UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

MÉTODO DE APRENDIZAJE PROFUNDO PARA LA CLASIFICACIÓN DE TWEETS CON RIESGO DE COVID-19

POR:

ALBERTO VALDÉS CHÁVEZ

TESIS PRESENTADA COMO REQUISITO PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERÍA EN COMPUTACIÓN

CHIHUAHUA, CHIH., MÉXICO **FEBRERO 2022**

Método de Aprendizaje Profundo para la Clasificación de Tweets con Riesgo de COVID-19. Tesis presentada por Alberto Valdés Chávez como requisito parcial para obtener el grado de Maestro en Ingeniería en Computación, ha sido aprobada y aceptada por:

M.I. Javier González Cantú
Director de la Facultad de Ingeniería

Dr. Alejandro Villalobos Aragón
Secretario de Investigación y Posgrado

M.S.I. Karina Rocío Requena Yáñez
Coordinador(a) Académico

Dr. Jesús Roberto López Santillán
Director(a) de Tesis

23 de febrero de 2022

Comité:
Dr. Jesús Roberto López Santllán
Dr. Manuel Montes y Gómez
Dra. Graciela María de Jesús Ramírez Alonso
Dr. Luis Carlos González Gurrola

Alberto Valdés Chávez
Circuito Universitario 1,
Campus Universitario 2
Chihuahuahua, Chih. México
Cp. 31125
Febrero 2022

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

24 de febrero de 2022.

ING. ALBERTO VALDÉS CHÁVEZ
Presente.

En atención a su solicitud relativa al trabajo de tesis para obtener el grado de Maestro en Ingeniería en Computación, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. Jesús Roberto López Santillán** para que lo desarrolle como tesis, con el título **"MÉTODO DE APRENDIZAJE PROFUNDO PARA LA CLASIFICACIÓN DE TWEETS CON RIESGO DE COVID-19"**.

Índice de Contenido

FACULTAD DE INGENIERÍA
Circuito No.1, Campus Universitario 2
Chihuahua, Chih., México. C.P. 31125
Tel. (614) 442-95-00
www.fing.uach.mx

3. Marco teórico
   3.1. Aprendizaje máquina (*Machine Learning*)
   3.2. Construcción y preparación de un conjunto de datos
   3.3. Aprendizaje profundo (*Deep Learning*)
   3.4. Métricas de evaluación
   3.5. Procesamiento de lenguaje natural

4. Conjuntos de datos
   4.1. #SMM4H Tarea 5: Identificación de casos potenciales de covid-19 en *Tweets*
   4.2. #SMM4H Tares 6: Clasificación de *Tweets* con síntomas de Covid-19

5. Método basado en BERT / CT-BERT
   5.1. Implementación de modelos
   5.2. Parámetros de entrenamiento
   5.3. Resultados
   5.4. Discusión

6. Método basado en arquitectura *WIDE & DEEP*
   6.1. Diseño y construcción del modelo
   6.2. Modelos de comparación *Baseline*
   6.3. Resultados modelos *Baseline*
   6.4. Resultados modelos *Wide & Deep*
   6.5. Discusión de resultados modelos *Wide & Deep*
   6.6. Atención en rama profunda
   6.7. Análisis de características anchas

7. Conclusiones y trabajo futuro
   7.1. Conclusiones
   7.2. Trabajo futuro

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

Referencias

Apéndice

Curriculum Vitae

Solicitamos a Usted tomar nota de que el título del trabajo se imprima en lugar visible de los ejemplares de las tesis.

A T E N T A M E N T E

"Naturam subiecit aliis"

EL DIRECTOR **SECRETARIO DE INVESTIGACIÓN Y POSGRADO**

FACULTAD DE INGENIERÍA U.A.CH. DIRECCIÓN

M.I. JAVIER GONZÁLEZ CANTÚ **DR. ALEJANDRO VILLALOBOS ARAGÓN**

FACULTAD DE INGENIERÍA
Circuito No.1, Campus Universitario 2
Chihuahua, Chih., México. C.P. 31125
Tel. (614) 442-95-00
www.fing.uach.mx

Dedicatoria

Este trabajo va dedicado a mi familia, por su ayuda al regresar a casa, lo que me permitió estudiar y concluir la maestría. A mi esposa, por su gran cariño y apoyo. A mis amigos por a través de los cuales conocí y me interesé por esta rama de la ciencia.

Agradecimientos

Agradezco a la Universidad Autónoma de Chihuahua y sus profesores que me acompañaron en este trayecto. A mis directores de tesis Roberto y Manuel por su dirección y apoyo. Al INAOE por permitirme utilizar su laboratorio de súper cómputo para llevar a cabo la experimentación. Por último, al Conacyt por la beca recibida durante estos dos años.

Resumen

Este trabajo se condujo con el propósito de analizar publicaciones de *Twitter* para la extracción de textos informativos sobre la enfermedad de Covid-19. Durante la pandemia de Covid-19, hubo mucha incertidumbre y falta de información sobre la evolución de la enfermedad. Una de las fuentes de información más grande son las redes sociales, y de ellas es posible obtener datos acerca del desarrollo de la pandemia mediante algoritmos de *Machine/Deep Learning*. El objetivo de esta investigación fue el de evaluar dos modelos de *Deep Learning* para clasificar *tweets* sobre Covid-19, y diseñar un método novedoso con arquitectura *Wide & Deep* para la misma tarea. Los modelos y métodos abordaron las tareas 5 y 6 de clasificación dadas por la competencia *Social Media Mining For Health* (#SMM4H) 2021. La tarea 5 buscaba identificar *tweets* de casos potenciales de un gran conjunto de *tweets*. La tarea 6 buscó clasificar *tweets* que contenían síntomas en las categorías de noticias, reportes personales y reportes de terceros.

Durante esta investigación se hizo una comparación de dos modelos basados en *Transformers* pre-entrenados en diferentes conjuntos de datos. Uno de temática general y gran tamaño, y otro de temática de Covid-19 de mucho menor tamaño. También se propuso un modelo experimental con arquitectura *Wide & Deep*. Este modelo combina dos ramas de un modelo alimentadas de diferentes características para mejorar la clasificación. La rama profunda utiliza un modelo tipo BERT. La rama ancha utiliza características tipo *term frequency-inverse document frequency* (TF-IDF), una distribución de tópicos obtenidos por el algoritmo *non-negative matrix factorization* (NMF) y una métrica experimental que mide que tan personal es una palabra en una oración (EXPEI). Se entrenaron varios modelos con las combinaciones de estas características.

Los resultados muestran que el modelo pre-entrenado en temática médica superó por mucho al modelo de temática general. Mostrando que es más importante tener datos de calidad y del área de interés que muchos datos de un área general. Este modelo entrenado obtuvo el 1er lugar en la competencia para la tarea 6. También se encontró que los modelos *Wide & Deep* obtuvieron mejores resultados que las ramas implementadas de forma individual. Llegando a superar al modelo, basado en técnicas de ensamble de modelos tipo BERT, que ganó el 1er lugar de la competencia en la tarea 5.

Índice de Contenido

Índice de Tablas

Índice de Figuras

1. INTRODUCCIÓN

1.1 LA PANDEMIA DE COVID-19

En el transcurso de este siglo han surgido enfermedades con alta capacidad de transmisión. Las dos más recientes, catalogadas como pandemias por el *Center for Disease Control and Prevention* (CDC), son la influenza en el año 2009, y la enfermedad Covid-19 provocada por el virus SARS-COV-2. Esta última fue declarada pandemia el 30 de enero de 2020. Se sospecha que su origen ocurrió en la ciudad de Wuhan, China, al registrarse un brote inusual de neumonía que puso en alerta a las autoridades médicas a nivel mundial en diciembre de 2019 (Taylor, 2020).

Según la actualización No. 6 del reporte epidemiológico de la *World Health Organization* (WHO), en septiembre de 2020 habían muerto alrededor de 954 mil personas a nivel mundial. Las consecuencias de la pandemia han sido devastadoras no solo en el ámbito de la salud, sino también en lo económico y educativo. (Fetzer et al., 2020). Una forma de disminuir estas consecuencias radica en la oportunidad de mejora de los sistemas de monitoreo de enfermedades. De esta forma se puede incrementar la cantidad de información de la que disponen.

En México y en otros países los sistemas de monitoreo de enfermedades funcionan de forma jerárquica (Office of Public Health Scientific Services, 2018). La primera línea son los lugares donde se realizan consultas médicas, como hospitales, centros de salud, empresas o consultorios privados. Cada uno de estos lugares tiene la obligación de llevar un registro de casos e informar al organismo de vigilancia de la localidad. A su vez, estos organismos locales reportan los hallazgos a otro a nivel estatal y de este al nivel nacional (Secretaría de Salud, 2020). Esto quiere decir que, de todos los casos potenciales de una enfermedad, solo se detectan aquellos que llegan a consulta. Para ayudar en la detección de casos potenciales de la enfermedad, algunos se han enfocado en el análisis de redes sociales. A través del análisis de publicaciones de usuarios, podría ser posible identificar casos potenciales de

enfermedades y síntomas. Todo esto debido a la abundante información disponible en estos sitios.

1.2 LAS REDES SOCIALES PARA EL ANÁLISIS DE ENFERMEDADES

Una de las redes sociales con más popularidad es Twitter. Esta red permite a sus usuarios realizar publicaciones de imágenes, videos y, principalmente, texto. Cada *tweet* (publicación) permite un máximo de 280 caracteres. Se estima que diariamente se generan más de 500 mil *tweets* y 200 billones al año en todo el mundo (Internet Live Stats, 2021). Esta cantidad descomunal de información ha sido ampliamente utilizada en trabajos de investigación y en la industria para su análisis. Mediante algoritmos de *machine learning* y *deep learning*, los *tweets* pueden ser clasificados en categorías relacionadas a la salud. Una metodología como esta podría ser utilizada para identificar casos potenciales de Covid-19 a partir de publicaciones de los usuarios. A continuación, se mencionan algunas investigaciones que se han enfocado en el desarrollo de estos algoritmos clasificadores para identificar *tweets* en temas médicos.

Desde el 2013, se han desarrollado trabajos que buscan analizar publicaciones de redes sociales para extraer información sobre enfermedades como la Influenza, Zika o el Ébola. Un trabajo de investigación se encargó de recopilar la mayor cantidad de trabajos realizados en este ámbito. Los resultados demuestran que el análisis de redes sociales para el monitoreo de enfermedades es factible y puede complementar a los sistemas tradicionales de vigilancia (Al-Garadi et al., 2016).

Una convocatoria abierta organizada por el *National Institute of Informatics* de Tokyo, Japón (NTCIR-13), se enfocó a la vigilancia de enfermedades respiratorias en redes sociales. Se buscaba que los participantes diseñaran un modelo de *machine learning* que clasificara una publicación en una de dos clases, positiva o negativa. En la clase positiva deberían encontrarse todas aquellas publicaciones donde el autor expresa sin ambigüedad

que sufre de un síntoma o enfermedad. La clase negativa contendría todas las demás publicaciones. Los organizadores facilitaron un *dataset* de 1920 pseudo *tweets* en inglés ya etiquetados para el entrenamiento de los modelos. El modelo con puntaje más alto consiguió un *accuracy* del 88%. Este modelo utilizó un ensamble de redes convolucionales con módulos de atención jerárquica. Sin embargo, los modelos utilizados no representan el estado del arte actual. También, los textos proporcionados fueron artificiales, y carecían del desorden característico de los *tweets* reales. Como los errores de gramática, ortografía y coloquialismos. (Wakamiya et al., 2019).

1.3 CLASIFICACIÓN DE *TWEETS* SOBRE COVID-19

Relacionado a la pandemia del Covid-19, se han expuesto varios proyectos en el último año. Las tareas que estas investigaciones abordaron fue la de clasificación de *tweets* de Covid-19. Una de ellas busca identificar *tweets* donde se mencionan síntomas de Covid-19 que padece el autor y separarlos del resto. Esta investigación utilizó un modelo de estado del arte actual llamado *Bi-directional Encoding Representation from Transformers* (BERT). Obtuvo un *f1-score* de 0.71 para la clase de síntomas (Al-Garadi et al., 2020).

Otro trabajo busco algo similar al clasificar publicaciones en casos potenciales de la enfermedad y otros. Los casos potenciales podían incluir el padecimiento de síntomas, personas cercanas contagiadas, o experiencias donde hubo riesgo de contagio. Nuevamente se utilizó un modelo BERT. Logró un *f1-score* de 0.76 para la clase de caso potencial (Klein et al., 2021).

Por último, otro trabajo se enfocó en separar *tweets* de Covid-19 que contengan información relevante de los que no. La información relevante puede referirse a noticias o menciones que provean información sobre casos sospechosos, confirmados, recuperados y fallecimientos. En este trabajo se emplearon una gran cantidad de modelos de *machine learning* y *deep learning* con el fin de compararlos. Estos fueron modelos de regresión

logística, redes neuronales, redes neuronales convolucionales y basados en *Transformers*. El de mejor desempeño fue nuevamente BERT pre-entrenado en datos de Covid-19, con un *f1-score* de 0.894 (Magge et al., 2020).

1.4 ARQUITECTURA WIDE & DEEP

Analizando las investigaciones mencionadas para la clasificación de *tweets*, los modelos que han tenido mejor desempeño en la identificación de *tweets* médicos han sido los basados en *Transformers*. La mayoría de estos modelos poseen la misma arquitectura, y solo varían en los datos que se utilizaron para entrenarlos. De igual forma, los resultados no varían mucho al tratar de identificar *tweets* de casos potenciales, por lo que aún tienen un amplio margen de mejora.

Algunos intentos de mejorar los resultados de este tipo de modelos han tratado de combinar las capacidades de los enfoques tradicionales de *Machine Learning* con los modelos basados en atención. En el reconocimiento de entidades médicas, un intento ha sido fusionar las salidas de los modelos tipo BERT con otros tipos de características léxicas, donde las características resultantes se utilizan para entrenar un clasificador (Roitero et a., 2020). Una de las desventajas de este enfoque es que no se aprovechan al máximo las capacidades de aprendizaje de los modelos de atención, ya que sólo se utilizan como extractores de características independientes y no se entrenan en la tarea en cuestión.

Con esto en mente, esta investigación propone experimentar con la arquitectura *Wide & Deep*, inspirada en el trabajo de Cheng et al. (2016). La idea detrás de esta arquitectura es combinar dos ramas en un solo modelo para un entrenamiento conjunto. Estas son las ramas anchas y profundas. La rama profunda aporta una gran capacidad de generalización. La rama ancha alimentada con un conjunto de características léxicas ofrece una alta capacidad de discriminación durante la clasificación. La unión de ambas produce un modelo con

predicciones más acertadas. Esta idea puede ser aplicada para crear un modelo novedoso para la clasificación de *tweets* sobre Covid-19 utilizando modelos basados en *Transformers*.

1.5 PREGUNTAS DE INVESTIGACIÓN

Con lo expuesto hasta ahora, la presente investigación busca ayudar en la investigación y desarrollo de un modelo novedoso clasificador de *tweets* sobre el Covid-19. Las preguntas de investigación que guían este trabajo son las siguientes:

¿Un modelo pre-entrenado sobre datos de temática médica puede superar a otro pre-entrenado en muchos más datos pero de temática general?

¿Podría un modelo de clasificación híbrido, al estilo *wide and deep*, obtener mejores resultados que un modelo únicamente profundo?

Si el modelo *wide & deep* obtiene mejores resultados, ¿Cómo es que la unión de las dos ramas se complementa entre sí?

1.6 OBJETIVOS GENERAL Y ESPECÍFICOS

Para dar respuesta a las preguntas de investigación y ponerlas a prueba, se propone como objetivo general de la investigación el diseño de un método novedoso basado en técnicas de aprendizaje profundo para la identificación de *tweets* sobre Covid-19 y su categorización en distintos grupos.

Dentro de los objetivos específicos se proponen los siguientes:

1. Implementar y comparar dos modelos basados en *Transformers*, uno pre-entrenado en un corpus grande de temática general y otro en un corpus pequeño de temática médica.
2. Crear modelos con arquitectura *Wide & Deep* que identifiquen si el autor de un *tweet* está en una situación de riesgo de contagio y otro que clasifique en informe de contagio personal, informe de contagio de un tercero y noticias.

3. Analizar los resultados de los modelos, para comprender mejor el funcionamiento de las ramas profundas y anchas.

1.7 JUSTIFICACIÓN

La justificación de este trabajo se explica en dos puntos. Para el primero se toma en cuenta el tiempo que normalmente tarda en detectarse el caso de una enfermedad. En todas las enfermedades existe un lapso de tiempo entre el momento de contagio y la aparición de los primeros síntomas. Otro lapso de tiempo ocurre hasta el momento en que el afectado llega a consulta. Esto significa que pueden pasar días o semanas desde el inicio hasta el descubrimiento del caso en consulta. El análisis de redes sociales es posible de llevarse a cabo en cuestión de minutos, casi a tiempo real (Twitter, 2021), disminuyendo el tiempo de detección de forma considerable. Contar con una herramienta que ofrezca información sobre el desarrollo de la enfermedad ayudaría a una toma de decisiones mejor informadas y oportunas. El primer paso en el desarrollo de una herramienta como esta es la identificación de las publicaciones relevantes. Por esta razón, la creación de un modelo clasificador con buen desempeño es una etapa necesaria para el análisis de las publicaciones.

Se pretende que el segundo beneficio de esta investigación sea en el área académica. Como se pudo observar en las investigaciones descritas anteriormente, aún hay camino por recorrer para seguir mejorando los resultados de los modelos. Para mejorar los resultados, se pretende implementar un modelo con arquitectura *wide & deep*, que no ha sido utilizado en una aplicación como esta. Por esto, se espera que el modelo *wide & deep*, producto de esta investigación, logre mejorar los resultados de los modelos existentes.

2. TRABAJO RELACIONADO

En la última década, los medios electrónicos han sustituido a los medios físicos en la mayoría de los rubros, incluyendo la salud y las comunicaciones (Blank and Reisdorf, 2012). Algunos de los medios digitales más importantes y utilizados hoy en día para la comunicación, son las redes sociales. Una de las redes sociales con más popularidad se trata de Twitter. Como se mencionó anteriormente, Twitter permite a sus usuarios realizar publicaciones de imágenes, videos y, principalmente, texto. Cada *tweet* (publicación) permite un máximo de 240 caracteres. Mediante conocimientos y técnicas de *Natural Language Processing* (NLP), es posible clasificar estos *tweets* de forma automática. Esto es una tarea compleja, principalmente por la falta de estructura y gramática que es muy común en texto de este tipo. También, la complejidad puede radicar en el tipo de categorías que se busca identificar. Existen numerosas investigaciones que pretenden desarrollar metodologías y modelos especiales para categorizar *tweets* en distintas áreas. Entre ellas se encuentran los desastres naturales, la identificación de lenguaje ofensivo/racista, efectos secundarios de medicinas, identificación de síntomas, por mencionar algunos. A continuación, se mencionan algunos ejemplos.

Una investigación se enfocó en realizar un análisis de trabajo relacionado. Fue realizada por Al-Garadi et al. (2016), y se trata de una búsqueda de literatura entre los años 2004 y 2015 sobre redes sociales para el rastreo de pandemias. El producto del trabajo es una comparación sistemática de las investigaciones encontradas. Al tener un margen de 10 años para la búsqueda de estos artículos, se muestran los diferentes avances y modelos que han ido surgiendo, así como el análisis de distintos tipos de enfermedades. Algunos de ellos se enfocaron a las enfermedades de la Influenza, Zika o el Ébola. Las técnicas utilizadas para la extracción de características van desde acercamientos sencillos como bolsas de palabras hasta *embeddings* de palabras por medio del método Word2Vec con las técnicas *Continuous Bag*

of Words y *Skipgram* (Mikolov et al., 2013). Los resultados demuestran que el análisis de redes sociales para el monitoreo de enfermedades es factible y puede complementar a los sistemas tradicionales de vigilancia.

2.1 RECOPILACIÓN DE *TWEETS* E INFORMACIÓN SOBRE EL COVID-19

Otras investigaciones se han enfocado en el área de recopilación de *tweets* médicos, que puedan ser de utilidad para la comunidad científica. Se describen a continuación.

La primera fue desarrollada para recolectar *tweets* que hablen sobre Covid-19. Pueden contener sinónimos de la enfermedad o el virus. Su recolección comenzó desde enero de 2020, unos pocos meses antes de que se declarara como pandemia, y continúa hasta el presente. Esta base datos es tan extensa que se han recuperado alrededor de 1,187,394,119 *tweets* en inglés, y el número de identificación de cada uno de ellos está abierto al público. (Chen et al., 2020).

Otro trabajo de investigación buscó de forma similar, recolectar *tweets* en inglés que hablen sobre las vacunas contra Covid-19. Los *tweets* se recolectaron utilizando la técnica *snowball sampling technique*, donde comienzan una búsqueda inicial por medio de palabras clave y analizan las palabras que más se repiten en la primera búsqueda. Las palabras relacionadas se incluyen en una segunda búsqueda y así sucesivamente hasta que no haya palabras nuevas que se repitan con una determinada frecuencia. De igual forma, los números de identificación de los *tweets* encontrados se encuentran abiertos al público (DeVerna et al., 2021).

El trabajo por Wicke et al. (2020) se enfocó en recolectar *tweets* de Covid-19 por medio de una búsqueda de *hashtags* clave: #covid19, #coronavirus, #ncov2019, #2019ncov, #nCoV, #nCoV2019, #2019nCoV y #covid19. Recolectaron alrededor de 25,000 *tweets* por día que contuvieran al menos uno de los *hashtags*, obteniendo cerca de 203 millones de

tweets en total. Después realizaron un análisis de tópicos para obtener las temáticas principales del texto.

Por otro lado, el *dataset* CORD-19 es un conjunto de artículos científicos sobre la investigación del Covid-19. Está diseñado para facilitar el uso de técnicas de minería de texto y extracción de información. Contiene alrededor de 500,000 artículos científicos y se actualiza semanalmente. A partir de este conjunto de datos se han realizado numerosas investigaciones para agrupar los artículos en categorías clave y crear buscadores semánticos (Wang et al., 2020).

2.2 ANÁLISIS DE *TWEETS* MÉDICOS MEDIANTE MODELOS NO SUPERVISADOS

Los conjuntos de datos abiertos pueden ser utilizados para su análisis. Algunos tipos de análisis se llevan a cabo por medio de modelos no supervisados, es decir, donde los datos no están etiquetados. Uno de estos trabajos se llevó a cabo con el objetivo de detectar conversaciones en *Twitter* donde se mencionan síntomas de Covid-19 y el acceso a una prueba de la enfermedad. Para este trabajo se comenzó con un preprocesamiento del texto para remover *hashtags*, *stopwords*, y texto de noticias. Se empleó un modelo *Biterm topic model* para realizar una clusterización de *tweets* similares entre sí. De los grupos formados, se seleccionaron alrededor de 36 mil *tweets* para su revisión manual, y como resultado se encontraron 3 mil *tweets* relacionados al objetivo. (Mackey et al., 2020).

Otro análisis no supervisado tuvo como objetivo hacer un modelado de tópicos y el análisis de emociones de *tweets* con respecto a la vacunación contra el Covid-19. Para cada tarea se llevó a cabo un preprocesamiento distinto, que se ajustara mejor a cada uno de los modelos utilizados. Para el análisis de tópicos se utilizó el algoritmo *Latent Dirichlet Allocation* (LDA) y para el análisis de emociones se empleó el paquete *syuzhet* en lenguaje de programación R. En los resultados se encontraron 16 tópicos, que después se agruparon en 5 categorías, “opiniones y emociones de vacunación”, “conocimiento sobre las vacunas”,

“vacunas como problema global”, “administración de vacunas” y “progreso en el desarrollo de vacunas”. En el análisis de emociones se encontró que la emoción más popular fue la confianza, seguida por anticipación. (Lyu et al., 2021).

2.3 CLASIFICACIÓN DE *TWEETS* MÉDICOS MEDIANTE MODELOS SUPERVISADOS

Cuando se tienen conjuntos de datos etiquetados, es posible entrenar un modelo supervisado que aprenda a identificar a que categoría pertenece el *tweet*. Dos investigaciones distintas se enfocaron en clasificar de forma binaria un conjunto de *tweets* sobre Covid-19.

La primera los clasifica en “casos potenciales de la enfermedad” y “otros”. Comienzan por un preprocesamiento de los *tweets* aunque no mencionan los pasos. Después entrenan un modelo BERT sobre los datos y consiguen un *F1-score* de 0.70 para la clase “caso potencial”. Por último, aplican el modelo en un conjunto de 85 millones de *tweets* provenientes de Estados Unidos, donde encontraron alrededor de 13 mil “casos potenciales” en todo el país. Esta información podría combinarse con la ubicación geográfica de los *tweets* para obtener una distribución de casos por región. (Klein et al., 2021).

La segunda investigación realizó algo similar, donde las clases a identificar fueron “reporta síntomas” y “otros”. De igual forma realizan un preprocesamiento y entrenan un modelo BERT para clasificación. Los resultados muestran un F1-score de 0.96 para la clase “reportan síntomas” y un 0.71 para la clase “otros”. Nuevamente aplican el modelo a otro conjunto de *tweets*, donde utilizaron los *tweets* clasificados para crear un léxico de síntomas. (Al-Garadi et al., 2020).

La tarea 2 de la competencia *Workshop on Noisy User-generated Text* (WNUT) edición 2020 propuso desarrollar modelos clasificadores de *tweets* sobre Covid-19 en informativos y no informativos. Los organizadores se encargaron de realizar un etiquetado manual de datos y compartirlos con los participantes. Uno de los equipos con mejor desempeño realizó una comparación entre diferentes modelos de *machine learning* y

Transformers. Los modelos tradicionales de *machine learning* fueron *Logistic Regression*, *Feedforward Neural Networks*, y *Convolutional Neural Network* (CNN). Los datos fueron preprocesados para normalizar nombres de usuarios, URLS y emojis. Después, se realizó una extracción de características por medio de la técnica *Bag of Words* para alimentar los modelos de *machine learning*. El modelo de este tipo que mejor desempeño tuvo fue la CNN con un *F1-score* de 0.84 en el set de validación. De los modelos basados en *Transformers*, se emplearon 7 distintos. BERT, GPT, XLNET, RoBERTa, DistilBERT, BERTweet, BERT-Epi, donde los modelos tipo BERT varían en cuanto al número de *encoders*, cabezas de atención, tamaño de los *embeddings* y datos de preentrenamiento. Todos ellos puntuaron más alto que el modelo CNN, y el mejor fue el modelo BERT-Epi con un F1-score de 0.92 en el set de validación. Este modelo fue preentrenado en *tweets* sobre Covid-19. (Magge et al., 2020).

Una segunda competencia organizada por la Universidad de Pensilvania fue titulada *Social Media Mining for Health* (#SMM4H). Estuvo enfocada en el análisis de texto de redes sociales para la farmacovigilancia, el rastreo de enfermedades y de pacientes. En particular, las tareas 5 y 6 se trataron de clasificación de *tweets* sobre Covid-19. La tarea 5 buscaba identificar *tweets* de casos potenciales y otros. La tarea 6 clasificaba *tweets* que contenían síntomas en tres clases, “síntomas propios”, “síntomas de terceros” y “menciones de noticieros/literatura”. En el reporte final muestran los resultados de todos los participantes en cada tarea. Los equipos ganadores en todas las tareas utilizaron modelos basados en *Transformers*. Al igual que las investigaciones anteriores, la mayoría de los participantes llevó a cabo un *fine-tuning* de estos modelos. (Magge et al., 2021).

De acuerdo a las investigaciones mencionadas, los modelos basados en *Transformers* son el estado del arte en temas de texto. Son capaces de obtener resultados muy competitivos únicamente por medio de un *fine-tuning*. Sin embargo, hay otros trabajos que tratan de mejorar los resultados al realizar modificaciones a estos modelos, para que se ajusten mejor

a la tarea en cuestión. A continuación, se describen algunas investigaciones que realizaron este tipo de modificaciones buscando mejorar el desempeño de los modelos.

Uno de estos trabajos reciente buscó desarrollar y poner a prueba un modelo utilizando como referente el *dataset* TREC-IS. Estos datos son un conjunto de publicaciones de redes sociales que hablan sobre situaciones de emergencia por desastres naturales. Los datos contienen una doble anotación de 25 diferentes categorías. Entre las cuales se encuentran “contiene ubicación”, “llamada de auxilio”, entre otras. La segunda categoría es “nivel de prioridad”. El modelo desarrollado en este trabajo propone utilizar un modelo BERT. La arquitectura del modelo busca predecir los dos tipos de anotaciones simultáneamente. Esto lo logra utilizando dos cabezas de predicción entrenadas al mismo tiempo. Cada una recibe como entrada la salida del token especial de inicio [CLS] del modelo BERT. La primera cabeza clasifica el texto en las categorías dadas, y la segunda es una cabeza de regresión que predice el nivel de prioridad. Los resultados demuestran que su modelo obtiene un mejor desempeño que los modelos que abordan los problemas de etiquetado dual de forma individual. De igual forma, el tiempo de inferencia es más rápido que los modelos individuales. (Wang et al., 2021).

En la edición 2020 de la competencia #SMM4H se propusieron tres tareas de clasificación. La primera consistió en identificar *tweets* que contengan una medicación. La segunda identifica menciones de afectos adversos o no. La tercera clasifica en tres categorías la mención de defectos de nacimiento. Uno de los equipos participó en estas tres tareas, obteniendo el mejor desempeño en esta última. Su modelo consistió en un ensamble de cuatro modelos BioBERT. Cada modelo tenía una cabeza de clasificación distinta, y utilizaba una representación del texto proveniente de diferentes niveles del modelo BioBERT. El primer modelo utilizó la representación dada por la salida del *pooler*, del token especial [CLS] del último estado oculto del modelo. Su cabeza de clasificación fue una red neuronal recurrente tipo Bi-GRU y una capa densa de clasificación. El segundo utilizó como representación del

texto la concatenación de los últimos dos estados ocultos del token de inicio [CLS] más la salida del *pooler*, alimentado a una capa densa. El tercero concatenó los últimos tres estados ocultos del token [CLS] seguido de una red densa. Por último, el cuarto utilizó la concatenación de los tres últimos estados ocultos del token [CLS] y la salida del *pooler*, seguido de una red densa. Para la tarea de defectos de nacimiento obtuvieron un F1-score de 0.69, superando al score promedio de 0.65. (Bai et al., 2020).

En otra investigación se recolectaron *tweets* con temática médica, y se etiquetaron manualmente en "médicos" y "no médicos". Para la extracción de características se propuso dos formas distintas. Una de ellas fue por medio del modelo BERT, y la segunda a través de MetaMap. MetaMap es una herramienta que reconoce términos médicos en un texto. Se aplicó esta herramienta a los *tweets* para obtener un vector tipo *one hot encoded* que coloca un 1 cuando un concepto médico está presente. Las características se probaron de forma individual y en conjunto (la concatenación de ambas) por medio de varios algoritmos de *machine learning*. Los algoritmos utilizados fueron *Logistic Regression*, R*andom Forest*, *Naive Bayes* y *Support Vector Machine* (SVM). Los resultados muestran que el modelo BERT utilizado por sí mismo obtiene los mejores resultados, mientras que cuando se le agregan las características de MetaMap, su desempeño desciende. Los modelos SVM y *Random Forest* mejoran su desempeño cuando se utiliza la concatenación de *embeddings* de BERT y MetaMap. El *dataset* utilizado fue liberado para el acceso público. (Roitero et al., 2020).

A continuación, se muestra una Tabla que resume el objetivo, los modelos y los resultados de los trabajos anteriores donde se clasifican *tweets* por medio de aprendizaje supervisado.

Tabla 2.1: Síntesis de trabajos relacionados.

Objetivos	**Modelos**	**Resultados**
Identificar casos potenciales de Covid-19 en *tweets* de Estados Unidos.	BERT	F1-score de 0.70 para clase caso potencial.
Identificar *tweets* que contienen síntomas de Covid-19.	BERT	F1-score de 0.96 para clase positiva. F1-score de 0.71 para clase negativa.
Clasificar *tweets* de Covid-19 en informativos y no informativos.	*Logistic Regression*, *Feed-forward* NN, CNN con características BOW. BERT, GPT, XLNET, RoBERTa, DistilBERT, BERTweet y BERT-Epi.	Mejores modelos: CNN con F1-score de 0.84. BERT-Epi con F1-score de 0.894.
Categorización y priorización de *tweets* de desastres naturales. (doble anotación)	Dos modelos BERT para cada anotación y uno solo con dos cabezas de clasificación para cada anotación.	El modelo que aborda la doble anotación de forma simultanea obtiene mejores resultados que los modelos independientes.
Clasificar *tweets* que mencionen defectos de nacimiento.	Ensamble de 4 modelos BioBERT. Cada modelo utiliza estados ocultos de diferentes capas del modelo	Para el ensamble F1-score de 0.69. Puntaje promedio de F1-score de 0.65.

	como representación de la oración.	
Clasificar *tweets* en médicos y no médicos.	*Logistic Regression*, *Random Forest*, SVM y BERT. Se utilizó BERT como extractor de características junto con un vector MetaMap (BOW con palabras médicas).	BERT obtuvo un F1-score de 0.927 cuando se usó por sí mismo. En el resto de los modelos los resultados mejoraron cuando se utilizó la concatenación de las características dadas por BERT y MetaMap que cuando se empleaban por individual.

En la Tabla 2.1 se puede apreciar que la mayoría de los trabajos relacionados utilizan modelos basados en *Transformers*. Estos modelos pueden llegar a obtener buenos resultados por si mismos, pero hay muy pocas investigaciones que experimenten con la arquitectura de este tipo de modelos. Esto abre un área de investigación que este trabajo pretende explorar.

3. MARCO TEÓRICO

En esta sección se explicará a detalle las técnicas y algoritmos utilizados en este trabajo de investigación. Comenzando por explicar los conceptos clave de *Machine Learning* y las características que deben tener los conjuntos de datos para poder ser analizados con estas técnicas. Siguiendo con los algoritmos de *Deep Learning* utilizados para crear los modelos clasificadores de este trabajo y las métricas de evaluación de dichos modelos. Por último, se explican las técnicas específicas en procesamiento de lenguaje natural para la preparación y la extracción de características a partir de un texto.

3.1 APRENDIZAJE MÁQUINA (*MACHINE LEARNING*)

Machine Learning es una rama de la inteligencia artificial, su objetivo es lograr que las computadoras aprendan a partir de datos y experiencia previa. Busca lograr que un sistema pueda reconocer patrones en un conjunto de datos de forma automática (Alpaydin 2016).

Hay tres tareas principales que llevan a cabo estos algoritmos. Una de ellas es clasificar datos, es decir, identificar a qué clase pertenece una instancia o también encontrar el valor específico de una de sus propiedades. La otra tarea es la búsqueda de patrones para tratar de agrupar datos según sus similitudes. La última tarea es la de enseñar a una máquina cómo realizar una actividad de forma satisfactoria.

Los nombres de estos tres tipos principales de aprendizaje máquina son, supervisado, no supervisado y aprendizaje por refuerzo. En este trabajo se buscan desarrollar modelos clasificadores, por lo que se empleará el aprendizaje supervisado. Los algoritmos utilizados serán la regresión logística, redes neuronales y *Transformers*. También se utiliza un algoritmo de extracción de tópicos, que pertenece al aprendizaje no supervisado.

3.1.1 Aprendizaje Supervisado

Los algoritmos de aprendizaje supervisado se utilizan más comúnmente en tareas de regresión o de clasificación. Estos algoritmos aprenden a reconocer patrones en conjuntos de datos. Una vez que encuentran esos patrones, los aplican en datos nunca antes vistos para clasificarlos o predecir el valor de una de sus propiedades. Para que sean capaces de encontrar estos patrones, necesitan de datos ejemplo. Los ejemplos tienen que contener todas las características y la respuesta correcta sobre el valor buscado o la clase a la que pertenece. Así logra aprender las relaciones que hay entre las características dadas y el objetivo (Alpaydin 2016). A estos ejemplos etiquetados con el valor objetivo, se le conoce como datos de entrenamiento, y se utilizan para entrenar algoritmos supervisados.

Regresión Logística (Logistic Regression)

Este se trata de un modelo clasificador, el cual aprende reglas de generalización a partir de un conjunto de datos etiquetado, para poder clasificar datos nuevos. El modelo de Regresión Logística utiliza una ecuación de regresión lineal, que incluye una función sigmoide, que calcula la probabilidad de que un dato pertenezca a una clase (Géron, 2019).

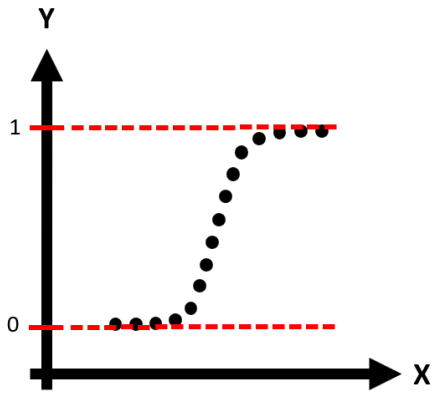

Figura 3.1: Función Sigmoide en *Logistic Regression*

La ecuación de regresión lineal se describe en la ecuación 3.1, donde su resultado dará un valor continuo, el cual no es aún suficiente para definir una clasificación.

$$\theta = \beta_0 + \beta_1 X \tag{3.1}$$

Para solucionar esto se utiliza en conjunto con la función sigmoide, esto ocasiona que cualquier valor continuo dado sea ajustado para dar un nuevo valor entre 0-1.

$$sig(\theta) = \frac{1}{1 + e^{-\theta}} \tag{3.2}$$

Si se define un valor umbral, por ejemplo, 0.5, se puede hacer una clasificación de datos. Los que den como resultado un valor menor o igual al umbral pertenecerán a una clase y los demás a la otra clase.

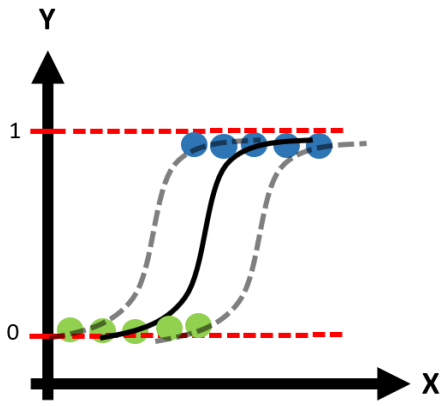

Figura 3.2: Curva sigmoide.

Para que el modelo tenga un buen desempeño, es necesario encontrar los parámetros que ajusten mejor la curva sigmoide a los datos ejemplo que se tienen para entrenamiento. Para lograr eso se define una función de costo que hay que minimizar. La función de costo penaliza los errores hechos por el clasificador y es mostrada en la ecuación 3.3, donde *hθ(x)* es la predicción hecha del ejemplo x, y *y* es la clasificación real del dato.

$$Loss(h_\theta(x), y) = -ylog\big(h_\theta(x)\big) - (1 - y)\log\,(1 - h_\theta(x)) \tag{3.3}$$

Este proceso se llama "*maximum likelihood*", y la ecuación de pérdida puede ser minimizada para encontrar los parámetros ideales por varias técnicas de optimización como gradiente descendente (Swaminathan, 2018).

Este es un modelo de clasificación binaria, pero que puede ajustarse para trabajar con una mayor cantidad de clases, utilizando el método "*one vs rest"* o "*multinomial logistic regression*". En el acercamiento "*one vs rest"* se entrena un modelo clasificador por cada clase existente. Para entrenar a cada modelo, se considera una clase objetivo y el resto de clases es tomado como una sola, convirtiéndolo así en un problema binario (Géron, 2019).

3.1.2 Aprendizaje No Supervisado

Muchas veces se cuentan con datos en bruto, es decir, sin ningún orden o clasificación específica, y lo que se busca es dar estructura a estos datos. A diferencia del aprendizaje supervisado, los algoritmos no supervisados no necesitan de datos de entrenamiento previamente etiquetados con el valor objetivo. Estos algoritmos buscan reconocer patrones en los datos y proponer una organización de forma que haga sentido. Ya que no existen respuestas concretas en estos datos, el algoritmo busca similitudes entre cada dato para agruparlos en grupos, y así obtener categorías de ellos (Alpaydin 2016). Un ejemplo de un algoritmo no supervisado es: se tiene una gran cantidad de documentos de texto, pero todos se encuentran mezclados entre sí. Se busca un algoritmo que ordene los documentos por temática, agrupando a todos los libros según los temas de política, economía, literatura, ingeniería, etc. Uno de estos algoritmos es el llamado *Non-Negative Matrix Factorizaction*, que se describe a continuación.

Factorización No Negativa de Matrices (Non-Negative Matrix Factorization NMF)

Este se trata de un modelo no supervisado comúnmente utilizado para la extracción de tópicos a partir de un conjunto de documentos. En cierta forma, busca el valor de

pertenencia que tiene un documento a un tema. En su forma más simple, el objetivo de este modelo es que dada una matriz A de tamaño m x n debe encontrar una matriz W de m filas con k columnas, y otra matriz H de tamaño k x n. Se busca que la multiplicación de W por H sea igual o similar a la matriz A.

$$A \cong W.H \tag{3.13}$$

$$\|A - WH\|^2 \tag{3.14}$$

Las matrices W y H son determinadas al minimizar la norma de Frobenius (ecuación 2.14). Al aplicar este modelo a texto, la dimensión k es especificada por el usuario, que corresponde al número de tópicos que debe encontrar. Recibe como entrada las características TF-IDF de los documentos, que deben ser siempre igual o mayores a 0. La matriz A corresponde a documentos por palabras, la H a documentos por tópicos y la W a tópicos por palabras. Cada palabra tiene asignado un valor en relación con cada tópico, y el tópico de un documento nuevo se define por las palabras que contiene.

3.2 CONSTRUCCIÓN Y PREPARACIÓN DE UN CONJUNTO DE DATOS

Con la llegada del internet y sobre todo de la web 2.0, la cantidad de datos disponibles nunca había sido tan grande como ahora, por lo que conseguir datos sobre un tema ya no requiere de una gran cantidad recursos como antes (Blank and Reisdorf, 2012), sin embargo, la complejidad radica en preparar estos datos para alimentar un modelo de aprendizaje.

Todos los algoritmos de *Machine Learning* y *Deep Learning* necesitan datos para aprender. Una regla general es que el desempeño del modelo será tan bueno como la calidad del conjunto de datos. Por eso, en el libro de *Aurélien Géron* (2017) y en el curso en línea de *Machine Learning* de Google (2020), se mencionan los siguientes temas a tomar en cuenta al crear y preparar un *dataset*.

3.2.1 Suficiente Cantidad de Datos para Entrenamiento

Se necesita una cantidad suficiente de datos para que el modelo pueda aprender. No hay una regla que diga la cantidad mínima que se debe tener, pero se recomienda que haya al menos diez veces más ejemplos que el número de características. También numerosas investigaciones concluyen que el desempeño de los modelos mejora mientras más datos se tengan (SUG, 2018).

3.2.2 Selección de Características Relevantes

Un *dataset* de calidad debe contener las características más importantes de los datos, propiedades que estén relacionadas al valor objetivo. Hay diversas formas de realizar esto. Una de ellas puede ser realizando un acercamiento empírico, a base de prueba y error con las características disponibles. También se puede realizar una ingeniería de características para crear nuevas medidas, después probar el desempeño del modelo y ver cuáles son las que más lo ayudan.

Otra forma consiste en utilizar algoritmos que realizan pruebas del conjunto de características para estimar su importancia con respecto a la variable objetivo. Existe una variedad de técnicas para la selección de características. A continuación, se describe el método de análisis de información mutua que fue utilizado en esta investigación.

Análisis de Información Mutua

Una vez que ya se tienen las características a partir de las cuales se entrena un modelo, se puede aplicar un análisis de información mutua seleccionar las características más discriminantes. Esto significa que se calcula que tan relacionadas están dos variables. En el caso de un conjunto de datos categorizados, mide la dependencia mutua de una variable categórica dada otra variable. (Witten et al. 2011)

Una forma de calcular el valor de información mutua es la siguiente:

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \tag{3.4}$$

Donde H(X) y H(Y) es la entropía para las variables X, Y. H(X,Y) es la entropía condicional para X dada Y.

El valor de información mutua es no negativo. Es igual a cero cuando las dos variables son independientes y mientras sea más alto, mayor dependencia tendrán ambas variables.

3.2.3 Limpieza de Datos

La limpieza de los datos consiste en verificar que no existan errores o situaciones fuera de lo común. Entre las situaciones a verificar está asegurar que el tipo de valor de cada característica sea el correcto y no mezclar formatos de datos. Que no queden campos vacíos, y si los hay, llenarlos de alguna forma o eliminarlos. Revisar si hay valores atípicos (que sean muy diferentes a los demás ejemplos) para evaluar si fue un error de medición o si son importantes.

Limpieza de Texto de Redes Sociales

A diferencia de otras fuentes de texto como lo son los libros y artículos, el texto de redes sociales puede ser muy caótico y requiere de pasos de limpieza específicos. Los usuarios de estas plataformas son libres de escribir de cualquier manera y en cualquier idioma. El texto producido por el usuario puede incluir faltas de ortografía, *links* URL, nombres de usuarios, y emojis. Los pasos más comunes en la limpieza de texto tienen que ver con lo mencionado anteriormente, dependiendo de las técnicas o algoritmos que se utilicen (Pradha et al., 2019).

3.2.4 Conjuntos de Entrenamiento, Validación y Prueba (*Training, Validation and Testing Sets)*

A la hora de entrenar un modelo, generalmente se separa el 80% del *dataset* para entrenamiento y el 20% restante para evaluarlo. A estas separaciones se les conoce como el *training set* y el *test set*, respectivamente. También existe otra partición llamada *validation set*, que es una pequeña parte del *training set* que se separa antes de hacer el entrenamiento, y se usa para probar el desempeño en ella y con base en los resultados, hacer el ajuste de hiperparámetros del modelo. Debido a que este último set se usa para hacer ajustes finos del modelo, con ello se evita tomar decisiones de entrenamiento sobre el set de prueba. El modelo con mejor desempeño en el *validation set*, puede ser elegido como el modelo que mejor generalizó los datos, y después se usará todo el conjunto de entrenamiento junto con el de validación para entrenamiento. Por último, la evaluación en el *test set* da una estimación final acerca del desempeño en datos nunca antes vistos.

3.3 APRENDIZAJE PROFUNDO (*DEEP LEARNING*)

Aprendizaje profundo se considera una de las ramas de aprendizaje máquina, y más específicamente, se utiliza como el concepto que abarca a los algoritmos basados en redes neuronales artificiales. Estos modelos son reconocidos por haber mejorado significativamente los resultados obtenidos en tareas de visión por computadora, reconocimiento de voz, procesamiento de lenguaje natural, inteligencia artificial de juegos, y otros campos (Ciregan et al., 2012; Krizhevsky et al., 2017). Existen varios tipos de arquitecturas que se ajustan mejor a diferentes aplicaciones. Entre ellas se encuentran las redes recurrentes, las convolucionales, basados en atención, entre otros. A continuación, se explican a detalle los métodos utilizados en esta investigación, incluyendo las redes neuronales artificiales, los mecanismos de atención, *Transformers* y la arquitectura *wide & deep*.

3.3.1 Red Neuronal Artificial

Es un modelo inspirado en el funcionamiento de la corteza cerebral, donde existen neuronas que están interconectadas entre sí, y se comunican por medio de impulsos eléctricos. El modelo imita esta arquitectura de una forma simplificada, dividida en tres partes. La capa de entrada, las capas escondidas y la capa de salida. En cada capa existen unidades o neuronas, que están conectadas con todas las unidades de la capa anterior y la capa siguiente y así sucesivamente, como se observa en la Figura 3.3. La primera capa de entrada (*input layer*) recibe los datos de entrada. Las unidades de la capa escondida (*hidden layer*) son la parte intermedia del proceso de aprendizaje. Las neuronas de la última capa son las responsables de determinar el resultado final dependiendo de la tarea (Goodfellow et al., 2016).

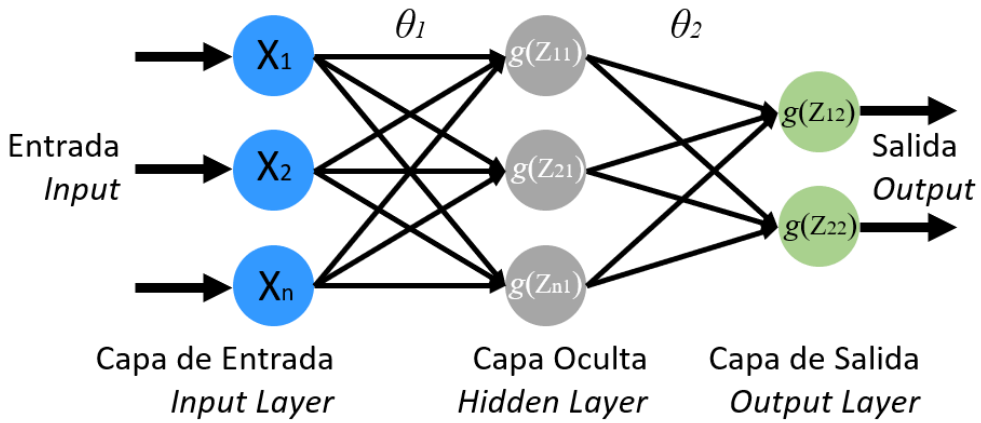

Figura 3.3: Red Neuronal MLP.

Algunos de los conceptos importantes para comprender el funcionamiento de los modelos de aprendizaje profundo según Marsland (2014) son:

- **Entradas (*inputs*)**: Las entradas del modelo, o el vector de entrada, son las características de un solo ejemplo de nuestro conjunto de datos.

- **Pesos (*weights*)**: El vector de pesos (θ) se refiere a los valores dados a las conexiones entre las unidades, para determinar la influencia que cada conexión tiene.
- **Función de Activación (*activation function*)**: A la salida de cada capa se aplica la función matemática que dicta bajo qué condiciones (o rango de valores) se activan las unidades de esa capa, algunos ejemplos *Rectified Linear Unit (ReLU)* y *Softmax* de las ecuaciones 3.4 y 3.5 respectivamente.

$$g(x) = max(0, x) \tag{3.4}$$

$$g(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{3.5}$$

- **Salidas (*outputs*)**: El vector de salida se refiere a la predicción del algoritmo, que se obtiene de la última capa sobre los datos que le fueron alimentados. Es el resultado final del algoritmo.
- **Objetivos (*targets*)**: Es el vector de los valores finales correctos de cada ejemplo. Solo se utilizan en algoritmos supervisados.
- **Error**: El error sirve para medir el desempeño del modelo. Se calcula mediante una función de costo, a partir del valor de salida predicho y el valor correcto.

Según el libro *Deep Learning for NLP and Speech Recognition* (Kamath et al., 2019), el entrenamiento de una red neuronal completamente conectada sigue cuatro pasos principales:

1. **Propagación hacia adelante (*Forward Propagation)***

   En la primera corrida se inicializan los vectores de pesos θ de forma aleatoria y una unidad llamada *bias* (presente en la capa de entrada y las capas escondidas).

La capa de entrada contiene las características de un solo ejemplo, y cada uno de estos valores está conectado a todas las neuronas de la capa siguiente. Cada neurona tiene su propio vector de pesos θ, valores asignados a cada uno de los valores de entrada X. Se multiplica cada valor de entrada por el peso correspondiente de la neurona y se suman, más la unidad de *bias*. La ecuación 3.6 muestra la operación, donde *i* denota el valor de entrada actual y *j* la capa.

$$Z = \sum_{i} X_i \theta_{ij} + b \tag{3.6}$$

Este cálculo dará un valor escalar que se ingresará a una función de activación, en este caso la función *ReLU* (ecuación 3.7). El resultado será la salida de esa unidad.

$$a = g(Z) = \max(0, Z) \tag{3.7}$$

Este proceso se repite por cada neurona de la capa. El conjunto de los valores de salida de todas las neuronas de esa capa se puede considerar como el nuevo vector de entrada de la siguiente. Por lo que se repiten las mismas operaciones de forma consecutiva hasta llegar a la última capa.

La última capa calcula de igual forma su salida, y aplica su propia función de activación. La función de activación de la última capa suele ser diferente a la de las capas escondidas, las más comunes en clasificación son la función sigmoide (para clasificación binaria) y la *softmax*.

2. Cálculo del Error (*Error Computation*)

Se calcula el error entre la predicción arrojada y la predicción real mediante una función de costo. Existen varias, y su elección depende de si se trata de una tarea de regresión o clasificación. Para regresión una función popular es la de *mean squared error* (ecuación 3.8), y *Binary Cross-Entropy* para clasificación binaria (ecuación 3.9).

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{3.8}$$

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{3.9}$$

Donde y_i es el *target* u objetivo y $\hat{y}_i$ es la predicción hecha por el modelo.

3. **Propagación hacia atrás (*Backpropagation)***

Este paso es para conocer la variación del error con respecto a cada uno de los pesos de las neuronas y poder realizar ajustes en la magnitud de los pesos que minimicen el error. Para esto se utiliza el método de gradiente descendente. Primero se calcula el gradiente del error, según la función de costo que se haya utilizado, con respecto a los pesos de la capa de salida. Después se propaga hacia atrás con la regla de la cadena para calcular el gradiente con respecto a los pesos θ. Este proceso se repite hasta llegar a la primera capa.

4. **Actualización de pesos (*Parameter Update)***

Ya que se tienen los gradientes calculados, el siguiente paso es actualizar los pesos con la siguiente formula:

$$\theta = \theta - \alpha \nabla_\theta E \tag{3.10}$$

Donde α que también es conocida como *learning rate* es la magnitud de que tanto se actualiza el peso en cada iteración. Si α es muy pequeño, el algoritmo tardará más tiempo en converger. Si es muy grande podría no llegar a converger.

Todo este proceso se repite por un determinado número de épocas (ciclos de entrenamiento), las cuales recorren la totalidad de los ejemplos del set de entrenamiento.

Por último, una técnica recomendada que utilizan estas redes para evitar el sobre-entrenamiento es la técnica de *dropout*. El *dropout* consiste en quitar temporalmente una

conexión elegida al azar con una probabilidad p de la capa escondida especificada durante un ciclo. Esto hace que la responsabilidad de las activaciones no recaiga en unas cuantas neuronas, y que las siguientes capas aprendan a corregir los errores de las capas anteriores.

3.3.2 Mecanismos de Atención

En un vector secuencial de características hay algunas que son más importantes que otras. Los mecanismos de atención permiten prestarles más atención a los elementos de una secuencia que sean más relevantes para cumplir la tarea del modelo. De igual forma, prestan menos atención a los elementos irrelevantes (Kamath et al., 2019).

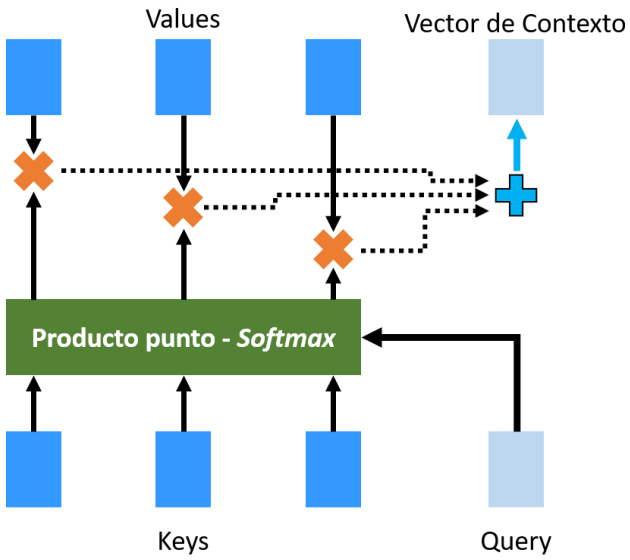

Figura 3.4: Mecanismo de Atención

La atención es una ponderación de cada uno de los elementos de la secuencia con todos los demás incluyéndose a si mismo, a esto se le llama vector de contexto. Se calcula utilizando un *score* de relevancia entre el estado actual y todos los anteriores. El *score* puede ser calculado utilizando diferentes funciones, una de ellas es el producto punto de los estados ocultos como se indica en la ecuación 3.11.

$$score(Keys, Query) = Keys^T Query \tag{3.11}$$

Donde *Keys* se refiere a los estados ocultos anteriores y *Query* el estado actual. Después que se ha calculado el vector *score*, es pasado por una función *softmax*, para que su sumatoria sea igual a uno, donde las posiciones con los valores mayor magnitud indican que el estado oculto correspondiente está más relacionado con el estado actual. Cada estado oculto (*Values*) se multiplica por su *score* correspondiente y la suma de todos los vectores resultantes producen el vector de contexto (Kamath et al., 2019). Este procedimiento se muestra en la Figura 3.4.

3.3.3 Transformadores (*Transformers*)

Este es uno de los modelos que ocupan el estado del arte en procesamiento de lenguaje natural, y está basado en mecanismos de atención. Es capaz de recibir una secuencia de datos y calcular una representación vectorial contextualizada. Consta de dos partes, llamadas *encoder* y *decoder* (Kamath et al., 2019).

La primera parte consiste en calcular un *embedding* (vector) de posicionamiento de cada elemento de la secuencia que va entrando y sumarlo al *embedding* original del elemento. Después, se tiene una capa de mecanismos de atención llamada *self-attention*. Esta capa recibe como entrada los *embeddings* y crea nuevos vectores de cada elemento, captando la relación entre ese elemento y todos los demás. El mecanismo de atención aprende a calcular un vector *score*. Este vector contiene un peso que indica la relevancia de la relación de ese elemento con todos los demás, incluyéndose a sí mismo. El vector de *score* se normaliza por medio de una función *Softmax* para obtener una distribución. Los pesos de este vector se multiplican con el *embedding* del elemento correspondiente, para ponderar su relación con el *embedding* actual, y se suman todos los resultados. El *embedding* resultante ahora contendrá la información original, su posición en la secuencia y la relación que tiene con

todos los demás elementos, es decir, el contexto. Este proceso se repite ocho veces en paralelo, en lo que denominan como las ocho cabezas de atención. Cada cabeza presta atención a diferentes elementos de la secuencia, por lo que cada una de ellas calcula un *embedding* contextualizado del mismo elemento, pero con cargas contextuales diferentes.

Las salidas de cada cabeza de atención son concatenadas para formar un vector de 512 dimensiones. A este vector se le suma una conexión residual del mismo vector que fue alimentado a la capa de *self-attention*. El vector resultante entra a una capa de normalización y a una capa densa. La red tiene una capa oculta de 2048 unidades y una capa de salida de 512. Esta capa tiene el fin de realizar una transformación lineal al vector de *embedding*. Después, a la salida de esta capa densa se le suma una conexión residual, el mismo vector de entrada de la red. El vector resultante es pasado nuevamente por una capa de normalización.

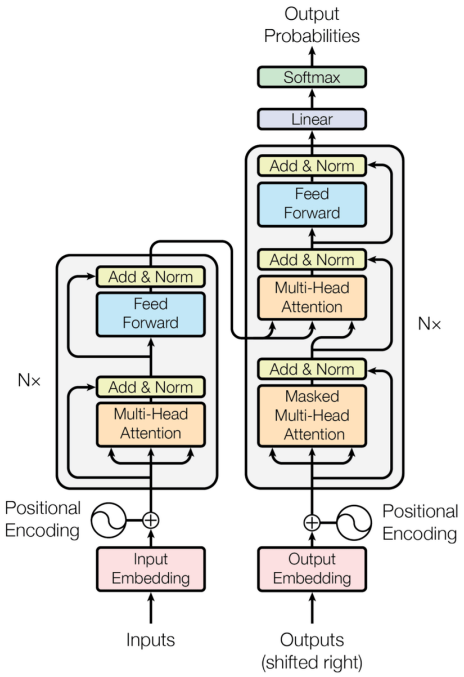

Figura 3.5: Arquitectura *Transformer* (Vaswani et al., 2017)

Todo este proceso se repite seis veces, ya que existen seis capas de *encoder* apiladas. Existen algunas variaciones en este modelo en cuestión de parámetros, como el tamaño de los vectores, el número de cabezas de atención, la capa oculta de la red densa, el número de

encoders, entre otros. La segunda parte, el *decoder*, funciona de forma similar, pero está diseñada para tareas *sequence to sequence* como la traducción de texto. Por esta razón, se utiliza únicamente la parte del *encoder* para tareas de clasificación (Vaswani et al., 2017).

3.3.4 Arquitectura Ancha y Profunda (*Wide & Deep*)

Esta arquitectura trata de combinar dos modelos distintos en uno solo para mejorar las predicciones. Un modelo es llamado ancho (*wide*) al utilizar características categóricas o dispersas, como características tipo *Bag-of-Words*, junto con transformaciones no lineales de ellas. Este modelo logra cierto nivel de memorización de las relaciones entre las características pero carece de generalización. El modelo profundo (*deep*) emplea otro tipo de características, densas a forma de *embeddings* y tiene capas ocultas. De esta forma, el modelo profundo puede lograr una mejor generalización de los datos. En la arquitectura *wide & deep*, se combinan estos dos modelos en uno solo para hacer la predicción. A diferencia de los ensambles, aquí ambos modelos se entrenan en conjunto, ya que sus salidas están conectadas a la última capa encargada de realizar la predicción. La Figura 3.6 muestra un ejemplo de la arquitectura (Cheng et al., 2016).

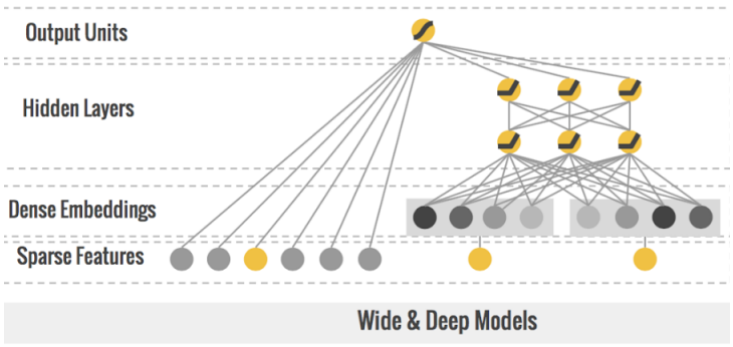

Figura 3.6: Arquitectura *Wide & Deep* (Cheng et al., 2016)

La idea general de esta arquitectura radica en combinar las ventajas que ofrecen las ramas anchas y profundas para obtener un mejor desempeño. Siguiendo esta idea, es posible

customizar ambas ramas para ajustarlas a una tarea de clasificación de *tweets*. Para esto es necesario elegir un modelo que actúe como la parte profunda. Para la parte ancha se puede llevar a cabo una ingeniería de características para crear nuevas métricas que se ajusten mejor a la clasificación de *tweets* sobre Covid-19.

3.4 MÉTRICAS DE EVALUACIÓN

Para evaluar el desempeño de modelos clasificadores se utilizan las métricas de *accuracy*, *precision*, *recall* y *f1-score*. Estos valores se calculan a partir de las medida mostradas en la Figura 3.7.

Matriz de Confusión		**Predicción**	
		Positivos	**Negativos**
Valores Reales	**Positivos**	Verdaderos Positivos (**VP**)	Falsos Negativos (**FN**)
	Negativos	Falsos Positivos (**FP**)	Verdaderos Negativos (**VN**)

Figura 3.7: Matriz de Confusión

De un lado se tienen los valores reales de un conjunto de datos, que corresponden a las clases a las que pertenecen cada ejemplo. Por otro lado, está la predicción hecha por el modelo sobre a qué clase pertenece cada ejemplo. Cuando coincide la clase predicha con el valor real, se cuentan como valores verdaderos, y cuando no coinciden, valores falsos. A continuación, se explican las métricas que se calculan a partir de estas medidas (Powers, 2008).

3.4.1 Accuracy

Accuracy es la medida que se refiere a la proporción de ejemplos clasificados correctamente. Se encuentra en un rango entre 0 a 1, donde 1 significa que todos los ejemplos fueron clasificados correctamente. Se calcula con la siguiente ecuación:

$$Acc = \frac{VP + VN}{VP + VN + FP + FN} \tag{3.15}$$

3.4.2 Precision

Se refiere a la fracción de valores clasificados correctamente de una sola clase entre el total de valores predichos en esa misma clase.

$$Precision = \frac{VP}{VP + FP} \tag{3.16}$$

Podría verse como una medida que indique cuantos ejemplos clasificados sí son relevantes. Va de un rango entre 0 a 1, donde 1 significa que todos los ejemplos predichos de una clase sí pertenecen a ella, sin importar si logró recuperar todos los ejemplos correctos.

3.4.3 Recall

Es la fracción de ejemplos de una clase predichos correctamente, entre la cantidad total de ejemplos pertenecientes a dicha clase.

$$Recall = \frac{VP}{VP + FN} \tag{3.17}$$

Podría verse como una medida que indique que tantos ejemplos de una clase puede recuperar. Va de un rango entre 0 a 1, donde 1 significa que logró recuperar la cantidad total de ejemplos de esa clase, sin importan los errores de otras clases.

3.4.4 F1-score

Es la media armónica de las medidas de *precision* y *recall*, calculado de la siguiente forma.

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.18}$$

Se encuentra en un rango de 0 a 1, donde 1 indica un desempeño perfecto del modelo.

3.5 PROCESAMIENTO DE LENGUAJE NATURAL

También conocido por su nombre en inglés, *Natural Language Processing* (*NLP*) es una rama de la inteligencia artificial relacionado con la lingüística, enfocada en interacción entre las computadoras y el lenguaje de los seres humanos. Su propósito es ayudar a las computadoras a comprender, interpretar y generar lenguaje humano (Goodfellow et al., 2016).

Algunas de las aplicaciones más comunes de NLP son: traducción, generación de texto, clasificación de sentimiento u otras clases, búsquedas semánticas y reconocimiento de entidades nombradas (Kamath et al., 2019). Para este trabajo nos enfocaremos en la clasificación de texto.

3.5.1 Representaciones de Texto

Las computadoras son incapaces de comprender directamente el lenguaje humano textual, y por eso necesitan una representación numérica de las palabras para poderlas analizar. Existen muchas técnicas para representar las palabras en forma numérica vectorial, y a continuación se mencionan dos de ellas que fueron empleadas en este trabajo, llamadas Bolsa de Palabras (*Bag-of-Words*) y *Embeddings* (Goodfellow et al., 2016).

3.5.2 Bolsa de Palabras (*Bag-of-Words*)

Las bolsas de palabras (*Bag of Words*) son una técnica de representación vectorial de texto. Consiste en crear un vocabulario a partir de todos los documentos o *tweets* del *dataset*, con las palabras más frecuentes y de tamaño N. Después, cada documento es transformado a un vector de longitud N, donde cada dimensión es la frecuencia con la que aparece una palabra en específico en ese documento. Por ejemplo, con la oración "*I have the corona virus*", su representación sería la siguiente:

Tabla 3.1: Ejemplo Bag-of-Words.

	I	***have***	***the***	***corona***	***virus***	**…**	**…**	**…**	***Word N***
Doc1	1	1	1	1	1	0	0	0	0

Una de las desventajas es que es una representación de características muy dispersa, con vectores de alta dimensionalidad que puede afectar el rendimiento del modelo.

Como se puede ver en el ejemplo anterior, esta representación solo indica si una palabra está presente o no. En algunos casos, conocer la sola presencia de la palabra no es suficiente para obtener buenos resultados. Por esta razón existe una variante normalizada llamada TF-IDF, que significa *Term Frequency – Inverse Document Frequency*. Esta variación evalúa que tan relevante es una palabra para un documento en una colección de documentos. Se calcula multiplicando las veces que una palabra aparece en un documento (en nuestro caso un *tweet*) por la frecuencia inversa del documento en un conjunto de documentos. (Aggarwal, 2018).

$$x_{tfidf} = x_i . \log\left(\frac{1+n}{1+n_i}\right) + 1 \tag{3.12}$$

Donde x_i es la frecuencia de la palabra en un documento, normalizada por el término logarítmico, donde n es el número total de documentos dividido entre n_i, número de documentos que contienen esa palabra.

3.5.3 *Document, Sentence y Word Embeddings*

Los *embeddings* son formas de representar letras, palabras, oraciones o documentos enteros de forma numérica vectorizada. Los valores de este vector actúan cómo las características de esa palabra u oración, y de esta forma pueden ser alimentadas a algoritmos de aprendizaje máquina. Algunos de los métodos más utilizados para calcular *embeddings* son: Word2Vec y fastText, ambos basados en las técnicas de *Continuous Bag-of-Words* y *Skip-gram*. Otro método efectivo (y el empleado en este trabajo) es *Bidirectional Encoders*

Representations from Transformers (BERT). Lo que tienen en común todos ellos es que utilizan algoritmos de *deep learning* para calcular las representaciones numéricas (Kamath et al., 2019). Los métodos Word2Vec y fastText utilizan redes neuronales *shallow* o poco profundas, y generan vectores únicos para cada palabra. Mientras los algoritmos tipo BERT generan vectores contextualizados de cada palabra, por lo que la misma palabra puede tener diferentes representaciones según su contexto.

Bidirectional Encoder Representations from Transformers

Bidirectional Encoding Representations from Transformers (BERT) es un modelo basado en *Transformers*. Busca aprovechar la técnica de *transfer learning*, al pre-entrenarse en tareas no supervisadas, y solo necesitar un *fine tuning* para nuevas aplicaciones. Está diseñado para ser flexible y adaptarse fácilmente a las diferentes tareas de NLP con cambios mínimos. Su función consiste en crear *embeddings* contextualizados, a diferencia de otras técnicas como C-BOW que calculan una única representación por palabra. Esto significa que puede crear definiciones diferentes de cada palabra según el contexto en que es utilizada.

Su arquitectura está basada en varias capas apiladas del *encoder* de los *transformers*. El modelo BERT base implementa 12 capas de *encoder* apiladas, con vectores de 768 dimensiones y 12 cabezas de atención. Estos parámetros cambian según la versión de BERT. Para su pre-entrenamiento, se utilizan la salida de la última capa del *encoder* para completar dos tareas diferentes en conjunto. La primera tarea consiste en predecir las palabras del texto que han sido enmascaradas de forma aleatoria por el modelo. La segunda tarea trata de predecir si una oración B es la que le sigue a una oración dada A. Este modelo puede ser pre-entrenado utilizando grandes cantidades de texto, y de temáticas generales o específicas. El modelo base desarrollado por Google fue entrenado en el corpus de Wikipedia y el *BookCorpus* en inglés, durante cuatro días en cuatro TPUs (Devlin et al., 2019).

Ya que el modelo ha ajustado sus parámetros para cumplir con las tareas anteriores, solo necesita de un *fine tuning* para dar buenos resultados en otras tareas. Para aplicar el modelo en diferentes aplicaciones, basta con utilizar la salida de la última capa del *encoder* para alimentar un modelo adaptado a la tarea en cuestión (que normalmente es una red neuronal densa). Para clasificación, se utiliza el primer elemento de la salida (vector de características), que corresponde al token de inicio de oración y que contiene la información contextualizada de toda la oración.

4. **CONJUNTOS DE DATOS**

Lo primero que se realizó en la investigación fue una búsqueda de conjuntos de datos etiquetados, relacionados al Covid-19. Durante la búsqueda se encontró la sexta edición de la competencia *Social Media Mining for Health Shared Tasks* (#SMM4H). Esta competencia fue organizada por el *Health Language Processing Lab* del *Institute of Biomedical Informatics de la Perelman School of Medicine, University of Pennsylvania*. Tiene el objetivo de promover la investigación en los métodos automáticos de colección, extracción, representación, análisis y validación de datos provenientes de redes sociales. Donde los participantes tienen que afrontar los retos de utilizar estos datos para la investigación médica (Magge et al., 2021).

Como parte de esta competencia, se propusieron ocho tareas a los participantes utilizando textos de Twitter.

1. Clasificación, extracción y normalización de efectos adversos mencionados en tweets en inglés.
2. Clasificación de tweets en ruso para detectar la presencia de efectos adversos.
3. Clasificación de tweets donde el usuario menciona que cambia su régimen de medicamentos.
4. Clasificación de tweets donde el usuario reporta efectos adversos de su embarazo.
5. Clasificación de tweets donde el autor tiene un caso potencial de COVID-19.
6. Clasificación de tweets de COVID-19 que contienen síntomas.
7. Identificación de profesiones y ocupaciones en tweets en español.
8. Clasificación de publicaciones de cáncer de mama de los propios usuarios en Twitter.

Cabe mencionar que como parte de esta investigación se participó en las tareas 5 y 6, donde se obtuvo acceso a un conjunto de datos para cada una, y se describen a continuación.

4.1 #SMM4H TAREA 5: IDENTIFICACIÓN DE CASOS POTENCIALES DE COVID-19 EN *TWEETS*

Esta tarea busca distinguir *tweets* en inglés que reportan casos potenciales de Covid-19 del resto de *tweets*. Un caso potencial etiquetado con un "1" se refiere a que el autor o alguien de su hogar está enfermo o padece un síntoma, que se les ha negado realizarse una prueba de la enfermedad, o que han sido expuestos a una situación riesgosa de contagio. Una situación riesgosa de contagio incluye haber estado presente en grandes reuniones de personas, estuvo cerca de alguien con un síntoma respiratorio, o no tomó las medidas de prevención adecuadas. El resto de *tweets* etiquetados con "0" corresponden a cualquier otro que haga referencia al Covid-19 pero sin ser considerado un caso potencial (Klein et al., 2021).

Tabla 4.1: Estructura de datos tarea 5.

tweet_id	user_id	tweet	label
Número identificador del *tweet*.	Número identificador del usuario.	*I think I have the coronavirus I've been coughing nonstop all day and I feel really warm*	1
		With coronavirus we certainly need more doctors and surgeons and nurses and sonographs and radiologists, let them in, quick!	0

Los datos fueron compartidos en formato .tsv (*tab-separated values*), y los campos de los datos se muestran en la Tabla 4.1.

Tabla 4.2: Conjuntos de datos tarea 5.

Datos	"1"	"0"	Totales
Entrenamiento	1,026	5,439	6465
Validación	122	594	716
Prueba	N/A	N/A	10,000

La Tabla 4.2 muestra la distribución de etiquetas de los datos de entrenamiento y validación. Los datos de prueba fueron compartidos sin etiquetas. La evaluación en el set de prueba debe realizarse a través de la página de la competencia en *Codalab* (https://competitions.codalab.org/competitions/28766), subiendo las predicciones hechas con los campos de *tweet_id* y *label*, para recibir automáticamente el *f1-score*, *precision* y *recall* de la clase de caso potencial. Las predicciones deben ser subidas en formato .tsv con los campos mencionados y dentro de un archivo .zip.

4.2 #SMM4H TAREA 6: CLASIFICACIÓN DE *TWEETS* CON SÍNTOMAS DE COVID-19

Esta tarea busca clasificar *tweets* en inglés en una de tres clases. La primera clase es llamada *self_reports*, y contiene aquellos *tweets* donde el autor sufre de un síntoma. En la segunda clase *non-personal reports* se incluyen los textos donde un tercero es el que padece el síntoma. Por último, la clase *literature/news mentions* son los *tweets* que describen síntomas pero que nadie los sufre, tales como noticias o descripciones de la enfermedad.

Tabla 4.3: Estructura de datos tarea 6.

tweet_id	tweet	label
Número identificador del *tweet*.	*The problem is I get shortness of breath from stomach acid and post-nasal drip. I never used to notice but I do now my lungs are crap. How're you feeling?*	Self_reports
	My daughter has low fever till yesterday What can we do	Nonpersonal_reports
	8 in 10 COVID-19 patients suffer neurological symptoms, study finds URL	Lit-News_Mentions

Los datos fueron compartidos en formato .tsv, y los campos se muestran en la Tabla 4.3.

Tabla 4.4: Conjuntos de datos tarea 6.

Datos	Lit-News	Self reports	Nonpersonal reports	Totales
Entrenamiento	4,277	1,348	3,442	9,067
Validación	247	73	180	500
Prueba	N/A	N/A	N/A	6,500

La Tabla 4.4 muestra la distribución de etiquetas de los datos de entrenamiento y validación. Al igual que en la tarea 5, los datos de prueba omiten las etiquetas y la evaluación se realiza por medio de *Codalab*. Subiendo las predicciones hechas en formato .tsv con los campos *tweet_id* y *label* y en un archivo comprimido .zip. El desempeño se mide por medio de *Micro f1-score*, *precision* y *recall*.

5. MÉTODO BASADO EN BERT / CT-BERT

Para responder a la pregunta de investigación si un modelo pre entrenado en muchos datos generales puede superar a otro entrenado en menos datos específicos, se implementaron dos modelos tipo BERT. En esta sección se explica a detalle la implementación completa de los modelos únicamente profundos BERT y CT-BERT. También se presentan los resultados obtenidos y su análisis.

5.1 IMPLEMENTACIÓN DE MODELOS

Existen distintos modelos de BERT, que varían en el número de capas del *encoder*, la cantidad de cabezas de atención y la dimensión de los *embedding*. La versión BERT_Large contiene 24 capas de *encoder*, 16 cabezas de atención en cada capa y *embeddings* con una dimensión de 1024. En total, tiene alrededor de 336 millones de parámetros entrenables. El modelo CT-BERT está basado en esta misma versión, pero varía en los datos de pre-entrenamiento. El modelo original de BERT por Google fue pre-entrenado en el corpus de Wikipedia en inglés y el *BookCorpus*, sumando un total de 4.3 billones de palabras (Devlin et al., 2018). El modelo CT-BERT fue pre-entrenado en un corpus compuesto por *tweets* que contienen palabras clave de Covid-19 con un total de 0.6 billones de palabras (Müller et al., 2020). Por lo que la cantidad de datos utilizados en el pre-entrenamiento de BERT_Large es aproximadamente siete veces mayor a comparación de CT-BERT.

Para determinar cuál de estos dos modelos tiene un mejor desempeño en las tareas 5 y 6 de la competencia #SMM4H 2021, se llevó a cabo un entrenamiento de ambos para su comparación. Los procedimientos para el preprocesamiento de texto y el entrenamiento fueron los mismos, y se describen a continuación:

5.1.1 Preprocesamiento de Texto

Tanto para los conjuntos de *tweets* de la tarea 5 como de la 6 se llevaron a cabo los siguientes pasos de limpieza por cada *tweet* por medio de expresiones regulares y las librerías *emoji*, *Unidecode* y *HuggingFace* de Python.

1. Reemplazo de los nombres de usuario @usuario con un *token* “user”.
2. Reemplazo de múltiples ocurrencias del *token* “user” a la forma “n user”, donde n es el número de ocurrencias de “user”.
3. Reemplazo de los links a páginas web con el *token* “URL”.
4. Reemplazo de múltiples ocurrencias del *token* “URL” a la forma “n URL”, donde n es el número de ocurrencias de “URL”.
5. Transformar emojis a texto descriptivo con la librería emoji, por ejemplo: 👍 → :thumbs_up:.
6. Estandarizar texto a formato ASCII con la librería *Unidecode*. Con esto se transforman o eliminan símbolos, puntuación y acentos extranjeros que no correspondan a la escritura inglesa.
7. Cambiar todo el texto a minúsculas.
8. Separar los *tweets* por palabras (*tokenization*) con la función *AutoTokenizer* de *HuggingFace* correspondiente a cada modelo (Wolf et al., 2020).
9. Insertar el *token* especial de inicio y final de oración, [CLS], [SEP], respectivamente.
10. Truncar los tweets a un máximo de 120 palabras. Para los textos con menos de 120 palabras, insertar *tokens* especiales de relleno (*padding*) a la derecha del texto con la función *AutoTokenizer*. Este número se definió según el promedio de número de *tokens* en los *dataset*.

Los últimos 3 pasos del preprocesamiento se realizaron de forma automática con la función *AutoTokenizer*, únicamente especificando los parámetros adecuados.

5.1.2 Entrenamiento de Modelos

Los modelos fueron implementados utilizando *Pytorch* y la librería *HuggingFace* (Wolf et al., 2020). Estos modelos se encuentran pre-entrenados, y solo hace falta la adición de una capa de clasificación de oración al final del modelo. Esta última capa se conoce como la cabeza del modelo, ya que la cabeza es intercambiable según la tarea a abordar.

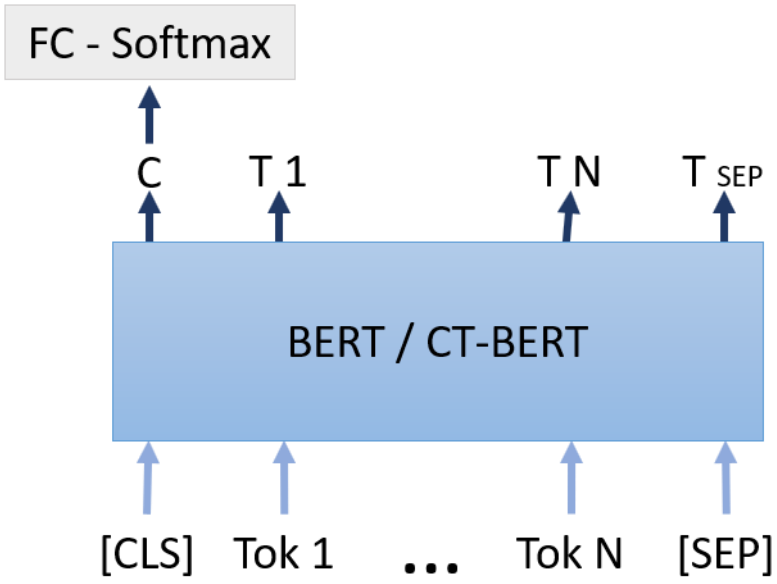

Figura 5.1: BERT / CT-BERT para clasificación de oración

El funcionamiento sigue un diseño estándar para las tareas de clasificación de oraciones con BERT (Figura 5.1). Considera el estado oculto C de la capa final calculado a partir del *token* especial de inicio de oración [CLS]. Esta salida actúa como la representación completa de la secuencia de entrada (las palabras del *tweet*). La salida C es alimentada a la cabeza del clasificador, que consiste en una red neuronal densa con las siguientes especificaciones:

- *Dropout* de 0.1.
- 1024 unidades de entrada.

- 2 unidades de salida para la tarea 5.
- 3 unidades de salida para la tarea 6.
- Función de activación *Softmax* para predecir la probabilidad de clase dada la representación del estado oculto.

Se hizo una búsqueda de hiperparámetros tipo *random search* de los modelos por medio de la librería *Optuna* (Akiba et al., 2019). Los parámetros buscados fueron la decadencia de peso (*weight decay*), tasa de aprendizaje (*learning rate*) y número de épocas de entrenamiento (e*pochs*). El espacio de búsqueda se muestra en la Tabla 5.1, eligiendo de forma aleatoria 10 combinaciones diferentes de hiperparámetros.

Tabla 5.1: Espacio de búsqueda de hiperparámetros BERT/CT-BERT

Weight Decay	*Learning Rate*	*Epochs*
0 – 0.3 con pasos de 1e-4	1e-5 – 5e-5 con pasos de 1e-8	1 – 4 con pasos de 1

Los mejores parámetros se definieron según el *F1-score* más alto sobre el set de validación para ambos modelos en las tareas 5 y 6.

5.2 PARÁMETROS DE ENTRENAMIENTO

Para los modelos profundos, así como la rama profunda de los modelos *Wide & Deep*, se realizó una búsqueda de hiperparametros por medio de la técnica *random search*, con el espacio de búsqueda definido en la Tabla 5.1.

Los mejores hiperparámetros encontrados, según el puntaje obtenido con la métrica F1-score en el set de validación, fueron los siguientes:

Para la tarea 5, los modelos finales se entrenaron con: un *batch_size* de 32, durante 3 épocas de entrenamiento, un *learning_rate* de 3.154e-5 y un *weight_decay* de 0.1328.

Para la tarea 6, los parámetros finales fueron: un *batch_size* de 32, durante 3 épocas de entrenamiento, un *learning_rate* de 3.278e-5 y un *weight_decay* de 0.1423.

Para todos los modelos, de ambas tareas, se empleó una función de pérdida *Cross Entropy*, con una probabilidad de *dropout* de 0.1 entre la salida del modelo y la capa de clasificación, así como una función de activación *Softmax*. También se empleó un optimizador tipo AdamW, y una función *linear schedule with warmup*, para decrecer el parámetro de *learning_rate* de forma lineal hasta llegar a 0 al final del entrenamiento.

Todos los modelos fueron implementados utilizando las librerías *Pytorch* y *Huggingface*.

5.3 RESULTADOS

En las Tablas siguientes se muestran los resultados de los modelos profundos entrenados, así como los modelos que obtuvieron el 1er lugar en la competencia #SMM4H 2021 como punto de comparación. Los primeros resultados mostrados corresponden a la tarea 5.

Tabla 5.2: Resultados BERT/CT-BERT tarea 5

Modelo	**Descripción**	***F1-score***	***Precision***	***Recall***
1er Lugar #SMM4H 2021	Ensamble de 5 modelos tipo BERT	**0.79**	**0.78**	**0.79**
Mediana	Mediana de todas las participaciones	0.74	0.73	0.74
BERT	*Fine-tuning*	0.68	0.71	0.65
CT-BERT	*Fine-tuning*	0.77	0.76	0.77

Los resultados muestran que el modelo CT-BERT, pre-entrenado en *tweets* sobre Covid-19, se desempeñan mejor que el modelo BERT de dominio general.

Los resultados de la tarea 6 se muestran en la Tabla 5.3.

Tabla 5.3: Resultados BERT/CT-BERT tarea 6

Modelo	**Descripción**	***F1-score***	***Precision***	***Recall***
Mediana	Mediana de todas las participaciones	0.93	0.93	0.93
BERT	*Finetuning*	0.94	0.93	0.93
CT-BERT (1er lugar)	*Finetuning*	**0.95**	**0.94**	**0.94**

Para esta tarea los resultados tienen poca variación entre ellos, probablemente se deba a que no se trate de un *dataset* muy complejo. Los detalles de la participación en la competencia de nuestro equipo fueron publicados, y se encuentran en el apéndice 1.

5.4 DISCUSIÓN

Ambos modelos BERT y CT-BERT comparten exactamente la misma arquitectura. Su única diferencia radica en los datos de pre-entrenamiento. Para el modelo BERT fue una cantidad de 4.3 billones de palabras de una temática general, obtenidos de Wikipedia. Para CT-BERT se utilizaron una cantidad de 0.7 billones de palabras provenientes de *tweets* sobre Covid-19. Aunque se considera que en modelo de *Deep learning* su desempeño es proporcional a la cantidad de datos de entrenamiento, en este caso fue lo contrario. La temática de los datos de pre-entrenamiento fue más importante que la cantidad de ellos. Para

la tarea 5 hubo una diferencia de 0.09 en el *f1-score* a favor del modelo CT-BERT. Para la tarea 6 la diferencia fue de 0.01.

5.4.1 Atención de los Modelos

Para tratar de comprender algunas diferencias entre ambos modelos, se hicieron mapas de calor que muestran la atención promediada de todas las cabezas y todos los *encoders* del modelo para un *tweet*. Cada uno de estos mapas de calor contiene la atención que cada una de las palabras le puso al resto para generar su representación contextualizada.

En las Figuras 5.2 y 5.3 se muestra un *tweet* de caso potencial donde el usuario sufre de un síntoma (*coughing*). El modelo BERT lo catalogó como "otro", mientras que CT-BERT lo catalogó como "caso potencial". BERT parece no prestar atención a ninguna palabra en específico, ya que la concentra en el símbolo "." al final de la oración. Tal vez por esta razón no logró catalogarlo de forma correcta. En cambio, el mapa de calor de CT-BERT muestra que gran parte de la atención se centra en el síntoma "*coughing*". Esto implica que la representación final del *tweet* contiene bastante información acerca del síntoma y logra catalogarlo de forma correcta.

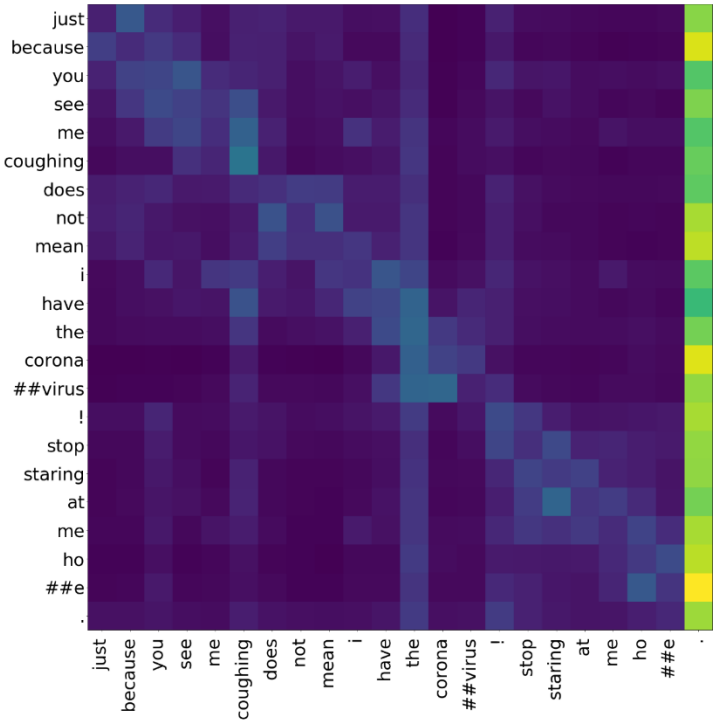

Figura 5.2: Atención promediada BERT

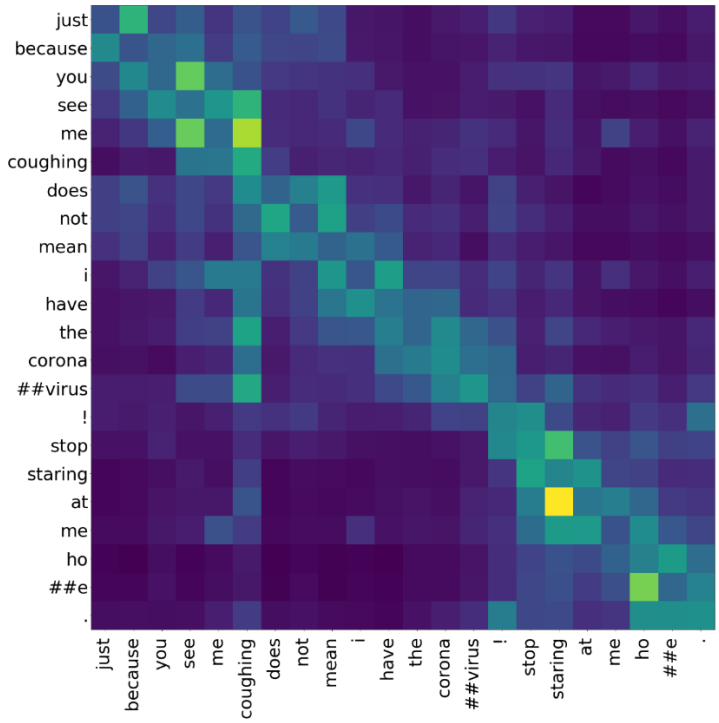

Figura 5.3: Atención promediada CT-BERT

6. MÉTODO BASADO EN ARQUITECTURA *WIDE & DEEP*

Un modelo con arquitectura *Wide & Deep* combina dos tipos de características para realizar la predicción. En la aplicación original de Cheng et al. (2016), que es un sistema de recomendaciones para *Google Store*, la parte ancha es alimentada con características dispersas. La parte profunda es una red neuronal con varias capas escondidas y alimentado con características densas. La unión de estas dos ramas obtiene mejores resultados que cada rama de forma individual en el sistema de recomendaciones.

En esta sección se explica a detalle el modelo de clasificación propuesto que implementa estos conceptos. Por último, se mostrarán los resultados obtenidos del modelo experimental y la discusión de ellos.

6.1 DISEÑO Y CONSTRUCCIÓN DEL MODELO

Aplicando la arquitectura *Wide & Deep* en una aplicación de clasificación de texto, se propone el modelo experimental de esta investigación. Para la rama profunda se eligió utilizar el modelo CT-BERT, ya que demostró haber conseguido resultados competitivos con el estado del arte aún y cuando se emplea solo por sí mismo. Esta rama utiliza CT-BERT para obtener un *embedding* contextualizado con información semántica de todo el *tweet*. La parte ancha es una combinación de tres tipos de características que son TF-IDF, EXPEI (valor personal de cada palabra) (Ortega et al., 2019) y una distribución de tópicos obtenidos por el modelo no supervisado NMF. Tanto las características profundas como las anchas son concatenadas para formar un nuevo vector de características. Este nuevo vector es alimentado a una red neuronal densa de clasificación, con los siguientes parámetros:

- *Dropout* de 0.1.
- 1024 + n unidades de entrada, donde n es el número de características de la rama ancha.
- 2 unidades de salida para la tarea 5.

- 3 unidades de salida para la tarea 6.

En la Figura 6.1 se muestra la arquitectura del modelo experimental. Varios modelos fueron entrenados utilizando las características anchas por individual y las diferentes combinaciones de ellas.

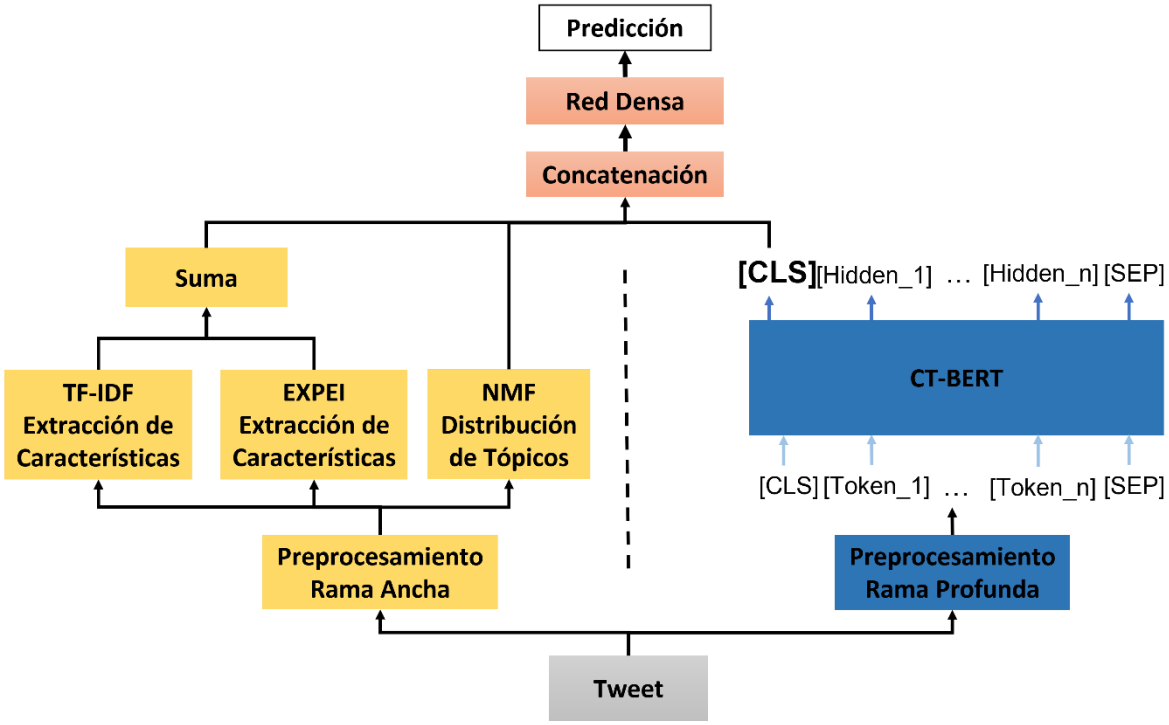

Figura 6.1: Arquitectura *wide & deep* experimental

6.1.1 Extracción de Características Tipo *Deep*

El modelo CT-BERT es alimentado con los *tokens* de los *tweets* preprocesados siguiendo los mismos pasos explicados en la sección 5.1.1. La salida del token especial de inicio [CLS] es utilizada como las características profundas, considerándose como el *embedding* contextualizado de la oración completa. Este *embedding* es de 1024 dimensiones y contiene información semántica de todo el *tweet*.

6.1.2 Extracción de Características Tipo *Wide*

Se llevaron a cabo tres métodos de características a partir del set de entrenamiento para la rama ancha. Para estos tres métodos se realizó un mismo preprocesamiento de texto que se describe a continuación:

- Se realiza una separación por palabras (tokenización) con la función TweetTokenizer de la librería NLTK.
- Se cambia todo el texto a minúsculas.
- Se estandarizan los símbolos HTML. Ejemplo &amp -> &.
- Se estandarizan los signos de puntuación y acentuación, cambiando o eliminando aquellos que no pertenezcan al idioma inglés.
- Se cambian emojis a sus palabras descriptivas con la librería emoji.
- Se reemplazan links de páginas web con la palabra url.
- Se reemplazan nombres de @usuario con la palabra user
- Se lematizan las palabras con la función WordNetLemmatizer de la librería NLTK.

En las siguientes subsecciones se explica el procedimiento para obtener cada una de las características, así como la motivación detrás de cada una ellas.

Carácterísticas TF-IDF

El método TF-IDF es muy común en aplicaciones de procesamiento de lenguaje natural, ya que ofrece una representación simple y básica de las palabras de un texto. Se eligió este método por qué representa una medida estadística de la colección de *tweets* que se calcula de forma sencilla. Este es un acercamiento que sigue siendo relevante y ofrece buenos resultados en aplicaciones de baja complejidad.

Este método asigna un valor a cada palabra según la frecuencia que aparece en el conjunto de datos. Esto trata de reflejar que tan importante es cada palabra en el conjunto. Para calcular este valor se utilizó la librería de Scikit Learn en Python, con la función TfidfVectorizer con los siguientes parámetros:

- *Analizer = 'word'* – Calcula el valor de frecuencia por palabra.
- *Use_idf= True* – Permite que las características sean normalizadas con la inversa de la frecuencia de documentos idf.
- *Smooth_idf= True* – Añade un 1 a las frecuencias de cada palabra al calcular el idf, para prevenir divisiones entre 0.

El modelo resultante contiene un diccionario que relaciona las palabras con el valor calculado. Ejemplo.

"sars-cov": 8.269, "ours": 7.669, "remembered": 2.154 ...

Estos valores son escalados utilizando la función StandardScaler de la librería Scikit Learn.

Después, se eligieron dos métodos para crear una representación de cada *tweet* a partir de este diccionario palabra – valor. Los métodos elegidos son los siguientes:

1. **Método Posicional.**

Este método simplemente intercambia las palabras de la oración por su valor escalar, ejemplo:

["we're", 'parking', 'at', 'the', 'airport', 'and', 'my', 'mom', 'rolled', 'down', 'the', 'window', 'to', 'speak', 'to', 'an', 'attendant', 'and', 'my', 'dad', 'immediately', 'said', '"', 'we', 'have', 'the', 'coronavirus', 'sir', '"']

[4.829, 7.481, 2.895, 1.426, 4.182, 1.786, 2.210, 5.535, 8.493, 4.536, 1.426, 7.576, 1.584, 7.240, 1.584, 3.877, 6.988, 1.786, 2.210, 5.836, 6.413, 4.357, 3.754, 2.518, 2.425, 1.426, 1.030, 6.788, 3.754 ... 0, 0, 0]

Si una palabra no se encuentra en el diccionario, se le asignará un 0. Se eligió este método ya que es la forma directa de utilizar el diccionario creado anteriormente para representar el *tweet*. Se llamará a este método posicional ya que el valor de cada dimensión depende de la palabra que esté en esa posición del *tweet*. Con esta representación se pretende que el modelo aprenda que tan importante es que existan ciertas palabras en ciertas posiciones, capturando así, el estilo de escritura.

2. **Método BOW.**

Un acercamiento por bolsa de palabras (BOW) suele utilizar un vector del tamaño del vocabulario entero del *dataset*. El tamaño del vocabulario de los conjuntos de datos fue de alrededor de 13,000 palabras únicas. Para evitar que el tamaño de este vector sea tan grande como el vocabulario, los experimentos se realizaron con una bolsa con las 5000 palabras más frecuentes del vocabulario. Se llegó a este tamaño después de entrenar varios modelos con una bolsa de tamaño entre 300 y 10,000, donde a partir de 5000 no se observa ninguna mejora en los resultados.

De esta forma el vector que representa a la oración será de tamaño 5000, donde cada una de las dimensiones está ordenada, y corresponde a una palabra en específico. Si el *tweet* contiene alguna de estas palabras más frecuentes en el diccionario, se coloca el peso de esa palabra en la dimensión correspondiente del vector. Si hay más de una ocurrencia de la misma

palabra, el peso de esa palabra se multiplica por el número de veces que aparece en la oración. La dimensión correspondiente a una palabra que no esté en la oración, tendrá un valor de 0. Este procedimiento se ejemplifica de la siguiente manera.

"I think i have the coronavirus #corona #worried"

[4.829, 7.481, 4.829, 1.426, 4.182, 2.321, 3.213, 1.547]

Tabla 6.1: Ejemplo Representación TF-IDF BOW.

	Palabras con más peso en diccionario TF-IDF								
	Word1	***I***	***Word3***	***have***	***coronavirus***	***Word6***	**…**	**…**	***Word N***
Tweet n	0	4.829 * 2	0	1.426	4.182	0	0	0	0

Se eligió este método porque es una representación tradicional de texto en NLP, y contiene información léxica discriminatoria. Sigue siendo utilizado ampliamente sobre todo para aplicaciones sencillas.

Características EXPEI

Estas características se refieren a que tan personal es cada palabra. Se asume que cuando una persona escribe utilizando pronombres personales, se refiere a si mismo o a alguien cercano. En el caso de la detección de casos potenciales, cuando se menciona un síntoma es importante saber quién lo sufre. Por esta razón se eligió utilizar la medida EXPEI, introducida por Ortega et al. (2018) y ampliada en López et al. (2022). Esta medida representa que tan personal es cada palabra presente en el conjunto de datos. Para esto se cuentan los *tweets* personales. Un *tweet* se considera personal cuando tiene un pronombre personal. Los

pronombres personales son *"i, i'd, i'll, i'v, i've, i'm, we, me, us, my, mine, our, ours, myself, ourselves"*. Con esto se mide que tan asociada está cada palabra con los *tweets* personales. Mediante algunas ecuaciones similares al cálculo de *precision*, *recall* y *F1-score* se calcula que tan asociada está cada término i con los *tweets* personales P.

$$P_{(t_i)} = \frac{\#(t_i, P)}{\#(t_i, S)} \tag{4.5}$$

$$\tau_{(t_i)} = \frac{\#(t_i, P)}{\#(P)} \tag{4.6}$$

En la ecuación 4.5 se calcula la relación entre el número de *tweets* personales en los cuales aparece la palabra i y el número de *tweets* en total S (personales y no personales) en los cuales aparece la misma palabra.

La ecuación 4.6 muestra la relación entre el número de *tweets* personales en los cuales aparece la palabra i y el número total de *tweets* personales.

Para combinar ambas ecuaciones y calcular una medida más estable de información personal, se calcula una métrica tipo *F1-score* con la ecuación 4.7.

$$F_{(t_i)} = 2\frac{P_{(t_i)} . \tau_{(t_i)}}{P_{(t_i)} + \tau_{(t_i)}} \tag{4.7}$$

Con esta medida tipo *F-score* se calcula un peso que mide que tan relacionada está el término con los *tweets* personales, y se muestra en la ecuación 4.8. El término del numerador es el número de veces que aparece cada palabra en todo el *dataset*, y el denominador es el número de palabras en total. Esta medida puede utilizarse por sí misma, o puede utilizarse para complementar los pesos TF-IDF, simplemente sumando los pesos de ambos como se muestra en la ecuación 4.9. A la suma de ambas se le llamará Term Specificty - Exponential Personal Rewards (TS-XPR).

$$EXPEI_{(t_i)} = \sqrt{\frac{\#(t_i)}{\#t}}^{1-F_{(t_i)}} \tag{4.8}$$

$$TS - XPR_{(t_i)} = TF - IDF_{(t_i)} + EXPEI_{(t_i)} \tag{4.9}$$

Con los pesos EXPEI o TS-XPR de cada palabra se representan los *tweets* de dos formas distintas, de igual forma y con los mismos procedimientos que con las características TF-IDF.

1. **Método Posicional –** Se reemplazan las palabras de cada *tweet* por su valor en el diccionario. Cuando se combinan con las características TF-IDF, se suman los pesos de ambos diccionarios.
2. **Método BOW.** – A diferencia de las características TF-IDF, se eligen las n palabras con mayor valor EXPEI en el diccionario para crear la bolsa. Para la combinación de estas características con el tipo TF-IDF, se utilizan las n palabras más importantes EXPEI para construir la bolsa de palabras. Para representar el *tweet* se le suma el valor TF-IDF al valor EXPEI de cada palabra presente en el texto. Este valor se multiplica según el número de ocurrencias de cada palabra en el *tweet.* Ejemplo.

"I think i have the coronavirus #corona #worried"

[4.829, 7.481, 4.829, 1.426, 4.182, 2.321, 3.213, 1.547]

Tabla 6.2: Ejemplo Representación TS-XPR BOW.

	Palabras con más peso en diccionario EXPEI								
	Word 1	***I***	**Word 3**	***have***	***coronavirus***	**Word 6**	**…**	**…**	**Word N**
Tweet n	0	4.829 * 2 + TF_IDF * 2	0	1.426 + TF_IDF	4.182 + TF_IDF	0	0	0	0

Características Tópicos

Para estas características se entrenó un modelo NMF no supervisado sobre los datos de entrenamiento de cada *dataset*. Este modelo aplicado en texto es ampliamente utilizado para la extracción de tópicos. Esto agrupa los *tweets* en categorías según qué tan similares sean entre sí. Con la idea de que los *tweets* de cada clase sean similares entre sí y tengan una distribución de tópicos similar, se propuso este método. Con ello, se obtuvo la probabilidad de que cada *tweet* perteneciera a cada tópico. Tratándose de una distribución de probabilidad que actúa como vector de características de cada *tweet*.

Se entrenaron varios modelos con diferentes números de tópicos para evaluar cuál de ellos da mejores resultados. Los números de componentes seleccionados fueron 200. Se experimentó con un número de componentes entre 50 a 300, y fue 200 tópicos los que obtuvieron mejores resultados. Los parámetros de entrenamiento para el modelo NMF fueron

random_state=1, *alpha*=0.1, *l1_ratio*=0.5 utilizando la función NMF de la librería de *Scikit learn*.

El modelo NMF recibe como entrada los vectores TF-IDF de cada *tweet*. Obtenidos de la misma forma y con los mismos parámetros que los mencionados en las características TF-IDF.

Con el modelo ya entrenado, se predijo la distribución de probabilidad entre tópico-*tweet*. Obteniendo un vector de características de longitud 200, donde 200 es el número de componentes del modelo.

6.1.3 Entrenamiento del Modelo

Se entrenó el modelo *Wide & Deep* con las combinaciones de características anchas para las tareas 5 y 6. Como se mencionó anteriormente, cuando se combinan las características TF-IDF y EXPEI, es la suma de ambas y se le llama TS-XPR. Cuando se utiliza la distribución de tópicos, es la concatenación con las demás. Las combinaciones son las siguientes:

Tabla 6.3: Combinaciones de características anchas

Combinaciones		
Método Posicional	**Método BOW**	
TF-IDF	TF-IDF	Tópicos
EXPEI	EXPEI	
TS-XPR	TS-XPR	
EXPEI, Tópicos	EXPEI, Tópicos	
TS-XPR, Tópicos	TS-XPR, Tópicos	

Se utilizaron los mismos hiperparámetros en el entrenamiento que los encontrados por medio de la búsqueda tipo *random search* de la implementación individual de los modelos. Se uso un optimizador *AdamW* con un *learning rate scheduler* linear. Por último, se empleó una función de pérdida *Cross Entropy*. Se entrenaron los modelos durante 3

épocas, ya que fue el número de épocas idóneo encontrado por medio del método *random search*.

De igual forma, para tener un punto de comparación, se entrenó un modelo simple de regresión logística utilizando únicamente las características anchas.

6.2 MODELOS DE COMPARACIÓN *BASELINE*

Para tener un punto de comparación entre las ramas *Wide* y *Deep* por individual, se entrenaron modelos simples de Regresión Logística con las características anchas.

Estos modelos fueron entrenados con las combinaciones mostradas en la Tabla 6.3, utilizando la librería Scikit-learn. Los parámetros utilizados en estos modelos fueron los siguientes: un número máximo de iteraciones de 5000, con un peso de clases de 1 a 4 para las clases “otros” y “caso potencial” respectivamente para la tarea 5. Para la tarea 6 se utilizó un peso de clases con la opción “balanced”.

6.3 RESULTADOS MODELOS *BASELINE*

En las siguientes Tablas se muestran los resultados que obtuvieron los modelos *baseline* de regresión logística para las tareas 5 y 6 del concurso #SMM4H 2021.

Tabla 6.4: Resultados baseline tarea 5

Modelo	**Tipo / Características**	**F1**	**P**	**R**
Regresión Logística	200 Topicos	0.42	0.49	0.36
-	TF-IDF (Posicional)	0.22	0.22	0.23
-	EXPEI (Posicional)	0.20	0.19	0.20
-	TS-XPR (Posicional)	0.21	0.21	0.21
-	TF-IDF (Posicional), 200 Tópicos	0.36	0.52	0.28
-	EXPEI (Posicional), 200 Tópicos	0.38	0.37	0.40
	TS-XPR (Posicional), 200 Tópicos	0.38	0.37	0.40

-	TF-IDF (BOW-5000)	0.52	0.47	0.58
-	EXPEI (BOW-5000)	0.48	0.42	0.57
-	**TS-XPR (BOW-5000)**	**0.52**	**0.47**	**0.58**
-	TF-IDF (BOW-5000), 200 Tópicos	0.49	0.50	0.48
-	EXPEI (BOW-5000), 200 Tópicos	0.48	0.42	0.56
-	TS-XPR (BOW-5000), 200 Tópicos	0.52	0.47	0.58

Tabla 6.5: Resultados baseline tarea 6

Modelo	**Tipo / Características**	**F1**	**P**	**R**
Regresión Logística	200 Tópicos	0.92	0.92	0.92
-	TF-IDF (Posicional)	0.55	0.55	0.55
-	EXPEI (Posicional)	0.55	0.55	0.55
-	TS-XPR (Posicional)	0.55	0.55	0.55
-	TF-IDF (Posicional), 200 Tópicos	0.78	0.78	0.78
-	EXPEI (Posicional), 200 Tópicos	0.78	0.78	0.78
-	TS-XPR (Posicional), 200 Tópicos	0.81	0.81	0.81
-	TF-IDF (BOW-5000)	0.55	0.55	0.55
-	EXPEI (BOW-5000)	0.61	0.61	0.61
-	**TS-XPR (BOW-5000)**	**0.93**	**0.93**	**0.93**
-	TF-IDF (BOW-5000), 200 Tópicos	0.92	0.92	0.92
-	EXPEI (BOW-5000), 200 Tópicos	0.92	0.92	0.92
-	TS-XPR (BOW-5000), 200 Tópicos	0.93	0.93	0.93

Para ambas tareas, los mejores resultados de los modelos *baseline* es utilizando las características TS-XPR con el método BOW (resaltados en las Tablas anteriores). Al agregar las características de tópicos a estos modelos no hay ninguna variación en los resultados. Se utilizarán los modelos resaltados como punto de comparación más adelante.

6.4 RESULTADOS MODELOS *WIDE & DEEP*

A continuación, se muestran los resultados obtenidos por las diferentes combinaciones del modelo experimental *Wide & Deep* en la Tabla 6.6.

Tabla 6.6: Resultados wide & deep tarea 5

Modelo	Tipo / Características	F	P	R
CT-BERT *Wide & Deep*	200 Tópicos	0.78	0.79	0.77
-	TF-IDF (Posicional)	0.77	0.82	0.74
-	EXPEI (Posicional)	0.78	0.82	0.75
-	TS-XPR (Posicional)	0.79	0.77	0.81
-	TF-IDF (Posicional), 200 Tópicos	0.79	0.77	0.81
-	**EXPEI (Posicional), 200 Tópicos**	**0.80**	**0.79**	**0.81**
-	TS-XPR (Posicional), 200 Tópicos	0.79	0.78	0.79
-	TF-IDF (BOW-5000)	0.79	0.81	0.77
-	EXPEI (BOW-5000)	0.78	0.75	0.82
-	TS-XPR (BOW-5000)	0.78	0.77	0.79
-	TF-IDF (BOW-5000), 200 Tópicos	0.79	0.78	0.80
-	EXPEI (BOW-5000), 200 Tópicos	0.79	0.81	0.78
-	TS-XPR (**BOW-5000**), 200 Tópicos	0.78	0.79	0.78

Tabla 6.7: Comparación de resultados tarea 5

Modelo	Tipo / Características	F	P	R
Regresión Logística	TS-XPR (BOW-5000)	0.52	0.47	0.58
BERT	Deep	0.68	0.72	0.65
CT-BERT	Deep	0.77	0.77	0.77
1er Lugar #SMM4H	Ensamble de 5 modelos tipo BERT	0.79	0.78	0.79
CT-BERT *Wide & Deep*	**EXPEI (Posicional), 200 Tópicos**	**0.80**	**0.79**	**0.81**

La Tabla 6.7 muestra la comparación de resultados del *baseline* con los métodos entrenados, así como con el modelo que obtuvo el mejor resultado en la tarea 5. El sistema con mejores resultados entrenó un total de 14 modelos basados en BERT durante una

búsqueda de hiperparámetros. A continuación, realizaron una búsqueda exhaustiva de todas las combinaciones posibles para encontrar el conjunto con la mayor puntuación de validación utilizando los 14 modelos entrenados anteriormente. El mejor conjunto estaba compuesto por 5 modelos. Si bien este enfoque logró la mejor puntuación, aumentó en gran medida el número de parámetros entrenables y la potencia computacional necesaria para realizar una predicción (Aji et al., 2021). En cambio, el modelo *Wide & Deep* tiene aproximadamente una quinta parte de parámetros entrenables, lo que lo hace mucho menos costoso y requiere una sola sesión de entrenamiento. Esto significa que es un método mucho más eficiente que consigue los mismos resultados. El modelo experimental logró superar al modelo únicamente profundo BERT por un margen de +0.12 *F1-score* y al modelo CT-BERT por +0.03. En cuanto al ensamble, logra superarlo por 0.01 de *F1-score*.

El modelo *Wide & Deep* que logró superar los resultados utilizó las características EXPEI empleadas con el método posicional concatenadas a la distribución de 200 tópicos.

Tabla 6.8: Resultados wide & deep tarea 6

Modelo	**Tipo / Características**	**F**	**P**	**R**
CT-BERT *Wide & Deep*	200 Tópicos	0.94	0.94	0.94
-	TF-IDF (Posicional)	0.95	0.95	0.95
-	EXPEI (Posicional)	0.95	0.95	0.95
-	TS-XPR (Posicional)	0.94	0.94	0.94
-	TF-IDF (Posicional), 200 Tópicos	0.94	0.94	0.94
-	EXPEI (Posicional), 200 Tópicos	0.95	0.95	0.95
-	TS-XPR (Posicional), 200 Tópicos	0.94	0.94	0.94
-	TF-IDF (BOW-5000)	0.94	0.94	0.94
-	EXPEI (BOW-5000)	0.94	0.94	0.94
-	TS-XPR (BOW-5000)	0.94	0.94	0.94
-	TF-IDF (BOW-5000), 200 Tópicos	0.94	0.94	0.94
-	**EXPEI (BOW-5000), 200 Tópicos**	**0.95**	**0.95**	**0.95**
-	TS-XPR (BOW-5000), 200 Tópicos	0.94	0.94	0.94

Tabla 6.9: Comparación de resultados tarea 6

Modelo	Tipo / Características	F	P	R
Regresión Logística	TS-XPR (BOW-5000)	0.93	0.93	0.93
BERT	Deep	0.94	0.94	0.94
CT-BERT / 1er lugar #SMM4H	**Deep**	**0.95**	**0.95**	**0.95**
CT-BERT *Wide & Deep*	EXPEI (BOW-5000), 200 Tópicos	0.95	0.95	0.95

En la Tabla 6.8 se muestran los resultados de las combinaciones de características para el modelo *Wide & Deep*. En la Tabla 6.9 muestra una comparación del modelo *baseline* con los modelos entrenados. En este caso, el modelo CT-BERT únicamente profundo, entrenado durante esta investigación, obtuvo el primer lugar de la competencia de un total de 20 participantes.

En el caso de la tarea 6, no hubo mejoras entre el modelo únicamente profundo y el *wide & deep*. Incluso con los resultados del mejor modelo *baseline* de regresión logística, solo hay una diferencia de 0.02 en el *F1-score*. Esto quiere decir que se trata de un *dataset* fácil de aprender para los modelos, y no se ve beneficiado por la arquitectura *wide & deep*.

6.5 DISCUSIÓN DE RESULTADOS MODELOS *WIDE & DEEP*

Los resultados obtenidos para la tarea 6 de clasificación entre reportes personales, reportes no personales y menciones de noticias muestran que se trata de un *dataset* de baja complejidad. Los resultados de *f1-score* de los modelos tanto profundos como con la rama ancha son los mismos. Aún con un acercamiento sencillo como los modelos *baseline* de regresión logística muestran resultados competitivos a comparación de los modelos profundos.

En cambio, los resultados de la tarea 5 muestran que se trata de un problema de alta complejidad. Existiendo una gran diferencia entre los modelos *baseline* y los modelos propuestos. Los modelos *baseline* muestran resultados muy por debajo de los modelos profundos. Por esta razón se eligió la tarea 5 para realizar una serie de análisis cualitativos para tratar de explicar el desempeño de los modelos propuestos.

6.6 ATENCIÓN EN RAMA PROFUNDA

La rama profunda de los modelos contiene un bloque de modelos tipo BERT. Este modelo está basado en cabezas de atención para generar la representación de la oración entera. Para tratar de comprender el funcionamiento interno, se hicieron mapas de calor que muestran la atención promediada de todas las cabezas y todos los *encoders* del modelo para ejemplos particulares. Cada uno de estos mapas de calor contiene la atención que cada una de las palabras le puso al resto para generar su representación contextualizada.

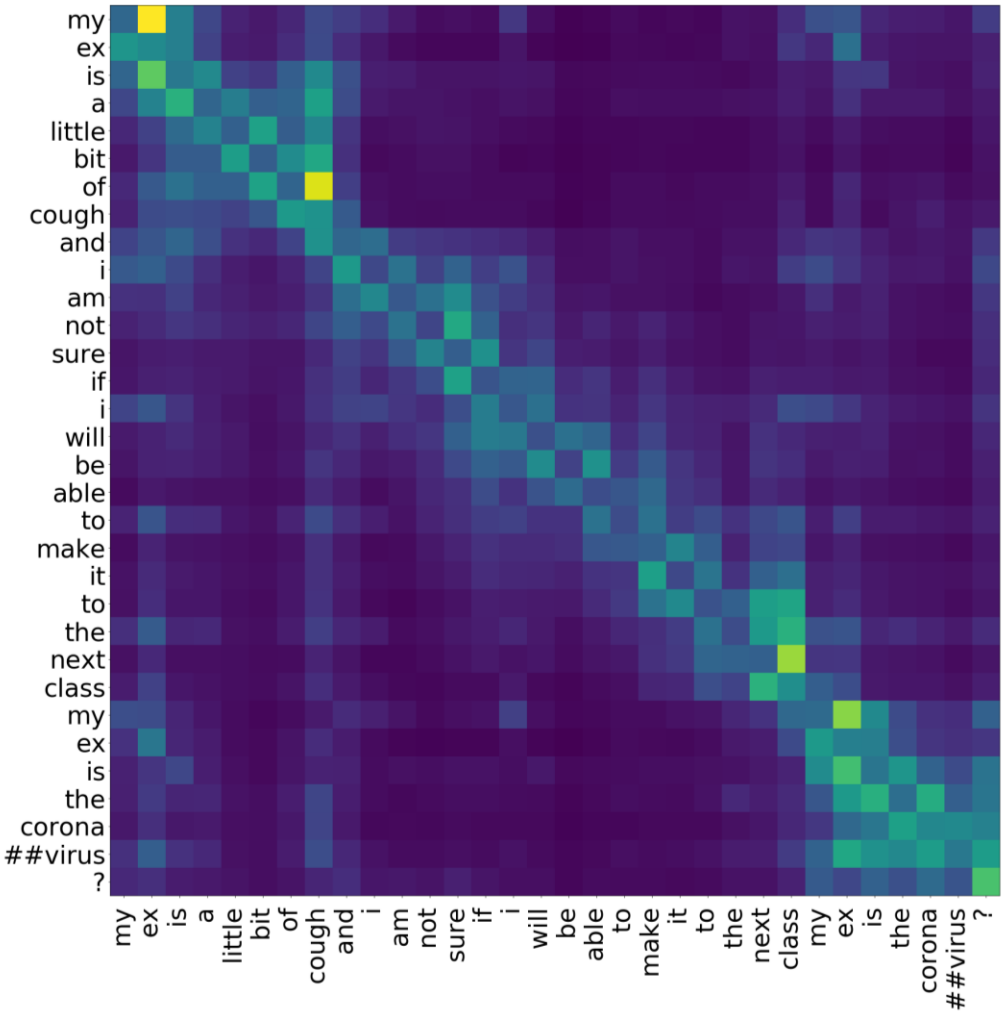

Figura 6.2: Atención promediada de *tweet* 1.

La Figura 6.2 muestra la atención en un *tweet* que no es caso potencial. Sin embargo, la rama profunda por sí sola lo catalogó como caso potencial, mientras que con la unión de la rama ancha el modelo logró identificarlo correctamente. Una posible razón por la cual el modelo profundo identificó al *tweet* como caso potencial puede ser por la atención que le asignó a la palabra "*cough"* al generar la representación del resto de las palabras. La palabra "*coronavirus*" también asigno una leve atención a "*cough*" para su representación, por lo que probablemente asoció estas dos palabras sin tomar en cuenta el sarcasmo del *tweet*.

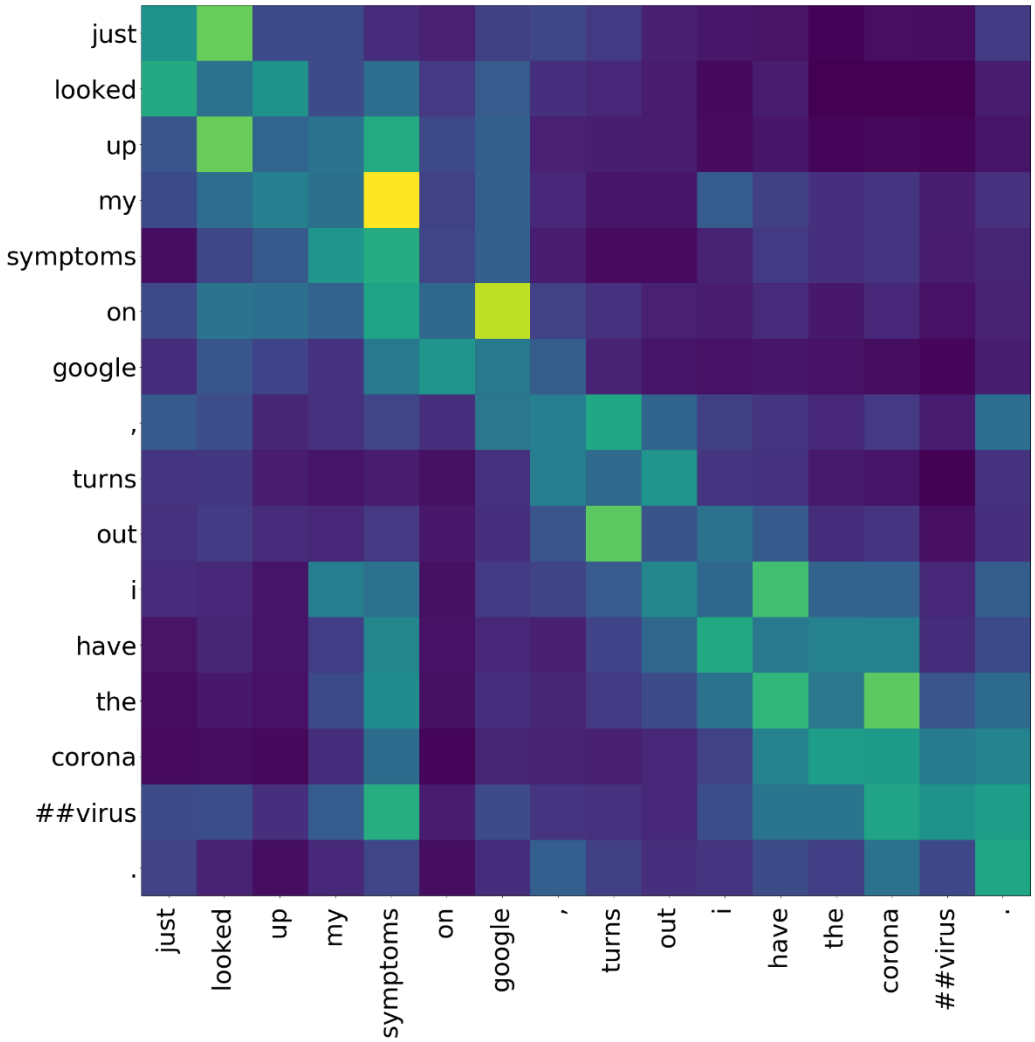

Figura 6.3: Atención promediada de *tweet* 2.

El *tweet* mostrado en la Figura 6.3 pertenece a la clase “caso potencial”. En esta ocasión la rama profunda lo catalogó correctamente, mientras que con la unión de la rama ancha lo clasificó de forma incorrecta. En este ejemplo se muestra que la mayoría de la atención se centra en la palabra “*symptoms”*. Nuevamente se puede observar una relación entre “*coronavirus”* y “*symptoms*”, así como con “*I*” y “*have*”. Esto parecería relevante ya que el autor expresa que tiene la enfermedad, ya que experimenta síntomas de ella.

En los ejemplos mostrados parece que la rama profunda asocia frecuentemente las palabras que se refieren a síntomas con la palabra coronavirus. Lo cual tiene sentido ya que

una es la causa de la otra. Aun así, parece que se le dificulta identificar ciertas formas de expresión, como el sarcasmo y la comedía. Y en algunos de estos casos, la unión de la rama ancha fue de ayuda para determinar la clasificación correcta.

6.7 ANÁLISIS DE CARACTERÍSTICAS ANCHAS

Según los resultados obtenidos, la unión de la rama profunda fue de utilidad para la clasificación correcta de los *tweets*. La tabla 6.10 muestra tres *tweets* de caso potencial de la tarea 5 que fueron clasificados por las ramas de forma individual. Los modelos elegidos para realizar la clasificación fueron los que obtuvieron mejores resultados en cada uno de sus respectivos grupos. En cada uno de ellos, la complementación de ambas ramas sirvió para la correcta clasificación. En un intento de comprender mejor como la rama ancha complementa al modelo profundo, a continuación, se muestran análisis cualitativos de las características anchas.

Tabla 6.10: Comparación de clasificaciones de ramas

Tweet	***Deep***	**Wide**	**W & D**
I got a runny nose today and started crying because I'm nervous about getting the coronavirus	"0"	"1"	"1"
I have all the symptoms of the Coronavirus so does that mean I have it???	"1"	"0"	"1"
We need more testing. Please. I may have coronavirus without showing symptoms. I live with my parents whom are at risk	"0"	"0"	"1"

6.7.1 Características TF-IDF

Según los resultados de la tarea 5, las características TF-IDF, empleadas por el método posicional, parecen no aportar mucho por sí mismas, sino solamente cuando se combinan con las otras características.

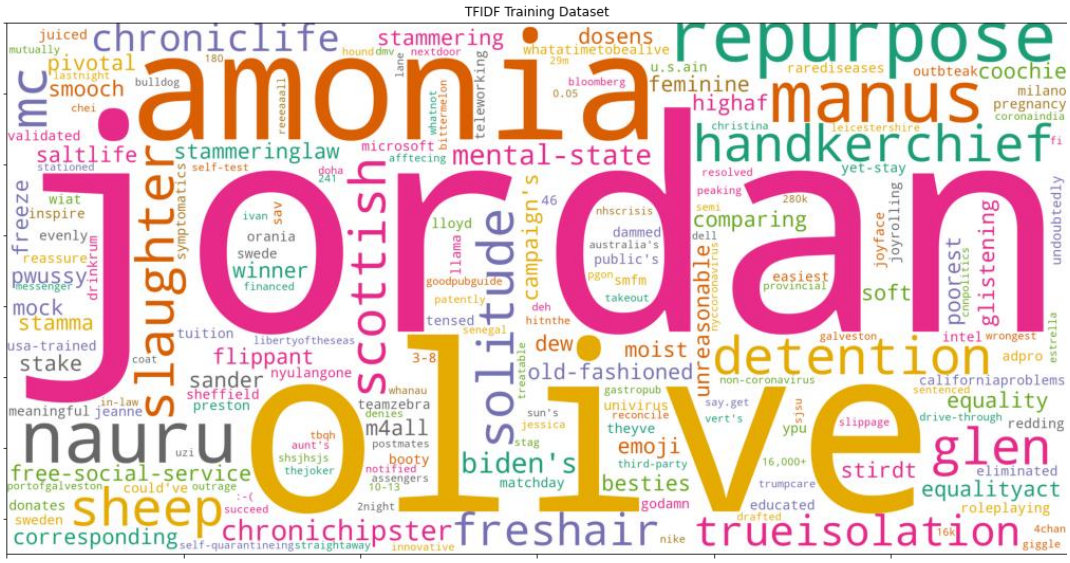

Figura 6.4: 50 palabras con mayor peso TF-IDF

En la Figura 6.4 se muestra una nube de palabras en donde su tamaño es proporcional al valor del diccionario TF-IDF. Como se puede observar, las palabras más recompensadas en su peso parecen no tener relación en un contexto de caso potencial de coronavirus. Por lo que probablemente esto sea la causa de su bajo desempeño cuando se utilizan de forma individual por el método posicional. En cambio, cuando se utiliza el método BOW, el puntaje aumenta. Esto puede ser debido a que solo se utilizan las 5000 palabras más frecuentes para construir la bolsa. Por lo tanto, no se consideran las palabras de rara ocurrencia que pudieran tener un peso elevado.

6.7.2 Características EXPEI

Las características EXPEI que denotan que tan personal es una palabra están presentes en la mayoría de los modelos *Wide & Deep* que igualaron al ensamble de 1er lugar, sea en su forma individual o combinada (TS-XPR). También forman parte del modelo que superó al ensamble. Por lo que se asume que deben contener información útil que complementa a la rama profunda.

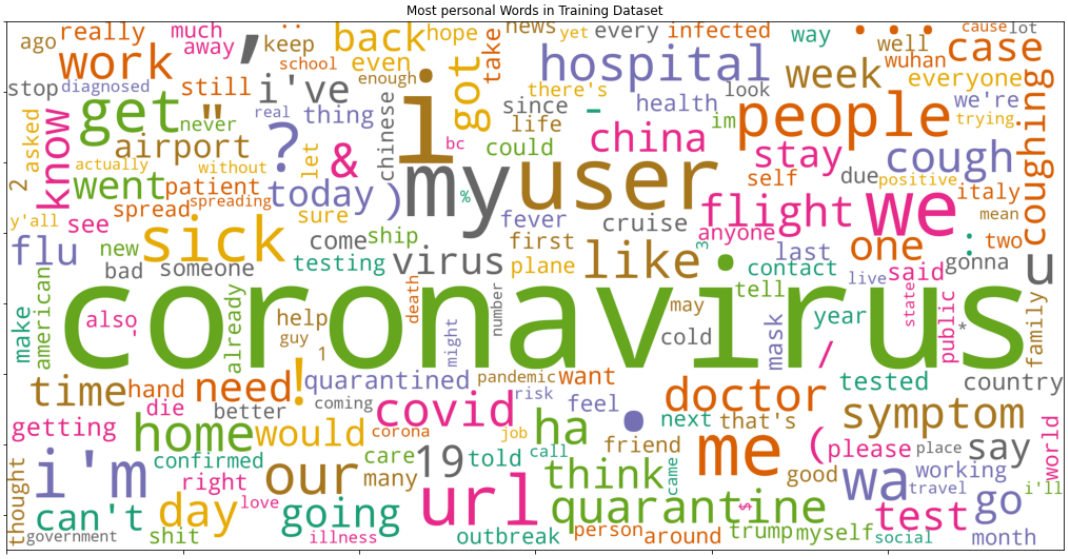

Figura 6.5: 50 palabras con mayor valor EXPEI

La Figura 6.5 muestra la nube de palabras donde el tamaño es proporcional al peso de la palabra del diccionario EXPEI. Las palabras que se pueden observar parece que están ampliamente relacionadas al contexto de caso potencial de coronavirus. Incluyendo palabras como *"I", "sick", "covid", "I've", "cough", "got", "virus", "symptom", "testing", "positive",* etc. Estas palabras son comúnmente utilizadas cuando alguien afirma estar enfermo o sufrir un síntoma. Tomando lo anterior en cuenta, parece que esta es la razón por la cual estas características aportan información complementaria al modelo *Wide & Deep*. Según los autores originales de esta arquitectura, la rama ancha ayuda en la memorización de ciertas características. Por lo que podría pensarse que cuando cierta combinación de estas palabras está presente, es más probable que el *tweet* pertenezca a la clase "caso potencial".

6.7.3 Características Tópicos

Estas características también están presentes en gran parte de los modelos de buen desempeño, incluyendo el mejor. Se tratan de una distribución de que tan probable un *tweet* pertenece a cada uno de los tópicos. Para tratar de analizarlos, se realizó un análisis de

información mutua con la distribución de tópicos. Este análisis muestra la relación que mantiene cada tópico con la variable objetivo. A continuación, se muestra en la Figura 6.6 el valor de información de cada tópico.

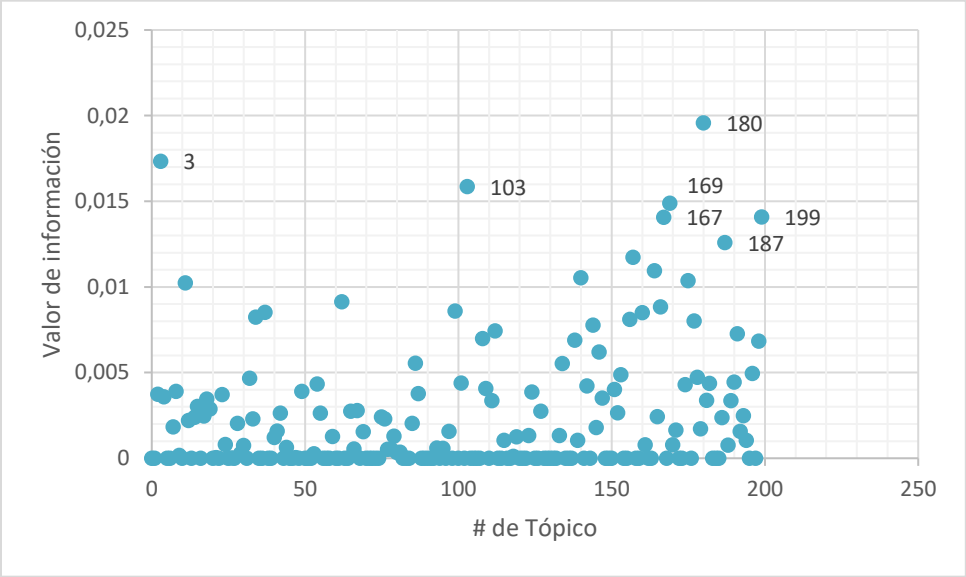

Figura 6.6: Valor de información de cada tópico

Tabla 6.10: Número de Tweets Caso Potencial por Tópico

Tópico	***# de Tweets***								
157	67	**142**	21	**150**	15	**165**	11	**104**	7
180	57	**169**	21	**2**	14	**177**	10	**166**	7
3	52	**126**	20	**87**	14	**194**	10	**187**	7
164	42	**145**	20	**185**	14	**84**	9	**186**	5
192	41	**167**	20	**5**	13	**163**	9	**17**	4
174	32	**121**	18	**19**	13	**172**	9	**175**	3
103	31	**171**	17	**190**	13	**189**	9	**178**	2

1	30	**173**	17	**191**	13	**198**	9	**199**	2
183	27	**196**	17	**108**	12	**152**	8	-	0
4	25	**179**	16	**168**	12	**155**	8	-	0
181	24	**188**	16	**176**	12	**182**	8	-	0
170	22	**195**	16	**197**	12	**184**	8	-	0
9	21	**22**	15	**161**	11	**193**	8	-	0

Como se puede observar en la Figura 6.6, los tópicos con mayor valor de información también se encuentran entre los tópicos que contienen *tweets* de caso potencial, mostrados en la Tabla 6.10.

A continuación, se muestran las palabras de mayor relevancia de cuatro de los tópicos con mayor valor de información. Mientras el *tweet* contenga más palabras pertenecientes a cada tópico, mayor probabilidad tendrá de pertenecer a este.

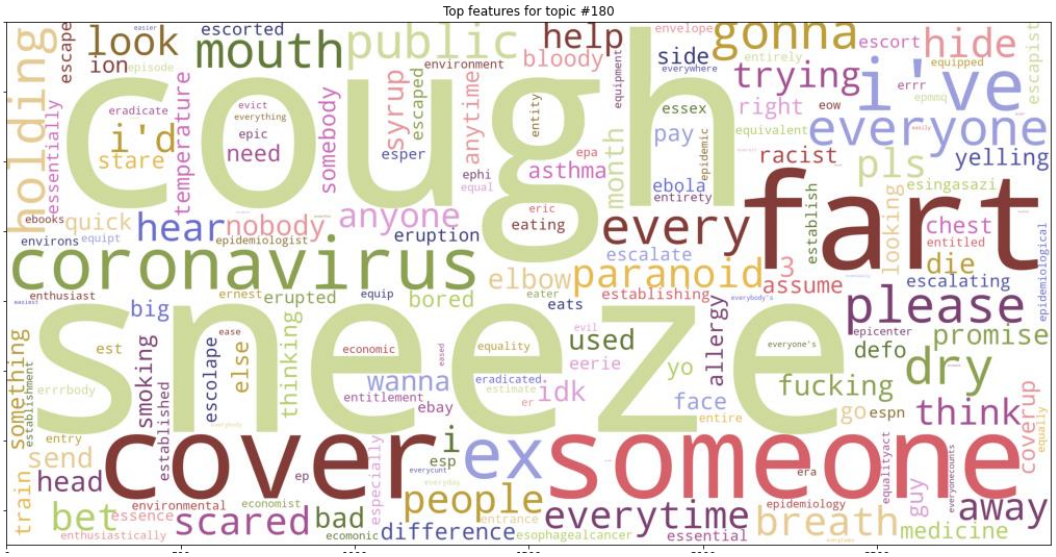

Figura 6.7: Palabras del tópico 180 con 57 casos potenciales

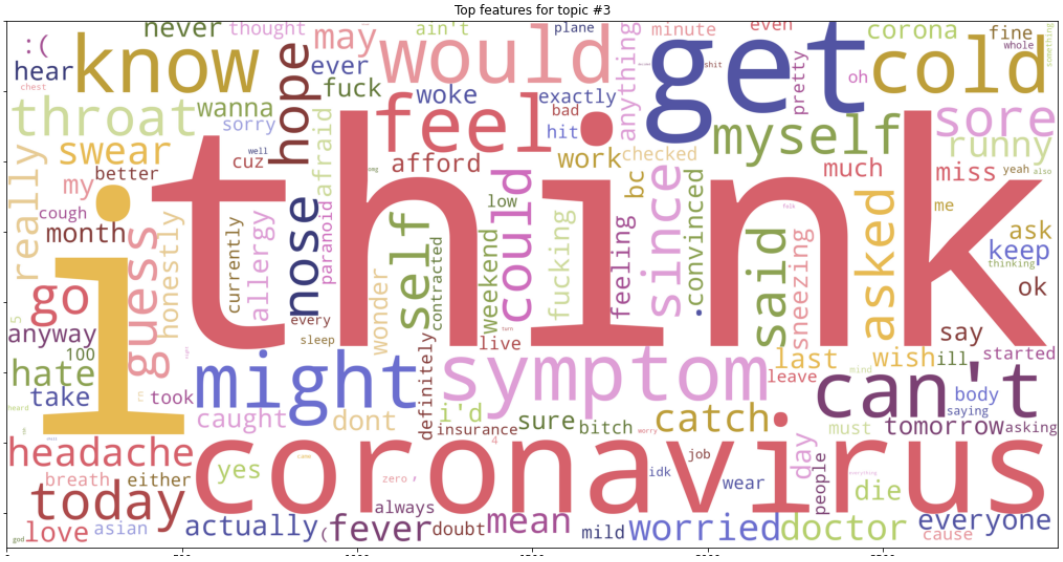

Figura 6.8: Palabras del tópico 3 con 52 casos potenciales

Figura 6.9: Palabras del tópico 103 con 31 casos potenciales

Figura 6.10: Palabras del tópico 199 con 2 casos potenciales

Como se puede observar en las Figuras 6.7, 6.8 y 6.9 los tópicos que contienen más *tweets* de casos potenciales también contienen palabras clave muy relacionadas al contexto de la enfermedad. Estos tópicos hacen énfasis en palabras como pronombres y síntomas, que se utilizarían comúnmente para describir un caso potencial. En cambio, los tópicos de la Figura 6.10, contiene palabras relacionadas también al coronavirus, pero en un contexto diferente. Se encuentran palabras que se refieren a la pandemia en general y medidas preventivas, así como palabras no conjugadas que frecuentemente se utilizan en noticieros.

Según el valor de información mutua (Figura 6.6), los tópicos que más contienen *tweets* de caso potencial, también están entre los que más información aportan para determinar su clase. De igual forma, se encuentran algunos tópicos con alta información que contienen pocos o ningún *tweet* de caso potencial. Con esto se podría pensar que cuando un *tweet* tiene mayor probabilidad de pertenecer a los tópicos con más casos potenciales, también tiene mayor probabilidad de ser un caso potencial y viceversa.

6.7.4 Análisis Método Posicional

El mejor modelo que alcanzó un *F1-score* de 0.80 utilizó las características TS-XPR codificadas por el método posicional. Para tratar de entender el funcionamiento de este método se aplicó un estudio de información mutua entre estas características y las clases. Este estudio muestra que tan relacionada esta cada dimensión del vector de características con la clase a la que pertenece ese vector. En el método posicional el vector de características está truncado a 120 dimensiones. Esto significa que cada dimensión contiene el valor TS-XPR de la palabra que está ubicada en esa posición. Y el estudio nos arroja que tan importante es que haya palabras personales en cada una de las 120 posiciones.

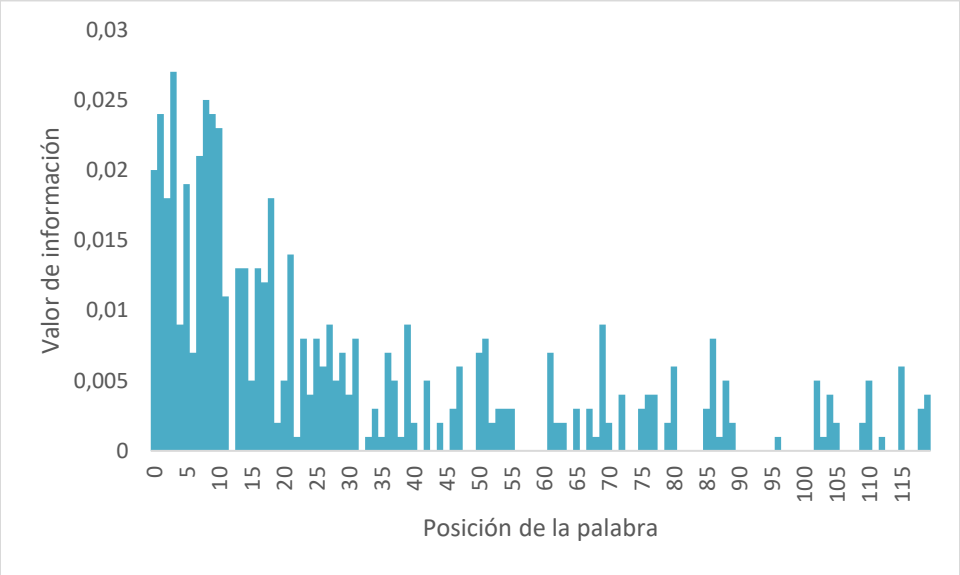

Figura 6.11: Valor de información de cada posición

La Figura 6.11 muestra el valor de importancia de cada posición del *tweet*. Mientras los valores se encuentren más cercanos a 0, menor es su importancia.

Los valores de información mutua muestran que es importante que existan palabras personales en las primeras 10 posiciones del *tweet*. Ya que contienen los valores más altos de información. Esto se puede interpretar como el estilo de escritura de las personas, donde

la mayor parte de *tweets* de casos potenciales comienzan utilizando palabras personales, como las mostradas en la Figura 6.5. La rama ancha aprende el estilo de escribir de las personas que sufren de la enfermedad o experimentan síntomas.

6.7.5 Análisis Método *BOW*

De igual forma se realizó un análisis de información mutua para tratar de comprender el método por bolsa de palabras de las características TF-IDF y EXPEI. Con este análisis se pudo obtener las palabras más importantes para definir a que clase pertenece el *tweet*.

Figura 6.12: Información mutua palabras TF-IDF

Para las características TF-IDF con una bolsa de palabras de tamaño 5000, las palabras más discriminantes se observan en la Figura 6.12. Según los modelos *baseline*, para que las características TF-IDF como bolsa de palabras sean informativas, la bolsa debe tener un tamaño grande. Parece que mientras más palabras sean utilizadas, mayor es la aportación de estas características. De igual forma, si se utilizan mayor cantidad de palabras, el modelo tendrá más parámetros entrenables y requerirá mayor poder de cómputo.

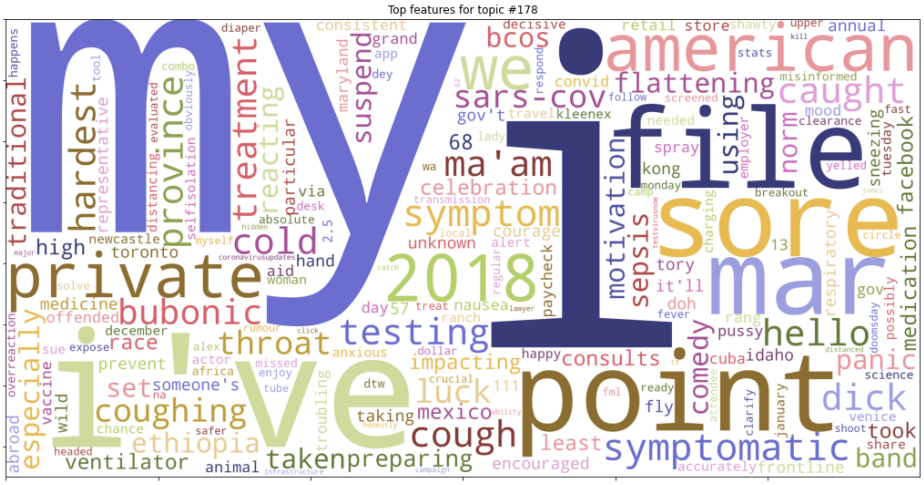

Figura 6.13: Información mutua palabras EXPEI

En la Figura 6.13 se muestran las palabras discriminantes EXPEI en su representación de bolsa de palabras. Las palabras mostradas podrían utilizarse para expresar si alguien sufre de síntomas. Con la unión de las características de tópicos, logró igualar al ensamble de 1er lugar.

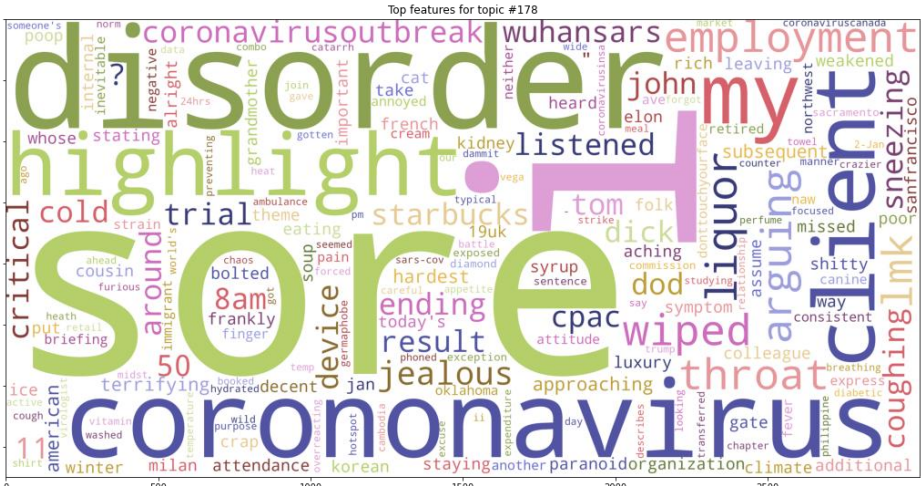

Figura 6.14: Información mutua palabras TS-XPR

La información mutua de la suma de ambas características se muestra en la Figura 6.14 Aunque la unión de ambas no igualó los resultados del ensamble, sí logró mejorar los resultados de la rama profunda. En la Figura 6.14 se observan algunas relacionadas con el

coronavirus y sus síntomas. También se encuentran otras que podrían considerarse como ruido que hacen referencia a lugares geográficos que a primera vista no tendrían relación con un caso potencial.

7. CONCLUSIÓNES Y TRABAJO FUTURO

Para esta investigación se utilizaron dos conjuntos de datos compuestos por *tweets* que hablaban sobre coronavirus. La tarea 5 buscó clasificar los *tweets* en dos categorías, “casos potenciales” y “otros”. La tarea 6 buscó clasificarlos en “reporte personal”, “reporte de terceros” y “noticias”. Primero se realizó una comparación entre dos modelos basados en *Transformers*, de arquitectura BERT, pero entrenados en dos conjuntos de datos de diferente temática. También se propuso un modelo experimental con arquitectura *Wide & Deep*, que buscó combinar las ventajas de un modelo profundo que tiene gran capacidad de generalización y un modelo ancho con capacidad de realizar predicciones más finas. Se entrenaron una serie de modelos que utilizaban diferentes características para la rama ancha. Entre ellas fueron características tipo TF-IDF, EXPEI (palabras personales) y una distribución de tópicos. Después se hizo un análisis de los resultados para tratar de comprender mejor el funcionamiento de la rama profunda y la rama ancha.

7.1 CONCLUSIONES

A continuación, se muestran las preguntas de investigación de la sección 1.3, junto con las respuestas obtenidas después de llevar a cabo este trabajo:

¿Un modelo pre-entrenado sobre datos de temática médica puede superar a otro pre-entrenado en muchos más datos pero de temática general?

Es bien sabido que en modelos de *deep learning* el desempeño está directamente relacionado a la cantidad de datos de entrenamiento. Mientras mayor cantidad de datos, mayor aprendizaje. Tras realizar la evaluación de dos modelos BERT con la misma arquitectura, pero diferentes datos de entrenamiento, se determinó que el modelo CT-BERT tuvo mejor desempeño. El modelo CT-BERT fue pre-entrenado en *tweets* sobre Covid-19, con una cantidad de palabras 7 veces menor que el modelo BERT original. Pero para la tarea

5, identificación de casos potenciales de Covid-19, superó con un margen de 0.09 en *f1-score* al modelo original. Esto quiere decir que la temática de los datos es tan importante para la aplicación como la cantidad de datos de entrenamiento.

Los resultados de la investigación de estos dos modelos fueron enviados a la competencia #SMM4H, de donde se obtuvieron los datos. El modelo CT-BERT entrenado en la tarea 6 de la competencia obtuvo el 1er lugar de un total de 20 participantes. Debido a esto se publicó un artículo que contiene la descripción de la participación (ver apéndice 1). También se dio una conferencia sobre el artículo en la *2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL 2021) (ver apéndice 2).

¿Podría un modelo clasificador híbrido, al estilo *wide and deep*, obtener mejores resultados que un modelo únicamente profundo?

Sí, en general, los modelos *Wide & Deep* lograron mejores resultados que un modelo únicamente profundo. En el caso de la tarea 5, en la competencia #SMM4H 2021, el modelo *Wide & Deep* superó al modelo ensamble que obtuvo el 1er lugar. El ensamble utilizó 5 modelos tipo BERT, incrementando por mucho la cantidad de parámetros entrenables y el poder de cómputo necesario para realizar una predicción. En cambio, el modelo *Wide & Deep* cuenta con aproximadamente una quinta parte de parámetros entrenables a comparación. Lo que significa que es una arquitectura mucho más eficiente y con mejores resultados.

¿Cómo es que la unión de los modelos *Wide & Deep* se complementan el uno al otro?

La rama profunda se enfoca en encontrar patrones que pueda generalizar para cada caso. Mientras que la rama ancha busca memorizar la interacción entre sus características. El mayor trabajo consiste en encontrar las características de la rama ancha que más información

discriminante contengan. Esto puede variar en cada aplicación. En el caso de la búsqueda de casos potenciales, la identificación de palabras personales y la distribución de tópicos, fueron características significativas que complementaron a la rama profunda. Estas características complementaron al modelo profundo para mejorar los resultados.

7.2 TRABAJO FUTURO

El desempeño de los modelos *Wide & Deep* depende mucho de la ingeniería de características para la rama ancha. Cada temática y cada aplicación requiere encontrar las características que más información aporten para realizar su tarea. Podrían utilizarse otras técnicas de pesado de palabras para encontrar otros métodos que se ajusten mejor a cada tarea. Estas pueden ser cualquier tarea de clasificación de texto.

Se puede seguir experimentando con arquitecturas y modelos diferentes, para aprovechar lo que cada uno de ellos puede ofrecer. Especialmente si los modelos son complementarios.

La aplicación de este modelo podría llegar a dar buenos resultados en otros estudios relevantes como la identificación de efectos adversos de medicinas o de las vacunas desarrolladas para Covid-19. Queda como trabajo a futuro aplicar este modelo a tareas similares como esta.

Referencias

[1] Aggarwal, C. C. (2018). Machine Learning for Text. doi:10.1007/978-3-319-73531-3

[2] Aji, A., Nityasya, M., Wibowo, H., Prasojo, R. & Fatyanosa, T. (2021). BERT Goes Brrr: A Venture Towards the Lesser Error in Classifying Medical Self-Reporters on Twitter, Proceedings of the Sixth Social Media Mining for Health Workshop 2021, June 10, 2021, pp. 58-64. https://doi.org/10.18653/v1/2021.smm4h-1.9

[3] Al-Garadi, M. A., Khan, M. S., Varathan, K. D., Mujtaba, G., and Al-Kabsi, A. M. (2016). Using online social networks to track a pandemic: A systematic review. Journal of biomedical informatics, 62:1–11.

[4] Al-Garadi, M. A., Yang, Y., Lakamana, S. and Sarker, A. (2020). A Text Classification Approach for the Automatic Detection of Twitter Posts Containing Self-reported COVID-19 Symptoms.

[5] Alpaydin, E. (2016). Machine learning: the new AI. MIT press.

[6] Bai, Y., & Zhou, X. (2020). Automatic Detecting for Health-related Twitter Data with BioBERT. Proceedings of the Fifth Social Media Mining for Health Applications Workshop & Shared Task 2020. Association for Computational Linguistics.

[7] Blank, G., & Reisdorf, B. C. (2012). The participatory web: A user perspective on Web 2.0. Information, Communication & Society, 15(4), 537-554.

[8] Chen, E., Lerman, K., & Ferrara, E. (2020). Tracking Social Media Discourse About the COVID-19 Pandemic: Development of a Public Coronavirus Twitter Data Set. JMIR Public Health and Surveillance, 6(2), e19273. 10.2196/19273

[9] Cheng, H., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., & Shah, H. (2016). Wide & Deep Learning for Recommender Systems.

[10] Ciregan, D., Meier, U., & Schmidhuber, J. (2012, June). Multi-column deep neural networks for image classification. In 2012 IEEE conference on computer vision and pattern recognition (pp. 3642-3649). IEEE.

[11] DeVerna, M. R., Pierri, F., Truong, B. T., Bollenbacher, J., Axelrod, D., Loynes, N., Torres-Lugo, C., Yang, K.-C., Menczer, F., & Bryden, J. (2021). CoVaxxy: A Collection of English-Language Twitter Posts About COVID-19 Vaccines. Proceedings of the International AAAI Conference on Web and Social Media, 15(1), 992-999. Retrieved from https://ojs.aaai.org/index.php/ICWSM/article/view/18122

[12] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[13] Fetzer, T., Hensel, L., Hermle, J., and Roth, C. (2020). Coronavirus Perceptions and Economic Anxiety. The Review of Economics and Statistics, 0:1–36.

[14] Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media.

[15] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning (Vol. 1, No. 2). Cambridge: MIT press.

[16] Google (2020). Machine Learning Crash Course with TensorFlow APIs [MOOC]. https://developers.google.com/machine-learning/crash-course

[17] Hyontai, S. U. G. (2018). Performance of machine learning algorithms and diversity in data. In MATEC Web of Conferences (Vol. 210, p. 04019). EDP Sciences.

[18] Internet Live Stats. (2021). Twitter Usage Statistics. https://www.internetlivestats.com/twitter-statistics/

[19] Kamath, U., Liu, J., & Whitaker, J. (2019). Deep learning for NLP and speech recognition (Vol. 84). Cham: Springer.

[20] Klein, A., Arjun, M., O'Connor, K., Flores, J, Weissenbacher, D. and Gonzalez, G. (2021). Toward Using Twitter for Tracking COVID-19: A Natural Language Processing Pipeline and Exploratory Data Set. J Med Internet Res 2021;23(1):e25314, URL: https://www.jmir.org/2021/1/e25314, DOI: 10.2196/25314

[21] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.

[22] López-Santillán, R., González, L., Montes-y-Gómez, M., López-Monroy, P., López-Santillán, M. (2022). A Transformer-based Wide & Deep Neural Network for Author Profiling.

[23] Lyu, J. C., Han, E. L., & Luli, G. K. (2021). COVID-19 Vaccine-Related Discussion on Twitter: Topic Modeling and Sentiment Analysis. Journal of medical Internet research, 23(6), e24435. https://doi.org/10.2196/24435

[24] Mackey, T., Purushothaman, V., Li, J., Shah, N., Nali, M., Bardier, C., Liang, B., Cai, M., & Cuomo, R. (2020). Machine Learning to Detect Self-Reporting of Symptoms, Testing Access, and Recovery Associated With COVID-19 on Twitter: Retrospective Big Data Infoveillance Study. JMIR public health and surveillance, 6(2), e19509. https://doi.org/10.2196/19509

[25] Magge, A., Klein, A., Flores, I., Alimova, I., Al-garadi, M., Miranda-Escalada, A., Miftahutdinov, Z., Farré-Maduell, E., López, S., Banda, J., O'Connor, K., Sarker, A., Tutubalina, E., Krallinger, M., Weissenbacher, D., & Gonzalez-Hernandez, G. (2021). Overview of the Sixth Social Media Mining for Health Applications (# SMM4H) Shared Tasks at NAACL 2021. In Proceedings of the Sixth Social Media Mining for Health Applications Workshop & Shared Task.

[26] Magge, A., Pimpalkhute, V., Rallapalli, D,. Siguenza, D. and Gonzalez, G. (2020). UPennHLP at WNUT-2020 Task 2 : Transformer models for classification of COVID19 posts on Twitter. 378-382. 10.18653/v1/2020.wnut-1.52.

[27] Marsland, S. (2015). Machine learning: an algorithmic perspective. CRC press.

[28] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[29] Müller, M., Salathé, M. & Kummervold, P. (2020). COVID-Twitter-BERT: A Natural Language Processing Model to Analyse COVID-19 Content on Twitter.

[30] Office of Public Health Scientific Services. Centers for Disease Control and Prevention (2018). Public health surveillance: Preparing for the future. Technical report.

[31] Ortega, R., Franco, A. & Montes, M. (2019). Identificación del perfil de autores en redes sociales usando nuevos esquemas de pesado que enfatizan información de tipo personal. Computación y Sistemas, 23(2), 501-510. Epub 10 de marzo de 2021.https://doi.org/10.13053/cvs-23-2-3005

[32] Powers, D. (2008). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Mach. Learn. Technol.. 2.

[33] Pradha, S., Halgamuge, M. and Tran, N. (2019). Effective Text Data Preprocessing Technique for Sentiment Analysis in Social Media Data, 2019 11th International Conference on Knowledge and Systems Engineering (KSE), pp. 1-8, doi: 10.1109/KSE.2019.8919368.

[34] Roitero, K., Bozzato, C., Mea, V.D., Mizzaro, S., & Serra, G. (2020). Twitter goes to the Doctor: Detecting Medical Tweets using Machine Learning and BERT. SIIRH@ECIR.

[35] Secretaría de Salud. (2020). Sistema Nacional de Vigilancia Epidemiológica. https://www.gob.mx/salud/acciones-y-programas/sistema-nacional-de-vigilancia-epidemiologica

[36] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena, 404, 132306.

[37] Swaminathan, S. (2018). Logistic Regression – Detailed Overview. Towards Data Science, https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc (Recuperado: 2020-04-15).

[38] Taylor, D. B. (2020).A Timeline of the Coronavirus Pandemic. The New York Times, https://www.nytimes.com/article/coronavirus-timeline.html.

[39] Twitter. (2021). Stream Tweets in Real-time. https://developer.twitter.com/en/docs/tutorials/stream-tweets-in-real-time

[40] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. arXiv preprint arXiv:1706.03762.

[41] Wakamiya, S., Morita, M., Kano, Y., Ohkuma, T., and Aramaki, E. (2019). Tweet Classification Toward Twitter-Based Disease Surveillance: New Data, Methods, and Evaluations. J Med Internet Res, 21

[42] Wang, C., Nulty, P., & Lillis, D. (2021). Transformer-based Multi-task Learning for Disaster Tweet Categorisation.

[43] Wang, L., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Eide, D., Funk, K., Kinney, R., Liu, Z., Merrill, W., Mooney, P., Murdick, D., Rishi, D., Sheehan, J., Shen, Z., Stilson, B., Wade, A. D., Wang, K., Wilhelm, C., Xie, B., … Kohlmeier, S. (2020). CORD-19: The Covid-19 Open Research Dataset. ArXiv, arXiv:2004.10706v2.

[44] WHO (2020).Coronavirus disease (COVID-19): weekly epidemiological, update 6.World Health Organization, https://www.who.int/docs/default-source/coronaviruse/situation-reports/20200921-weekly-epi-update-6.pdf?sfvrsn=d9cf94966.

[45] Wicke P, Bolognesi MM (2020) Framing COVID-19: How we conceptualize and discuss the pandemic on Twitter. PLOS ONE 15(9): e0240010. https://doi.org/10.1371/journal.pone.0240010

[46] Wikipedia (2010). Multilayer Neural Network.https://tinyurl.com/y6dygnzb.

[47] Witten, I., Frank, E. & Hall, M. (2016). Data Mining: Practical Machine Learning Tools and Techniques (Fourth Edition). Morgan Kaufmann.

[48] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., & Rush, A. (2020). HuggingFace's Transformers: State-of-the-art Natural Language Processing.

Apéndice

A.1

A.1.1 - Publicaciones

- **Valdés, A**., López, J., Montes, M. (2021, Jun). ***UACH-INAOE at SMM4H: a BERT based approach for classification of COVID-19 Twitter posts***. Proceedings of the Sixth Social Media Mining for Health (#SMM4H) Workshop and shared Task 2021 jun. Association for Computational Linguistics Mexico City, Mexico. 10.18653/v1/2021.smm4h-1.10 https://aclanthology.org/2021.smm4h-1.10 https://doi.org/10.18653/v1/2021.smm4h-1.10 P. 65-68
- **Valdés-Chávez, A**., López-Santillán, J., Gonzalez-Gurrola, C., Ramírez-Alonso, G., Montes-y-Gómez, M. (2022). A Wide & Deep Learning Approach for Covid-19 Tweet Classification. En revisión.

A.1.2 – Conferencia

- **Alberto Valdes-Chavez**, Jesús Roberto López-Santillán, Manuel Montes-y-Gómez (Jun 07 2021). **UACH-INAOE at SMM4H: a BERT based approach for classification of COVID-19 Twitter posts**, NAACL 2021, Underline Science Inc. Available from: https://underline.io/lecture/20426-uach-inaoe-at-smm4h-a-bert-based-approach-for-classification-of-covid-19-twitter-posts

Curriculum Vitae

Educación

Maestría en Ingeniería en Computación – En progreso

Universidad Autónoma de Chihuahua (2022)

Ingeniería en Mecatrónica

Universidad La Salle Chihuahua (2015)

Experiencia

Profesor

Universidad La Salle Chihuahua (2019 - 2021)

- Impartiendo materias de ética y ciencias religiosas.

Practicante de Mantenimiento

Safran (2019)

- Proyecto: Creación de un programa de ayuda visual para el ensamble de más de 2,000 interruptores de aeronaves.

Publicaciones

- UACH-INAOE at SMM4H: a BERT based approach for classification of COVID-19 Twitter posts. Proceedings of the Sixth Social Media Mining for Health (#SMM4H) Workshop and shared Task 2021.

Domicilio Permanente: Loma Vieja 1406 Lomas Karike. Chihuahua, Chihuahua, 31120.

Esta tesis/disertación fue mecanografiada por el autor.