deep learning. *Department Linguistik, Universit¨at Potsdam* (2017).

[45] Kelly, R. Twitter study. *PearAnalytics.* (2009).

[46] Ketkar, N. Training deep learning models. *Deep Learning with Python* (2017).

[47] Kim, P. Convolutional neural network. *MATLAB Deep Learning* (2017).

[48] Koller, D., e Friedman, N. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[49] Lev Ratinov, D. R. Design challenges and misconceptions in named entity recognition. *Proceeding CoNLL '09 Proceedings of the Thirteenth Conference on Computational Natural Language Learning Pages 147-155* (2009).

[50] Levy, Omer; Goldberg, Y. Linguistic regularities in sparse and explicit word representations. *CoNLL. pp. 171–180* (2014).

[51] Llu´is-F. Hurtado, Ferran Pla, D. B. Elirf-upv en tass 2015: An´alisis de sentimientos en twitter. *TASS 2015* (2015).

[52] Lluís-F. Hurtado, Ferran Pla, J.- G. Elirf-upv en tass 2017: Análisis de sentimientos en twitter basado en aprendizaje profundo. *TASS 2017* (2017).

[53] Manaswi, N. Rnn and lstm. *Deep Learning with Applications Using Python* (2018).

[54] Manning, C. D., e Schutze, H. Foundations of statistical natural language processing. *MIT Press, Cambridge, MA, USA.* (1999).

[55] Maria Pontiki, Dimitrios Galanis, H. P.-S. M. I. A. Semeval-2015 task 12: Aspect based sentiment analysis. *SemEval* (2015).

[56] Marlies Santos Deas, Or Biran, K. M. S. R. Spanish twitter messages polarized through the lens of an english system.

[57] Martinez, A. Natural language processing. *Wiley Interdisciplinary Reviews: Computational Statistics* (2010).

[58] Martinez-Camara, E., M.-V. M. M. M. P.-O.-J. Integrating spanish lexical resources by meta-classifiers for polarity classification. *J. Inf. Sci* (2014).

[59] María del Pilar Salas-Zárate, José Medina-Moreira, P. J. -S. K. L.-O. M. A. P.-V., e Valencia-García1, R. Sentiment analysis and trend detection in twitter. *2do Congreso Internacional de Tecnología e Innovación CITI 2016* (2016).

[60] Mikolov, T., C. K.-C. G., e Dean, J. Efficient estimation of word representations in vector space. *CoRR, abs/1301.3781* (2013).

[61] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., e Joulin, A. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)* (2018).

[62] Moreno-Ortiz. Lingmotif: Sentiment analysis for the digital humanities. *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics* (2017).

[63] Neubig, G. Neural machine translation and sequence-to-sequence models: A tutorial. *Language Technologies Institute, Carnegie Mellon University* (2017).

[64] Okazaki, N. Crfsuite: a fast implementation of conditional random fields (crfs), 2007.

[65] Oscar Araque, Ignacio Corcuera, C. R. C. A. I. y. J. F. S.-R. Aspect based sentiment analysis of spanish tweets. *TASS 2015* (2015).

[66] Oscar S. Siordia, Daniela Moctezuna, M. G. S. M.-J. E. S. T.-E.-A. V. Sentiment analysis for twitter: Tass 2015. *TASS 2015* (2015).

[67] Padró, L., e Stanilovsky, E. Freeling 3.0: Towards wider multilinguality. *Proceedings of the Language Resources and Evaluation Conference (LREC 2012) ELRA.* (2012).

[68] Pang, B., e Lee, L. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval: Vol. 2: No. 1–2, pp 1-135* (2008).

[69] Paredes Valverde, M.A., C. P. R. S. Z. M.-V. G. R. Sentiment analysis in spanish for improvement of products and services. *A deep learning approach. Scientific Programming 6* (2017).

[70] Paskin, M. Introduction to probability theory. *A Short Course on Graphical Models*.

[71] Pedregosa, F., G. V. A. G. V. M. B. T. O. G. M. B.-P. P. R. W.-V. D. J. V. A. P. D. C. M. B. M. P., e Duchesnay., E. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12:2825–2830.* (2011).

[72] Psycholinguistics), C. C. L. . https://www.clips.uantwerpen.be/. *Computational Linguistics Psycholinguistics* (2017).

[73] Reese, R. Natural language processing with java. *Community Experience Distilled. Packt Publishing.* (2015).

[74] reviews from www.muchocine.net, A. Corpus cine (spanish cinema).

[75] Rosa María Montañés Salas, Rafael del-Hoyo Alonso, J. V.-M. M. R. A. G. F. J. L.-P. Bag of tricks for efficient text classification. *TASS 2017* (2017).

[76] Rosa María Montañés Salas, Rafael del-Hoyo Alonso, J. V.-M. M. R. A. G. F. J. L.-P. Fasttext como alternativa a la utilización de deep learning en corpus pequeños. *TASS 2017* (2017).

[77] Rosset, D. N.-M. E.-S. Evaluating named entity recognition. *Named Entities for Computational Linguistics* (2016).

[78] Sang, E. F. T. K., e Meulder, F. D. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *Proceeding CONLL 03 Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4* (2003).

[79] Schölkopf, A. J. S. *A tutorial on support vector regression.* Springer, 2004.

[80] Segura-Bedmar, I., Q. A. M.-P. Exploring convolutional neural networks for sentiment analysis of spanish tweets. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. pp. 1014–1022. Association for Computational Linguistics* (2017).

[81] Shapiro, M. A.-A. R.-E. L-cnn: Exploiting labeling latency in a cnn learning framework. *23rd International Conference on Pattern Recognition (ICPR)* (2016).

[82] Socher, Richard; Perelygin, A. W. J. C. J. M. C. N. A. P. C. Recursive deep models for semantic compositionality over a sentiment treebank. *EMNLP* (2013).

[83] Socher, Richard; Bauer, J. M. C. N. A. Parsing with compositional vector grammars. *Proc. ACL Conf.* (2013).

[84] Sutton, C., e McCallum, A. An introduction to conditional random fields. *Foundations and Trends in Machine Learning* (2011).

[85] Tenne, Y. Machine–learning in optimization of expensive black–box functions. *International Journal of Applied Mathematics and Computer Science.* (2017), 105.

[86] the 16th Ibero-American Conference on Artificial Intelligence. Iberamia-2018.

[87] Theodoridis, S. Probabilistic graphical models. *Machine Learning* (2015).

[88] Triola, M. F. Bayes' theorem. *UW Faculty Web Server*.

[89] Turney, P. D. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia* (2002).

[90] Vilares, D., D.-Y. A. M. G.-R. C. Lys at tass 2015: Deep learning experiments for sentiment analysis on spanish tweets. *Proceedings of TASS 2015: Workshop on Sentiment Analysis at SEPLN.* (2015).

[91] Wikipedia, e. https://dumps.wikimedia.org/eswiki/latest/. *Wikipedia* (2017).

[92] Ying Zhang, S. V. S. machine translation. *https://doi.org/10.1007/s10590-010-9073-6* (2010).

[93] Zong, H. Y.-C. Multi-predicate semantic role labeling. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014).