Clasificación temprana de texto con redes neuronales

por

Dante López Rosas

Propuesta de tesis sometida como requisito parcial para obtener el grado de

MAESTRO EN CIENCIAS EN LA ESPECIALIDAD DE CIENCIAS COMPUTACIONALES

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica
©Coordinación de Ciencias Computacionales

Febrero, 2020
Santa María de Tonantzintla, Puebla.

Asesores:
Dr. Hugo Jair Escalante Balderas, INAOE
Dr. Manuel Montes y Gómez, INAOE

Índice general

Índice de figuras

Índice de tablas

Resumen (Abstract)

El uso de las redes sociales es cotidiano para la mayoría de los usuarios de internet, aunque en éstas se presentan diferentes situaciones de riesgo que pueden afectarlos fuertemente. Muchas de estas situaciones podrían ser evitadas si se consideran mecanismos de filtrado o moderación, sin embargo, es difícil dado el contenido masivo que se encuentra en las redes sociales. Para atacar lo anterior, actualmente se han desarrollado diversos métodos que permiten detectar contenido en el que los usuarios muestren algún tipo de síntoma de alguna enfermedad mental así como de conductas ofensivas, sin embargo, dichos mecanismos son aplicables para detección (cuando algo ya pasó), no para prevención (cuando algo aún no ocurre).

Considerando lo dicho, en este trabajo se propone el desarrollo de métodos basados en redes neuronales que sean útiles para la clasificación o detección temprana de situaciones de riesgo en redes sociales usando texto. Para ello se hace uso de diversos enfoques en el manejo del texto al momento de alimentar las redes neuronales durante las fases de entrenamiento o prueba. Además de que únicamente se utiliza la característica cronológica del texto, por lo que se tiene la ventaja de que los modelos creados son independientes de características específicas de un problema en concreto. Así mismo, se adapta y explora el uso de una nueva función de pérdida en las redes neuronales, de modo que se fomente la clasificación

temprana usando poca información.

Los resultados obtenidos muestran diferentes desempeños con la función de pérdida propuesta al usar diferentes enfoques al momento de alimentar la red, sin embargo, se obtienen métricas favorables en los casos de alimentación de la red con cronologías secuenciales de palabras o *embeddings* promedio de pedazos, es decir, se logra clasificar correctamente de manera temprana.

Capítulo 1

Introducción

Hoy en día el uso de redes sociales tales como Facebook[1], Twitter[2], Reddit[3], entre otras, ha ido en aumento debido a que en éstas los usuarios tienen una gran libertad de expresión. Sin embargo, el intercambio de contenido no filtrado y la limitada protección de información ha traído consigo diferentes situaciones de riesgo tales como problemas de acoso sexual (D. Liu y cols., 2017), bullying (Dadvar y Eckert, 2018), lenguaje agresivo (Escalante y cols., 2017), entre otros. Así como casos en los que usuarios muestran comportamiento de enfermedades mentales como pueden ser depresión, anorexia o en casos extremos, suicidio.

Considerando lo anterior se han desarrollado diferentes técnicas de procesamiento de lenguaje natural que permiten identificar estos comportamientos aprovechando que los usuarios comparten sus opiniones, sentimientos, experiencias y demás detalles acerca de sus actividades diarias (Marwa y cols., 2018). Sin embargo, la mayoría de estos trabajos tienen el problema de ser aplicables una vez que se ha

[1]https://www.facebook.com
[2]twitter.com/
[3]https://www.reddit.com

presentado la problemática (Escalante y cols., 2017), y aunque son de ayuda en diferentes contextos, dejan de lado el concepto de prevención.

Es por ello que los escenarios de predicción temprana han llamado la atención de la comunidad científica cuyo objetivo es prevenir la mayor cantidad de amenazas en situaciones prácticas utilizando evidencia textual. A este campo emergente se le conoce como "Early Text Classification"(ETC) y tiene el objetivo de identificar de forma anticipada las diferentes categorías de riesgo utilizando la menor cantidad de texto.(López-Monroy y cols., 2018)

Sin embargo, esta área no ha sido explorada a pesar de la relevancia que tiene. Por ejemplo, el problema de detectar la depresión (o alguna otra enfermedad mental) basándose en los *posts* que hacen los usuarios en sus redes sociales (Losada y cols., 2018). Aquí la idea consiste en leer los *posts* cronológicamente y tratar de determinar de la forma más rápida posible si el usuario presenta algún síntoma. Por lo que el reto presente es el incremento del texto a lo largo del tiempo y lograr una correcta clasificación con la evidencia textual que se tenga en determinados lapsos.

1.1. Problemática

Si bien existen algunas soluciones propuestas a las tareas de detección siguiendo el concepto de ETC,la mayoría deja de lado la importancia del aspecto cronológico secuencial presente en el texto, además de que en general se enfrentan a los siguientes problemas:

1. La mayoría de los trabajos se centra en la extracción de características de

un dominio en específico o representaciones de texto, por lo que lograrlos de una forma adecuada en estados tempranos de clasificación dificulta esta labor debido a que se depende en gran medida de la cantidad de texto que se tenga.

2. En su gran mayoría hacen uso de clasificadores tradicionales, que, si bien son efectivos, no toman en consideración la naturaleza secuencial de los textos, que en sí son ideas cronológicas.

Considerando lo anterior se deben buscar alternativas que no dependan de características en específico de un dominio, sino más bien de explotar características generales del texto como son los aspectos temporales y secuenciales, además de mejorar la clasificación en estados tempranos.

1.2. Motivación

Los métodos de ETC propuestos hasta ahora tienen la desventaja de ser especializados en un dominio, por lo que necesitan extraer características específicas o buscar representaciones que se adapten al problema. Lo anterior no es una mala práctica, sin embargo, es bueno buscar alternativas más generales que no dependan de ello, ya que muchas características específicas presentes en un dominio pueden no estar en otro y afectar el desempeño de los clasificadores. Una solución puede ser el uso de redes neuronales, ya que éstas presentan diferentes ventajas sobre los clasificadores clásicos, tales como el hecho de que en éstas es más común evitar el uso de características definidas manualmente, por lo que se construyen redes neuronales que toman palabras como entrada y aprenden a inducir características como parte del proceso de aprendizaje, además de que pueden lidiar con historias más largas y pueden generalizarse sobre contextos de palabras similares

(Jurafsky y Martin, 2019).Por otro lado, es necesario buscar mecanismos en los clasificadores de modo que se fomente la clasificación temprana, así como buscar características independientes del dominio que ayuden en ello.

Teniendo en cuenta lo anterior, en este trabajo se hace uso de diferentes tipos de redes neuronales como clasificadores, las cuales son alimentadas de diversas maneras, sin embargo, en todas ellas se explota la característica cronológica secuencial presente en el texto considerando que el aspecto temporal es una característica poco usada. Así mismo se busca la optimización de las redes neuronales usadas en este trabajo por medio de la implementación de una nueva función de pérdida. Teniendo en consideración que las funciones de pérdida comúnmente usadas solamente miden el error cuadrado o absoluto entre la salida de una red y la salida deseada, en este trabajo se propone usar una nueva función de pérdida con la que se pueda clasificar correctamente usando la menor cantidad de información posible, fomentando con ello la clasificación temprana.

1.3. Objetivos

1.3.1. General

Desarrollar un método basado en redes neuronales con función de pérdida específica para la clasificación temprana de texto que obtenga desempeño comparable o superior que el estado del arte.

1.3.2. Específicos

1. Determinar las posibles representaciones de texto a usar como entrada para los modelos de aprendizaje basado en redes neuronales.

2. Establecer posibles configuraciones de redes neuronales para la clasificación temprana.

3. Agregar una función de pérdida a la red neuronal que fomente la clasificación temprana.

1.4. Contribuciones y principales resultados obtenidos

Las contribuciones de este trabajo son las siguientes:

1. Modelos de redes neuronales con diferentes enfoques de alimentación de datos aplicables a la clasificación temprana sin importar el dominio.

2. Modelos de redes neuronales con función de pérdida específica que fomenta la clasificación temprana, es decir pueden clasificar con poca información obteniendo resultados favorables.

1.5. Alcance y limitaciones

En este trabajo de tesis se abarcó el diseño y evaluación de diferentes modelos de redes neuronales que permiten la clasificación temprana sin importar el dominio aprovechando la característica cronológica secuencial del texto. Los experimentos se realizaron únicamente en corpus del idioma inglés. De igual modo, no se garantiza que las ideas propuestas en esta tesis sean aplicables a otro tipo de redes neuronales.

1.6. Organización de la tesis

El presente documento de tesis se organiza de la siguiente forma:

- Capítulo 2: Marco teórico. En esta sección se describen los conceptos que permiten una compresión adecuada para la solución propuesta en la tesis.
- Capítulo 3: Estado del arte: Se hace una revisión a los trabajos relacionados y los diferentes enfoques de éstos en la clasificación temprana.
- Capítulo 4: Método propuesto. Se hace una descripción detallada de los modelos y enfoques que se usaron en esta tesis, así como las diferentes configuraciones de las redes neuronales.
- Capítulo 5: Evaluación y resultados. Se describen y muestran las diferentes evaluaciones hechas a los modelos, así como los resultados obtenidos en cada uno de los corpus usados con los diferentes enfoques planteados en la tesis.
- Capítulo 6: Análisis y discusión. En esta sección se hace un análisis del desempeño obtenido por los diferentes enfoques propuestos en la tesis.
- Capítulo 7: Conclusiones y trabajo futuro. Se describen los aportes obtenidos en este trabajo y el trabajo futuro que se puede hacer.

Capítulo 2

Marco teórico

En este capítulo se hace una descripción de las representaciones de texto, los tipos y configuraciones en las redes neuronales, así como de las métricas de evaluación usadas en este trabajo.

2.1. Clasificación de texto

La clasificación de texto tiene como objetivo identificar de manera automática los diferentes textos que uno tenga en un conjunto etiquetado apoyándose de técnicas de aprendizaje de máquina.

Es necesario tener en consideración que el proceso de clasificación de textos comienza con el indexado del documento, que consiste en mapear el documento a una representación compacta de su contenido. Los métodos de indexación normalmente utilizados en la categorización de textos utilizan una representación del documento mediante un modelo de espacio vectorial donde un conjunto de documentos en lenguaje natural son representados mediante vectores de términos. Finalmente, para clasificar un documento se calcula la similitud entre el vector de términos característicos de la categoría y el vector de términos del documento.(Quinteiro y cols., 2011).

2.2. Representaciones de texto

Existen diferentes representaciones de texto que varían en su complejidad y utilidad dependiendo de la tarea. En general el objetivo de estas representaciones es mapear documentos a un espacio vectorial, para poder aplicar métodos de clasificación. A continuación, se hace una descripción de las representaciones usadas comúnmente, así como en este trabajo.

2.2.1. Bolsa de palabras (BoW)

Considerada como la representación más popular que se puede hacer del texto, fácil de usar y comprender, su principio consiste en una matriz que representa al vocabulario del texto de forma desordenada, pero manteniendo la frecuencia que se tiene de cada palabra (Jurafsky y Martin, 2019). Bajo este concepto se pueden calcular cantidades que son derivadas directamente de las palabras, como puede ser la longitud de las oraciones en términos del número de letras o número de palabras. De igual modo, cuando se consideran palabras individuales, se pueden extraer características basadas directamente de ellas, como puede ser contar el número de palabras que tengan un prefijo o sufijo específico o calcular la proporción de palabras cortas o largas (con una longitud inferior a una longitud dada) (Goldberg, 2017).

Así mismo, cuando se hace uso de la bolsa de palabras, es común calcular la relevancia de las palabras. Para este fin, uno de los métodos más extendidos es la obtención del tf-idf. El tf*idf comprende la frecuencia del término en el documento (tf) y la inversa de la frecuencia de documentos que poseen (idf). El tf representa la importancia local que el término posee en el documento, es decir, cuanto más aparezca un término en un documento, más relevante será ese término para ese documento (Quinteiro y cols., 2011). El tf se define como

$$tf_w = \frac{Count(w)}{\sum_{i=0}^{n} Count(w_i)} \tag{2.1}$$

Donde $Count(w)$ representa el número de ocurrencias de una palabra en el documento, mientras que $\sum_{i=0}^{n} Count(w_i)$ indica la suma de el número de ocurrencias de todas las palabras en el documento. (Chen y cols., 2016/11).

Por otro lado, el idf representa la importancia global de un término en relación inversa, es decir, cuantos más documentos incluyan el término, este término será menos relevante (Quinteiro y cols., 2011). El idf se define como

$$idf(w) = log(\frac{Count(docs)}{Count(w, docs)} + 0{,}01) \tag{2.2}$$

Donde $Count(docs)$ es el número total de documentos, mientras que $Count(w, docs)$ es el total de número de documentos que contienen esa palabra.(Chen y cols., 2016/11).

Por lo que la relevancia de una palabra al final se calcula como

$$Relevancia(w) = tf(w) * idf(w) \tag{2.3}$$

2.2.2. *Word Embeddings*

Una de las principales desventajas que presentan la mayoría de los modelos de texto es el hecho de que generan vectores de alta dimensionalidad, puesto que dependen del vocabulario, así como de la cantidad de documentos que se tengan, que, regularmente son grandes cantidades. Así mismo, la relación que existe entre las palabras se pierde debido a que no se conserva la relación semántica o sintáctica que éstas presentan en el texto, como en el caso de la bolsa de palabras.

Una solución a lo anterior es la representación basada en *embeddings*, que tienen la característica de mapear el texto en vectores de baja dimensionalidad. Así mismo codifica la relación semántica y sintáctica de las palabras, donde la información semántica se relaciona con el significado de las palabras, mientras que la sintáctica con el rol estructural.(Li y Yang, 2018)

Existen diferentes modelos de *embedding* de palabras, sin embargo, en este trabajo se utilizan los modelos de Word2Vec (Mikolov y cols., 2013) y FastText (Joulin y cols., 2017)(Bojanowski y cols., 2017) que se describen a continuación.

Word2Vec

Word2Vec es una arquitectura de representación de palabras vectorial de baja dimensionalidad basada en redes neuronales con bolsas de palabras o n-gramas desarrollada por (Mikolov y cols., 2013). Su gran capacidad se aprecia en problemas de analogías o en agrupamiento de términos relacionados. Como se menciona en (Montejo-Ráez y Díaz-Galiano, 2016), el método consiste en proyectar las palabras a un espacio n-dimensional, cuyos pesos se determinan a partir de una estructura de red neuronal mediante un algoritmo recurrente. El modelo se puede configurar para que utilice una topología de bolsa de palabras (CBoW) donde se busca predecir una palabra dado el contexto, o la topología Skip Gram, cuya idea consiste en dada una palabra predecir el contexto en el que aparecerá (Meyer, 2016). La arquitectura de Word2Vec con dichas topologías se muestran en la figura 2.1 en la que p_i representa las palabras del contexto, mientras que P representa la palabra objetivo (CBoW) o la palabra con la que se produce el contexto(Skip Gram). Con estas topologías, si se dispone de un volumen de texto suficientemente grande, esta arquitectura puede llegar a capturar la semántica de cada palabra. La longitud de los vectores resultantes puede elegirse libremente, aunque en muchos casos se recomiendan dimensionalidades bajas si se tienen pocos ejemplos.

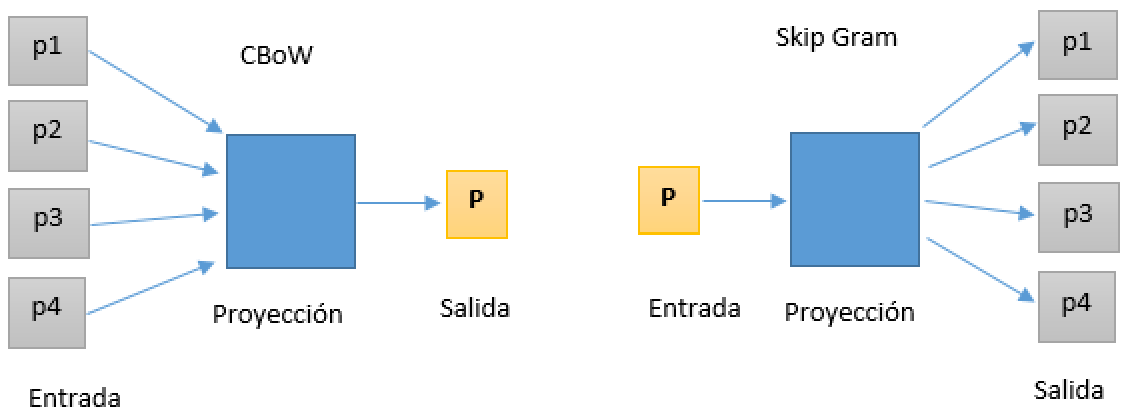

Figura 2.1: Idea de la arquitectura de Word2Vec en las topologías CBoW y Skip Gram. En ellas P es la palabra objetivo o con la que se desea obtener el contexto.

FastText

Como se menciona en (Lacueva y cols., 2017), FastText se basa en el modelo de bolsa de palabras mejorado a través del uso de clasificadores basados en redes neuronales apiladas, en lugar de los tradicionales clasificadores lineales cuya capacidad de generalización suele ser pobre. El uso de redes neuronales apiladas permite compartir información entre las variables y las clases a través de la capa oculta. La arquitectura implementada se asemeja a la de Word2Vec, para la representación de palabras en un espacio vectorial n-dimensional.

En concreto, FastText se basa en las mismas dos arquitecturas mencionadas, sin embargo, se diferencia en representar las palabras como bolsa de n-gramas de caracteres, lo cual permite obtener vectorizaciones de palabras que son incluso desconocidas.(Trotzek y cols., 2019)

2.3. Redes neuronales

Es bien sabido que el uso de redes neuronales ha mostrado grandes resultados en la clasificación de señales, imágenes, entre otras tareas. Así mismo su uso se ha ido popularizando en el procesamiento de lenguaje natural mejorando en resultados a los métodos de clasificación tradicional sin hacer uso de demasiadas características. A continuación, se hace una descripción de las redes neuronales que se usaron en este trabajo.

2.3.1. Redes recurrentes

Para este trabajo se hizo uso de redes neuronales Long Short Term Memory, sin embargo, antes de pasar a su definición es necesario aclarar que estas redes son del tipo recurrente, las cuales son de especial utilidad para el procesamiento secuencial de datos como el sonido, procesamiento de señales o el texto.

Las redes recurrentes son una clase de modelos de aprendizaje supervisado hechos con neuronas con uno o más bucles de retroalimentación, los cuales son ciclos recurrentes a lo largo del tiempo o de la secuencia de entrada. El entrenamiento de la red recurrente requiere de un conjunto de datos de entrenamiento etiquetados y se busca como objetivo minimizar la diferencia entre los datos y sus etiquetas al optimizar los pesos de la red.

La arquitectura de la red recurrente está compuesta por tres capas: capa de entrada, capa oculta recurrente y capa de salida (figura 2.2). En el caso de la capa de entrada, ésta tiene N unidades de entrada, los cuales son una secuencia de vectores a través del tiempo t tal que $\{..., x_{t-1}, x_t, x_{t+1}, ...\}$, donde $x_t = (x_1, x_2, ..., x_N)$.

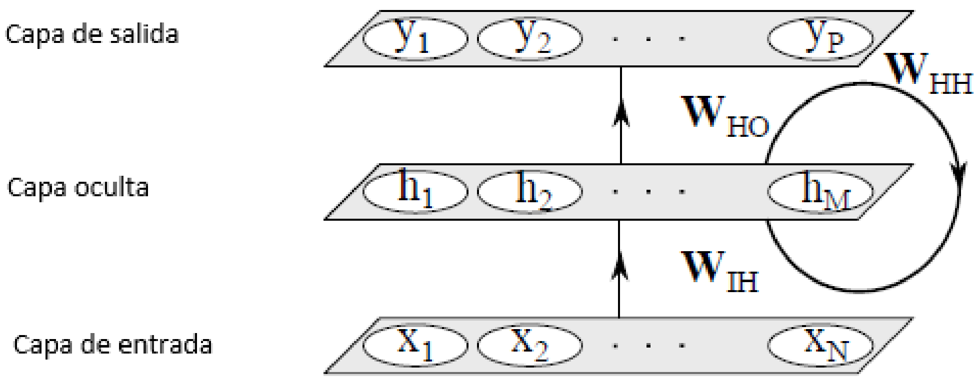

Figura 2.2: Estructura de una red recurrente simple. En ella se pueden observar las tres capas que componen a este tipo de red. Imagen tomada de (Salehinejad y cols., 2017)

Las unidades de la capa de entrada están conectadas a las unidades de la capa oculta, donde las conexiones están definidas como una matriz de pesos W_{IH}. La capa oculta tiene M unidades ocultas $_t = (h_1, h_2, ..., h_M)$, que están conectadas entre si. De igual modo, la capa oculta define el espacio de estado o "memoria" como:

$$h_t = f_H(o_t) \tag{2.4}$$

donde

$$o_t = W_{IH}x_t + W_{HH}h_{t-1} + b_h \tag{2.5}$$

$f_H(.)$ es la función de activación de la capa oculta y b_h es el vector sesgo de las unidades ocultas. Las unidades ocultas están conectadas a su vez a la capa de salida con conexiones con peso W_{HO}. La capa de salida tiene P unidades $y_t = (y_1, y_2, ..., yP)$ que son calculadas como

$$y_t = f_O(W_{HO}h_t + b_o) \tag{2.6}$$

donde $f_O(.)$ es la función de activación y b_o es el vector sesgo de la capa de salida. Dado que los pares de entrada-etiqueta son secuenciales a través del tiempo, los pasos anteriores se repiten consecutivamente sobre un tiempo $t = (1, ..., T)$. En cada paso de tiempo los estados ocultos proveen una predicción a la capa de salida basada en el vector de entrada. El estado oculto en una red recurrente es un conjunto de valores, que aparte del efecto de cualquier factor externo, resume toda la información necesaria sobre los estados pasados de la red. Esta información integrada puede definir el comportamiento futuro de la red y hacer predicciones precisas en la capa de salida (Salehinejad y cols., 2017)

2.3.2. Redes de memoria a largo plazo (LSTM)

Una dificultad que presentan las redes neuronales recurrentes es preservar la "memoria" a puntos distantes, sin embargo, muchas aplicaciones de lenguaje requieren recordar información a largos plazos. Para solucionar este problema se han desarrollado diferentes arquitecturas con la intención de mantener la "memoria"de la red por más tiempo, logrando con ello mantener la información importante a lo largo de la secuencia.

Una de estas redes es la LSTM(Hochreiter y Schmidhuber, 1997), la cual está

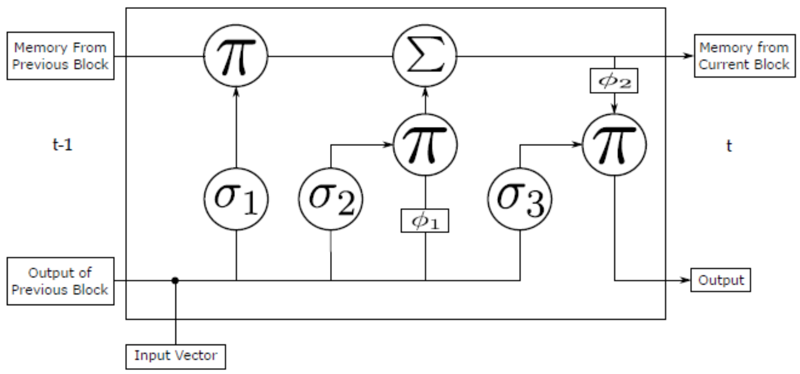

Figura 2.3: Módulo repetitivo de una red LSTM. Imagen tomada de (Kumar y cols., 2018)

diseñada para recordar información durante más tiempo, lo cual soluciona el problema de "memoria" a largo plazo que enfrentan las redes recurrentes. El módulo LSTM típico, llamado módulo repetitivo, tiene cuatro capas de redes neuronales que interactúan de manera única, como se muestra en la figura 2.3.

El módulo tiene tres funciones de activación de compuerta σ_1, σ_2 y σ_3 y dos funciones de activación de salida ϕ_1 y ϕ_2. El símbolo π y Σ representan la multiplicación y adición respectivamente. La operación de concatenación es representada por el símbolo ($\bullet$). El componente fundamental de la LSTM es el estado de celda, una línea que va desde la memoria del bloque anterior(S_{t-1}) a la memoria del bloque actual(S_t), lo cual permite que la información fluya en línea recta. Con esto la red puede decidir la cantidad de información previa a fluir. La operación llevada a cabo por esta capa es dada por la ecuación 2.7.

$$cf_t = \sigma_1(W_{cf} \cdot [O_{t-1}, X_t] + b_{cf}) \tag{2.7}$$

$$I_t = \sigma_2(W_I \cdot [O_{t-1}, X_t] + b_I) \tag{2.8}$$

$$\tilde{S}_t = tanh(W_s \cdot [O_{t-1}, X_t] + b_s) \tag{2.9}$$

$$S_t = cf_t \times S_{t1} + I_t \times \tilde{S}_{t-1} \tag{2.10}$$

La nueva información a ser almacenada en el estado de celda es calculada usando dos capas de red. Una capa σ_2 que decide los valores para actualizar (I_t)(ecuación 2.8) y una capa $tanh\ \phi_1$ que desarrolla un vector de nuevos valores candidatos $\tilde{S}_t$(ecuación 2.9), la combinación de ambas capas se agrega al estado. Finalmente el estado de celda se actualiza (ecuación 2.10) (Kumar y cols., 2018).

2.3.3. LSTM apilada (Stacked)

Una red recurrente LSTM consiste en leer secuencias y posteriormente usar la salida para clasificar o predecir. Sin embargo, también se puede usar esa salida como entrada para otra red, lo cual se conoce como LSTM stacked o apilada.

Se ha demostrado con diferentes trabajos que las redes neuronales recurrentes apiladas pueden superar el desempeño de aquellas que sólo usan una capa, tal como se puede ver en el trabajo de (D. Liu y cols., 2017) o en el caso de (Sadeque y cols., 2017). Una razón para este éxito se debe a la habilidad de las redes para inducir representaciones en diferentes niveles de abstracción a través de las capas. (Jurafsky y Martin, 2019). Un ejemplo de una red LSTM apilada aplicada en texto se puede observar en la figura 2.4, en ella se puede apreciar que la red neuronal esta compuesta por tres capas: la primera es una capa *embedding*, que se encarga de crear vectores a partir de oraciones de texto, posteriormente dichos *embeddings* son pasados a dos capas LSTM para finalmente clasificar usando una función sigmoide.

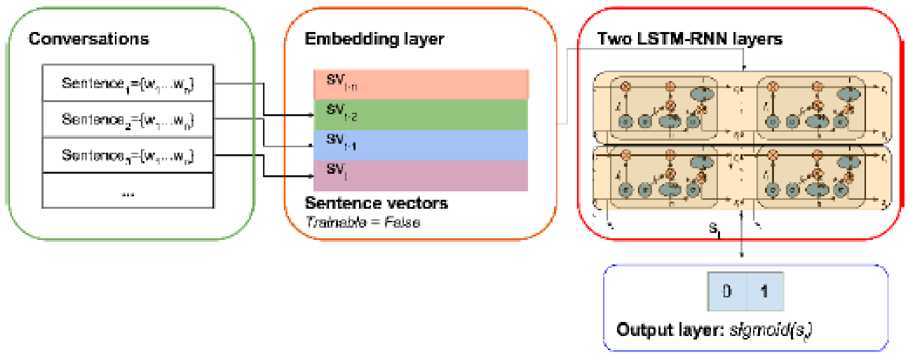

Figura 2.4: Ejemplo de LSTM apilada aplicada en texto. Imagen tomada de (D. Liu y cols., 2017)

2.3.4. Redes convolucionales (CNN)

Las redes convolucionales son un tipo especial de red que están diseñadas para identificar predictores locales indicativos en una estructura grande, y combinarlos para producir una representación vectorial de tamaño fijo, capturando con ello los aspectos locales que son más informativos para las tareas de predicción. Es decir, que por ejemplo, para el caso de texto, la arquitectura convolucional identificará los n-gramas que son predictivos para la tarea en cuestión, sin la necesidad de pre-especificar un *embedding* para cada posible n-grama. De igual modo, la arquitectura convolucional también permite compartir comportamientos predictivos entre n-gramas que comparten componentes similares, incluso si el n-grama exacto no se encuentra en los datos de la prueba.

Las redes convolucionales no constituyen una red independiente y útil por si sola, sino que están destinadas a integrarse a una red más grande para trabajar en conjunto y producir un resultado final. La responsabilidad de la capa convolucional es extraer sub-estructuras de datos significativos que son útiles para la tarea de predicción.

Las arquitecturas de convolución y agrupamiento (*pooling*) ayudaron a la comunidad de visión por computadora, donde mostraron un gran éxito para la detección de objetos (reconocer un objeto de una categoría predefinida independientemente de su posición en la imagen) (Goldberg, 2017).

Es necesario mencionar que las redes convolucionales pueden hacer uso de estructuras internas de datos como vectores 2D de las imágenes a través de las capas de convolución, donde cada unidad de cálculo responde a una pequeña región de los datos de entrada (Johnson y Zhang, 2015). En el caso de texto se hace uso de una estructura 1D (nivel palabra o frase) de modo que cada unidad en la capa de convolución responda a una pequeña región del texto, que sería una secuencia de palabras o frases. Un ejemplo de una red CNN aplicada en texto se puede observar en la figura 2.5.

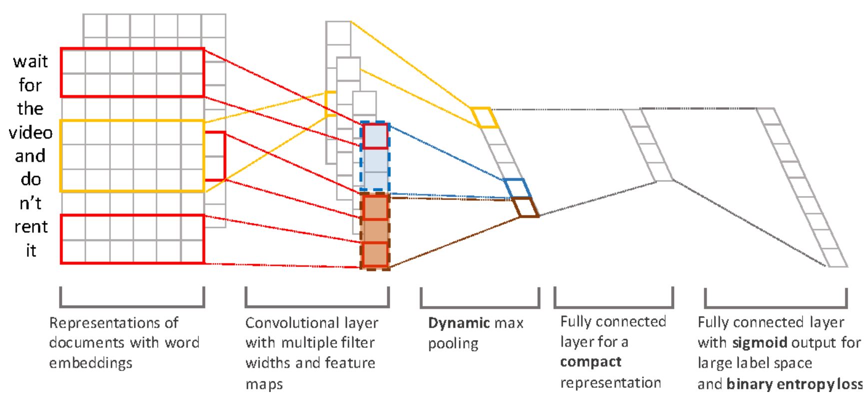

Figura 2.5: Red convolucional aplicada en texto con maxpooling.Imagen tomada de (Kim, 2014)

Agrupación máxima (*Max Pooling*)

La idea principal detras de una arquitectura de convolución y agrupación para tareas de lenguaje es aplicar una función no lineal sobre cada instanciación de una ventana deslizante de k-palabras sobre una oración. Esta función (filtro) transforma una ventana de k-palabras en un vector escalar. Se pueden aplicar varios filtros de este tipo, lo que resulta en un vector dimensional que captura propiedades importantes de las palabras en la ventana. Luego se utiliza la operación de agrupamiento (*pooling*) para combinar los vectores resultantes de las diferentes ventanas en un solo vector unidimensional, tomando el valor máximo observado en cada una de las dimensiones sobre las diferentes ventanas. La intención es centrarse en las características más importantes de la oración, independientemente de su ubicación. El vector dimensional resultante se alimenta luego en una red que se usa para la predicción. (Goldberg, 2017).

2.4. Función de pérdida

En el aprendizaje supervisado se tiene un conjunto de datos de entrada $x_{1:n} = x_1, x_2, ..., x_n$ junto con sus correspondientes etiquetas $y_{1:n} = y_1, y_2, ..., y_n$, el objetivo es encontrar una función f que mapee de forma precisa las entradas a sus etiquetas deseadas, es decir una función $f(x) = \hat{y}$, donde $\hat{y}$ es lo predicho por el algoritmo de aprendizaje supervisado. Sin embargo, durante el proceso de entrenamiento existe cierta pérdida cuando se predice $\hat{y}$, por lo que para medirla se hace uso de una "función de pérdida" $L(\hat{y}, y)$, la cual se encarga de asignar un puntaje numérico (un escalar) al valor predicho $\hat{y}$ por el algoritmo dado su valor real y. Téngase en cuenta que el proceso de aprendizaje es iterativo, por lo que los parámetros del algoritmo, regularmente una matriz W, son actualizados para minimizar el valor arrojado por la función de pérdida, es decir, que mientras más pequeño sea el valor de la función de pérdida mejor sera la eficacia del algoritmo.

Existen diferentes funciones de pérdida aplicables según sea el problema que se quiera resolver, tales como error cuadrático medio, entropía cruzada categórica, error logarítmico cuadrático medio, entre otros. Sin embargo, en este trabajo se hará uso de la función de entropía cruzada binaria como *baseline* y de la función propuesta en (Aliakbarian y cols., 2017), la cual es usada con resultados favorables en video, esta función de pérdida, así como su adaptación en este trabajo se encuentra descrita a detalle en la sección 4.4.

2.4.1. Entropía Cruzada Binaria

La función de pérdida de entropía cruzada binaria (BCE) es usada en problemas de clasificación con salidas de probabilidad condicional. Para su uso se asume un conjunto de dos clases etiquetadas con 0 y 1, $y \in \{0,1\}$. La salida del clasificador $\tilde{y}$ es transformada usando la función sigmoide $\sigma(x) = 1/(1+e^{-x})$ para el rango [0,1], y es interpretado como la probabilidad condicional $\hat{y} = \sigma(\tilde{y}) = P(y=1|x)$. Con lo anterior se obtiene que la regla de predicción es:

$$Predicción = \begin{cases} 0, \hat{y} < 0{,}5 \\ 1, \hat{y} \geq 0{,}5 \end{cases}$$

De tal modo, en el caso de las redes neuronales, estas son entrenadas para maximizar la probabilidad condicional logarítmica $P(y=1|x)$ para cada ejemplo del conjunto de entrenamiento. Teniendo con esto que la función de entropía cruzada binaria esta definida como:

$$BCE(\hat{y}, y) = -ylog\hat{y} - (1-y)log(1-\hat{y}) \tag{2.11}$$

Es por ello que la entropía cruzada binaria es útil cuando se desea que una red neuronal produzca una salida de probabilidad condicional en problemas de clasificación binaria.(Goldberg, 2017)

2.5. Métricas de evaluación

Para validar los resultados de las diferentes configuraciones y representaciones usadas en este trabajo se hace uso de métricas de evaluación clásicas, además de la medida Early Risk Detection (ERDE)(Losada y cols., 2017). Las cuales se describen a continuación.

Es necesario indicar que, en este trabajo, como en la mayoría de los problemas de aplicación, la clase que se desea detectar es la que tiene menor cantidad de ejemplos en los corpus, por lo que en este caso las medidas clásicas se aplican sobre la clase minoritaria. De igual modo se necesitan explicar unos términos antes de definir las medidas clásicas.

Primeramente, supóngase un problema binario en el que se tiene la clase positiva y negativa, al momento de clasificar los ejemplos se obtienen los siguientes casos:

- Verdadero positivo: se llama de este modo a aquellos ejemplos positivos que son clasificados correctamente como clase positiva.
- Verdadero negativo: se dice de este modo a los ejemplos negativos que son clasificados correctamente como clase negativa.
- Falso positivo: se dice de este modo a los ejemplos negativos que son clasificados erróneamente como clase positiva.
- Falso negativo: se llama de este modo a los ejemplos positivos que son cla-

sificados como clase negativa.

Una vez que se han aclarado los roles que puede tomar un ejemplo o item al clasificarse, ya se pueden explicar las medidas clásicas.

2.5.1. Precisión

La precisión mide el porcentaje de ejemplos que un modelo clasificó como positivos y son realmente de la clase positiva (Hossin y Sulaiman, 2015). La precisión se define como:

$$Precision = \frac{verdaderos\,positivos}{verdaderos\,positivos + falsos\,positivos} \tag{2.12}$$

2.5.2. Recuerdo (recall)

El recuerdo mide el porcentaje de ejemplos de la clase positiva que son clasificados correctamente sobre los que debieron ser clasificados en realidad (Hossin y Sulaiman, 2015). El recuerdo se define como:

$$Recuerdo = \frac{verdaderos\,positivos}{verdaderos\,positivos\; +\; falsos\,negativos} \tag{2.13}$$

2.5.3. Medida F

La métrica F, a veces conocida como F-score o (incorrectamente) como F1, es una media armónica ponderada del recuerdo y la precisión. La forma más general de esta formula esta dada en la ecuación:

$$F_\beta = (1+\beta^2) * \frac{Precisión * Recuerdo}{(\beta^2 * Precisión) + Recuerdo} \tag{2.14}$$

El término β permite darle más importancia al recuerdo cuando es mayor a uno, caso contrario, cuando es menor, se le da más importancia a la precisión. Sin embargo cuando β es igual a 1, se le da la misma importancia a precisión y recuerdo, de ahí el porque F1 (Powers, 2015). Para este trabajo se usara el valor de $\beta = 1$

2.5.4. Early Risk Detection Error (ERDE)

Esta medida está diseñada para determinar qué tan rápido un sistema logra clasificar correctamente un ejemplo. Antes de definir su funcionamiento es necesario indicar que se sugiere tener un corpus organizado cronológicamente (por ejemplo un conjunto de *posts* de una red social) además de saber el número de escritos hechos en cada ejemplo del corpus(Losada y cols., 2018).

ERDE es una medida binaria que considera la exactitud de la clasificación y la demora del sistema en hacer dicha decisión. La demora es medida contando el numero (k) de diferentes *posts* o comentarios vistos por el clasificador antes de tomar la decisión sobre qué clase es el ejemplo a clasificar. Como ejemplo, considérese un usuario u, quien ha posteado un total de 250 *posts*. Si el clasificador toma la decisión después de haber visto 50 *posts*, entonces k sería 50.

Otro factor importante es que los datos son desbalanceados, por lo que la medida de evaluación necesita ponderar diferentes errores de una manera diferente. Entonces considerando los casos de verdadero positivo, verdadero negativo, falso positivo y falso negativo, la medida ERDE se defina como:

$$ERDE = \begin{cases} 1, \text{cuando es falso negativo} \\ \frac{VerdaderosPositivos}{TotalEjemplos}, \text{cuando es falso positivo} \\ lc_o(k), \text{cuando es verdadero positivo} \\ 0, \text{cuando es verdadero negativo} \end{cases}$$

El factor $lc_o(k)(\in [0, 1])$ representa un costo asociado al retraso en detectar verdaderos positivos. La función es monotónicamente creciente de k:

$$lc_o(k) = 1 - \frac{1}{1 + e^{k-o}} \tag{2.15}$$

La ecuación es parametrizada por **o**, la cual controla el lugar en el eje X donde el costo crece más rápidamente. El comportamiento de la ecuación ERDE se puede observar en la figura 2.6.

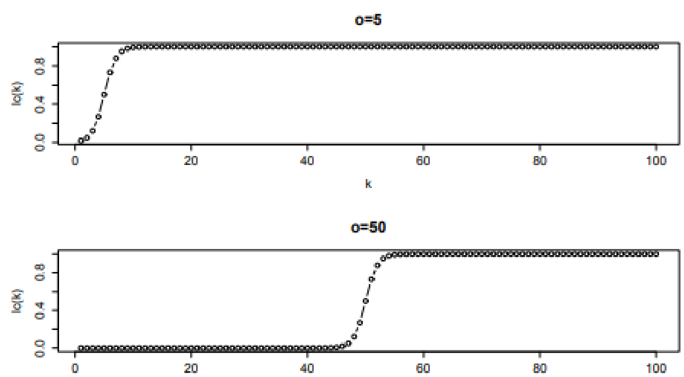

Figura 2.6: Comportamiento de medida ERDE.Imagen tomada de (Losada y cols., 2018)

Así mismo debe de tenerse en cuenta que el resultado es mejor entre más bajo sea el valor obtenido.

2.5.5. Prueba t de student

Una manera de comprobar si existe diferencia entre dos clasificadores sobre los resultados en diferentes conjuntos de datos y que dicha diferencia no es random es haciendo uso de una prueba t de student, la cual verifica si la diferencia promedio en los desempeños de los clasificadores sobre los conjuntos de datos es significativamente diferente de cero(Demšar, 2006).

Para ello se requiere la distribución t, la cual es un conjunto de curvas estructurada por un grupo de datos de unas muestras en particular. La primera presunción es formular la hipotesis nula y la hipotesis alterna, que establece que no hay diferencias en la media de las dos muestras y que de existir esta diferencia, sólo se debe al azar. Si la t calculada que se origina es mayor a una significancia p establecida (regularmente 0.05) entonces se rechaza la hipotesis nula(Sánchez Turcios, 2015).

La metodología de la prueba t de student es la siguiente:

1. Probar que cada una de las muestras tiene una distribución normal.
2. Obtener para cada uno de las muestras:
   - El tamaño de las muestras n_1 y n_2
   - Sus respectivas medias m_1 y m_2
   - Sus varianzas v_1 y v_2
3. Probar que las varianzas sean homogéneas.
4. En caso de homogeneidad en esas varianzas
   - Establecer la diferencia entre las medias: $m_1 - m_2$

- Calcular la varianza común de las dos muestras:

  $vc = ((n_1 - 1)v_1 + (n_2 - 1)v_2)/(n_1 + n_2 - 2)$

  Es decir, la varianza común (vc) es igual a un promedio pesado de las varianzas de las dos muestras en donde los pesos para ese promedio son iguales al tamaño, menos uno ($n - 1$) para cada una de las muestras

- Con esa varianza común, se calcula el error estándar de la diferencia de las medias $ESM = \sqrt{((vc)(n_1 + n_2)/(n_1 n_2)}$

5. Finalmente, la prueba t de student es igual al cociente de la diferencia de medias entre el ESM anterior.

6. De acuerdo a las hipótesis nula y alterna se debe mostrar que existe diferencia entre las medias de las muestras.

En este trabajo se hace una comparativa de desempeño en las redes neuronales entre la función de entropía cruzada y la nueva función de pérdida enfocada en clasificación temprana, es decir, se tiene un problema de antes y después del uso de la nueva función en las redes neuronales, por lo que la prueba t de student se utiliza para comprobar que existe diferencia al usar la nueva función de pérdida.

Capítulo 3

Estado del arte

A continuación, se hace una revisión de trabajos en los que se maneja el concepto de ETC, así como de algunos en los que no se clasifica de manera anticipada, sin embargo, éstos utilizan redes neuronales para resolver algún problema en los que sería de utilidad clasificar de forma temprana.
Los trabajos pueden ser agrupados de la siguiente forma independientemente de la representación de texto o características que utilicen:

1. Trabajos en los que se utilizan clasificadores tradicionales.

2. Trabajos en los que se utilizan redes neuronales.

De igual modo es necesario aclarar que la mayoría de estos trabajos se enfoca en la detección de depresión y anorexia.

3.1. Métodos basados en técnicas estándar de clasificación

Se caracterizan por hacer uso de clasificadores como pueden ser SVM, Naïve Bayes, árboles de decisión, regresión logística, entre otros. De igual modo la mayoría de éstos requieren de la búsqueda de características específicas del dominio o en su caso de buscar nuevas representaciones del texto. A continuación, se describen estos trabajos y los resultados obtenidos.

En la detección de depredadores sexuales se tiene el trabajo de (Escalante y cols., 2016), cuyo enfoque se basa en una modificación al algoritmo de Naïve Bayes de modo que se pueda clasificar temprano sin importar la cantidad de información que se tenga. La idea consiste en utilizar una distribución multinomial para la clasificación, que en el caso de texto se puede obtener usando una representación de bolsa de palabras (BoW) con un pesado de frecuencia de términos para los documentos. La fase de entrenamiento no difiere al realizado en el procedimiento estándar con Bayes. Sin embargo, durante la fase de clasificación la lectura de los términos del documento se hace de forma secuencial, y en determinados lapsos se obtiene la representación de texto mencionada anteriormente. Para probar la eficacia del método se hace una comparativa contra el clasificador SVM usando el corpus de depredadores sexuales presentado en PAN 2012[1] y descritos a detalle en (Inches y Crestani, 2012), así como la métrica F1 con diferentes cantidades de información durante la fase de prueba. Los resultados reportados indican que el desempeño del algoritmo Bayes modificado supera SVM, ya que cuando se maneja 10 % de la información se obtiene un valor de F1 del 0.53 y 0.31 respectivamen-

[1] https://pan.webis.de/clef12/pan12-web/author-identification.html

te. Nótese que en el trabajo anterior no se depende de alguna característica del dominio, sin embargo, se utiliza una representación en específico para el correcto funcionamiento del algoritmo.

Por otro lado, existen trabajos en los que se proponen nuevas representaciones de texto, por ejemplo en el caso de (Escalante y cols., 2017), se propone aplicar una representación de texto que consiste en aprender patrones de simples estadísticas de ocurrencia. La idea consiste en generar representaciones vectoriales de baja dimensionalidad, similares a los *embeddings*, por cada documento. Para probar su eficacia utilizan diferentes clasificadores clásicos (como SVM o Naïve Bayes), el corpus de depredadores sexuales presentado en PAN 2012, además de usar diferentes cantidades de información y la métrica F1. Los resultados obtenidos indicaron que si bien el método no aporta nada cuando se trata de clasificación utilizando información completa (se obtienen valores de F1 del 85 %), su desempeño mejora de forma considerable al usar información de forma parcial puesto que obtiene un valor de F1 de 0.60 al usar 10 % de información superando a las representaciones tradicionales de texto que obtienen un 0.25 en F1 al usar la misma cantidad de información.

Otro trabajo en el que se utilizan nuevas representaciones de texto se ve en (López-Monroy y cols., 2018), en donde se intenta detectar de forma temprana problemas de depresión y depredadores sexuales, para ello se presenta la idea de generar representaciones vectoriales de las palabras del corpus y después aplicar un algoritmo de clustering para generar un determinado número de clusters, posteriormente se crea una representación del documento usando las frecuencias de palabras que aparecen en los grupos generados. Para probar su método se utilizan los corpus presentados en eRisk 2017[2] descritos en (Losada y cols., 2017) y PAN 2012, diferentes números de clusters con cantidades variadas de información en un clasificador SVM comparándolo con otras representaciones vectoriales. Los resultados obtenidos en este trabajo fueron favorables tanto en depresión como en depredadores sexuales, ya que se obtienen valores de F1 de 0.48 y 0.71 respectivamente; mientras que con representaciones como la bolsa de palabras se obtienen valores de 0.37 y 0.27

Un ejemplo más se puede ver en (Funez y cols., 2018), en donde se pretende detectar depresión y anorexia de forma temprana conforme las reglas y datos presentados en eRisk 2018[3] y descritos en (Losada y cols., 2018). Para ello se presenta una representación textual que no depende de un dominio en específico, ya que se propone el uso de dos métodos: el primero consiste en usar una representación vectorial reducida que interpreta palabras y fragmentos de texto en un espacio de conceptos que son cercanos a las clases que se desea clasificar; el segundo consiste en crear un diccionario de palabras almacenando las frecuencias y usarlas posteriormente para calcular la relación que existe con las clases. Los mejores resultados se obtuvieron usando un clasificador SVM para el problema de depresión con un 8.78 de la métrica $ERDE_5$, 7.24 de $ERDE_{50}$ y 0.60 en F1. Para el caso de anorexia los resultados fueron de 11.40 en $ERDE_5$,7.82 en $ERDE_{50}$ y 0.79 en F1 usando regresión logística.

Por otra parte, se tienen ejemplos de trabajos en el que se utilizaron características específicas de un dominio; como en el caso de (Ramiandrisoa y cols., 2018), que igual fue presentado en eRisk 2018. En este ejemplo se extraen más de 50 ca-

[2]https://early.irlab.org/2017/index.html
[3]https://early.irlab.org/2018/index.html

racterísticas que van desde las léxicas como son signos de puntuación, promedios de *posts*, pronombres, entre otras. Así como características específicas de los males a detectar como son síntomas de depresión, sentimientos, medicamentos relacionadas y otras. De igual modo se generan representaciones vectoriales de texto a nivel frase que son combinadas con las características extraídas, y, ambas son utilizadas como entrada para clasificadores random forest y regresión logística. Los resultados obtenidos demostraron que el uso de características específicas mejora el desempeño en comparación a solamente usar representación de texto ya que obtienen resultados en las mátricas $ERDE_5$ de 9.46, $ERDE_{50}$ de 7.09 y F1 de 0.55 para el caso de depresión, mientras que para el caso de anorexia se obtienen resultados de $ERDE_5$ de 12.78, $ERDE_{50}$ de 10.33 y F1 de 0.76.

Otro ejemplo de trabajo de depresión y anorexia del eRisk 2018 que se basa en la extracción de características específicas se puede ver en (Ortega-Mendoza y cols., 2018), cuya idea principal recae en el concepto de que la información personal tiene una importante relevancia para el modelado de comportamientos o desórdenes mentales. Para probar lo dicho utilizan un clasificador SVM alimentado por los términos más relevantes presentes en frases personales en primera persona. De igual modo hacen una modificación a la representación de texto dándole mayor valor a los términos de frases personales. Los resultados obtenidos para depresión son $ERDE_5$ de 10.07, $ERDE_{50}$ de 7.22 y F1 de 0.45, mientras que para anorexia se tiene valores de $ERDE_5$ de 12.41, $ERDE_{50}$ de 7.79 y F1 de 0.67, por lo que demostraron que el uso de información personal mejora los resultados en la clasificación temprana, incluso resulta mejor la selección de dichos términos que la modificación a la representación textual con diferentes valores.

3.1.1. Discusión

Como se puede ver en el resumen anterior, la mayoría de estos trabajos dependen de la extracción de características en específico (de un dominio en particular o del texto) o en su caso de nuevas representaciones para su correcto funcionamiento (Jurafsky y Martin, 2019).

Cabe destacar que el uso de características específicas de un dominio limita el funcionamiento de estos trabajos, debido a que no se garantiza que éstas se presenten en todos los problemas de clasificación temprana.

Por otro lado, aquellos trabajos que proponen una nueva representación se enfrentan a un problema en su propia solución, es decir, encontrar la configuración adecuada para la representación del texto, aunado a la selección del clasificador a usar. Además de que pierden la característica secuencial del texto.

Tratando de evitar las situaciones anteriores, en este trabajo se hace uso de representaciones tipo *embedding* en los que se aprovecha la característica secuencial del texto sin depender de un dominio.

3.2. Métodos basados en redes neuronales

Estos trabajos se caracterizan por usar redes neuronales como clasificadores, así como de buscar estructuras y configuraciones adecuadas para el problema que intentan resolver. Además de que algunos de éstos hacen comparativas de desempeño de las redes contra clasificadores clásicos.

En el caso de la detección temprana de depresión usando los datos de eRisk 2017, se tiene en (Sadeque y cols., 2017) la propuesta de usar tres enfoques diferentes: el primero consiste en implementar un modelo de tipo no secuencial con un clasificador SVM utilizando características léxicas y médicas a través de un lexicón que contiene una lista de las palabras más relacionadas directamente con depresión. El segundo consiste en el uso de una red neuronal recurrente que recibe como entrada las características indicadas en el enfoque anterior, pero de manera secuencial. Por último, el tercer enfoque consiste en combinar las dos ideas anteriores en un ensamble usando Naïve Bayes. Los resultados arrojados indicaron que el uso del ensamble resulta mejor, ya que se obtienen valores de $ERDE_5$ con 14.73, $ERDE_{50}$ de 10.23 y F1 de 0.45, los cuales resultan superiores a los reportados cuando se usan enfoques individuales.

Otro análisis para detección temprana de depresión se realizó en (Maupomé y Meurs, 2018) con los datos de eRisk 2018, con la idea de extraer tópicos del texto basándose en el hecho de que los temas de discusión en los que una persona participa indican su estado mental. Para probar la eficacia de la propuesta se utilizó un perceptrón apilado sin ninguna configuración en especial. Los resultados obtenidos usando diferentes cantidades de información indicaron que los tópicos son una buena herramienta para la detección de depresión, sin embargo, el poder predictivo del perceptrón se vio afectado al tener pocos ejemplos de entrenamiento ya que se obtienen resultados en $ERDE_5$ de 10.04, $ERDE_{50}$ de 7.85 y F1 de 0.42, los cuales son inferiores a los mejores reportados en eRisk 2018.

Por otro lado, se tiene el análisis realizado en (N. Liu y cols., 2018) con el objetivo de clasificar depresión y anorexia de forma temprana usando los datos de eRisk 2018. Para ello se proponen tres enfoques: el primero consiste en detectar palabras clave presentes en el texto para poder clasificar, siendo la palabra clave “body” en el caso de anorexia y “depression” en el caso de depresión; el segundo enfoque hace uso de una representación de bolsa de palabras con un clasificador SVM, y finalmente el tercer enfoque representa las palabras en vectores de baja dimensionalidad que alimentan de forma secuencial una red convolucional, cuya salida es procesada a su vez por una red recurrente. Los resultados obtenidos mostraron que el detectar palabras clave es de mayor utilidad en el caso de depresión, mientras que el uso de la red neuronal mostró mejor desempeño en la detección de anorexia. Sin embargo, ninguno de los enfoques resulto favorable para clasificación temprana cuando se tiene poca información, ya que se obtienen resultados de $ERDE_5$ de 10.4, $ERDE_{50}$ de 9.54 y F1 de 0.27 para depresión, mientras que $ERDE_5$ de 13.53, $ERDE_{50}$ de 12.57 y F1 de 0.36 para el caso de anorexia, lo cual es inferior a los reportados por otros trabajos.

Otro trabajo al que se puede hacer mención, en el que igual se hace uso de los datos de eRisk 2018, es el de(Wang y cols., 2018), en el cual se proponen crear vectores que representen las oraciones del texto. Para ello se utiliza una representación de bolsa de palabras, y se conservan sólo las 300 palabras con los mayores valores en la representación. Posteriormente se utiliza una red convolucional alimentada secuencialmente por los vectores de las palabras seleccionadas. Es necesario destacar que para determinar si un usuario padece algún mal se utilizan diferentes umbrales de confianza al momento de clasificar con diferentes cantidades de información. Los resultados obtenidos mostraron un poco de inconsistencia, según los autores, ya que aquellos trabajos que tienen un buen resultado en métrica F1, obtienen un desempeño bajo en $ERDE_5$. Los resultados obtenidos para la tarea de depresión para $ERDE_5$ y $ERDE_{50}$ fueron de 10.81 y 9.22 respectivamente con un valor de F1 de 0.37, mientras que para anorexia en $ERDE_5$ y $ERDE_{50}$ fueron de 12.93 y 9.85 respectivamente y con un valor de F1 de 0.67.

Con una solución diferente para la misma tarea de detección temprana de depresión y anorexia del eRisk 2018, en (Trotzek y cols., 2018) se propone el uso de una red convolucional introduciendo palabras de forma secuencial representadas por vectores de baja dimensionalidad. Lo interesante de este trabajo recae en la selección de texto que se aplica, ya que preserva emociones, caracteres especiales y en general todas aquellas palabras que ocurren al menos dos veces en los ejemplos. Los resultados reportados indicaron un buen desempeño de la red convolucional para la detección temprana en ambas tareas a pesar de los pocos ejemplos que se tenían para el entrenamiento de la red, ya que se obtienen valores de $ERDE_5$ y $ERDE_{50}$ de 9.21 y 6.44, así como un valor F1 de 0.64 para el caso de depresión; mientras que para el caso de anorexia se reportan valores de $ERDE_5$ y $ERDE_{50}$ de 11.75 y 5.96 respectivamente, además de un F1 de 0.85, los cuales son de los mejores valores reportados en el eRisk 2018. Cabe destacar que anteriormente los autores hicieron un trabajo similar en (Trotzek y cols., 2017) con los datos de eRisk 2017, sin embargo, en ese caso hacen uso de una red recurrente con la idea de explotar la información secuencial. Los resultados obtenidos con la red recurrente son de $ERDE_5$ de 12.7, $ERDE_{50}$ de 9.69 y F1 de 0.64. Sin embargo, decidieron cambiar la red recurrente debido a que la red convolucional es menos complicada en su configuración, sin embargo no por ello se desmerita el buen desempeño alcanzado por la red recurrente.

Los trabajos anteriores se centraron en la detección temprana, sin embargo, existen otras investigaciones de clasificación usando redes neuronales que vale la pena mencionar, ya que tratan de resolver problemas que bien pueden ser atacados con el concepto de ETC o que tienen cierta relación al momento de buscar la solución.

Por ejemplo, se tiene el trabajo de (Shrestha y cols., 2017), el cual se enfoca en la detección de autoría con textos cortos, es decir, se tiene poca información para lograrlo, lo cual lo relaciona con uno de los problemas que tiene ETC. La idea de este trabajo es usar una red convolucional que recibe como entrada una secuencia de representaciones vectoriales de n-gramas de caracteres, lo cual lo diferencia de los métodos tradicionales ya que comúnmente se usan palabras completas o sólo caracteres. Para comprobar la efectividad de su método se utiliza un corpus de extraido de Twwiter[4] mostrado en (Schwartz y cols., 2013), además de hacer diferentes experimentos variando la cantidad de información y los clasificadores, como redes recurrentes, convolucionales o regresión logística, así como diferentes representaciones como secuencias vectoriales de palabras o secuencias vectoriales de caracteres. Los resultados obtenidos mostraron que la clasificación con CNN utilizando secuencia de n-gramas de caracteres en vez de sólo caracteres es mejor a pesar de variar la cantidad de información.El mejor resultado reportado en este trabajo es de un 0.76 de exactitud.

Por otro lado, se tienen trabajos en los que se quiere detectar conversaciones sospechosas sin importar la cantidad de información que se tenga, por ejemplo en el caso de (D. Liu y cols., 2017) se pretende detectar depredadores sexuales. Para ello primeramente generan un modelo de lenguaje usando secuencias de palabras representadas con vectores de baja dimensionalidad en una red neuronal

[4]https://twitter.com/

recurrente LSTM con el objetivo de que aprenda la dependencia de las palabras en las oraciones. Posteriormente se utiliza la salida de dicha red recurrente para generar vectores que representen a las oraciones. Para la parte correspondiente a la clasificación nuevamente se utiliza una red recurrente que recibe como entrada los vectores generados. Para probar la eficacia de su modelo, hacen uso del corpus de PAN 2012, comparando sus resultados con los mejores reportados en dicha conferencia. Los resultados obtenidos fueron favorables, ya que se obtiene un F1 de 0.89, con lo cual determinaron que el reducir una conversación a oraciones más simples mejora considerablemente los resultados al momento de clasificar.

Un último ejemplo se puede ver en (Husseini Orabi y cols., 2018), en cuyo trabajo se pretende detectar depresión en textos cortos utilizando redes convolucionales y recurrentes LSTM bidireccionales con diferentes configuraciones y usando representaciones vectoriales de baja dimensionalidad optimizados. Para probar la eficacia de su propuesta, hacen uso de datos de Twitter mostrados en (Coppersmith y cols., 2015). Los resultados arrojados muestran desempeños satisfactorios en ambas redes neuronales, sin embargo, la red convolucional supera a la recurrente, al obtenener estas métricas de F1 de 0.87 y 0.78 respectivamente, por lo que los autores sugieren la búsqueda de diferentes mecanismos y configuraciones que exploten el potencial de las redes recurrentes.

Para finalizar el capítulo se puede decir que con las descripciones de los trabajos relacionados se ha logrado obtener una visión de los diferentes enfoques con los que ha sido tratado el problema de clasificación temprana en diferentes tareas, así como el uso que se la ha dado a las redes neuronales en ello. La tabla 3.1 muestra un resumen con las principales características de los trabajos citados.

Estado del arte					
Autor	Representación del texto o características	Clasificadores	Tareas	Datos usados	Resultados reportados
(Escalante y cols., 2016)	Bolsa de palabras	Naive Bayes	Detección de depredadores sexuales	PAN 2012	F1=0.53 (10 % de informacion)
(Escalante y cols., 2017)	Representaciones tipo embedding	SVM Naive Bayes	Detección de depredadores sexuales	PAN 2012	F1= 0.60 (10 % de información)
(López-Monroy y cols., 2018)	Vectores basado en frecuencia de palabras	SVM	Depresión y depredadores sexuales	PAN 2012 eRisk 2017	F1=0.48 F1=0.71 (10 % de de información)
(Funez y cols., 2018)	Vectores reducidos y diccionario de palabras	SVM Regresión logística	Depresión y anorexia	eRisk 2018	ERDE5=8.78 ERDE50=0.60 F1= 0.60 ERDE5=11.40 ERDE50= 7.82 F1= 0.79
(Ramiandrisoa y cols., 2018)	Vectores de frases y características de dominio	Random forest regresión logística	Depresión y anorexia	eRisk 2018	ERDE5=9.46 ERDE50=7.09 F1=0.55 ERDE5=12.78 ERDE50=10.33 F1=0.76
(Ortega-Mendoza y cols., 2018)	Selección de términos relevantes	SVM	Depresión y anorexia	eRisk 2018	ERDE5=10.07 ERDE50= 7.22 F1=0.45 ERDE5=12.41 ERDE50=7.79 F1=0.67
(Sadeque y cols., 2017)	Características léxicas y médicas	SVM Red recurrente Naive Bayes	Depresión	eRisk 2017	ERDE5=14.73 ERDE50=10.23 F1=0.45
(Maupomé y Meurs, 2018)	Extracción de tópicos	Perceptrón multicapa	Depresión	eRisk 2018	ERDE5=10.04 ERDE50=7.85 F1=0.42
(N. Liu y cols., 2018)	Selección de palabras claves Bolsa de palabras	SVM Red convolucional Red recurrente	Depresión y anorexia	eRisk 2018	ERDE5=10.4 ERDE50=9.54 F1=0.27 ERDE5=13.53 ERDE50=12.57 F1=0.36
(Wang y cols., 2018)	Selección de palabras frecuentes y bolsa de palabras	Red convolucional	Depresión y anorexia	eRisk 2018	ERDE5=10.81 ERDE50=9.22 F1=0.37 ERDE5=12.93 ERDE50=9.85 F1=0.67
(Trotzek y cols., 2017)	Selección de emociones, caracteres especiales, embeddings	Red recurrente	Depresión	eRisk 2017	ERDE5=12.7 ERDE50=9.69 F1=0.64
(Trotzek y cols., 2018)	Selección de emociones, caracteres especiales, embeddings	Red convolucional	Depresión y anorexia	eRisk 2018	ERDE5= 9.21 ERDE50=6.44 F1=0.64 ERDE5=11.75 ERDE50=5.96 F1=0.85
(Shrestha y cols., 2017)	Embeddings de n-gramas	Red convolucional	Autoría de textos cortos	Twitter	Accuracy=0.76
(D. Liu y cols., 2017)	Embeddings	Red recurrente	Conversaciones sospechosas	PAN 2012	F1=0.89
(Husseini Orabi y cols., 2018)	Embeddings optimizados	Red convolucional Red recurrente	Depresión en textos cortos	Twitter	F1=0.87

Tabla 3.1: Trabajos relacionados y sus principales características

3.2.1. Discusión

Como se puede ver en los trabajos anteriores, varios de éstos se apoyan nuevamente de la extracción de características específicas del dominio o de nuevas representaciones de texto, aunque en este caso algunos sí explotan el concepto secuencial al momento de alimentar las redes neuronales, sin embargo, en la mayoría de los casos lo manejan como secuencias de palabras (el enfoque comúnmente usado).

Por otro lado, se aprecia que los modelos propuestos en estos trabajos buscan diferentes configuraciones y combinaciones de redes neuronales tratando de encontrar la mejor; en este trabajo de tesis se hace algo similar al buscar diferentes configuraciones tanto en las redes neuronales como en la forma de alimentarlas con los *embeddings* del texto. Además de que se considera la nueva función de pérdida para la mejora en la clasificación temprana sin tener que depender de características de un dominio.

Capítulo 4

Método propuesto

Como ya se ha mencionado, es necesaria la búsqueda de nuevas alternativas de clasificación de texto de modo que sean aplicables en estados tempranos cuando se tiene poca información. Es por ello que en este trabajo se hace uso de redes neuronales alimentadas de diferentes formas considerando la característica cronológica secuencial que existe en el texto.

De igual modo, considerando que la mayoría de los trabajos con redes neuronales del estado del arte hacen uso de redes con *embeddings* a nivel palabra debido a su simplicidad, buen desempeño, fácil configuración y que además conserva el orden de las palabras, en este trabajo de tesis se decidió explorar el uso de los *embeddings* en diferentes niveles: *embeddings* de palabras (el comúnmente usado), *embeddings* de fragmentos (promedio de los *embeddings* de las palabras dado un fragmento), *embeddings* de historial de *posts* o pedazos(promedio de todos los *embeddings* de las palabras dado un historial de *posts*). Un diagrama con las diferencias de dichos enfoques se muestra en la figura 4.1. Así mismo, el motivo por el que se decidió probar promedios de *embeddings* se debe a que como se menciona en (Kenter y cols., 2016), el promediar todos los *embeddings* de las palabras ha demostrado ser una técnica muy eficiente, además de mostrar buenos resultados en cuestiones de semántica y analogías, y con la ventaja de ser un *embedding* más facil de procesar (Coates y Bollegala, 2018), aunque si se considera el hecho de que se desea aprovechar la característica cronológica secuencial del texto, se decidió probar diferentes cantidades de palabras al momento de promediar de modo que se preservara dicha característica del texto.

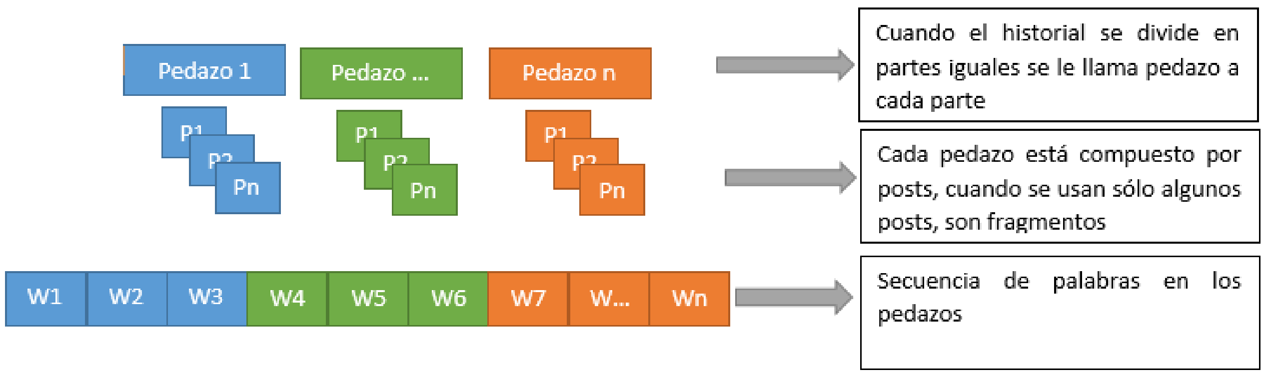

Figura 4.1: Diferencias en los diferentes enfoques usados en esta tesis.

Así mismo, se propone la modificación de las redes neuronales convolucionales y LSTM aplicando una nueva función de pérdida que fomenta la clasificación temprana cuando se tiene poca información, ya que en estados iniciales el poco texto presente puede ocasionar diversas interpretaciones que afecten las decisiones de las redes al clasificar. Con esta solución se evita el enriquecimiento manual de características, por lo que se permite su aplicación en diversos dominios.

Téngase en cuenta que para la aplicación de los métodos propuestos se considera que se posee un corpus ordenado cronológicamente y que los ejemplos que se tienen son lo suficientemente extensos para poder dividirlos en subpartes y poder generar embeddings de diversos tamaños. Los corpus utilizados para probar los métodos propuestos se encuentran descritos a detalle en la sección 5.1 y se centran en los problemas de anorexia, depresión y depredadores sexuales.

En el presente capítulo se describen los diferentes enfoques propuestos. Primeramente, se inicia con el enfoque nivel palabra, en el cual se hace uso de los *embeddings* de las palabras de manera secuencial alimentando las redes con diferentes cantidades. El siguiente enfoque propuesto considera el promediar fragmentos a manera de secuencias de ideas. El último enfoque genera nuevamente *embeddings* promedio, pero en este caso con el objetivo de crear secuencias a nivel pedazo a manera de documentos. Así mismo, en todos los casos se hace uso de los *embeddings* generados con Word2Vec y FastText, siendo que el caso del primero se decidió utilizar debido a ser uno de los modelos más populares de *embeddings*, mientras que FastText por su utilidad en los casos de vocabulario de redes sociales. Es necesario indicar que para los enfoques nivel palabra y pedazo se hace uso de los corpus de depresión, anorexia y depredadores sexuales. Mientras que el nivel fragmento sólo usa los corpus de depresión y anorexia.

4.1. Enfoque a nivel palabra

Este enfoque es el comúnmente usado en redes neuronales aplicadas a texto debido a su fácil configuración, buenos resultados y por conservar el orden de las palabras, por lo que se decidió hacer uso de éste como primera aproximación. La idea consiste en crear secuencias de un determinado tamaño con los *embeddings* de las palabras. Este enfoque ha sido usado en otros trabajos de clasificación de texto, sin embargo, depende del número de ejemplos que se tenga al momento de entrenar, además de que tiene la limitante del número de palabras en secuencia que se puede usar, por lo que se puede presentar pérdida de información. Un diagrama de alimentación a la red neuronal (LSTM o CNN) se muestra en la figura 4.2, en la que se aprecia como las palabras son mapeadas a una representación vectorial a través de la capa *embedding* para posteriormente pasar a la red LSTM o CNN y clasificar el ejemplo como positivo o negativo (ya sea de depresión, anorexia, depredador sexual) según sea el caso. Cabe destacar que usando este enfoque se realizaron dos versiones, los cuales se describen a continuación.

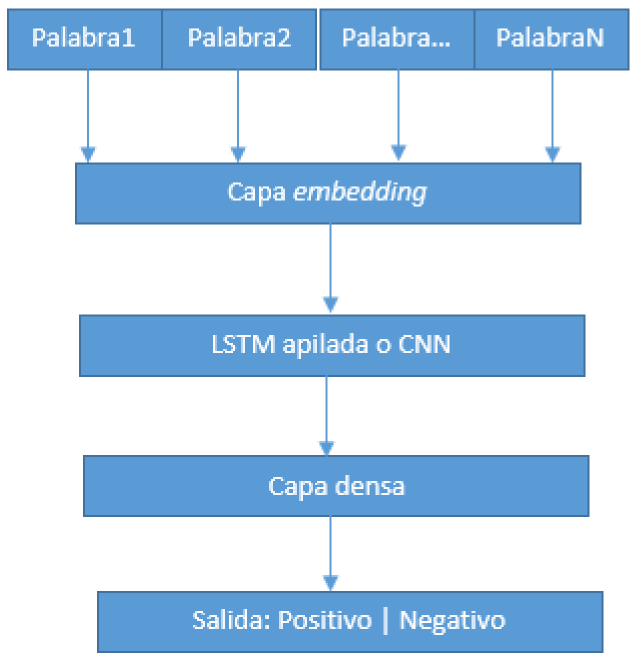

Figura 4.2: Alimentación de la red con enfoque a nivel palabra.

4.1.1. Secuencia larga de palabras

El texto es una serie cronológica de palabras, en si una secuencia, por lo que manejar el texto completo como una secuencia larga de datos es algo intuitivo de hacer. Por tal motivo, este experimento se centra en la idea de usar la mayor cantidad de palabras en secuencia posibles para alimentar la red neuronal. Sin embargo, las redes neuronales usadas en este trabajo tienen la limitante de que es necesario indicar un número fijo de palabras a usar y considerando el hecho de que el texto tiene un tamaño variable, al fijar una cantidad de palabras en algunos casos se trunca información mientras que en otros casos falta.

Teniendo en cuenta lo anterior, fue necesario determinar el número de *embeddings* que alimentan la red. Para ello, se calculó el número de palabras promedio de los ejemplos y se redondeó a una cantidad aproximada, siendo que para el caso de los corpus de depresión y anorexia se usaron secuencias de 10000 palabras, mientras que en el caso de depredadores sexuales se usaron secuencias de 1000. Un diagrama de cómo se alimenta la red con este enfoque se muestra en la figura 4.3

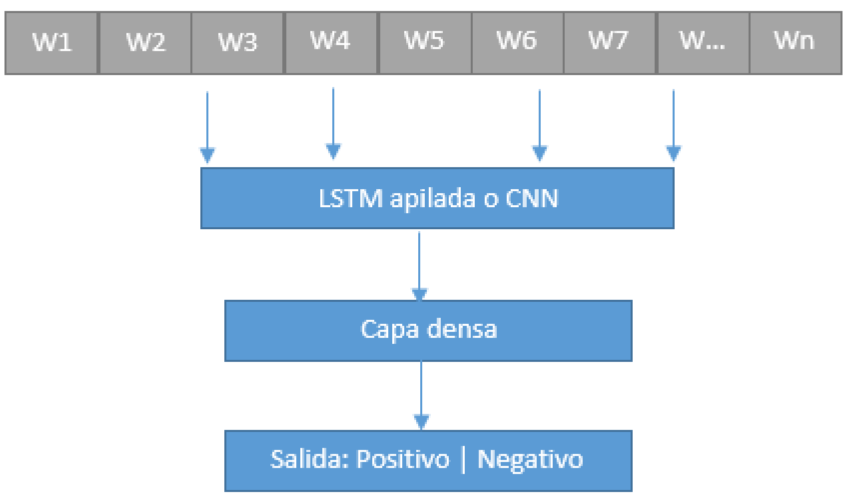

Figura 4.3: Alimentación de la red con secuencia larga de palabras. Nótese que se utiliza una secuencia de palabras del historial completo.

Al momento de generar las secuencias que alimentarán la red durante la fase de entrenamiento se van indicando los *embeddings* de las palabras, ignorando aquellas que no cuentan con un *embedding* asociado.

Por otra parte, debido a que se desea determinar qué tan bien clasifica la red neuronal de forma temprana durante la fase de prueba, es necesario clasificar con información parcial. Esto se logra aprovechando la división en pedazos de los ejemplos de los corpus. Para ello se generan 10 secuencias de 1000 (para los casos de depresión y anorexia) y 100(para el caso de depredadores sexuales) palabras, que representen los pedazos de los ejemplos de los corpus. Posteriormente la red es alimentada con cada una de esas secuencias de palabras de modo que vaya clasificando y acumulando la información que va recibiendo; es decir, primero se intenta clasificar generando una secuencia de 1000 de un solo pedazo, posteriormente con dos pedazos y una secuencia de 2000 y así sucesivamente hasta llegar a los 10000 para los casos de anorexia y depresión, téngase en consideración que al ir alimentando la red con información parcial la entrada de la red no queda cubierta completamente, es decir, por ejemplo, en el caso del primer pedazo, se generan 1000 palabras, pero la entrada de la red requiere 10000, por lo que los 9000 espacios de entrada deben de ser completados con 0. Para el caso de depredadores sexuales la idea es la misma, pero va incrementando de 100 en 100 hasta llegar a las 1000.

4.1.2. Secuencia corta de palabras

Debido a que las secuencias de palabras eran demasiado grandes al momento de procesar, además de que en los casos anorexia y depresión no se cuenta con una gran cantidad de ejemplos para el entrenamiento de las redes, se decidió crear subsecuencias (a modo de subejemplos) de palabras para alimentar la red, explotando el hecho de que los ejemplos de los corpus están divididos.

La idea consiste en crear subsecuencias de palabras aprovechando que los corpus estaban divididos en 10 partes iguales. Por lo que por cada parte o pedazo de los ejemplos de los corpus se crearon subsecuencias de un determinado número, siendo que para el caso de depresión y anorexia se manejaron subsecuencias de tamaño 1000, mientras que en el caso de depredadores sexuales subsecuencias de 100. Al hacer esto se creaban 10 subejemplos por cada ejemplo del corpus para la fase de entrenamiento. Un diagrama de como se crean subsecuencias y se alimenta la red se aprecia en la figura 4.4

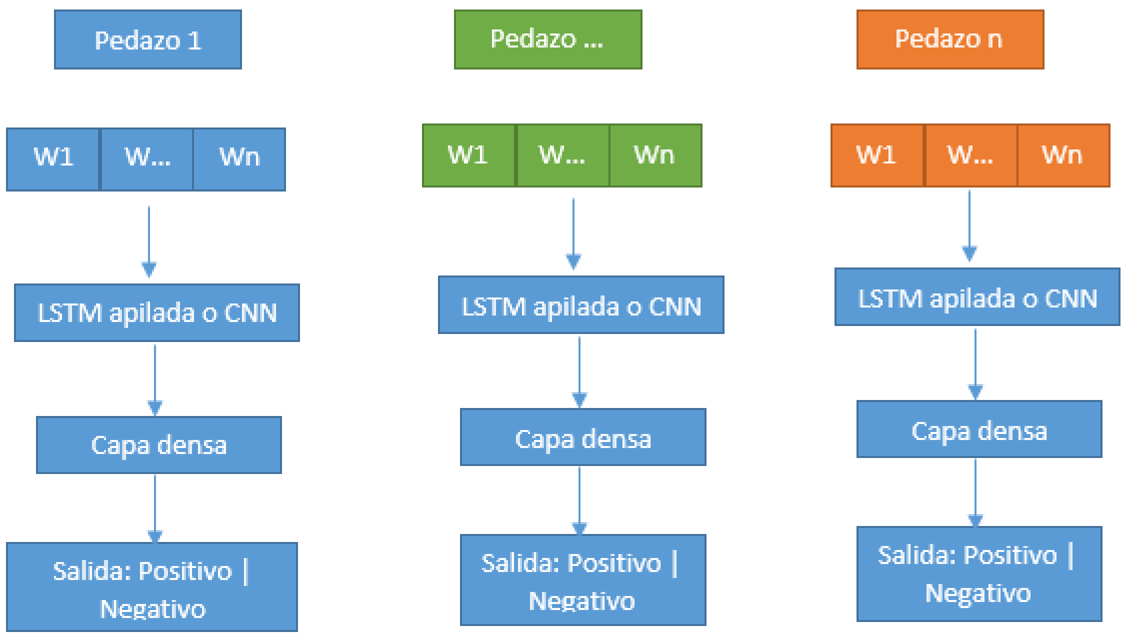

Figura 4.4: Alimentación de la red con secuencia corta de palabras. Nótese que para cada pedazo se crea una secuencia determinada de palabras que sirve como ejemplo para entrenar la red..

Para esta versión la forma de alimentar la red durante la fase de entrenamiento es prácticamente igual al anterior enfoque, con la única diferencia de que son secuencias más pequeñas y más ejemplos para entrenar la red, por lo que el procesamiento de las secuencias es más eficaz , además de que al dividir la secuencia original en subsecuencias, se pretende que en éstas se encuentre información relevante que pueda ser ignorada cuando se usan secuencias largas.

Así mismo, como en el caso de la versión anterior, la red neuronal debe de clasificar con información parcial, por lo que de igual modo se generan secuencias de 1000 y 100 palabras que representen los pedazos de los ejemplos de la prueba, sin embargo, a diferencia de la versión anterior, la información no se va acumulando, debido a que la información de entrada de la red se limpia en cada pedazo.

4.2. Enfoque a nivel fragmento

En el enfoque anterior se utilizaron secuencias de palabras con la idea de que todo el texto está relacionado bajo un mismo tema, sin embargo, si se considera la naturaleza de los corpus de anorexia y depresión, los cuales están divididos por *posts*, se tiene entonces secuencias de ideas que pueden estar o no relacionadas, por lo que el enfoque a nivel fragmento trata de explotar esta característica al crear *embeddings* promedio para alimentar la red neuronal.

De una forma general la idea consiste en crear varios *embeddings* de una determinada cantidad de texto y organizarlos secuencialmente para alimentar la red neuronal. Cabe destacar que este enfoque no se utilizó en el problema de depredadores sexuales debido a que este corpus está dividido en 10 partes iguales considerando las palabras totales, mientras que los corpus de anorexia y depresión se dividen en consideración de los *posts* hechos por los usuarios. Un diagrama de cómo se alimenta la red en este enfoque se puede ver en la figura 4.5.

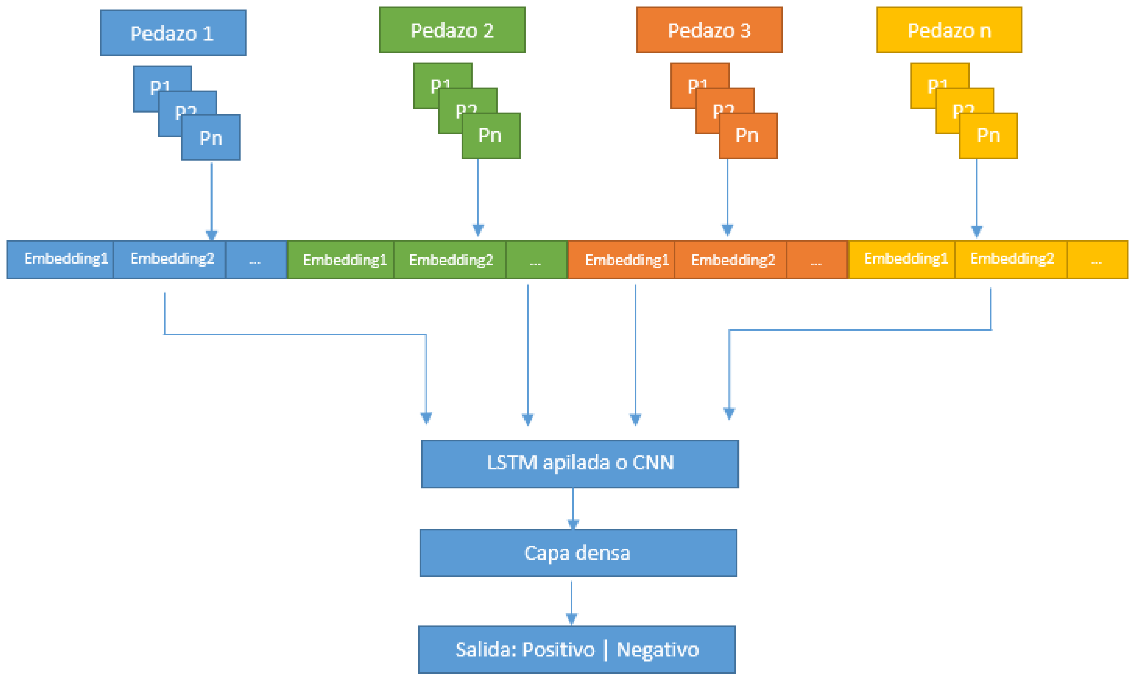

Figura 4.5: Alimentación de la red cuando se usan embeddings promedio con los fragmentos creados.

Téngase en cuenta que el principal problema que se presenta en este enfoque, es determinar el número de *embeddings* promedio a generar, ya que, en este trabajo se hace considerando el número de *posts* o en partes más pequeñas como se describe más adelante, sin embargo, se tiene el problema de determinar el número de promedios a generar, el cual podría ser basándose en la mayor cantidad de oraciones, el promedio de oraciones, un determinado número de palabras, entre otros.

4.2.1. Promedio de *posts*

La idea de esta versión es crear secuencias de *embeddings* promedio considerando la media aritmética de *posts* que hacían los usuarios etiquetados como positivos en anorexia o depresión, siendo en ambos casos 37 el número de *posts* en cada pedazo.

El objetivo es que, para cada uno de las 10 partes de los corpus, detectar 37 *posts* hechos por los usuarios y por cada uno de estos *post* sumar y promediar los *embeddings* de las palabras correspondientes, de modo que al final se tenga una secuencia total de 370 *embeddings* promedio que alimentan a la red de manera secuencial.

Por otro lado, durante la fase de prueba, así como en el enfoque anterior y sus versiones, es necesario clasificar usando información parcial, por lo que se generan 37 *embeddings* por cada pedazo, los cuales se van acumulando conforme se va clasificando con la red neuronal. Sin embargo debe de tenerse en cuenta que la entrada de la red es de 370 y que se va llenando con los 37 *embeddings* de cada pedazo, por lo que los espacios vacios que sobran en la entrada de la red se llenan con 0.

4.2.2. División en fragmentos

Considerando que el número *posts* realizados podría variar en cada ejemplo y que habría casos en los que se perdería o faltaría información, se decidió dividir cada uno de los 10 pedazos en 10 fragmentos más pequeños y promediar los *embeddings* de las palabras correspondientes, de modo que la red recibiera 100 *embeddings* promedio de manera secuencial durante la fase de entrenamiento.

Posteriormente durante la fase de prueba se crean 10 *embeddings* promedio por cada uno de los pedazos para poder clasificar de manera parcial, de modo que la red intenta clasificar usando los 10 fragmentos de forma acumulativa, llenando con 0 los espacios vacios que no se cubren en la entrada de la red.

4.3. Enfoque a nivel pedazo

En los dos enfoques anteriores se trataron secuencias de *embeddings* de una forma un poco más particular tratando que las secuencias proporcionaran información temporal que podría estar presente, sin embargo, en ambos enfoques se requirió truncar el texto, por lo que existe pérdida o falta de información. Por un lado el enfoque a nivel palabra considera secuencias de palabras al momento de alimentar la red, sin embargo tiene la limitante del número de palabras que se pueden usar; por otro lado el enfoque a nivel fragmento genera n *embeddings* que representan un determinado número de *posts* o palabras, sin embargo,igualmente se ve afectado al momento de determinar la cantidad de texto a usar. Teniendo esto presente y para evitar la falta o pérdida de información, se decidió generar *embeddings* promedio que representaran a los pedazos en su totalidad al considerar todas sus palabras y posteriormente alimentar a las redes neuronales de forma secuencial. Téngase en cuenta que la idea es simple y es usada de diversas formas en otros trabajos de clasificación, sin embargo, no se tiende a hacerlo de forma secuencial como se hace en esta tesis. Un diagrama de cómo se alimenta la red en este enfoque se puede ver en la figura 4.6. A continuación, se describen los métodos propuestos en este caso.

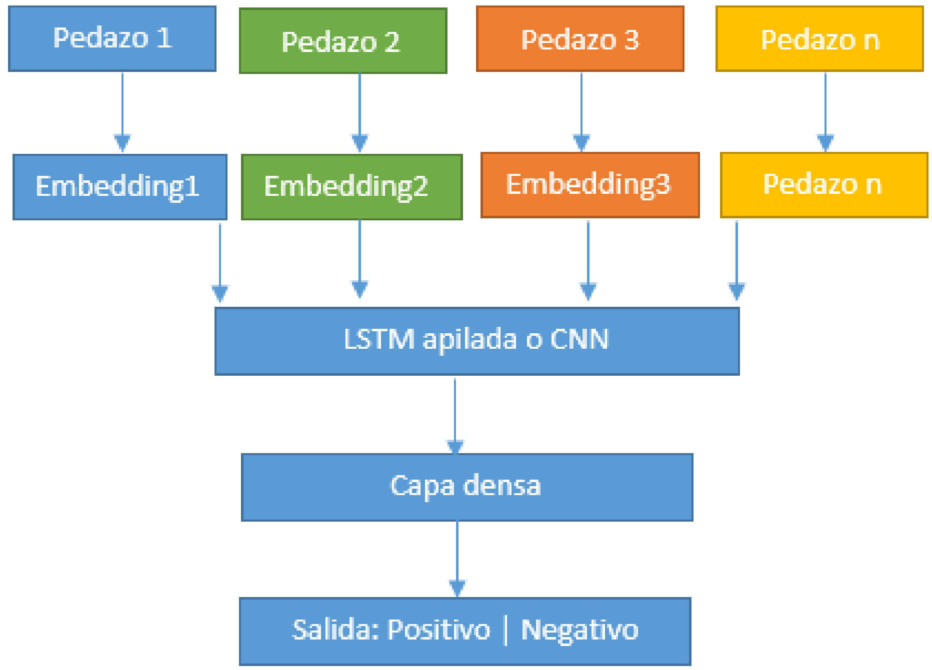

Figura 4.6: Alimentación de la red cuando se usan embeddings promedio por cada pedazo de los corpus.

4.3.1. Promedios a nivel pedazo

La idea de esta versión es sencilla, se trata de sumar y promediar los *embeddings* de todas las palabras correspondientes a cada pedazo, de modo que al final se generen 10 *embeddings* promedio en secuencia que alimentan a la red neuronal durante la fase de entrenamiento.

Para la fase de prueba, la idea es similar a los enfoques anteriores, ya que se debe de alimentar la red con información parcial de cada parte, de modo que por cada uno se crea su *embedding* promedio que alimenta a la red neuronal de forma secuencial, clasificando y acumulando la información que va recibiendo y llenando los espacios vacios con 0.

4.3.2. Promedios a nivel pedazo con entrada ajustada

Esta versión maneja la misma forma de entrenar la red neuronal que la mencionada en "promedios a nivel pedazo", es decir, crear 10 *embeddings* de promedio simple en secuencia que representan a los 10 partes de los ejemplos.

Lo diferente y nuevo de esta versión se presenta al momento de hacer la prueba, puesto que, a diferencia de las versiones anteriores, en las que con la información textual iba llegando y se creaban y agregaban los *embeddings* promedio a la entrada de la red hasta cubrir completamente la entrada de la red, en este caso se propuso cubrir por completo la entrada de la red de modo que no queden espacios vacíos que representen falta de información y con ello dificulten la clasificación. Para lograr, en la fase de prueba cuando se recibe una determinada cantidad de texto, éste es dividido en 10 partes de las cuales se crean sus *embeddings* correspondientes y se procede a la clasificación. Posteriormente cuando se recibe más información textual, esta es agregada al texto que ya se tenía y se divide nuevamente en 10 partes creando nuevos *embeddings* promedio para clasificar, es decir, siempre se mantienen 10 embeddings de entrada. Con esto se logra evitar que haya espacios que no se alimentan en la red sin importar la cantidad de información que se tenga, además de que no se presenta alguna perdida de información debido a que se utilizan todas las palabras presentes sin importar su cantidad. En el diagrama 4.7 se aprecia como se alimentan los enfoques anteriores por cada pedazo de forma parcial, nótese que la entrada de la red debe de ser llenada de 0 al faltar la información. Por otro lado, en la figura 4.8 se encuentra un diagrama de cómo se alimenta la red durante la fase de prueba, que sin importar la cantidad de texto que se tenga, se ajusta a la entrada de la red.

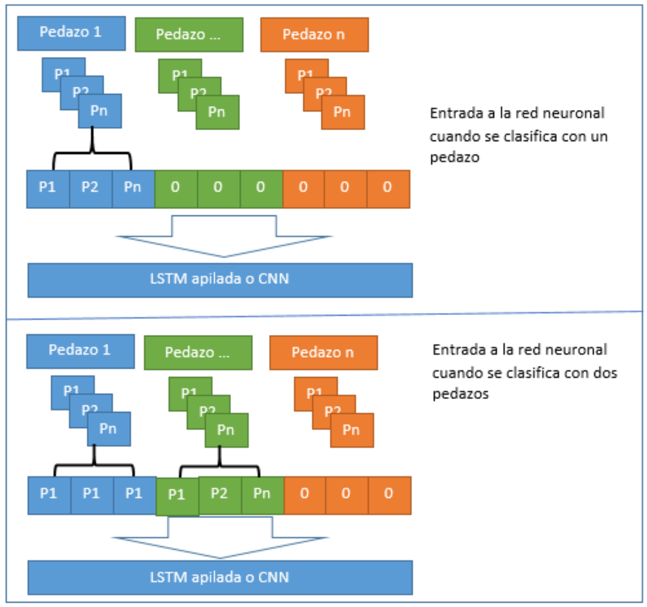

Figura 4.7: Entrada a la red cuando se clasifica de manera parcial, nótese que se deben llenar de 0 cuando falta información.

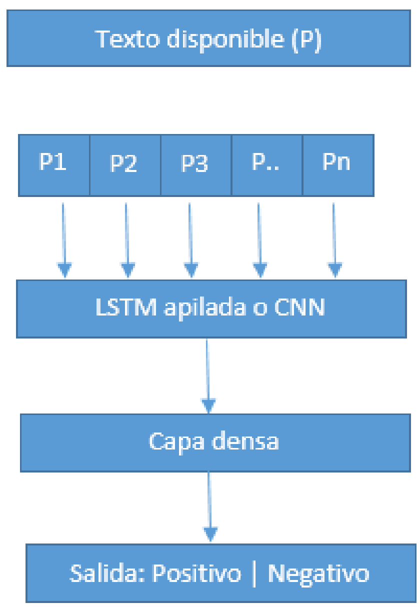

Figura 4.8: Propuesta de alimentación de la red neuronal ajustando la entrada durante la fase de prueba. Nótese que el texto disponible siempre se ajusta a la entrada de la red.

En general en la tabla 4.1 se indican de forma resumida los enfoques y las versiones que se aplicaron en cada corpus.

Enfoques y su aplicación en los corpus					
Enfoque	Versión	Depresión	Anorexia	Depredador Sexual	Ventajas
Nivel palabras	Secuencia larga de palabras	*	*	*	Fácil implementación, el comúnmente usado
	Secuencia corta de palabras	*	*	*	Incremento en el número de ejemplos para entrenar
Nivel fragmento	Promedio de posts	*	*		Secuencias más cortas
	División en partes más pequeñas	*	*		Secuencias más cortas
Nivel pedazo	Promedio de palabras de cada pedazo	*	*	*	Evita el truncamiento de información
	Promedio de palabras de cada pedazo con entrada ajustada en la red	*	*	*	Evita el truncamiento de información y ajusta la información que se tiene al momento de alimentar la red

Tabla 4.1: Enfoques y sus versiones aplicadas en los corpus.

4.4. Función de pérdida propuesta

Como se ha dicho anteriormente, se usó una función de pérdida que ha tenido resultados favorables en detección anticipada de acciones en video. Dicha función se caracteriza por fomentar la clasificación temprana, basándose en la idea de que algunas acciones son altamente ambiguas cuando al inicio sólo se ha visto poca información, por lo que falsos positivos no deberían de ser vigorosamente penalizados en estados tempranos.

Teniendo en mente esto, la función de pérdida es la que se propone en (Aliakbarian y cols., 2017) y está definida en la ecuación 4.1.

$$L(y,\hat{y}) = -\frac{1}{N}\sum_{k=1}^{N}\sum_{t=1}^{T}\left[y(k)log(\hat{y}(k)) + \frac{t(1-y(k))}{T}log(1-\hat{y}(k)\right] \tag{4.1}$$

En dicha función se establece que $y^t(k)$ codifica la clase o valor real que se quiere alcanzar, mientras que $\hat{y}(k)$ indica lo predicho por el modelo, además de que N es el número de clases y T el tamaño de la secuencia de entrada.

Esta función de pérdida consiste en dos términos: el primero penaliza los falsos negativos con la misma fuerza en cualquier punto en el tiempo; mientras que la segunda se enfoca en los falsos positivos, y su penalización incrementa linealmente en el tiempo hasta alcanzar el mismo peso que los falsos negativos. Además, el peso relativo del primer término comparado con el segundo es mayor en el inicio de la secuencia. Todo junto mejora la predicción de un puntaje alto para la clase correcta lo antes posible, es decir, se previenen los falsos negativos y a su vez se tiene en consideración las ambigüedades al comienzo de la secuencia, que dan lugar a los falsos positivos. Sin embargo, a medida que se recibe información, se eliminan las ambigüedades y con ello los falsos positivos (Aliakbarian y cols., 2017).

Téngase en cuenta que la otra función de pérdida que se usa en este trabajo es entropía cruzada binaria, y que a pesar de ser muy popular en cuestiones de clasificación, ésta mide únicamente la distribución de probabilidad que existe entre el valor real y el valor predicho de la red neuronal sin considerar en ningún sentido los falsos positivos o negativos ni las ambigüedades que pudieran surgir en las predicciones cuando se tiene poca información.

4.5. Configuración de las redes neuronales

Hasta el momento se ha descrito la forma en la que se alimentan las redes neuronales en sus diferentes enfoques, sin embargo, no se ha indicado las configuraciones que éstas requirieron, así como la estructura que se manejó en ellas.

4.5.1. Configuración LSTM

Como se ha mencionado anteriormente, en este trabajo se quiere explotar la idea de la información secuencial presente en el texto. Para ello se decidió hacer uso de redes tipo LSTM de forma apilada de modo que se mejorara el nivel de abstracción de los *embeddings* de las palabras. Para lograrlo se utilizó la parte correspondiente al entrenamiento de los corpus de anorexia y depresión y se dividió en tres pliegues para hacer una validación cruzada. Así mismo, se hizo una combinatoria de los parámetros de unidades, dropout y tasa de aprendizaje indicados en la tabla 4.2, resultando con ello 27 posibles configuraciones a la red neuronal.

Configuración de LSTM	
LSTM	Número de capas: 2 Unidades: 50, 100, 200
Dropout	0.2, 0.3, 0.4
Dense	Salida: 1
Función de activación	Sigmoide
Optimizador	Adam
Taza de aprendizaje	0.0005, 0.001, 0.015
Función de pérdida	Propuesta / Entropía cruzada binaria

Tabla 4.2: Parámetros y valores necesarios en la configuración de la red LSTM.

Después de realizar la validación cruzada se optó por utilizar 50 unidades en cada capa, con un dropout de 0.2 y una tasa de aprendizaje de 0.0005 dado que fue la combinación que mejor resultados arrojó en el promedio de los tres pliegues.

Cabe señalar que la selección de estos parámetros se realizó para los casos de secuencia de palabras y secuencia de *embeddings* promedio, coincidiendo en ambas situaciones en ser los mejores parámetros.

4.5.2. Configuración CNN

La red convolucional se usó considerando que la parte clave en la que se pudiera presentar la información necesaria para clasificar de forma correcta podría estar en cualquier punto de la secuencia del texto.

Por tal motivo se probaron diferentes configuraciones para los casos de secuencias de palabras y secuencias de fragmentos, utilizando de igual modo validación cruzada en tres pliegues con los parámetros de filtros, kernel y *max pooling* (indicados en la tabla 4.3), resultando con ello 16 posibles configuraciones para la red neuronal.

Configuración CNN	
Conv1D	Filtros= 25, 50, 100, 200 Kernel= 2, 5 Número de capas: 1
MaxPooling	2, 5
Flatten	-
Dense	Salida=1
Función de activación	Sigmoide
Optimizador	Adam
Tasa de aprendizaje	0.0005
Función de pérdida	Propuesta / Entropía cruzada binaria

Tabla 4.3: Parámetros y valores necesarios en la configuración de la red CNN.

Después de hacer la validación cruzada para la selección de los parámetros, se determinó que para el caso de secuencia de palabras era mejor el uso de 25 filtros, con un kernel de 5 y maxpolling de 2; mientras que para el caso de secuencias de *embeddings* promedio era mejor el uso de 100 filtros, con un kernel de 5 y maxpolling de 2.

4.5.3. Configuración de red combinada CNN+LSTM

Para este caso se hizo una combinación de ambas redes, usando los mejores parámetros de ambas y combinándolos, de modo que la salida de la red CNN fuera procesada por la LSTM.

4.5.4. Configuración de las funciones de pérdida

Como se indica en las tablas de los parámetros de las redes neuronales, se hicieron pruebas usando dos funciones de pérdida.

Para el caso de la función de entropía cruzada binaria, se utilizaron los parámetros por default que ofrece la librería de Keras[1], mientras que para el caso de la función propuesta a usar, ésta posee parámetros que pueden ser modificados: el número de clases (N) y el tamaño de la secuencia de entrada (T). Debido a la naturaleza binaria de los problemas a clasificar se estableció N=2; mientras que en el caso de las secuencias se probaron diferentes tamaños $T = 5, 10, 20, 40, 80$, siendo que se obtuvieron resultados favorables con $T = 20$. Así mismo, téngase en consideración que los valores de T fueron probados durante la validación cruzada de los parámetros de las redes neuronales, además de que se requirió una leve modificación al código proporcionado originalmente por el autor, ya que inicialmente está diseñada para el uso de características de dos dimensiones, por lo que fue necesario hacer el cambio a una dimensión.

[1] https://keras.io

Capítulo 5

Resultados y evaluación

En este capítulo se presentan los resultados obtenidos para cada uno de los enfoques y sus versiones presentadas. De igual modo se hace una descripción de los corpus en los que se pusieron a prueba las redes neuronales así como la forma de evaluación que se llevó a cabo en cada uno de ellos.

5.1. Corpus usados

Para los experimentos presentados en este trabajo se decidió usar tres corpus relacionados a los problemas de depresión, anorexia, y depredadores sexuales.

5.1.1. Depresión

Este corpus fue el presentado en la competencia eRisk 2017[1] y se encuentra descrito a detalle en (Losada y cols., 2017), y consiste en *posts* extraídos de la plataforma Reddit. Los datos son una colección de *posts* realizados por los usuarios de manera secuencial divididos en dos categorías: sufren depresión, no presentan depresión.

[1] https://early.irlab.org/2017/index.html

La colección consiste en *posts* ordenados cronológicamente y después divididos en 10 pedazos, donde el primer 10 % de los *posts* corresponden al primer pedazo, el segundo 10 % corresponde al segundo y así sucesivamente.

La descripción de los datos de este corpus se presenta en la tabla 5.1 y un par de ejemplos del texto presente en el corpus se muestra en la figura 5.1.

	Train		Test	
	Deprimidos	No Deprimidos	Deprimidos	No Deprimidos
Num. de ejemplos	83	403	52	349
Num. de posts	30851	264172	18706	217665
Promedio de posts por ejemplo	371.7	655.5	359.7	623.7
Promedio de días del primer al último post	572.7	626.6	608.31	623.2
Promedio de palabras por post	27.6	21.3	26.9	22.5

Tabla 5.1: Descripción del corpus de depresión

Positivos	I have great friends, an amazing boyfriend, all that jazz. It doesn't matter, I still feel lonely. They all care about me and love me, yet, I still feel so isolated from everyone.
	Hi :) It is awful to feel 'not normal'. I don't ever remember feeling anything other than a freak for a long time.
	I tried pretty much every SSRI and tricyclical antidepressant in existence over the last years. Its hard to give you a full list because I know only the local names
Negativos	To keep calm I breathe in through my nose and slowly exhale through my mouth for about 15 seconds
	Thanks for this info. :) So talent agents look for people who are singing and dancing at public places such as restaurants?

Figura 5.1: Ejemplos positivo y negativo del corpus de depresión.

Como se puede ver en la tabla, se tienen pocos ejemplos, lo cual es una limitante si se considera el hecho de que las redes neuronales requieren de un gran número de ejemplos al momento de entrenamiento, por lo que enfoques de secuencias largas de texto podrían verse afectados al tener un número reducido de ejemplos donde aprender los patrones de secuencia. Sin embargo, si se considera el hecho de cada uno de los ejemplos puede dividirse en subejemplos, se crea una solución a dicho problema, lo cual está indicado en la tabla 5.2. Por tal motivo se considera que en corpus con características similares a éste, es mejor hacer uso de representaciones cortas, ya sea de *embeddings* de palabras o *embeddings* promedio.

	Entrenamiento		Prueba	
	Deprimidos	No deprimidos	Deprimidos	No deprimidos
Num. ejemplos	830	1209	520	3490
Num. total de posts	30851	264172	18706	217665
Num. de posts por pedazo	37	65	35	62
Promedio de palabras por pedazo	1021.2	1384.5	941.5	1395
Promedio de palabras por posts	27.6	21.3	26.9	22.5

Tabla 5.2: División del corpus de depresión cuando se crean subejemplos.

5.1.2. Anorexia

Este corpus es el usado por la competencia eRisk 2018[2] y descrtitos a detalle en (Losada y cols., 2018), que de igual modo es extraído de la plataforma Reddit. La

[2]https://early.irlab.org/2018/index.html

organización de los datos se realiza del mismo modo que el corpus de depresión. La descripción de este corpus se presenta en la tabla 5.3 y un par de ejemplos del texto presente en el corpus se muestra en la figura 5.2.

	Train		Test	
	Anorexia	No Anorexia	Anorexia	No Anorexia
Num. de ejemplos	20	132	41	279
Num. de posts	7452	77514	17422	151364
Promedio de posts por ejemplo	372.6	587.2	424.9	542.5
Promedio de días del primer al último post	803.3	641.5	798.9	670.6
Promedio de palabras por post	41.2	20.9	35.7	20.9

Tabla 5.3: Descripción del corpus de anorexia.

Positivos	As far as food, I have been watching my macros like a hawk and I've read just about everything I can find on the proper kinds of food. I have spoken to many professionals
	Breakfast: Skipped Lunch: Skipped Dinner: Skipped Total: 0 calories Yeah I totally get this too. As a vegan, I have dreams/nightmares all the time where I'm eating meat
	I don't eat regularly enough. I never have any energy as a result of it, but I have a hard time convincing myself to eat
Negativos	Which is exactly why I love stand up comedy. It's my own little safe space away from PC culture.
	Good try woodtick. But, this person is obviously worshiping the idol of Trump, and won't be dissuaded with facts.

Figura 5.2: Ejemplos positivo y negativo del corpus de anorexia.

Similar al caso de depresión, este corpus cuenta con pocos ejemplos para el entrenamiento, por lo que el desempeño de los métodos propuestos que hacen uso de secuencias largas se puede ver afectado al momento del entrenamiento. Sin embargo, al crear secuencias cortas se ataca dicho problema, lo cual está indicado en la tabla 5.4.

	Entrenamiento		Prueba	
	Anorexia	No anorexia	Anorexia	No anorexia
Num. ejemplos	200	1320	410	2790
Num. total de posts	30851	264172	18706	217665
Num. de posts por pedazo	37	58	42	54
Promedio de palabras por pedazo	1524.4	1212.2	1499.4	1215
Promedio de palabras por posts	41.2	20.9	35.7	20.9

Tabla 5.4: División del corpus de anorexia cuando se crean subejemplos.

5.1.3. Depredadores sexuales

Este corpus es el que se presentó en la conferencia del PAN 2012[3] para la tarea de identificación de depredadores sexuales y está descrito a detalle en (Inches y Crestani, 2012), cuyo objetivo es detectar depredadores sexuales en chats. A diferencia de los dos corpus anteriores, éste solamente se encuentra organizado de manera cronológica en la secuencia de preguntas y respuestas típicas de un chat. Por lo que fue necesario dividir el corpus de manera manual en 10 partes. Para ello se calculó el número de palabras totales y se obtiene el 10 % de ellas para cada uno de los pedazos. Nótese la diferencia con los dos corpus anteriores que están divididos a nivel *post* mientras que este a nivel palabra. La descripción del corpus

[3]https://pan.webis.de/clef12/pan12-web/author-identification.html

	Train		Test	
	Chat con depredador	Chat sin depredador	Chat con depredador	Chat sin depredador
Detección de depredador sexual	798	5790	1466	13863

Tabla 5.5: Descripción del corpus de depredadores sexuales.

Positivo	hey | hi | what ya doin? | nothin | u? | sittin here naked | cool | u here? | yep eating | oh k | lunch time | its early | 1216 here | its 11:19 | 12:20 now
Negativo	hi | hey | whats up? | nm andu? | same | kindda bored | me too | how was ur day/ | ? | it was ok | crappy | weather lol | yeah same here | :(

Figura 5.3: Ejemplos positivo y negativo del corpus de depredadores sexuales.

se encuentra en la tabla 5.5 y un par de ejemplos del texto presente en el corpus se muestra en la figura 5.3.

A diferencia de los otros corpus, este cuenta con suficientes ejemplos para el entrenamiento de las redes neuronales usando secuencias largas, aunque debe considerarse que los chats que se manejan son cortos en comparación al historial de *posts* de los otros, por lo que el desempeño usando secuencias largas no tiene problema.

5.1.4. Generación de *embeddings*

Los corpus están divididos en ejemplos de entrenamiento y prueba, por lo que para cada uno de los problemas se generó un modelo de *embeddings* de palabras usando Word2Vec y FastText con la modalidad de SkipGram usando solamente la parte de entrenamiento.

Los modelos de *embeddings* utilizaron los siguientes parámetros en ambos casos:

- min_count=2. Este parámetro ignora todas las palabras con una frecuencia menor al indicado.
- windows=5. Indica la máxima distancia entre una palabra y la pronosticada

dentro de una oración.

- size=100. Indica el tamaño de los vectores de las palabras.
- epocas=15. Indica el número de iteraciones sobre el corpus.

5.2. Forma de evaluación

Como se ha dicho anteriormente, los corpus de depresión y anorexia son los presentados en las competencias del eRisk 2017,eRisk 2018 respectivamente, y están divididos considerando el número de *posts* realizados por los sujetos en cada ejemplo, por lo que la forma de evaluarlos se llevó a cabo considerando la regla presentada en dicha competencia, la cual indica que si al momento de clasificar un ejemplo con información parcial se decide que es un ejemplo positivo, la decisión ya no puede ser cambiada aun después de que llegue más información que haga al clasificador cambiar de decisión. De igual modo en estos corpus se aplica la medida de evaluación ERDE con valores de o en 5 y 50, además de las medidas precisión, recuerdo y F1.

Para los corpus de depresión y anorexia la estrategia que se siguió fue la siguiente: inicialmente todos los ejemplos se consideran negativos, al momento de clasificar una parte, si la red considera que el ejemplo es positivo y tiene una confiabilidad mayor a 0.5, dicho ejemplo se considera positivo y se mantiene así hasta el final sin importar si llega más información. Por otro lado, para los ejemplos que son clasificados como negativos, éstos pueden cambiar su valor conforme llega más información, siendo que si se termina de recibir información y la red neuronal no cambia su decisión de dejarlo negativo, entonces se queda como tal.

Así mismo, inicialmente se planteó usar como *baseline* las secuencias largas de palabras con la función entropía cruzada binaria, sin embargo, debido al bajo desempeño que mostró en los casos de depresión y anorexia, se decidió usar las secuencias cortas en estos corpus.

Por otro lado, para el corpus de depredadores sexuales, debido a que se encuentra dividido considerando el número de palabras, no es posible aplicar la medida ERDE, de modo que para determinar el desempeño de las redes neuronales en este corpus se clasifica acumulando la información parcial que va recibiendo y se va calculando la métrica F1, por lo que en este caso un ejemplo clasificado como positivo si puede pasar su valor a negativo si recibe más información que ocasione el cambio de decisión de la red.

Así mismo, en este caso si se usaron secuencias largas de palabras con cross entropy como baseline.

5.3. Resultados en corpus de depresión

5.3.1. Depresión: nivel palabra

A continuación, se presentan los resultados obtenidos con las versiones del enfoque a nivel palabra usando Word2Vec y FastText.

Como se puede apreciar en las tablas 5.6 y 5.7, en el caso de depresión, el enfoque de palabras con secuencias largas no obtiene resultados que superen el baseline que se estableció, pero si comparables en algunas medidas. Por otro, lado en el caso de crear secuencias de palabras cortas en subjemplos, se obtienen resultados favorables en la mayoría de los casos para la medida F1 en Word2Vec y FastText. En el caso de las medidas $ERDE_5$ y $ERDE_{50}$ los resultados resultan nuevamente favorables en comparación del *baseline*. Por último, se puede observar que la red convolucional es la que muestra mejor desempeño tanto en Word2Vec como en FastText, obteniendo los mejores resultados.

Depresión- Word2Vec (Nivel palabra)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	16.97	11.87	0.21	0.92	0.34
	CNN	14.83	10.08	0.32	0.88	0.47
	CNN+LSTM	16.96	11.32	0.22	0.9	0.35
Función custom (Secuencia larga de palabras)	LSTM	13.51	12.54	0.39	0.21	0.28
	CNN	13.86	11.43	**0.45**	0.48	0.47
	CNN+LSTM	16.37	10.2	0.21	0.58	0.3
Función custom (Secuencia corta de palabras)	LSTM	17.29	11.95	0.22	**0.94**	0.35
	CNN	**13.47**	**9.16**	0.42	0.83	**0.55**
	CNN+LSTM	16.52	11.42	0.23	0.9	0.37

Tabla 5.6: Resultados nivel palabra con Word2Vec en depresión

Depresión- FastText (Nivel palabra)						
Enfoque	Tipo de red	ERDE 5	Erde 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	16.47	11.69	0.26	0.94	0.41
	CNN	16.86	11.53	0.22	0.9	0.36
	CNN+LSTM	17.17	12.4	0.23	0.94	0.37
Función custom (Secuencia larga de palabras)	LSTM	13.34	12.75	0.17	0.21	0.19
	CNN	13.34	12.84	0.16	0.1	0.12
	CNN+LSTM	**13.21**	11.28	0.26	0.38	0.31
Función custom (Secuencia corta de palabras)	LSTM	18.29	12.66	0.19	**0.98**	0.31
	CNN	14.96	**9.71**	**0.3**	0.83	**0.44**
	CNN+LSTM	15.83	11.64	0.27	0.88	0.42

Tabla 5.7: Resultados nivel palabra con FastText en depresión

En general se puede observar lo siguiente:

- Tanto para el caso de Word2Vec como para el de FastText, la red tipo CNN es la que obtiene mejores resultados tanto en métricas ERDE como en F1.
- El enfoque de usar secuencias cortas de palabras tiene un mejor desempeño en comparación al uso de secuencias largas.
- Los resultados obtenidos usando la función de pérdida custom son comparables, y en algunos casos mejores a los de entropia cruzada binaria.

5.3.2. Depresión: nivel fragmento

Los resultados a nivel fragmento se muestran en la tabla 5.8 para el caso de Word2Vec y 5.9 para FastText. Como puede apreciarse, este enfoque obtiene resultados favorables en la mayoría de los resultados de los dos enfoques en el caso de la medida F1. Para el caso de la medida $ERDE_5$, se obtienen resultados en su mayoría mejores a los del baseline cuando se comparan las redes neuronales similares. Sin embargo, esto no ocurre en $ERDE_{50}$, donde no se supera en la mayoría de los casos el baseline ni en Word2Vec ni en FastText.

Depresión-Word2Vec (Nivel fragmento)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencias corta de palabras)	LSTM	16.97	11.87	0.21	**0.92**	0.34
	CNN	14.83	**10.08**	0.32	0.88	0.47
	CNN+LSTM	13.96	11.32	0.22	0.9	0.35
Función custom (Promedio de posts)	LSTM	14.42	12.93	0.37	0.5	0.42
	CNN	14.32	12.33	0.36	0.46	0.41
	CNN+LSTM	17.13	14.65	0.25	0.85	0.39
Función custom (División en subpartes)	LSTM	14.81	11.48	0.38	0.67	0.48
	CNN	14.82	11.68	0.4	0.75	**0.52**
	CNN+LSTM	**13.93**	13.68	**0.44**	0.46	0.45

Tabla 5.8: Resultados nivel fragmento con Word2Vec en depresión

Depresión-FastText (Nivel fragmento)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	16.47	11.69	0.26	**0.94**	0.41
	CNN	16.86	**11.53**	0.22	0.9	0.36
	CNN+LSTM	17.17	12.4	0.23	**0.94**	0.37
Función custom (Promedio de posts)	LSTM	14.84	13.59	0.3	0.48	0.37
	CNN	14.66	12.5	0.38	0.63	0.47
	CNN+LSTM	14.35	12.61	0.37	0.48	0.42
Función custom (División en subpartes)	LSTM	14.87	13.12	0.35	0.62	0.45
	CNN	**13.67**	11.93	**0.56**	0.54	**0.55**
	CNN+LSTM	14.06	13.56	0.39	0.42	0.41

Tabla 5.9: Resultados nivel fragmento con FastText en depresión

En general se puede observar lo siguiente en las tablas:

- Como en el caso del enfoque a nivel palabra, el desempeño de la red tipo CNN otorga los mejores resultados.
- El uso de la función propuesta no presenta mejoría en la mayoría de los casos respecto a las métricas ERDE.
- El uso de la función propuesta presenta mejoría respecto al baseline en la métrica F1.

5.3.3. Depresión: nivel pedazo

Como se puede apreciar en la tabla 5.10 y 5.11, los resultados al manejar la información a nivel pedazo en sus dos versiones, superan en su mayoría a los establecidos en el *baseline*; siendo que el caso de promediar los pedazos presenta mejores resultados en $ERDE_5$ y F1, mientras que promediar los pedazos y ajustar la entrada en $ERDE_{50}$, siendo que la red convolucional es la que tiene los mejores resultados en ambos casos.

Depresión-Word2Vec (Nivel pedazo)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencias corta de palabras)	LSTM	16.97	11.87	0.21	**0.92**	0.34
	CNN	14.83	10.08	0.32	0.88	0.47
	CNN+LSTM	13.96	11.32	0.22	0.9	0.35
Función custom (Promedios a nivel pedazo)	LSTM	14.5	11.59	0.39	0.6	0.47
	CNN	**13.42**	11.21	**0.41**	0.65	**0.51**
	CNN+LSTM	14.44	11.02	**0.41**	0.65	**0.51**
Función custom (Promedios con entrada ajustada)	LSTM	13.73	9.31	0.33	0.73	0.46
	CNN	13.92	9.04	0.32	0.75	0.45
	CNN+LSTM	14.5	**8.97**	0.3	0.81	0.43

Tabla 5.10: Resultados nivel pedazo con Word2Vec en depresión

Depresión-FastText (Nivel pedazo)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	16.47	11.69	0.26	**0.94**	0.41
	CNN	16.86	11.53	0.22	0.9	0.36
	CNN+LSTM	17.17	12.4	0.23	0.94	0.37
Función custom (Promedios a nivel pedazo)	LSTM	14.03	11.42	0.4	0.48	0.44
	CNN	14.24	11.08	**0.45**	0.65	**0.53**
	CNN+LSTM	14.31	11.7	0.44	0.63	0.52
Función custom (Promedios con entrada ajustada)	LSTM	14.91	9.5	0.26	0.88	0.4
	CNN	**14.21**	**8.96**	0.31	0.85	0.45
	CNN+LSTM	14.52	9.02	0.28	0.87	0.42

Tabla 5.11: Resultados nivel pedazo con FastText en depresión

En general se puede observar lo siguiente:

- El uso del promedio simple otorga buenos resultados en lo que se refiere a la métrica F1 cuando se usa la función de pérdida propuesta en comparación al *baseline*.
- El ajustar la entrada de información a la red neuronal proporciona mejores resultados en lo que refiere a las métricas ERDE, por lo que se hace una mejoría en la clasificación temprana.

5.3.4. Comparación con el estado del arte.

Si bien los resultados anteriores se ven prometedores en varios casos tanto en las medidas ERDE como en F1, son comparativas hechas a nivel local, por lo que es necesario hacerlo respecto al estado del arte, siendo que para ello se decidió utilizar el modelo de CNN con el enfoque promedio a nivel pedazo con entrada ajustada y FastText, ya que en el caso de las medida $ERDE_5$ es superior a la del *baseline*, en el caso de $ERDE_{50}$ es la mejor de todas las versiones, y en el caso de F1, su resultado es competitivo.

La comparativa del modelo seleccionado se realizó usando un diagrama de caja y bigotes que incluye todos los resultados reportados en la competencia eRisk 2017 para depresión tanto para las métricas ERDE (figura 5.4) como en F1 (figura 5.6).

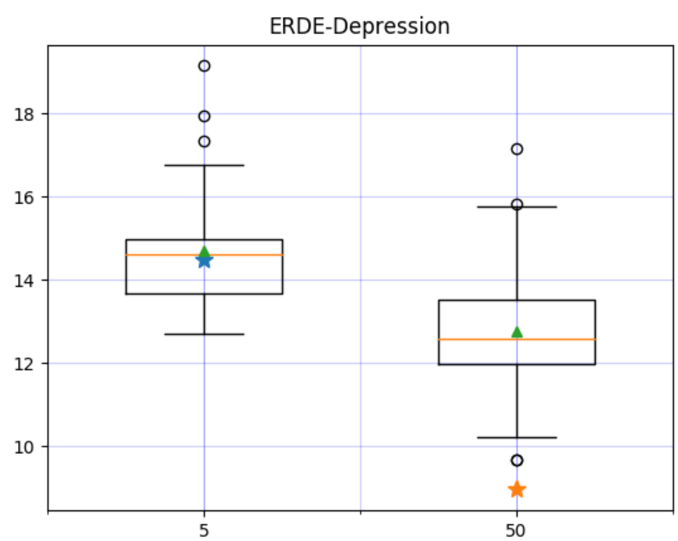

Figura 5.4: Desempeño de la red neuronal respecto a los reportados en la competencia eRisk2017. Los resultados del modelo CNN se indican con una estrella, mientras que el promedio de los resultados con un triángulo.

Como se aprecia en la gráfica, los resultados del modelo seleccionado en comparación al promedio general del estado del arte son bastante competitivos, ya que se encuentra abajo del promedio para los casos de $ERDE_5$ y $ERDE_{50}$ (recordar que entre más bajo mejor), mientras que F1 se encuentra encima del promedio.

De una forma general los resultados que obtiene el modelo seleccionado respecto a los reportados a los 30 reportados en eRisk 2017 son lo siguientes:

- 15a posición en $ERDE_5$
- 1a posición en $ERDE_{50}$
- 14a posición en F1

Por otro lado, en la tabla 5.12 se hace una comparativa contra los mejores resultados reportados en eRisk2017 para la tarea de detección temprana de depresión. En dicha tabla se aprecia que, si bien no supera a todos los casos, obtiene un desempeño superior en $ERDE_{50}$ comparado al reportado.

	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
eRisk 2017	12.70	9.68	0.69	0.92	0.64
Modelo CNN	14.21	8.96	0.31	0.85	0.45

Tabla 5.12: Mejores resultados reportados para depresión en eRisk 2017.

5.4. Resultados en corpus de anorexia

5.4.1. Anorexia: nivel palabra

Los resultados para el corpus de anorexia con Word2Vec y FastText se muestran en las tablas 5.13 y 5.14 respectivamente. Como se puede apreciar el uso de secuencias largas no muestra mejoría respecto al *baseline* seleccionado, sin embargo, en el caso de secuencia de palabras cortas obtiene resultados favorables en la mayoría de las métricas, siendo que la combinación de CNN+LSTM con Word2Vec obtiene el mejor $ERDE_{50}$ y F1, además de un $ERDE_5$ mejor a su similar con entropía cruzada binaria.

Anorexia-Word2Vec (Nivel palabra)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencias corta de palabras)	LSTM	**12.47**	9.69	0.72	0.44	0.55
	CNN	13.46	7.62	0.43	**0.9**	0.58
	CNN+LSTM	13.57	7.66	0.43	0.88	0.58
Función custom (Secuencia larga de palabras)	LSTM	15.12	13.29	0.18	0.32	0.23
	CNN	12.86	10.74	**0.87**	0.32	0.46
	CNN+LSTM	13.21	12.9	0.52	0.27	0.35
Función custom (Secuencia corta de palabras)	LSTM	13.2	**6.59**	0.51	**0.9**	0.65
	CNN	12.71	7.44	0.61	0.83	0.7
	CNN+LSTM	12.75	6.8	0.59	**0.9**	**0.71**

Tabla 5.13: Resultados a nivel palabra con Word2Vec en anorexia

Anorexia- FastText (Nivel palabra)						
Enfoque	Tipo de red	ERDE 5	Erde 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	16.47	11.69	0.26	0.94	0.41
	CNN	16.86	11.53	0.22	0.9	0.36
	CNN+LSTM	17.17	12.4	0.23	0.94	0.37
Función custom (Secuencia larga de palabras)	LSTM	18.73	17.47	0.14	0.61	0.23
	CNN	**12.83**	11.6	**0.88**	0.17	0.29
	CNN+LSTM	13.61	13.61	0.2	0.12	0.15
Función custom (Secuencia corta de palabras)	LSTM	13.23	8.5	0.5	0.9	**0.64**
	CNN	13.12	**7.83**	0.55	0.71	0.62
	CNN+LSTM	14.06	8.89	0.35	**0.95**	0.51

Tabla 5.14: Resultados nivel palabra con FastText en anorexia

En general se puede observar lo siguiente:

- El uso de secuencias largas no muestra mejoría respecto al *baseline* en cuestiones de la métrica ERDE ni de F1.
- El uso de secuencias cortas muestra resultados favorables para las métricas ERDE y F1, tanto en Word2Vec como en FastText.
- El uso de la función custom mejora el desempeño de la red neuronal en lo que se refiere a la métrica ERDE cuando se hace la comparativa respecto al *baseline*.

5.4.2. Anorexia: nivel fragmento

Los resultados a nivel fragmento se muestran en la tabla 5.15 y 5.16 para Word2Vec y FastText respectivamente. Como se puede apreciar los resultados que se obtienen muestran un comportamiento desfavorable para ambas versiones del enfoque, ya que son pocos los casos en los que se aprecia una mejora en las métricas ERDE en ambas versiones de los *embeddings*. Sin embargo, se destaca el modelo convolucional con división en partes más pequeñas, ya que obtiene el F1 más alto.

Anorexia-Word2Vec (Nivel Fragmento)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencias corta de palabras)	LSTM	**12.47**	9.69	0.72	0.44	0.55
	CNN	13.46	**7.62**	0.43	**0.9**	0.58
	CNN+LSTM	13.57	7.66	0.43	0.88	0.58
Función custom (Promedio de posts)	LSTM	14.81	13.86	0.25	0.41	0.31
	CNN	13.05	11.78	0.74	0.41	0.53
	CNN+LSTM	13.21	12.9	0.66	0.46	0.54
Función custom (División en partes más pequeñas)	LSTM	13.53	10.35	0.24	0.66	0.35
	CNN	13.22	10.13	0.73	0.73	**0.73**
	CNN+LSTM	13.01	12.43	**0.77**	0.41	0.54

Tabla 5.15: Resultados a nivel fragmento con Word2Vec en anorexia

Anorexia-FastText (Nivel fragmento)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	13.21	12.9	0.52	0.27	0.35
	CNN	14.16	**8.34**	0.37	**0.85**	**0.51**
	CNN+LSTM	14.38	10.39	0.38	0.66	0.48
Función custom (Promedio de posts)	LSTM	15.09	13.83	0.27	0.51	0.35
	CNN	**13.01**	12.06	**0.74**	0.34	0.47
	CNN+LSTM	14.45	14.14	0.32	0.46	0.38
Función custom (División en subpartes)	LSTM	13.61	11.68	0.2	0.54	0.29
	CNN	13.53	12.27	0.53	0.49	**0.51**
	CNN+LSTM	15.33	14.44	0.3	0.66	0.41

Tabla 5.16: Resultados nivel fragmento FastText en anorexia

En general se puede observar lo siguiente:

- Los resultados obtenidos a lo que se refiere en métricas ERDE no superan al *baseline* establecido.
- Los resultados respecto a la métrica F1, son comparables y algunos casos mejores cuando se usa promedio de *posts* o división en subpartes.
- La red CNN es la que obtiene mejores resultados en varias métricas.

5.4.3. Anorexia: nivel pedazo

Los resultados del enfoque nivel pedazo del corpus de anorexia para Word2Vec y FastText se muestran en las tablas 5.17 y 5.18 respectivamente. Como se puede apreciar los resultados son favorables para las métricas ERDE y F1 en comparación del *baseline* establecido, siendo que, como en el caso de depresión, la versión de promediar pedazos de forma simple obtiene mejores resultados en F1, mientras que el promediar y ajustar la entrada los obtiene en $ERDE_{50}$.

Anorexia-Word2Vec (Nivel pedazo)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencias corta de palabras)	LSTM	12.47	9.69	0.72	0.44	0.55
	CNN	13.46	7.62	0.43	0.9	0.58
	CNN+LSTM	13.57	7.66	0.43	**0.88**	0.58
Función custom (Promedios a nivel pedazo)	LSTM	**12.02**	9.03	0.61	0.56	0.58
	CNN	12.65	10.32	**0.74**	0.56	**0.64**
	CNN+LSTM	12.66	9.89	0.63	0.63	0.63
Función custom (Promedios con entrada ajustada)	LSTM	12.41	7.7	0.4	0.71	0.51
	CNN	12.53	7.37	0.51	0.71	0.59
	CNN+LSTM	12.03	**7.33**	0.5	0.66	0.57

Tabla 5.17: Resultados a nivel pedazo con Word2Vec en anorexia

Anorexia-FastText (Nivel pedazo)						
Enfoque	Tipo de red	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
BCE (Secuencia corta de palabras)	LSTM	13.21	12.9	0.52	0.27	0.35
	CNN	14.16	8.34	0.37	**0.85**	0.51
	CNN+LSTM	14.38	10.39	0.38	0.66	0.48
Función custom (Promedios a nivel pedazo)	LSTM	12.81	10.74	0.59	0.46	0.52
	CNN	12.81	10.31	**0.74**	0.49	0.59
	CNN+LSTM	13.02	9.93	0.68	0.73	**0.71**
Función custom (Promedios con entrada ajustada)	LSTM	13.17	8.28	0.39	0.68	0.5
	CNN	**12.22**	**7.6**	0.51	0.63	0.57
	CNN+LSTM	12.46	7.8	0.45	0.61	0.52

Tabla 5.18: Resultados nivel pedazo con FastText en anorexia

En general se puede observar lo siguiente:

- El uso del promedio simple obtiene mejores resultados en la métrica F1 respecto al *baseline* establecido.
- El uso del promedio simple con entrada ajustada obtiene resultados favorables para las métricas ERDE y comparables en la métrica F1 respecto al *baseline* establecido.

5.4.4. Comparación con el estado del arte.

Así como en el caso de depresión , los resultados para anorexia en métricas ERDE y F1 fueron prometedores en experimentos locales, por lo que para comparar su desempeño fue necesario hacerlo contra el estado del arte. Para este caso se utilizó la red neuronal CNN+LSTM en secuencias cortas de palabras con Word2Vec ya que posee un $ERDE_5$ y $ERDE_{50}$ bajo en comparasión a su *baseline* y el mejor F1.

De igual modo que en el caso de depresión se hizo uso de diagramas de caja y bigotes que incluyen todos los resultados reportados en la competencia eRisk 2018 para anorexia en las métricas ERDE (figura 5.5) y F1 (figura 5.6).

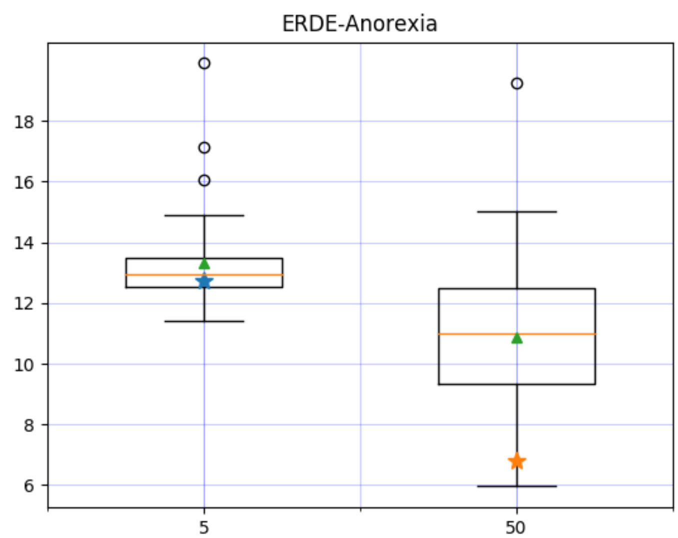

Figura 5.5: Resultados obtenidos en la métrica ERDE por el modelo secuencia corta de palabras con CNN+LSTM (estrella) en comparación con el estado del arte y su promedio (triángulo)

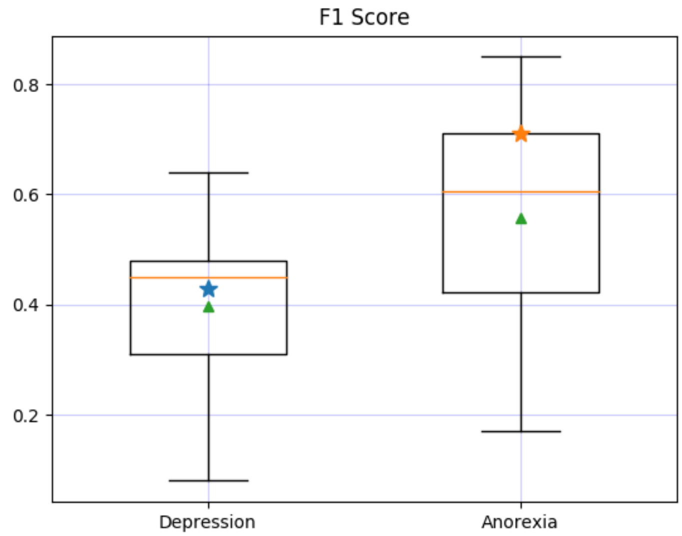

Figura 5.6: Resultados obtenidos para la métrica F1 respecto en los problemas de depresión y anorexia. Con estrella marcados los resultados obtenidos por las redes neuronales propuestas, con triangulo el promedio de los resultados del estado del arte.

Como se puede apreciar en las gráficas el resultado para las métricas ERDE es favorable, ya que se encuentran abajo del promedio (recordar que entre más bajo mejor), además que el desempeño en el caso de $ERDE_{50}$ se ve bastante prometedor. Mientras que en el caso de F1 se encuentra por encima del promedio.

De una forma general los resultados que obtiene el modelo seleccionado respecto a los reportados a los 35 reportados en eRisk 2018 son lo siguientes:

- 11a posición en $ERDE_5$
- 3a posición en $ERDE_{50}$
- 9a posición en F1

Por otro lado, en la tabla 5.19 se hace una comparativa contra los mejores resultados reportados en eRisk 2018 para la tarea de detección temprana de anorexia.

En dicha tabla se aprecia que, si bien no supera a todos los casos, obtiene un resultado decente en $ERDE_{50}$, además de tener un recuerdo mejor al reportado.

	ERDE 5	ERDE 50	Precisión	Recuerdo	F1
eRisk 2018	11.4	5.96	0.91	0.88	0.85
Modelo CNN+LSTM	12.75	6.8	0.59	0.9	0.71

Tabla 5.19: Mejores resultados reportados en anorexia en eRisk 2018.

5.5. Resultados en corpus de depredadores sexuales

5.5.1. Depredadores: nivel palabra

Los resultados en F1 para cada una de las partes a nivel palabra del corpus de depredadores se pueden ver en la tabla 5.20 para Word2Vec y 5.21 para FastText. Como se puede apreciar, el uso de secuencias más largas en este corpus muestra un buen desempeño a diferencia de los corpus anteriores debido a la longitud de los ejemplos.

De igual modo, se observa en las tablas que el uso de secuencias cortas en subejemplos muestra un buen desempeño en el primer pedazo de clasificación, sin embargo, este decae en los siguientes, debido a que esta versión de enfoque no acumula información, por tanto se entiende que en el caso de este corpus la información importante para clasificar se encuentra en el primer pedazo. Por otro lado, en el caso de usar secuencias más largas acumulando la información los resultados son favorables en el caso de la red CNN, ya que supera de forma favorable al *baseline*, siendo en este caso la versión con Word2Vec la que otorga mejores resultados.

Depredadores Sexuales-Word2Vec (Nivel palabra)											
Enfoque	Tipo de red	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
BCE (Secuencia larga de palabras)	LSTM	0.51	0.68	0.76	0.82	0.82	0.85	0.88	0.91	0.93	0.95
	CNN	0.62	0.83	0.89	0.92	**0.94**	0.94	0.95	**0.96**	0.95	0.96
	CNN+ LSTM	0.58	0.78	0.86	0.9	0.92	0.93	0.94	**0.96**	**0.96**	**0.97**
Función custom (Secuencia corta de palabras)	LSTM	**0.86**	0.83	0.8	0.79	0.79	0.78	0.78	0.78	0.79	0.81
	CNN	0.85	0.81	0.79	0.78	0.76	0.74	0.75	0.76	0.77	0.78
	CNN+ LSTM	0.8	0.76	0.72	0.72	0.71	0.7	0.7	0.7	0.71	0.78
Función custom (Secuencia larga de palabras)	LSTM	0.42	0.62	0.72	0.78	0.81	0.85	0.87	0.88	0.91	0.93
	CNN	0.81	**0.88**	**0.91**	**0.93**	**0.94**	0.94	0.95	**0.96**	**0.96**	0.96
	CNN+ LSTM	0.51	0.78	0.87	0.92	**0.94**	**0.95**	**0.96**	**0.96**	**0.96**	**0.97**

Tabla 5.20: Resultados a nivel palabra con Word2Vec en depredadores sexuales en métrica F1.

Depredadores Sexuales-FastText (Nivel palabra)											
Enfoque	Tipo de red	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
BCE (Secuencia larga de palabras)	LSTM	0.5	0.7	0.77	0.82	0.85	0.88	0.9	0.91	0.92	0.95
	CNN	0.73	0.85	0.89	**0.92**	0.93	0.94	0.94	0.95	0.95	0.96
	CNN+ LSTM	0.6	0.82	0.89	**0.92**	**0.94**	**0.95**	**0.96**	**0.96**	**0.97**	**0.97**
Función custom (Secuencia corta de palabras)	LSTM	**0.85**	0.8	0.77	0.76	0.75	0.74	0.75	0.74	0.75	0.74
	CNN	0.83	0.78	0.77	0.75	0.73	0.72	0.74	0.74	0.74	0.77
	CNN+ LSTM	0.83	0.78	0.75	0.74	0.73	0.72	0.72	0.73	0.73	0.75
Función custom (Secuencia larga de palabras)	LSTM	0.34	0.47	0.53	0.6	0.63	0.66	0.7	0.73	0.78	0.89
	CNN	0.82	**0.87**	**0.89**	0.91	0.92	0.93	0.94	0.95	0.95	0.96
	CNN+ LSTM	0.44	0.73	0.83	0.88	0.9	0.91	0.91	0.91	0.91	0.9

Tabla 5.21: Resultados a nivel palabra de FastText en depredadores sexuales en métrica F1.

De forma general se puede apreciar lo siguiente:

- El enfoque usando secuencias largas de palabras presenta un mejor desempeño en comparación al uso de secuencias cortas, esto es contrario a lo que ocurre en los casos de depresión y anorexia debido a que el número de palabras en dichos corpus es mayor cada ejemplo en comparación a los ejemplos que se tienen en depredadores sexuales.
- El uso de la función de pérdida custom en secuencias largas, muestra un mejor desempeño en los pedazos iniciales en comparación al *baseline* esta-

blecido, por lo que se muestra una mejoría en la clasificación temprana.

- El uso de secuencias cortas muestra resultados favorables en los primeros pedazos, sin embargo dado que el enfoque no va acumulando información, su desempeño decae.

5.5.2. Depredadores: nivel pedazo

Los resultados a nivel pedazo están indicados en las tablas 5.22 y 5.23 para Word2Vec y FastText respectivamente. En dichas tablas se observa que el uso de promedios simples a nivel pedazo no es útil en clasificaciones tempranas, ya que su desempeño es malo en los pedazos iniciales. Por otro lado, en el caso de usar promedios con entrada ajustada, el desempeño es comparable al *baseline* en la mayoría de los casos, siendo que la versión con FastText es la que muestra mejores resultados.

Depredadores Sexuales-Word2Vec (Nivel pedazo)											
Enfoque	Tipo de red	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
BCE (Secuencia larga de palabras)	LSTM	0.51	0.68	0.76	0.82	0.82	0.85	0.88	0.91	0.93	0.95
	CNN	0.62	**0.83**	**0.89**	**0.92**	**0.94**	**0.94**	**0.95**	0.96	0.95	0.96
	CNN+ LSTM	0.58	0.78	0.86	0.9	0.92	0.93	0.94	0.96	0.96	0.97
Función custom (Promedios a nivel pedazo)	LSTM	0	0.02	0.24	0.6	0.82	0.91	0.95	**0.97**	**0.97**	**0.98**
	CNN	0	0	0	0.06	0.22	0.57	0.8	0.89	0.94	0.97
	CNN+ LSTM	0	0	0	0.01	0.22	0.82	0.95	0.96	0.96	0.96
Función custom (Promedios con entrada ajustada)	LSTM	**0.66**	0.82	0.88	0.88	0.92	0.93	0.94	0.95	0.96	0.97
	CNN	0.63	0.76	0.85	0.9	0.92	0.93	0.94	0.95	0.96	0.97
	CNN+ LSTM	0.58	0.71	0.81	0.87	0.9	0.92	0.94	0.95	0.95	0.96

Tabla 5.22: Resultados a nivel pedazo de Word2Vec en depredadores sexuales en métrica F1.

Depredadores Sexuales-FastText (Nivel pedazo)											
Enfoque	Tipo de red	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
BCE (Secuencia larga de palabras)	LSTM	0.5	0.7	0.77	0.82	0.85	0.88	0.9	0.91	0.92	0.95
	CNN	**0.73**	**0.85**	**0.89**	**0.92**	**0.93**	0.94	0.94	0.95	0.95	0.96
	CNN+ LSTM	0.6	0.82	0.89	0.92	0.94	**0.95**	**0.96**	**0.96**	**0.97**	**0.97**
Función custom (Promedios a nivel pedazo)	LSTM	0	0.01	0.18	0.51	0.72	0.84	0.9	0.94	0.96	**0.97**
	CNN	0	0	0	0.1	0.02	0.46	0.79	**0.96**	**0.97**	**0.97**
	CNN+ LSTM	0	0	0	0	0.08	0.78	0.94	**0.96**	0.96	0.96
Función custom (Promedios con entrada ajustada)	LSTM	0.63	0.78	0.85	0.89	0.91	0.92	0.94	0.94	0.95	0.96
	CNN	0.66	0.78	0.85	0.89	0.91	0.92	0.94	0.95	0.96	0.96
	CNN+ LSTM	0.61	0.74	0.83	0.88	0.9	0.92	0.93	0.94	0.95	0.95

Tabla 5.23: Resultados a nivel pedazo con FastText en depredadores sexuales en métrica F1.

En general se puede apreciar en las tablas lo siguiente:

- El uso de promedios simples obtiene resultados poco favorables en los primeros pedazos.
- El uso de promedios simples con entrada ajustada en las redes neuronales obtiene resultados comparables al *baseline* establecido.
- En la mayoría de los casos la red tipo CNN es la que obtiene los mejores resultados.

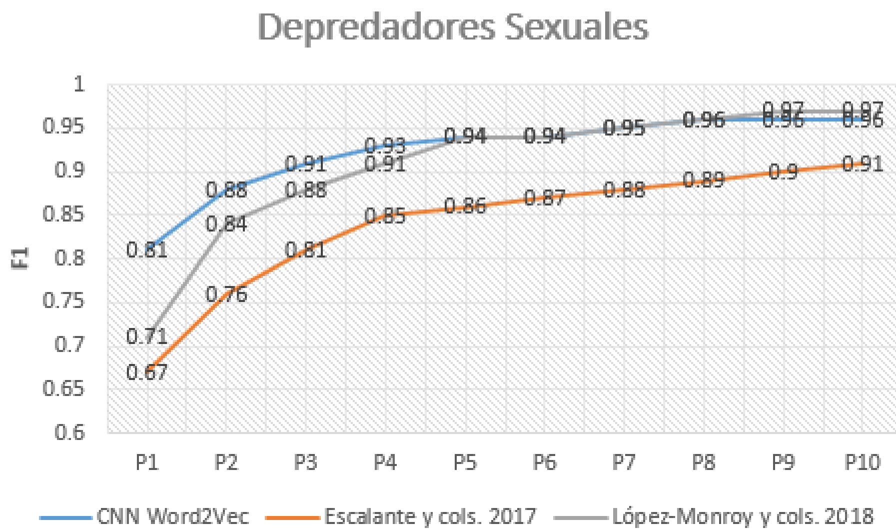

Figura 5.7: Métrica F1 por pedazo en comparativa a las diferentes métodologias del estado del arte.

5.5.3. Comparación con el estado del arte.

Los experimentos hechos para este corpus mostraron resultados prometedores de manera local, sin embargo, es necesario hacer la comparativa respecto al estado del arte. Para ello se seleccionó el modelo de secuencia larga de palabras con CNN y Word2Vec en comparación con los resultdos reportados por (Escalante y cols., 2017) y (López-Monroy y cols., 2018). La comparativa se muestra en la figura 5.7. Como se puede apreciar, los resultados de la red CNN propuesta en este trabajo obtienen mejores valores a los reportados en el estado del arte. Cabe destacar, que, si bien el modo de evaluación de este corpus fue diferente respecto a los anteriores, el desempeño de los enfoques propuestos fue competitivo.

Capítulo 6

Análisis y discusión

En este capítulo se hace un análisis del desempeño de los enfoques propuestos respecto a los resultados obtenidos en el capítulo anterior. De igual modo se analizan las ventajas y desventajas que presentaron, así como puntos a considerar.

6.1. Análisis de enfoque: nivel palabra

Este enfoque a pesar de ser simple, obtiene resultados bastante buenos en comparación al baseline. Sin embargo, se enfrenta a un problema que claramente lo afecta dependiendo de los datos de la tarea, y eso es la longitud de palabras que recibe, ya que se vió con los experimentos que entre más larga es la secuencia su desempeño en clasificación reduce en cualquier tipo de red así como en las métricas ERDE para los casos de depresión y anorexia que fueron los corpus que poseen una gran cantidad de palabras.

En general se aprecia que de las dos versiones de enfoque sucede que:

- Cuando son corpus con ejemplos de secuencias muy largas (como en los corpus de anorexia y depresión), el desempeño de las redes neuronales decae, además de que también puede verse afectado debido al poco número de ejemplos disponibles de los corpus. Caso contrario, su desempeño es bastante favorable en casos de corpus con ejemplos no tan extensos (corpus depredadores sexuales) y con un buen número de elementos para la fase de entrenamiento. En cuestiones de clasificación temprana, se puede decir que este depende del número de ejemplos para entrenar la red, así como de la longitud de la secuencia que se maneje. Ya que presenta buenos resultados cuando se trata con redes convolucionales.

- El dividir las secuencias largas en secuencias más cortas y de ese modo crear subejemplos, ayuda a mitigar el problema anterior, sin embargo, al truncar las secuencias se puede alterar la información relevante que ayude a la clasificación correcta del ejemplo, como es en el caso de depredadores sexuales, donde se aprecia un desempeño decreciente a medida que va cambiando de pedazos. De igual modo se puede decir que esta versión cumple con el concepto de clasificación temprana, ya que la clasificación de los primeros pedazos resulta favorable en los tres corpus, aunque en el caso de depredadores sexuales va decayendo.

- Se requiere determinar una mejor estrategia en la división de secuencias que permitan mantener la consistencia de la información o en su caso buscar una estrategia de clasificación que considerer este problema.

6.2. Análisis de enfoque: nivel fragmento

De los tres enfoques manejados, este parece ser el que menos se ajusta para cuestiones de clasificación temprana, ya que, si bien muestra resultados que pueden ser competitivos con el baseline establecidos respecto a la métrica F1, no se aplica lo mismo en medidas ERDE en las que no obtiene resultados que pueden destacarse. Sin embargo, se considera que no es un enfoque que sea malo, simplemente se necesita encontrar alguna configuración aún más adecuada para cuestiones de clasificación temprana.

En general se aprecia en este enfoque lo siguiente:

- El determinar el número de *posts* realizados por los sujetos del corpus, resulta una idea apropiada, sin embargo, tiene la desventaja de que puede verse afectado si el número de *posts* de los ejemplos varia demasiado, por lo que la idea de truncar el número de *posts* se ve afectada de forma similar a cuando se trunca el número de palabras en total que se usan como en el enfoque a nivel palabra.

- Dividir el corpus en fragmentos más pequeñas mejora el desempeño en cuestiones de F1 y ERDE en comparasión a la versión de determinar el número de *posts*. Sin embargo, no supera el desempeño del baseline establecido.

6.3. Análisis de enfoque: nivel pedazo

Este enfoque es simple y tiene la ventaja de no desperdiciar información al usar todas las palabras de cada pedazo, además de que al reducir las secuencias permite a las redes procesar la información de forma más rapida.

De forma general este enfoque se puede ver de la siguiente forma:

- La versión de promediar las palabras y alimentar la red neuronal con información parcial dejando parte de la entrada de la red sin información, muestra mejoría en la métrica F1 respecto al *baseline* establecido, sin embargo, no sucede esto en el caso de las medidas ERDE, esto para la forma de evaluación de los corpus de depresión y anorexia. Por otro lado, en el caso del corpus de depredadores sexuales, la red no es capaz de clasificar de forma correcta en los primeros pedazos respecto a la medida F1, por lo que si bien parece ser un buen enfoque cuando se utiliza información completa, no es el caso para la clasificación temprana.

- La versión propuesta de promediar las palabras de cada pedazo e ir ajustando la entrada de la red en la etapa de prueba obtiene buenos resultados en las métricas ERDE respecto al baseline establecido, aunque en el caso de la métrica F1 no parece haber gran mejoría, sin embargo, se mantiene competitivo en todas las redes neuronales.

6.4. Análisis de las redes neuronales

Si bien los enfoques tuvieron resultados favorables o comparables de forma general, el desempeño de estos varió en cada una de las redes neuronales, de lo cual se puede deducir lo siguiente:

- La red LSTM con la función propuesta muestra un comportamiento favorable para las métricas ERDE o F1 en comparación a la red LSTM con función de entropía cruzada del *baseline*, sin embargo, su desempeño se ve superado por las redes CNN o CNN+LSTM.

- La red CNN muestra un comportamiento bastante destacable, ya que en la mayoria de los casos logra obtener valores ERDE 5 o 50 superiores al baseline o a las otras configuraciones. Además de lograr clasificar de manera temprana en el caso de depredadores sexuales de forma muy favorable en comparación a las otras redes y el estado del arte.

- La red CNN+LSTM combina las mejores configuraciones de las otras dos y obtiene resultados que en su mayoría son comparables y en algunos casos superiores. Sin embargo, no se podría decir que es la mejor debido a que en el caso de depredadores sexuales, su desempeño fue similar al de la red LSTM, siendo superada con facilidad por la red CNN.

Ahora bien, también es necesario determinar que el comportamiento de las redes no haya sido al azar y que realmente exista un desempeño significativo respecto al *baseline*. Para hacer esto se decidió aprovechar la división en 10 partes de los corpus, ya que, visto en cierto modo, cada uno de estos vendría representando un corpus diferente que al momento de ser evaluado otorga una nueva métrica, siendo en este caso F1. Es necesario indicar que en el capítulo anterior en los experimentos de depresión y anorexia no se reportaron los valores de F1 por pedazo, sino uno general, debido a que la forma de evaluación se basó a las reglas establecidas en eRisk 2018, por lo que se necesitaba hacer la comparación con el estado del arte, sin embargo, la métrica F1 si fue calculada para cada pedazo y serán usados para este caso y se muestran en la tabla 6.1.

Depresión											
Método	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	significancia $p < 0{,}05$
Baseline-BCE CNN-FastText (Secuencia corta de palabras)	0.37	0.33	0.37	0.41	0.3	0.42	0.41	0.42	0.39	0.51	0.000075
Custom CNN-FastText (Promedio con entrada ajustada)	0.46	0.46	0.43	0.49	0.48	0.47	0.49	0.5	0.51	0.53	Cumple
Anorexia											
Baseline-BCE CNN+LSTM-Word2Vec (Secuencia corta de palabras)	0.51	0.48	0.44	0.59	0.55	0.48	0.48	0.52	0.53	0.6	0.015039
Custom CNN+LSTM-Word2Vec (Secuencia corta de palabras)	0.58	0.58	0.48	0.56	0.66	0.55	0.55	0.71	0.47	0.62	Cumple
Depredadores sexuales											
Baseline-BCE CNN-Word2Vec (Secuencia larga de palabras)	0.62	0.83	0.89	0.92	0.94	0.94	0.95	0.96	0.95	0.96	0.083925
Custom CNN-Word2Vec (Secuencia larga de palabras)	0.81	0.88	0.91	0.93	0.94	0.94	0.95	0.96	0.96	0.96	No Cumple

Tabla 6.1: Prueba t de student con significancia $p < 0{,}05$

Considerando lo anterior, se tiene un problema en el que se puede comparar el desempeño de la función entropía cruzada con la función propuesta sobre cada pedazo de los corpus, es decir, se tienen elementos pareados, por lo que se decidió usar la prueba t de student para verificar que existe un cambio significativo respecto al uso de ambas funciones de pérdida en las redes neuronales. Teniendo esto en consideración se plantearon las siguientes hipótesis:

- Hipotesis nula: ambos clasificadores tienen un desempeño similar.
- Hipótesis alterna: ambos clasificadores tienen un desempeño diferente, y por consecuente la red con la función custom tiene un mejor desemepeño.

Como se puede apreciar en la tabla 6.1, en el caso de depresión y anorexia, se cumple la condición de que el valor p sea menor que 0.05, por lo que se rechazaría la hipótesis nula aceptando la alterna, dejando en claro que el uso de la función custom mejora los resultados.

Por otra parte, en el caso de depredadores sexuales, no se cumple la condición $p < 0{,}05$ por lo que se rechaza la hipotesis alterna, sin embargo, esto no quiere decir que no exista la mejoría respecto al uso de la función entropía cruzada o la función propuesta, sino más bien se debe a que la mejoria se encuentra en los pedazos iniciales (pedazos 1 a 4), mientras que en los siguientes los resultados en comparación a cross entropy son practicamente iguales.

Por parte de los *embeddings*, si bien hubo casos destacables para Word2Vec y FastText, de una manera general se puede observar un mejor desempeño por parte del primero. Sin embargo, esto no descarta la eficacia de FastText, ya que la cantidad de ejemplos de entrenamiento para generar los *embeddings* pudo afectar su desempeño.

Capítulo 7

Conclusión y trabajo futuro

7.1. Conclusiones

La clasificación temprana es un área que debe de ser investigada con mayor profundidad debido a las diferentes aplicaciones que ésta puede tener en la prevención de situaciones de riesgo así como de comportamientos de enfermedades mentales que los usuarios muestran en redes sociales. En este trabajo se abordó esta área usando redes neuronales con diferentes enfoques, de los cuales se concluye lo siguiente:

- El enfoque de alimentación a nivel palabra a pesar de ser simple, obtiene resultados favorables cuando se manejan secuencias cortas de palabras, sin embargo su desempeño se ve afectado cuando se trata de secuencias largas de palabras con pocos ejemplos.
- El enfoque de alimentación a nivel fragmento no resulta útil en la clasificación temprana, sin embargo, presenta buenos resultados globales de clasificación.

- La propuesta de alimentación a nivel pedazo con entrada ajustada a la red, resultó ser bastante favorable para clasificación temprana sin importar la información que se tenga disponible.
- La configuración extensa de las redes neuronales permite encontrar diferentes alternativas para diferentes usos, lo cual es muy conveniente, siendo que para este caso se buscó encontrar la adecuada para la clasificación temprana.
- Optimizar la red neuronal para que clasifique de forma temprana por medio de la función de pérdida sin importar un dominio en específico resulta favorable, ya que se logra obtener un modelo general aplicable a diferentes ámbitos.

7.2. Trabajo futuro

Con los resultados obtenidos se planea mejorar aquellos enfoques y configuraciones que mejor desempeño obtuvieron en cuestión de clasificación temprana, esto se podría lograr probando otros tipos de *embeddings*, además del uso de ambas topologías, Skip Gram y CBoW. Por parte de las redes se planea probar nuevos filtros y configuraciones que pudieran aumentar el rendimiento y desempeño de las redes, como el usar redes combinadas en la entrada de datos o hacer incluso combinaciones con los mejores enfoques que se obtuvieron. Además, no se descarta el agregar nuevas características generales del texto que pudieran enriquecer los resultados.

Por último, al ver el buen desempeño que se obtuvo al modificar la red neuronal en su función de pérdida, se planea investigar nuevas funciones o idealmente desarrollarlas y que sean específicas para texto en la clasificación temprana o incluso otras tareas.

Referencias

Aliakbarian, M. S., Saleh, F. S., Salzmann, M., Fernando, B., Petersson, L., y Andersson, L. (2017, Oct). Encouraging lstms to anticipate actions very early. , 280-289. doi: 10.1109/ICCV.2017.39

Bojanowski, P., Grave, E., Joulin, A., y Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. Descargado de https://www.aclweb.org/anthology/Q17-1010 doi: 10.1162/tacl_a_00051

Chen, J., Chen, C., y Liang, Y. (2016/11). Optimized tf-idf algorithm with the adaptive weight of position of word. En *2016 2nd international conference on artificial intelligence and industrial engineering (aiie 2016).* Atlantis Press. Descargado de https://doi.org/10.2991/aiie-16.2016.28 doi: https://doi.org/10.2991/aiie-16.2016.28

Coates, J., y Bollegala, D. (2018, junio). Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. En *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 2 (short papers)* (pp. 194–198). New Orleans, Louisiana: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/N18-2031 doi: 10.18653/v1/N18-2031

Coppersmith, G., Dredze, M., Harman, C., Hollingshead, K., y Mitchell, M. (2015, junio 5). CLPsych 2015 shared task: Depression and PTSD on twitter. En *Proceedings of the 2nd workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality* (pp. 31–39). Denver, Colorado: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/W15-1204 doi: 10.3115/v1/W15-1204

Dadvar, M., y Eckert, K. (2018). Cyberbullying detection in social networks using deep learning based models; A reproducibility study. *CoRR*, *abs/1812.08046*.

Demšar, J. (2006, diciembre). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, *7*, 1–30. Descargado de http://dl.acm.org/citation.cfm?id=1248547.1248548

Escalante, H. J., Montes y Gomez, M., Villasenor, L., y Errecalde, M. L. (2016, junio). Early text classification: a naïve solution. , 91–99. Descargado de https://www.aclweb.org/anthology/W16-0416 doi: 10.18653/v1/W16-0416

Escalante, H. J., Villatoro-Tello, E., Garza, S. E., López-Monroy, A. P., y Gómez, M. M., y Villaseñor-Pineda, L. (2017). Early detection of deception and aggressiveness using profile-based representations. *Expert Systems with Applications*, *89*, 99 - 111. Descargado de http://www.sciencedirect.com/science/article/pii/S0957417417305171 doi: https://doi.org/10.1016/j.eswa.2017.07.040

Funez, D. G., Ucelay, M. J. G., Villegas, M. P., Burdisso, S., Cagnina, L. C., Montes-y-Gómez, M., y Errecalde, M. (2018). Unsl's participation at erisk 2018 lab. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_137.pdf

Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, *10*(1), 1–309.

Hochreiter, S., y Schmidhuber, J. (1997, noviembre). Long short-term memory. *Neural Comput.*, *9*(8), 1735–1780. Descargado de http://dx.doi.org/10.1162/neco.1997.9.8.1735 doi: 10.1162/neco.1997.9.8.1735

Hossin, M., y Sulaiman, M. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), 1.

Husseini Orabi, A., Buddhitha, P., Husseini Orabi, M., y Inkpen, D. (2018, junio). Deep learning for depression detection of twitter users. En *Proceedings of the fifth workshop on computational linguistics and clinical psychology: From keyboard to clinic* (pp. 88–97). New Orleans, LA: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/W18-0609 doi: 10.18653/v1/W18-0609

Inches, G., y Crestani, F. (2012, septiembre). Overview of the International Sexual Predator Identification Competition at PAN-2012. En P. Forner, J. Karlgren, y C. Womser-Hacker (Eds.), *CLEF 2012 Evaluation Labs and Workshop – Working Notes Papers, 17-20 September, Rome, Italy.* CEUR-WS.org. Descargado de http://www.clef-initiative.eu/publication/working-notes

Johnson, R., y Zhang, T. (2015, mayo–junio). Effective use of word order for text categorization with convolutional neural networks. En *Proceedings of the 2015 conference of the north American chapter of the association for computational linguistics: Human language technologies* (pp. 103–112). Denver, Colorado: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/N15-1011 doi: 10.3115/v1/N15-1011

Joulin, A., Grave, E., Bojanowski, P., y Mikolov, T. (2017, abril). Bag of tricks for efficient text classification. En *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 2, short papers* (pp. 427–431). Valencia, Spain: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/E17-2068

Jurafsky, D., y Martin, J. H. (2019). *Speech and language processing (third edition draft)*. Descargado de https://web.stanford.edu/~jurafsky/slp3/

Kenter, T., Borisov, A., y de Rijke, M. (2016, agosto). Siamese CBOW: Optimizing word embeddings for sentence representations. En *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 941–951). Berlin, Germany: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/P16-1089 doi: 10.18653/v1/P16-1089

Kim, Y. (2014, octubre). Convolutional neural networks for sentence classification. En *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1746–1751). Doha, Qatar: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/D14-1181 doi: 10.3115/v1/D14-1181

Kumar, J., Goomer, R., y Singh, A. K. (2018). Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. *Procedia Computer Science*, *125*, 676 - 682. Descargado de http://www.sciencedirect.com/science/article/pii/S1877050917328557 (The 6th International Conference on Smart Computing and Communications) doi: https://doi.org/10.1016/j.procs.2017.12.087

Lacueva, F., Vea-Murguía, J., del Hoyo-Alonso, R., y Salas, R. M. (2017, 09). Fasttext as an alternative to using deep learning in small corpus. En *Proceedings of tass 2017: Workshop on sentiment analysis at sepln co-located with 33nd sepln conference (sepln 2017)* (p. 65-69).

Li, Y., y Yang, T. (2018). Word embedding for understanding natural language: A survey. En S. Srinivasan (Ed.), *Guide to big data applications* (pp. 83–104). Cham: Springer International Publishing. doi: 10.1007/978-3-319-53817-4_4

Liu, D., Suen, C. Y., y Ormandjieva, O. (2017). A novel way of identifying cyber predators.

Liu, N., Zhou, Z., Xin, K., y Ren, F. (2018). TUA1 at erisk 2018. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_121.pdf

López-Monroy, A. P., González, F. A., Montes, M., Escalante, H. J., y Solorio, T. (2018, junio). Early text classification using multi-resolution concept representations. En *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers)* (pp. 1216–1225). New Orleans, Louisiana: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/N18-1110 doi: 10.18653/v1/N18-1110

Losada, D. E., Crestani, F., y Parapar, J. (2017). erisk 2017: Clef lab on early risk prediction on the internet: Experimental foundations. En G. J. Jones y cols. (Eds.), *Experimental ir meets multilinguality, multimodality, and interaction* (pp. 346–360). Cham: Springer International Publishing.

Losada, D. E., Crestani, F., y Parapar, J. (2018). Overview of erisk: Early risk prediction on the internet. En P. Bellot y cols. (Eds.), *Experimental ir meets multilinguality, multimodality, and interaction* (pp. 343–361). Cham: Springer International Publishing.

Marwa, T., Salima, O., y Souham, M. (2018, Oct). Deep learning for online harassment detection in tweets. En *2018 3rd international conference on pattern analysis and intelligent systems (pais)* (p. 1-5). doi: 10.1109/PAIS.2018.8598530

Maupomé, D., y Meurs, M. (2018). Using topic extraction on social media content for the early detection of depression. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_173.pdf

Meyer, D. (2016). How exactly does word2vec work?

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., y Dean, J. (2013). Distributed representations of words and phrases and their compositionality. En *Proceedings of the 26th international conference on neural information processing systems - volume 2* (pp. 3111–3119). USA: Curran Associates Inc.

Montejo-Ráez, A., y Díaz-Galiano, M. C. (2016). Participación de SINAI en TASS 2016. En *Proceedings of TASS 2016: Workshop on sentiment analysis at SEPLN co-located with 32nd SEPLN conference (SEPLN 2016), salamanca, spain, september 13th, 2016* (pp. 41–45). Descargado de http://ceur-ws.org/Vol-1702/tass2016_proceedings_v212.pdf

Ortega-Mendoza, R. M., López-Monroy, A. P., Franco-Arcega, A., y Montes-y-Gómez, M. (2018). PEIMEX at erisk2018: Emphasizing personal information for depression and anorexia detection. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-*

14, 2018. Descargado de http://ceur-ws.org/Vol-2125/paper_122.pdf

Powers, D. M. (2015). What the f-measure doesn't measure: Features, flaws, fallacies and fixes. *arXiv preprint arXiv:1503.06410*.

Quinteiro, J., Martel-Jordán, E., Hernández Morera, P., Antonio Ligero Fleitas, J., y López Rodríguez, A. (2011, 12). Clasificación de textos en lenguaje natural usando la wikipedia. *Revista Ibérica de Sistemas y Tecnologías de la Información*, *8*, 39-52. doi: 10.4304/risti.8.39-52

Ramiandrisoa, F., Mothe, J., Benamara, F., y Moriceau, V. (2018). IRIT at e-risk 2018. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_102.pdf

Sadeque, F., Xu, D., y Bethard, S. (2017). Uarizona at the CLEF erisk 2017 pilot task: Linear and recurrent models for early depression detection. En *Working notes of CLEF 2017 - conference and labs of the evaluation forum, dublin, ireland, september 11-14, 2017.* Descargado de http://ceur-ws.org/Vol-1866/paper_58.pdf

Salehinejad, H., Sankar, S., Barfett, J., Colak, E., y Valaee, S. (2017). Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*.

Schwartz, R., Tsur, O., Rappoport, A., y Koppel, M. (2013, octubre). Authorship attribution of micro-messages. En *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1880–1891). Seattle, Washington, USA: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/D13-1193

Shrestha, P., Sierra, S., González, F., Montes, M., Rosso, P., y Solorio, T. (2017, abril). Convolutional neural networks for authorship attribution of short texts. En *Proceedings of the 15th conference of the European chapter of* *the association for computational linguistics: Volume 2, short papers* (pp. 669–674). Valencia, Spain: Association for Computational Linguistics. Descargado de https://www.aclweb.org/anthology/E17-2106

Sánchez Turcios, R. A. (2015, 03). t-Student: Usos y abusos. *Revista mexicana de cardiología*, *26*, 59 - 61.

Trotzek, M., Koitka, S., y Friedrich, C. M. (2017). Early detection of depression based on linguistic metadata augmented classifiers revisited - best of the erisk lab submission. En *Experimental IR meets multilinguality, multimodality, and interaction - 9th international conference of the CLEF association, CLEF 2018, avignon, france, september 10-14, 2018, proceedings* (pp. 191–202). Descargado de https://doi.org/10.1007/978-3-319-98932-7_18 doi: 10.1007/978-3-319-98932-7_18

Trotzek, M., Koitka, S., y Friedrich, C. M. (2018). Word embeddings and linguistic metadata at the CLEF 2018 tasks for early detection of depression and anorexia. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_68.pdf

Trotzek, M., Koitka, S., y Friedrich, C. M. (2019). Utilizing neural networks and linguistic metadata for early detection of depression indications in text sequences. *IEEE Transactions on Knowledge and Data Engineering*, 1-1. doi: 10.1109/TKDE.2018.2885515

Wang, Y., Huang, H., y Chen, H. (2018). A neural network approach to early risk detection of depression and anorexia on social media text. En *Working notes of CLEF 2018 - conference and labs of the evaluation forum, avignon, france, september 10-14, 2018.* Descargado de http://ceur-ws.org/Vol-2125/paper_126.pdf