UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

Detección automática de pseudociencia

T E S I S

QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A :

Víctor Martinell García

TUTOR

Dra. Gemma Bel Enguix
Instituto de Ingeniería

Ciudad Universitaria, Cd. Mx. diciembre, 2019

Para el animal

Agradecimientos

Diana, mis papás, hermana y mi maestro.

También agradezco el apoyo del proyecto PAPIIT-IA401219.

Índice general

Agradecimientos II

1. Introducción 1

2. Justificación 3

3. Trabajos previos 6

4. Datos 11

4.1. ¿Qué es verdad? 11

4.2. Fuente de datos 12

4.2.1. Extracción de datos 14

4.2.2. Distribución de los Datos 15

5. Marco Teórico 22

5.1. Redes neuronales profundas 24

5.1.1. Aprendizaje 24

5.1.2. Perceptrón 26

5.1.3. Redes neuronales 27

5.2. Embeddings 29

5.3. Redes recurrentes 30

5.3.1. LSTM 31

5.3.2. AWD-LSTM 34

5.4. Universal Language Model Fine Tuning 35

5.4.1. Pre-entrenamiento del LM 36
5.4.2. Entrenamiento del LM 36
5.4.3. Entrenamiento de la tarea objetivo 36

6. Metodología **38**

7. Resultados **42**

8. Conclusiones **49**

Capítulo 1

Introducción

Hoy en día, debido al cada vez mayor uso del Internet, la difusión de la información ha alcanzado escalas y velocidades sin precedentes. A pesar de las muchas ventajas que esto tiene también, inevitablemente, hay problemas que surgen. La diseminación de información falsa es tan rápida y no cuenta con ningún filtro, por esto corre el riesgo de ser usada para manipular a la sociedad (Howell, 2013).

Esto, aunado con la cantidad de información que se genera diariamente, imposibilita su administración y verificación manual por instituciones confiables. Así ha surgido la necesidad de crear herramientas que faciliten o automaticen la tarea de verificación.

En este trabajo se obtiene un sistema que logra identificar de manera automática si un sitio de Internet está promoviendo las creencias pseudocientíficas y teorías de conspiración o, por el contrario, está hablando de ciencia confiable.

Es una contribución novedosa y útil por varias razones.

Primero, no se ha hecho ningún sistema que se centre en el problema de la divulgación de la ciencia. La mayoría de los sistemas similares están siempre enfocados en política. Pero, la difusión de creencias preudocientíficas y de conspiración puede tener un profundo impacto social, sobre todo en el campo de la salud donde el rechazo a vacunas o tratamientos puede resultar en miles de muertes que se podían prevenir (Moore, 2009).

Segundo, se están utilizando técnicas modernas de aprendizaje profundo como

embeddings y redes recurrentes. En particular se recurre a ULMFiT (Howard and Ruder, 2018), una arquitectura que recientemente ha superado al estado del arte en múltiples tareas del procesamiento del lenguaje natural.

Tercero, para este trabajo se fabricaron tres corpus, uno de tendencia política y dos enfocados en sitios a favor y en contra de la ciencia. Estos se crearon a partir de un "crawler" que exploró cientos de sitios de Internet extraídos de las listas que provee *MediabiasFactcheck*. Gracias a esto los sistemas que se obtuvieron fueron entrenados con datos del mundo real y se espera que se puedan usar en cualquier otro sitio para ayudar a un usuario a decidir si debe confiar en el contenido.

El sistema final que se obtuvo logra identificar con un 0.88 de precisión si un sitio de Internet es poco confiable con su información científica.

Capítulo 2

Justificación

En la cultura actual, donde el uso de Internet ha adquirido una posición dominante, las diferencias entre el contenido creado por instituciones (Traditional Media Content TMC) y el creado por individuos (User Generated Content UGC) se vuelve cada vez más borroso. Esto ha empoderado a los individuos, permitiéndoles ser escuchados a pesar de no contar con los recursos de una gran institución. Sin embargo estos medios individuales generalmente están administrados por gente aficionada y su contenido suele ser parcial, sensacionalista, desinformado o sin verificar (Chen et al., 2015).

Este cambio en el modelo de comunicación de masas ha impactado principalmente al campo de las noticias y la información. En este ámbito, la falta de control permite la proliferación de páginas cuya prioridad es algún interés económico o político más que la calidad o veracidad de su contenido. Esto no es algo nuevo, los periódicos amarillistas han existido por décadas e incluso los rumores y chismes, que cumplen con muchas de las mismas características de las noticias falsas, son tan viejos como la humanidad. Pero la actual era digital ha empeorado la situación. La ganancia promedio generada por anuncios en páginas de Internet, "ad", aumentó hasta los 50 mil millones de dolares del 2015 al 2016 (Mitchell and Page, 2015) fomentando aun más el ser llamativo sobre la calidad del contenido.

La mayoría de la atención y los estudios sobre el impacto de la falsedad informativa se han centrado alrededor de noticias, específicamente del ámbito político (Mitchell

and Page, 2015; Baly et al., 2018a; Pérez-Rosas et al., 2017; Conroy et al., 2015; Janze and Risius, 2017). Incluso se les ha otorgado su propio nombre, las llamadas "fake news". Las noticias falsas pueden tener un profundo impacto social. En 2013 en el *World Economic Forum* (WEF) se señaló a la diseminación de información falsa como un riesgo geopolítico mayor (Howell, 2013). Este miedo fue parcialmente concretado durante las elecciones para la presidencia de los Estados Unidos de América, que fueron fuertemente influenciadas por el gobierno ruso utilizando "fake news" (Blake, 2018).

Pero este problema no solo afecta al ámbito político, hay otro tipo de información falsa que también tiene un fuerte impacto social negativo: la pseudociencia y las teorías de conspiración. La simple definición de lo que es una teoría de conspiración o la pseudociencia puede ser compleja y controversial, por lo que en este trabajo se tomará la definición utilizada en Lobato et al. (2014):

Se definirá una **teoría de conspiración** como la atribución de la causa última de un acontecimiento o evento (o el ocultamiento de dicho acontecimiento) a una plan secreto e ilegal orquestado por múltiples actores trabajando juntos.

Una teoría o explicación **pseudocientífica** es aquella descripción del mundo físico que presume ser ciencia pero utiliza evidencia no científica para sostenerse.

Las teorías de conspiración pueden tener un fuerte impacto social. Algunos estudios sugieren que las personas que creen en ellas son menos propensas a participar en los sistemas políticos o confiar en otras personas. Además, en muchos casos puede evitar o alentar el progreso de políticas sociales como campañas para contrarrestar el cambio climático, vacunación o la administración de tratamientos a médicos (Douglas, 2017; M. J. Lazer et al., 2018; Moore, 2009). En general, una vez que una persona está convencida por una teoría de conspiración el conocer nueva información que la refute no la hará cambiar de parecer. Pero informar a las personas antes de ser expuestos a la teoría sí puede prevenir su aceptación (Douglas, 2017).

Las creencias de conspiración suelen estar estrechamente ligadas con explicaciones pseudocientíficas. Es común que cuando las teorías de conspiración utilizan conocimiento falso para defenderse, este provenga de una teoría pseudocientífica. De igual

forma las teorías pseudocientíficas se suelen apoyar en teorías de conspiración para explicar el rechazo por la comunidad científica. Por esto no siempre es fácil separar una de la otra (Lobato et al., 2014).

Mucha de esta información falsa suele ser presentada imitando la ciencia auténtica e incluso a veces científicos de universidades publican información falsa. Bohannon (2015) explica cómo un estudio falso se reportó en todos los noticieros del mundo. El autor hizo esto para denunciar lo fácil que es publicar datos no científicos en el contexto académico actual.

Esto, aunado con la presión que las instituciones imponen sobre los investigadores, llega a fomentar la publicación de ciencia de mala calidad o falsa. Por esto no siempre es fácil diferenciar entre ciencia y pseudociencia, y para el público en general puede resultar un verdadero reto.

Debido a la enorme cantidad de información disponible y el tiempo finito con el cual cuentan los expertos de cada tema correspondiente, en la actualidad, resulta imposible curar de forma manual todo el contenido que está disponible en Internet. Es así como la verificación pasa a ser un problema del consumidor.

Según Gottfried and Shearer (2016) el 62 % de los adultos en los Estados Unidos obtienen sus noticias de las redes sociales. Esto no es tan alarmante como suena, ya que muchos periódicos y noticieros reconocidos se apoyan en redes sociales para facilitar el acceso de las noticias a los usuarios. Es común que muchas noticias en redes sociales citen o refieran a una "fuente" externa. Por esta razón es posible atacar el problema de la detección de noticias falsas no directamente en las redes sociales sino enfocándose en los sitios de Internet fuente que las generan.

Capítulo 3

Trabajos previos

Se han intentado desarrollar sistemas para identificar de forma automática las noticias falsas (Mitchell and Page, 2015; Baly et al., 2018a; Pérez-Rosas et al., 2017; Conroy et al., 2015; Janze and Risius, 2017; Rubin et al., 2015; Horne et al., 2018b,a; Posadas-Durán, 2019) pero, debido a la complejidad del problema, estos sistemas suelen ser específicos a un tema o trabajan con un corpus fabricado a mano, que no necesariamente representa lo que hay disponible en el mundo real. En general, todos los trabajos son académicos y casi todos centrados en la política (Shvets, 2015). No se ha desarrollado ningún sistema que ayude a los usuarios a discernir entre un contenido científico cierto y aseveraciones pseudocientíficas.

Conroy et al. (2015) examina dos distintas estrategias para detectar noticias falsas que denomina la "aproximación lingüística" y la "aproximación de red", cada una con sus ventajas y desventajas. La primera utiliza el texto y aplica cualquier técnica de procesamiento de lenguaje natural (análisis semántico, estructuras retóricas, clasificadores, etc) para discernir sobre la veracidad del texto. Esta, generalmente, busca detectar patrones lingüísticos presentes en textos de noticias falsas. Tiene muchas limitaciones, por ejemplo, los patrones pueden ser específicos a cierto autor o género periodístico. La segunda estrategia utiliza fuentes externas al texto. Aprovechando la interconexión del Internet usa la información de recursos como *Wikipeida*, *Twitter* o *Facebook* para evaluar la credibilidad de la página fuente. La limitante de esta perspectiva sería que no exista ninguna de las fuentes externas que se utilizan,

algo bastante común para sitios informales. Ambas estrategias, al no counsultar directamente información del mundo real para tomar una decisión, sufren del mismo problema: es imposible verificar la veracidad de algunas afirmaciones, por ejemplo el acontecimiento de un hecho, si sólo se cuenta con el contexto del texto. Más adelante se habla sobre lo que esto implica.

Esta distinción entre aproximación lingüística y de red puede resultar insuficiente para describir las diversas técnicas utilizadas. Algunos trabajos toman una aproximación híbrida extrayendo características tanto del texto como de fuentes externas (Baly et al., 2018a). Otros, extraen características de diversas fuentes no textuales, por ejemplo, Janze and Risius (2017) intenta identificar automáticamente las noticias falsas en publicaciones de *Facebook*, usando información del texto de la publicación (título y contenido), comentarios de la publicación, análisis de brillo y presencia de caras en las imágenes de la publicación, "emojis" de reacción a la publicación e interacciones como "likes", etiquetados y las veces que fue compartido. Todos estos datos se utilizan para entrenar diversos clasificadores de los cuales el que obtuvo el mejor desempeño fue una SVM.

También se han hecho esfuerzos para estandarizar los corpus sobre los cuales se trabaja. Rubin et al. (2015) propone nueve puntos que debe cumplir un corpus para detección de "fake news".

1. Disponibilidad de ejemplos de todas las clases a predecir.
2. Contar con la información en formato textual. El vídeo, imágenes o audio no es conveniente.
3. Una forma fiable de verificar lo que es "verdad"
4. Homogeneidad en la longitud de los documentos.
5. Homogeneidad en tema, género, y autores.
6. Intervalo de tiempo predefinido.
7. Género periodístico. Por ejemplo crónica, sátira, sensacionalista, etc.

8. Costo de obtención, incluyendo dinero, tiempo y disponibilidad, etc.

9. Idioma y contexto cultural del corpus.

Estas propuestas son apropiadas en el ámbito académico, donde el estudio del impacto o existencia de características específicas requiere un control estricto sobre las macro características del corpus, pero resultan sumamente restrictivas si se busca utilizar el corpus en una tarea general del mundo real. La restricción de género limita fuertemente el alcance de cualquier aplicación relacionada con el corpus. De igual forma los requisitos de homogeneidad en el estilo de escritura y la longitud restringen aun más, a tal grado que trabajos como Pérez-Rosas et al. (2017) deben generar la mayoría de sus datos de forma manual.

Pérez-Rosas et al. (2017) construyeron dos corpus. Para ambos se siguió el mismo procedimiento: Primero se recolectaron varios artículos de noticias verdaderas, para cada artículo, se verificaron todos los hechos reportados manualmente utilizando diversas fuentes. Luego, utilizando *Amazon Mechanical Turk* se fabricaron versiones falsas de las noticias originales, cuidando mantener el mismo estilo de escritura y longitud de los documentos. Estos corpus luego fueron evaluados utilizando extracción de características con una SVM obteniendo un 0.74 de precisión.

El gran costo de contruir este tipo de corpus junto con la limitada utilidad del corpus resultante, hacen poco atractivo el trabajar con este estilo de datos. Por esto otros trabajos utilizan datos reales directamente obtenidos de sitios de Internet. Horne et al. (2018b) genera un corpus extrayendo la información utilizando cuatro "scrapers" o "crawlers" distintos dependiendo del estilo de la página fuente. Los sitios los extraen de dos fuentes: *Wikipedia* y *opensources.co* y algunas cuantas seleccionadas manualmente. Posteriormente utilizan *MediabiasFactcheck* para verificar y obtener las etiquetas de confianza de los sitios. Los datos se extraen diariamente a partir del mediodía, esto para capturar artículos que son eliminados a lo largo del día (esta práctica es común en sitios de noticias falsas). El corpus final cuenta con artículos de 92 sitios distintos para los cuales se recolectaron las siete características:

1. contenido

2. título
3. fuente
4. autor
5. fecha de publicación
6. liga del artículo
7. texto html crudo

Este corpus es posteriormente utilizado por los mismos autores en Horne et al. (2018a) donde, entre otras cosas, entrenan un clasificador *Random Forest* para predecir qué tan confiable es un artículo.

Otro trabajo que utiliza un corpus sin un estricto control es Baly et al. (2018a). Donde se extrae información de más de mil sitios de Internet con variados autores, estilos y temas. Estos fueron previamente recopilados y verificados por la misma institución independiente *MediabiasFactcheck*. Este provee etiquetas de veracidad de hechos reportados y tendencia política para cada uno de los sitios. Para cada sitio se seleccionaron algunos artículos, y se calcularon características (las mismas que reportan en Horne et al. (2018a)) para representarlo. Además, se utilizó la información de su URL y las páginas de *Wikipedia*, *Twitter* y *Alexa Rank* para obtener características, tanto binarias (la existencia de la página o si la cuenta está verificada) como vectores de representación, fechas y números. Todo esto se utilizó para entrenar múltiples clasificadores SVM, uno para cada una de las características y uno final utilizándolas todas, que predicen tanto la tendencia política como el reporte de hechos. El mejor resultado para el reporte de hechos fue 65.48 de exactitud, se obtuvo con el clasificador que utiliza todas las características. El modelo que sólo utilizó el cuerpo del artículo alcanzó 64.35 de exactitud.

Finalmente en todos los trabajos revisados se utilizan técnicas de aprendizaje superficial, como máquinas de soporte vectorial o *Random Forest*, que reciben como entrada vectores de características seleccionadas y calculadas previamente. Además

el foco de atención suele ser la política. La utilización de técnicas de aprendizaje profundo sigue siendo poco utilizada en el campo así como el enfoque de noticias de ámbito científico.

Capítulo 4

Datos

4.1. ¿Qué es verdad?

Antes de hablar sobre las técnicas y metodologías utilizadas en este trabajo vale la pena hablar sobre los datos, específicamente sobre lo que representan.

En el campo del procesamiento del lenguaje natural es tentador querer resolver tareas como detección de mentiras. Este es un planteamiento que debe tomarse con precaución. La veracidad de una frase no está contenida en el texto, si no en el mundo real al que se refiere este (por ejemplo *Yo estoy sosteniendo una manzana* es cierto a medida que la afirmación esté acorde con la realidad). Cuando las frases son de tipo cualitativo el problema es todavía mayor ya que la veracidad o falsedad se vuelve subjetiva (por ejemplo *Esta manzana es sabrosa*).

Por esto es imposible tener un sistema que te diga la veracidad de un texto sin consultar el mundo real. Los sistemas que dicen hacerlo en realidad simplemente están relacionando características del texto con las etiquetas que se les asignaron a los textos. Aun así, esto puede resultar útil, esta relación entre texto y etiquetas puede llegar a coincidir muy bien con la realidad, pero siempre debe tenerse presente que el sistema no sabe nada sobre la verdad. Por esta razón es posible utilizar ingeniería inversa para crear un "deep fake" es decir un texto que cumpla con todas las características que el sistema identifique como verdad pero su contenido sea falso Zhu et al. (2017).

Con respecto al ámbito científico, que es el foco principal de este trabajo, a lo que coloquialmente se suele referir como "la explicación científica" es simplemente la explicación que es aceptada por la mayoría de la comunidad científica, lo que no es aceptado sería "pseudociencia" o "ciencia falsa".

En este trabajo todos los resultados obtenidos simplemente toman las etiquetas extraídas de Baly et al. (2018a) y MediabiasFactcheck como la *verdad* y todos los modelos entrenados simplemente buscan predecirlas lo mejor posible. Múltiples de los sitios evaluados fueron revisados manualmente para verificar que, a menos a la consideración subjetiva del autor, las etiquetas concordaran en buena medida con las aceptadas por la comunidad científica y así evitar conflictos morales.

4.2. Fuente de datos

Todos los datos utilizados en este trabajo fueron extraídos utilizando la página MediabiasFactcheck y sus etiquetas como ciertas. Este es un sitio independiente que se dedica a la verificación de hechos. Como se mantiene por donaciones y no está afiliado con ninguna institución pública no es completamente reconocida por la comunidad de periodistas. Ha sido descrito como un "intento amateur" (Wilner, 2018) y criticado por reducir el complejo problema de parcialidad a sólo una calificación entera, sin nunca abordar el problema de que quien sea que decida esa cifra no puede hacerlo de forma imparcial.

No obstante este sitio ha sido utilizado en diversos trabajos de universidades como fuente de datos. (Baly et al. (2018a); Horne et al. (2018a)) Además es el único sitio que se logró encontrar que cuenta con una una clasificación enfocada a ciencia, con una lista de sitios "pro ciencia" y otra de teorías de conspiración y pseudociencia.

Mediabiasfactcheck cuenta con diversas listas de sitios que para los cuales provee diversas calificaciones en varias categorías:

- Left Bias
- Left-Center Bias

- Least Biased
- Right-Center Bias
- Right Bias
- Pro-Science
- Conspiracy-Pseudoscience

Las calificaciones que se otorgan a cada sitio son son un promedio de los artículos que lo componen. No necesariemente representan de manera exacta un artículo individual extraido del sitio.

Todos los sitios cuentan con una calificación de reporte de hechos que puede tomar cinco valores distintos: VERY HIGH, HIGH, MIXED, LOW, VERY LOW, con VERY HIGH indicando un reporte de hechos muy alto y VERY LOW uno muy bajo. Las dos clases extremas (VERY HIGH y VERY LOW) están poco representadas y en todos los experimentos se juntaron con la del valor más cercano (HIGH y LOW respectivamente)

Conspiracy-Pseudoscience cuenta con una lista de sitios que promueven las teorías de conspiración y la pseudociencia. Cada sitio cuenta con una calificación de conspiración y pseudociencia que puede tomar cinco valores distintos: None, Mild, Moderate, Strong, Tin Foil Hat/Quackery. Pro-Science cuenta con un listado de sitios a favor de la ciencia. El resto de los sitios tienen noticias políticas con tendencias de derecha o izquierda como lo indican sus nombres.

De estas listas dos de ellas (*Pro-Science* y *Conspiracy-Pseudoscience*) se utilizaron completamente como fuente de datos para fabricar los corpus en los que se enfoca este trabajo. El resto de las listas fueron indirectamente utilizadas para fabricar un corpus auxiliar para validar la eficiencia de ULMFiT (el método de aprendizaje profundo utilizado) como predictor de la fiabilidad de una noticia. Para esto se intentaron reproducir los resultados reportados en Baly et al. (2018a,b) con ULMFiT en el lugar de una SVM que se utiliza en el trabajo original. Por desgracia fue imposible conseguir los datos exactos utilizados en el trabajo por lo que se elaboró una lista de 1067 sitios

con sus etiquetas obtenida directamente de los autores Baly et al. (2018b) de la cual se extrajo el corpus auxiliar intentando así apegarse los más posible al artículo original.

4.2.1. Extracción de datos

Se extrajeron tres corpus, los primeros dos fueron de las listas que provee MediabiasFactcheck.com. Uno de la lista *Pro-Science* otro de *Conspiracy-Pseudoscience*, nos referiremos a ellos con los nombres de su lista fuente. El último se extrajo de la lista Baly et al. (2018b), que a su vez fue obtenido de las listas restantes de *MediabiasFactcheck*. A este último nos referiremos como *Validación*

Las etiquetas de los primeros corpus se tomaron de MediabiasFactcheck.com. Para el corpus de *Validación* las etiquetas se extrajeron directamente de las tabla reportadas en el artículo Baly et al. (2018b).

El contenido se extrajo utilizando un "crawler" que, para cada sitio, recorrió todas las ligas dentro del domino y del HTML crudo extrajo todo el "texto válido" que encontró.

Se consideró como "texto válido" las líneas de texto que no contuvieran caracteres '{' o '<', y terminaran con un signo de puntuación válido en ingles: ., ? o !. La primera regla fue para evitar tomar código javaScript, HTML o algún otro lenguaje de "scripting". La segunda fue para intentar restringirse a texto formal (se espera que si el texto es formal estará puntuado de forma correcta) y evitar cualquier tipo de texto libre, principalmente ligas a encabezados de otros artículos. Estas medidas buscan ser extremadamente simples y generales para así poderse aplicar a cualquier sitio de noticias, cada uno con su estilo y formato particular, y esperar capturar la mayor cantidad de información posible. Se evitaron utilizaron reglas demasiado especificas (utilizando elementos o etiquetas de HTML) ya que su creación es muy costosa en tiempo y muchos de los sitios no formales siguen pocas o ninguna de estas convenciones.

Gracias a esto se lograron extraer noticias de más de 800 sitos distintos. Pero, por este mismo enfoque permisivo, se capturaron varios documentos con texto no representativo que posteriormente se buscó mitigar. Además las reglas utilizadas nunca

captarán ni el título del artículo ni el autor de este, ya que incluso en un texto formal estos no suelen llevar signos de puntuación.

Cada liga fue considerada como un documento independiente del corpus a pesar de que todas las ligas provenientes del mismo sitio tienen las mismas etiquetas. Con todo esto se espera que, en su mayoría, cada documento del corpus sea una noticia con sus respectivas etiquetas.

Cada documento del corpus *Pro-Science* y *Conspiracy-Pseudoscience* cuenta con seis entradas:

- Sitio fuente. Dominio general del URL.
- URL. Dirección URL específica de donde se encontró el artículo.
- Veracidad de hechos. Valore de veracidad de hechos reportado por Mediabias-Factcheck.com.
- Conspiración.Valore de conspiración reportado por MediabiasFactcheck.com.
- Pseudociencia. Valore de pseudociencia reportado por MediabiasFactcheck.com.
- Texto. Texto válido extraído del URL.
- Notas. Notas que provee MediabiasFactcheck.com sobre el sitio.

A todos los elementos de la lista *Pro-Science* se les asignó una etiqueta de Conspiración y Pseudociencia de “None”.

Los documentos del corpus *Validación* son muy similares solo cambiando las etiquetas Conspiración y Pseudociencia por tendencia política.

4.2.2. Distribución de los Datos

Para ofrecer una idea general de cómo se distribuyen los datos se muestran a continuación algunas gráficas.

En total se obtuvieron artículos de 480 sitios para el corpus de *Validación* y 339 entre los corpus de *Pro-Science* y *Conspiracy-Pseudoscience*.

El análisis se centra en el conteo de documentos y la longitud en caracteres de cada documento.

Las tres tablas 4.1, 4.2 y 4.3 muestran cuántos documentos hay para cada una de las clases con las que se entrenó.

La gráfica 4.1 muestra la distribución de los sitios en corpus *Conspiracy-Pseudoscience*. Cada pareja de barras verde y azul representan un sitio, la altura de cada barra indica su valor de conspiración y pseudociencia, respectivamente. Abajo el color rojo o amarillo indican si el sitio que está por arriba tiene reporte de hechos bajo o mixto, respectivamente.

La tabla 4.4 y las tres gráficas 4.3, 4.2 y 4.4 muestran, con la altura y, la distribución de longitud en caracteres de todos los documentos para los tres corpus que se construyeron. Los documentos que provienen del mismo sitio se agruparon juntos, para facilitar su distinción se agregó un cambio de color de fondo cada que comienza un nuevo sitio. Además estos grupos se ordenaron de acuerdo a cantidad de documentos, con los sitios con más ejemplos a la izquierda.

Tabla 4.1: Distribución de veracidad para corpus de *Validación*

Etiqueta de veracidad	**Número de documentos**
HIGH	15,320
MIXED	5,853
LOW	4,478

Tabla 4.2: Distribución de veracidad para *Pro-Science* y *Conspiracy-Pseudoscience*

Etiqueta de veracidad	**Número de documentos**
VERY HIGH	3,402
HIGH	2,512
MIXED	5,325
LOW	7,229

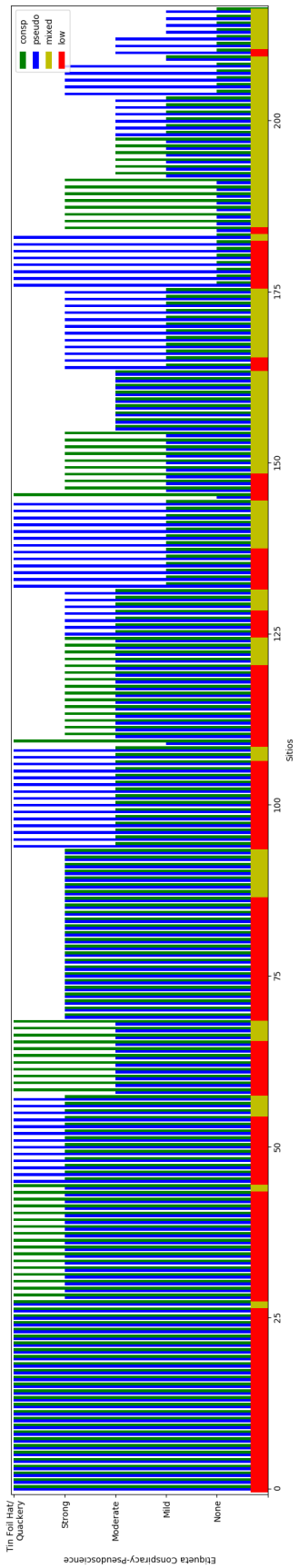


Figura 4.1: En verde y azul, de 1 a 5, se representa el valor de conspiración y pseudociencia con la altura en y. Abajo el color amarillo y rojo indican el reporte de hechos.

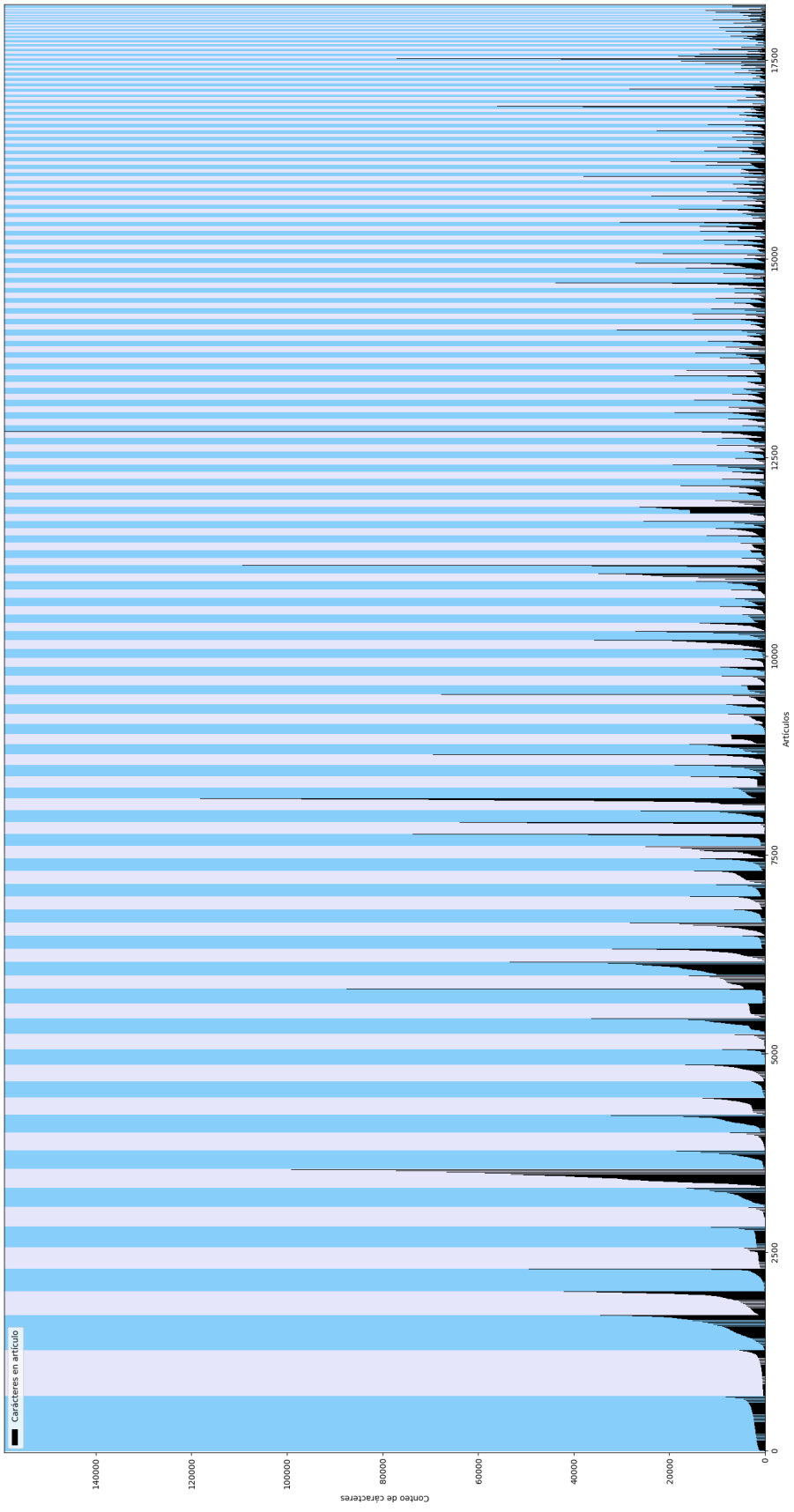


Figura 4.2: Distribución en longitud de caracteres para el corpus *Conspiracy-Pseudoscience*. La gráfica se cortó a una altura de 140,00 caracteres para que fuera legible.

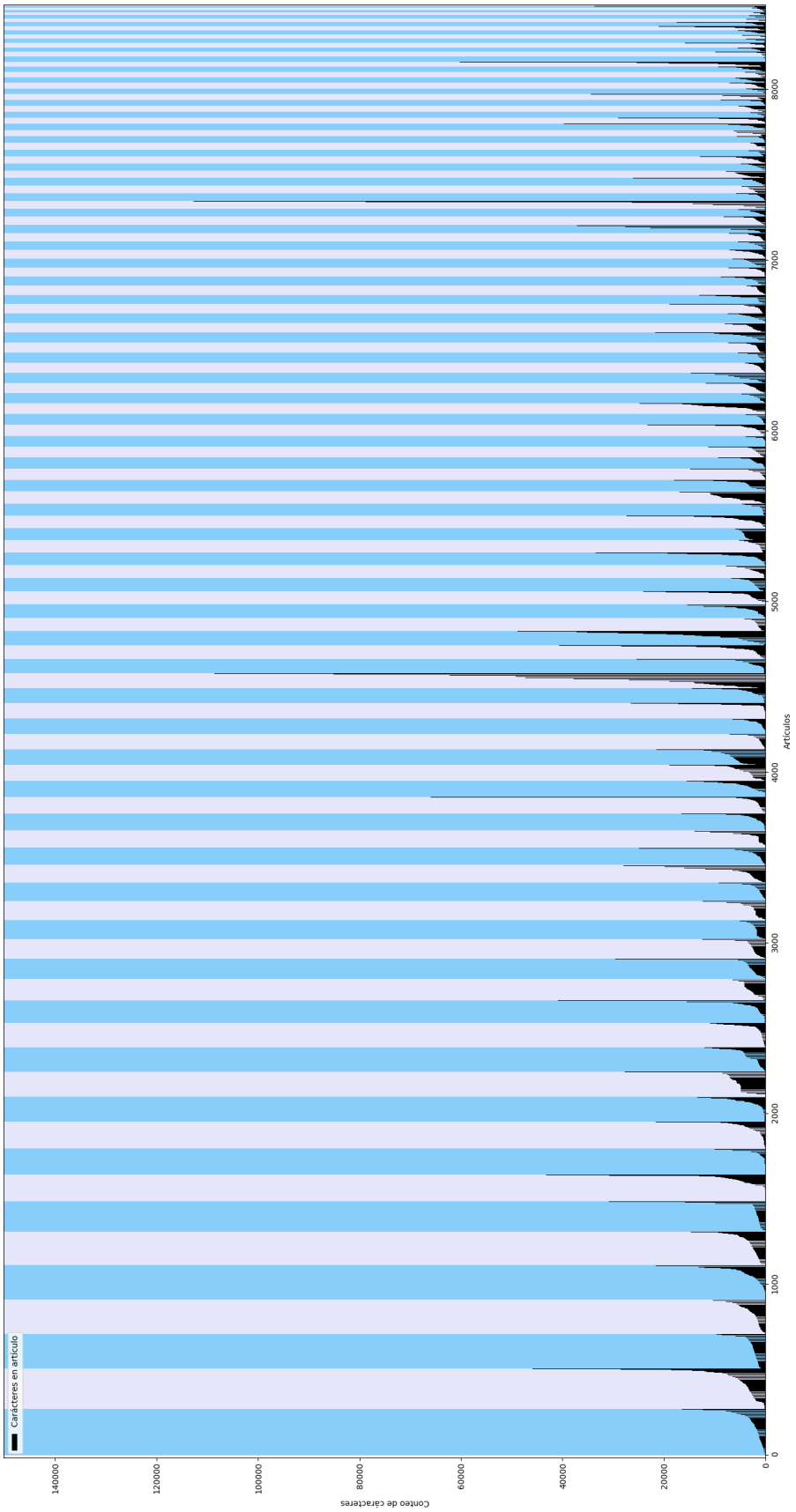


Figura 4.3: Distribución en longitud de caracteres para el corpus *Pro-science*. La gráfica se cortó a una altura de 140,00 caracteres para que fuera legible.

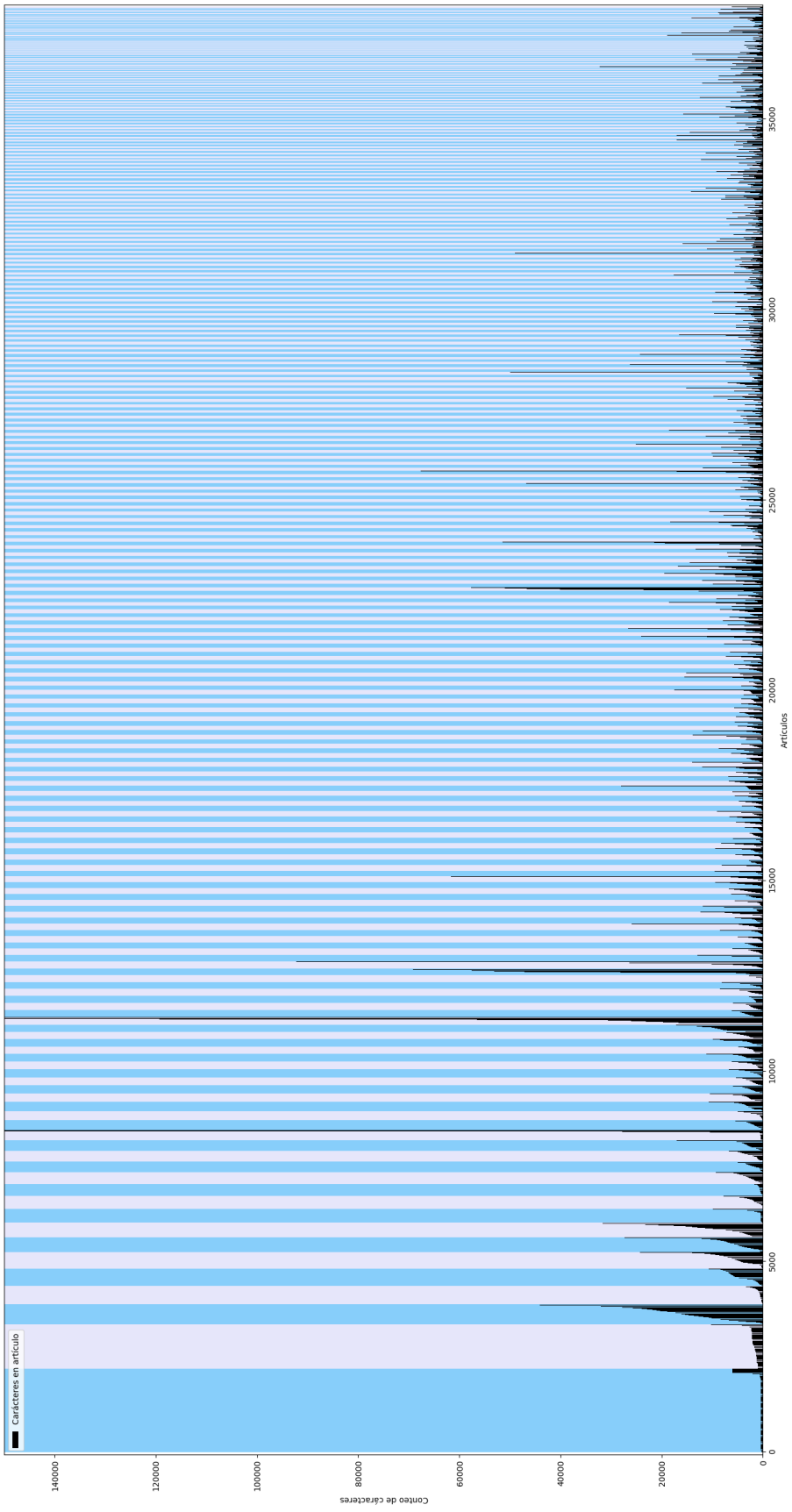


Figura 4.4: Distribución en longitud de caracteres para el corpus *Validación*. La gráfica se cortó a una altura de 140,00 caracteres para que fuera legible.

Tabla 4.3: Distribución corpus *Pro-Science* y *Conspiracy-Pseudoscience*

Etiqueta de tendencia cintífica	**Número de documentos**
Conspiracy-Pseudoscience	17,936
Pro-science	8,297

Tabla 4.4: Estadística descriptiva de la longítud en caracteres de los corpus.

	Validación	*Pro-Science* y *Conspiracy-Pseudoscience*
Varianza	499,857,808	136,465,003
Promedio	4,244	4,588
Desviación estándar	22,357	11,681

Capítulo 5

Marco Teórico

El utilizar máquinas para resolver problemas ha sido un sueño de la humanidad desde tiempos ancestrales. En la antigüedad se construyeron máquinas de complejidad cada vez mayor para resolver distintas tareas mecánicas y a mediados del siglo XX, cuando se construyeron las primeras computadoras, se intentó y especuló sobre su utilización para resolver problemas que se consideraban reservados para el intelecto humano, la llamada Inteligencia Artificial (IA).

Irónicamente, en la actualidad las computadoras han logrado resolver problemas que son extremadamente complicados para las personas (aritmética y calculo numérico); en cambio, problemas que, a nuestro parecer, son triviales resultan muy difíciles para las computadoras.

Esto es porque las computadoras son eficientes resolviendo problemas que se pueden describir con precisión de manera formal, como un juego de ajedrez u operaciones matemáticas. En cambio muchos problemas, a pesar de lo simples que nos puedan parecer, resultan ser muy difíciles de describir numéricamente. Cosas como reconocer objetos en una imagen o entender el habla.

Debido a esta dificultad de obtener una representación numérica, durante muchos años, la técnica para resolver tareas complejas se apoyaban en expertos que manualmente idearan e hicieran abstracciones que la computadora pudiese utilizar. A esta aproximación se le conoce como "basada en conocimiento". En general no dio muy buenos resultados, ya que la creación de reglas, su cálculo y obtención resultaba

muy costosa para problemas complejos y los resultados obtenidos nunca eran completamente satisfactorios. Además, las reglas, al ser creadas específicamente para un problema en particular, raramente eran generales; por ello era necesario crear nuevas reglas para problemas similares.

Todo esto sugiere que es preferible dejar a las computadoras obtener su propia representación. A esto se le denomina "aprendizaje de máquinas". Existen muchas técnicas distintas de aprendizaje, la regresión logística, clasificación Bayesiana ingenua o las máquinas de soporte vectorial (SVM) son técnicas de optimización matemática que se utilizaron inicialmente para "aprender" una distribución y resolver problemas.

El desempeño de estas técnicas depende principalmente en la representación numérica inicial de los datos. Por ejemplo, el desempeño de un sistema detector de correos basura (spam) no será muy bueno si se utilizan como entrada los valores ASCII del texto. En cambio si se le entrega el conteo de palabras clave y ligas a sitios externos sus resultados serán muy superiores.

Muchas tareas muy complejas pueden resolverse eligiendo el conjunto de características iniciales apropiado, pero, en la mayoría de los casos, esto es un problema muy complicado como para resolverse manualmente utilizando la estrategia basada en el conocimiento. Una solución es utilizar aprendizaje para obtener estas características de manera automática, a esto se le conoce como aprender una representación del conocimiento. Existen muchas técnicas distintas para automatizar la representación del conocimiento, una de las más simples y útiles son los "autoencoders".

Si se utiliza aprendizaje para obtener una representación de la información y posteriormente esto se alimenta a un sistema de aprendizaje para analizarlo se obtiene a lo que se denomina "aprendizaje profundo". Si se concatenan múltiples copias del modelo, normalmente capas de perceptrones (sección 5.1.3), se espera que las primeras capas trabajen en simplificar la representación progresivamente de forma que las últimas capas puedan resolverlo con facilidad.

5.1. Redes neuronales profundas

Una red neuronal esta compuesta de múltiples capas de perceptrones (sección 5.1.2) que son un mecanismo matemático inspirado por el funcionamiento de las neuronas.

5.1.1. Aprendizaje

Según Mitchell (1997) se puede decir que un programa aprende de una experiencia E sobre una tarea T con medida de desempeño D cuando el desempeño D al hacer T mejora con la experiencia E. Esta definición general permite una increíble variedad de métodos para el aprendizaje usando cualquier tipo de tareas, experiencias y medidas de desempeño.

En este trabajo en particular la tarea T que nos interesa es la de **clasificación**. Esta consiste en asignarle una clase $k \subset K$ a la entrada del programa, para esto normalmente se utiliza la codificación categórica definiendo una función del tipo $f : \mathbb{R}^n \to \{h_1, ..., h_k\}$ (con n la dimensión del espacio de entrada) donde h_k es la probabilidad de que la entrada pertenezca a la clase k.

La experiencia de la que se busca aprender es de tipo **supervisado**. En esta se cuenta con un conjunto J de ejemplos de parejas de entradas y salidas que describen una distribución sobre el espacio de las entradas. El aprendizaje se lleva a cabo observando todos los elementos de J e intentando que la función f replique la distribución lo mejor posible

Para las tareas de clasificación se suelen utilizar dos medidas de desempeño D: **precisión** y entropía cruzada categórica o **softmax**, la primera como medida final del desempeño, la otra durante el proceso de aprendizaje junto con la codificación categórica para evaluar con precisión las predicciones. La exactitud simplemente es la proporción de ejemplos para los cuales el modelo da una predicción correcta. El *softmax* se define como $CE = -\sum_i^C v_i log(p_i)$ con C el conjunto de clases a predecir, v la verdad (con 0 para todo menos la clase a la que pertenece el ejemplo) y p la predicción.

Con todo esto el aprendizaje se puede reducir a un problema de optimización, utilizar muchas experiencias para maximizar el desempeño. Normalmente esta optimización se logra utilizando una técnica matemática llamada *gradiente en descenso* en el que se calcula la derivada de la función de desempeño con todas las experiencias disponibles y se reajustan los parámetros de la tarea en dirección al gradiente. En particular se suele usar la versión estocástica de esta técnica SGD en la que se toman al azar subconjuntos de E para avanzar gradualmente.

Pero el verdadero reto para el aprendizaje de máquinas (por lo menos para el tipo de aprendizaje que le es de interés a este trabajo) no es simplemente conseguir una precisión muy alta, lo que se busca es que el modelo conserve una precisión alta incluso con datos con los que nunca se había encontrado antes, a esto se le conoce como **generalización**.

Hasta ahora el aprendizaje de máquinas ha sido descrito como un problema de optimización: se cuenta con un conjunto de ejemplos de entradas y salidas y se intenta obtener una función que describa lo mejor posible este conjunto. Pero cuando el conjunto de ejemplos C es simplemente un subconjunto finito de uno mucho más grande CM (por ejemplo si CM es todas las posibles imágenes de perros y gatos, sólo es posible conseguir un subjonjunto C de estas) el sólo tener una buena precisión en el conjunto de entrenamiento no es suficiente, incluso a veces puede ser dañino. Por esto no es ideal aproximar el aprendizaje sólo como un problema de optimización. De lo contrario es posible que un modelo termine aprendiendo particularidades del conjunto C de entrenamiento, prácticamente "memorizándolo", descuidando características generales que servirían para predecir elementos del súper conjunto CM. A esto se le conoce como sobre ajuste (overfitting).

Para evitar esto se utilizan diversa técnicas de **regularización**. La regularización se define como: Cualquier tipo de modificaciones al proceso de aprendizaje cuyo objetivo es mejorar el poder de generalización de un modelo (Goodfellow et al.).

Existen muchas técnicas de regularización, varias de estas son las causantes de grandes mejoras en el desempeño de diversas tareas y modelos. A continuación se describen muy brevemente algunas de las técnicas que son de interés para este trabajo.

- Dropout. Cada peso de la red tiene una probabilidad p de anularse (es decir $W_{ij} = 0$). Así se fuerza a la red a apoyarse en múltiples características para hacer sus predicciones.

- Early Stopping. Si el desempeño sobre el conjunto de validación baja o deja de aumentar se detiene el entrenamiento, esto indica que el modelo se está sobre ajustando al conjunto de entrenamiento por lo que es deseable detenerse.

- L_2. Es una de muchas técnicas de regularización que penalizan la aparición de valores muy grandes dentro del modelo, esto promueve un modelo más simple y menos propenso a sobre ajustarse.

Además de todos lo parámetros que se modifican durante el aprendizaje existe toda otra familia de valores que se eligen previamente, a estos se les conoce como **hiperparámetros**. Estos comprenden el tamaño del subconjunto que se utiliza en el SGD (batch size) y la velocidad con la que avanza en cada iteración (velocidad de aprendizaje), así como los valores de cualquier técnica de regularización o preprocesamiento de los datos que se lleve a cabo. Los hiperparámetros son cruciales durante el aprendizaje, en muchos casos la diferencia entre un modelo bueno y uno malo radica sólo en los valores de estos.

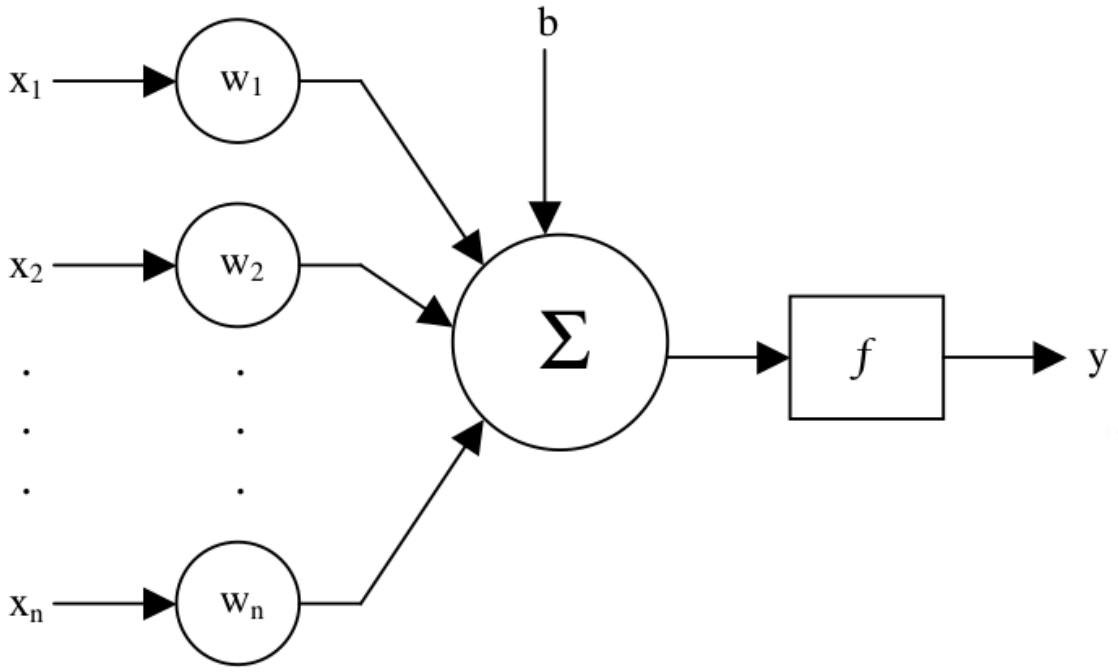
5.1.2. Perceptrón

Un perceptrón es un algoritmo de optimización matemática inspirado por el funcionamiento de las neuronas. Este toma n valores de entrada y genera una salida. En concreto cada una de las entradas es multiplicada por un peso w_n, luego todas son sumadas, y finalmente se le aplica una función f para calcular el valor de salida. A todo esto se le puede sumar una constante b.

$$f(\sum^{n} x_n w_n + b) \tag{5.1}$$

Los valores de los pesos w y la constante b se ajustan para así obtener la salida deseada en la optimización.

Figura 5.1: Diagrama de un perceptrón



La función f (llamada función de activación) y la constante b no aparece en la definición original del perceptrón pero son esenciales para la construcción de redes profundas, que son capaces de aproximar cualquier funcion. Los perceptrones por si solos sólo pueden representar funciones no lineales en n dimensiones (Csáji, 2001).

5.1.3. Redes neuronales

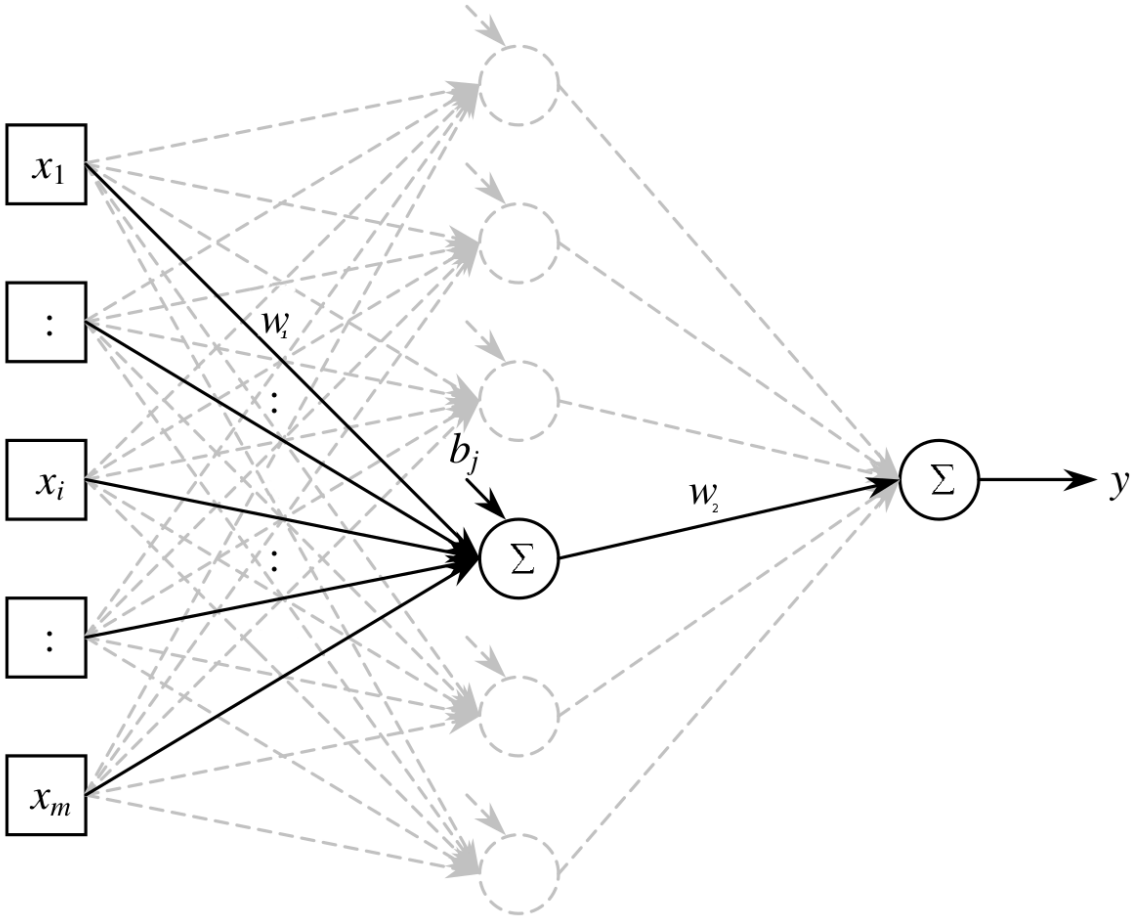
Un perceptron sólo puede aproximar funciones lineales. Para representar funciones más complejas se alimenta la misma entrada $\overline{x^0}$ a i perceptrones. Esto genera una salida $\overline{x^1}$, a la que se le conoce como "capa de salida", de tamaño i que una vez más puede ser alimentada a j perceptrones y así sucesivamente. Las conexiones entre perceptores se puede representar como una matriz W_{ni}, indicando que recibe una capa de tamaño n y genera una nueva capa de tamaño i. Así, al vector de entrada se le denomina la capa de entrada, el vector de salida la capa de salida y a todos los vectores intermedios como capas ocultas.

A esta arquitectura de red neuronal se le conoce como "Feedforward neural net-

work". En esta cada capa de perceptrones está completamente conectada con la siguiente capa, la información fluye en un sólo sentido y cada capa es utilizada una vez. Esta arquitectura simple se puede representar como una serie de multiplicaciones de vectores y matrices:

$$f_2(f_1(\overline{x}W_1 + \overline{b_1})W_2 + \overline{b_2})\ldots. \tag{5.2}$$

Figura 5.2: Diagrama de una red de dos capas de perceptrones



Se puede ver que las funciones de activación f_n no deben ser lineales, de lo contrario cualquier cantidad de capas se podrían colapsar en una sola.

Actualmente la función más utilizada es la unidad lineal rectificada o *ReLU*, ya que es extremadamente simple y da muy buenos resultados, pero anteriormente se han utilizado la función sigmoidal y *tanh*.

Este modelo, aunque simple, puede ser muy poderoso. De hecho se ha demostrado que pueden aproximar cualquier función.

El teorema de aproximación universal (Csáji, 2001) afirma que cualquier función continua sobre los reales ($C(\mathbb{R}^m)$) puede ser aproximada utilizando una red neuronal con una capa oculta y un número finito de perceptrones.

Teorema de aproximación universal

Sea $\phi()$ una función de activación arbitraria. Sea $X \subseteq \mathbb{R}^m$ con X compacta. Dado el espacio de funciones continuas sobre X denotado por $C(X)$. Entonces $\forall f \in C(X), \forall \epsilon > 0 : \exists n \in \mathbb{N}, a_{ij}, b_i, w_i \in \mathbb{N}, i \in \{1..n\}, j \in \{1..m\}$:

$$(A_n f)(x_1, ..., x_n) = \sum_{i=1}^{n} w_i \phi(\sum_{j=1}^{m} a_{ij} x_j + b_i) \tag{5.3}$$

Con:

$$||f - A_n f|| < \epsilon \tag{5.4}$$

A pesar de este impresionante resultado teórico en la práctica se requieren más de dos capas de perceptrones cuando se intenta aprender funciones complejas. En muchos casos, se requieren arquitecturas de redes más elaboradas que una "feedforward neural network" para lograr relver ciertos problemas (Csáji, 2001).

5.2. Embeddings

Al trabajar con lenguaje natural una aproximación usual es contar con una representación numérica del texto, para así poder aplicarles algoritmos matemáticos. Antes del reciente auge del aprendizaje profundo se recurría a técnicas estadísticas para representar textos y palabras numéricamente, como bolsa de palabras, TF/IDF o PMI, donde se utilizan contextos, frecuencias y conteos de palabras para representar vectorialmente ya sea una palabra o todo un texto. Más adelante, influenciados por la aproximación distribucional de las redes neuronales, se desarrollaron los llamados

word embeddings, representaciones vectoriales en un espaio $\mathbb{R}^N$ de todo un vocabulario para la cual palabras relacionadas se encuentran cerca en el espacio N dimensional de la representación.

Esta representación se obtiene naturalmente entre la primera y segunda capa de cualquier red que recibe texto como entrada, pero si no se cuenta con suficientes datos se puede recurrir a un embedding pre-entrenado.

Para obtener un embedding a partir de un texto de gran tamaño no etiquetado se entrena una red simple con una sola capa oculta de tamaño d sobre todo el texto. Primero se elige una ventana k_i alrededor de una palabra i para la cual se va a considerar que la palabra está relacionada con sus vecinos, luego se entrena la red para aprender esta relación.

Existen muchas formas de entrenar la red, se puede tomar todo el intervalo de palabras k_i como entrada e intentar predecir la palabra i o se puede elegir sólo una palabra al azar del intervalo y entrenar a relacionarla con i o cualquiera de estas dos anteriores pero utilizando i como entrada y prediciendo k_i. Para esto las palabras se deben codificar de alguna forma, normalmente se utiliza la llamada categórica o *one-hot encoding*, es decir se utiliza un vector de tamaño $|V|$, donde V es el vocabulario del corpus, con 1 en la posición correspondiente a la palabra y 0 en el resto.

Una vez entrenada esta red se utiliza la salida de la capa oculta como una codificación vectorial de tamaño d del vocabulario. Se ha demostrado que esta codificación cumple con las propiedades deseadas de conservar palabras similares cerca y en la práctica ha dado muy buenos resultados (Mikolov et al., 2013).

5.3. Redes recurrentes

Las redes recursivas son un tipo particular de redes neuronales especializadas en procesar datos secuenciales. Principalmente se han utilizado en el procesamiento de texto, ya que es un problema de alto interés e inherentemente secuencial.

El procesamiento secuencial de datos, normalmente, se representa como una serie temporal $t_1, t_2, ...$ En el tiempo t se le da como entrada el vector x_t junto con la salida

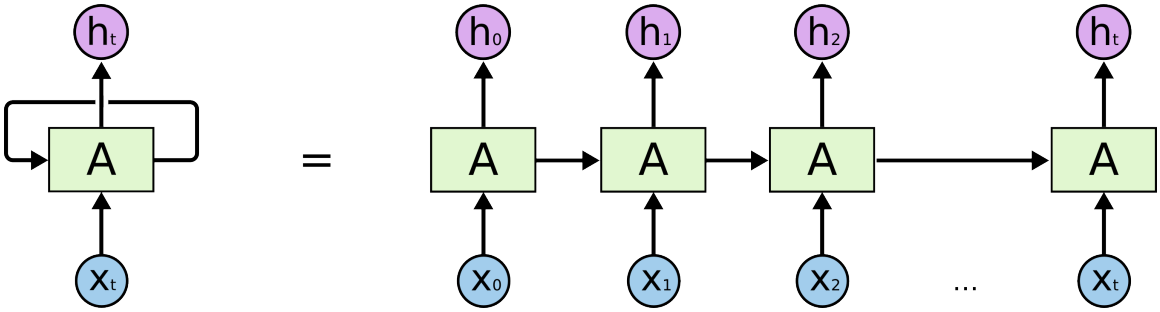


Figura 5.3: Equivalencia de representación de redes recursivas. Tomado de Olah (2015)

de memoria c_{t-1} del tiempo anterior $t-1$, estas son procesadas por la red y generan una salida h_t y una salida de memoria c_t. Así, a reserva de que la entrada pueda ser partida en fragmentos de un tamaño fijo, pueden ser utilizadas para procesar datos de tamaño variable, entregando cada uno de los fragmentos en orden a la red.

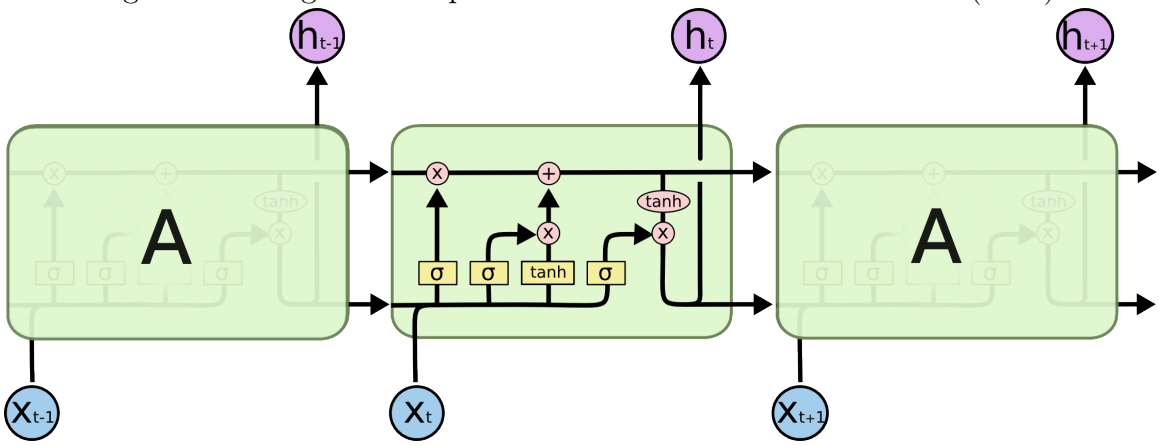
Se espera que la red aprenda características generales de los datos y complemente esta con la información de la salida anterior, así puede relacionar características relevantes en cualquier instante de tiempo.

En un caso real entre mayor sea la separación entre dos instancias de tiempo le es cada vez más difícil a la red relacionarlas. Los datos que se recuerdan están restringidos por el poder de representación de la red y el tamaño de la salida de memoria C_t, a menos que la representación sea extremadamente eficiente, es inevitable que la información vieja se vaya olvidando a medida que avanza el tiempo.

5.3.1. LSTM

Las redes "long short-term memory" (LSTM) (Hochreiter and Schmidhuber, 1997) intentan solucionar el problema de pérdida de memoria en el tiempo.

Figura 5.4: Diagrama temporal de red LSTM. Tomado de Olah (2015).



$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{5.5a}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{5.5b}$$

$$\tilde{c}_t = tanh(W_c[h_{t-1}, x_t] + b_{\tilde{c}}) \tag{5.5c}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{5.5d}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5.5e}$$

$$h_t = o_t * tanh(c_t) \tag{5.5f}$$

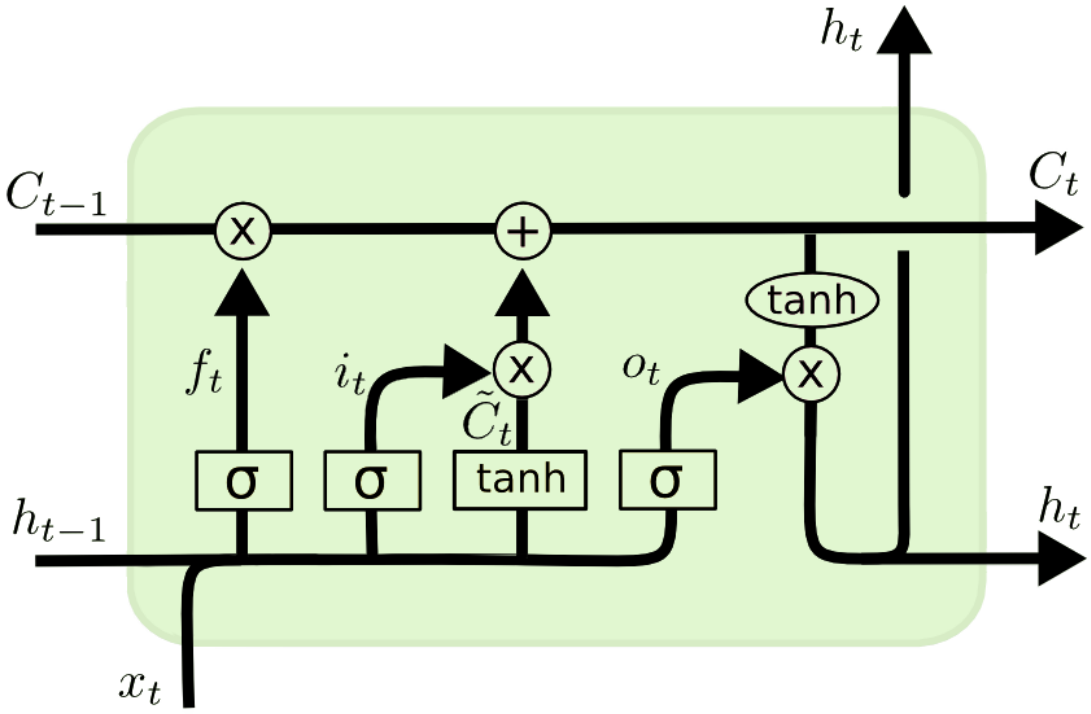
Las ecuaciones 5.5 describen una red LSTM. Las figuras 5.5 y 5.4 son representaciones visuales de las distintas operaciones vectoriales que se llevan a cabo. Vamos a desglosar las ecuaciones 5.5 para entenderlas mejor.

Primero podemos ver que las funciones 5.5a 5.5b 5.5c 5.5e son pequeñas redes neuronales, cuentan con parámetros W y b que van aprendiendo. Sus funciónes de activación son o una sigmoidal, con valores entre 0 y 1, o una tangente hiperbólica, con valores entre -1 y 1.

Las dos funciones restantes 5.5d y 5.5f calculan las salidas c_t y c_t que son la salida de la red LSTM y la salida de memoria respectivamente.

En la ecuación 5.5d la memoria c_t simplemente es una actualización de la memoria

Figura 5.5: Vista interna de una red LSTM. Tomado de Olah (2015).



previa c_{t-1}. Al multiplicar por f_t se olvida o refuerzan ciertos valores, luego se le suma nueva información en la forma de $i_t * \tilde{c}_t$.

Con esto se puede ver que la red 5.5a decide qué información conservar. Donde la salida 0 indica olvidar por completo y 1 conservar totalmente.

La información nueva que se va a agregar se trabaja en dos partes. La red 5.5b, llamada la compuerta de entrada, determina qué valores de $\tilde{c}_t$ serán actualizados, al igual que el caso anterior utiliza valores entre 0 y 1 para indicar el grado de actualización. Luego la red 5.5c calcula los nuevos valores de la memoria. Estos dos se multiplican y suman a la memoria en 5.5d.

Finalmente la red 5.5e genera una salida o_t de valores entre 0 y 1 que se utiliza para seleccionar los valores aplanados de la memoria $tanh(c_t)$ y produce la salida final como se ve en la función 5.5f.

Gracias a este elaborado sistema el desempeño de las LSTM suele ser superior al de una RNN simple. A pesar de esto ninguno de estos modelos es inmune al overfitting, para evitar esto se requieren diversas técnicas de regularización que permitan el

aprendizaje de la red a la vez que prevenga el sobre ajuste.

5.3.2. AWD-LSTM

Merity et al. (2017) propusieron diversas técnicas de regularización que juntas logran mejorar el desempeño de las redes LSTM.

Se hacen tres ajustes al dropout para mejorar el desempeño de la red. Primero se utiliza *Drop Connect* que aplica el dropout a los pesos internos W en lugar de a las salidas y las conexiones entre instantes de tiempo. En las conexiones donde no se utiliza *Drop Connect* en su lugar se aplica *Variational Dropout*. Este regulariza las máscaras de dropout para todo un ciclo de entrenamiento, calculando un único conjunto de valores que utiliza cada que la función de dropout es llamada. De esta forma los mismos pesos son anulados en toda la red. De igual manera aplican dropout a la capa inicial de *embedding* manteniéndola durante todo el ciclo.

ASGD es una variante SGD en la que a partir de un umbral T el gradiente que se utiliza para el aprendizaje se promedia con todos los gradientes previos obtenidos. NT-ASGD es una técnica para determinar el valor de T sobre la marcha.

Si la longitud de un documento es igual o mayor que el *batch size* este nunca se entrenará junto con otros documento, afectando el poder de generalización sobre esos datos. Por esto se varia el *batch size* así asegurando que los datos siempre se mezclen.

El *Weight tying* fuerza a la primera capa de *embedding* y la última antes del *softmax* a compartir sus valores disminuyendo sustancialmente la cantidad de valores de la red, al evitar que el modelo tenga que aprender la correspondencia entre entrada y salida.

Normalmente las capas internas de la red son del mismo tamaño que el *embedding* de palabra. Para librarse de esta restricción modifican las capas iniciales y finales para que adapten el tamaño de *embedding* a la red.

Finalmente se aplica un tipo modificado de regularización L_2 a las capas finales h_t, al que denominan *Activation Regularization* y *Temporal Activation Regularization*, que penaliza los valores mucho mayores a 0 y los cambios abruptos entre intervalos de tiempo.

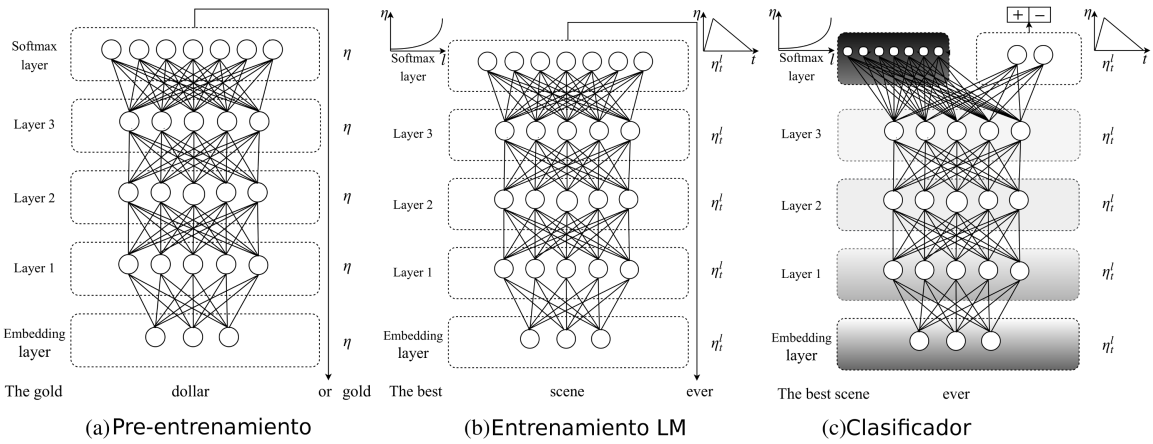
5.4. Universal Language Model Fine Tuning

Universal Language Model Fine Tuning (ULMFiT) es una técnica de transferencia del conocimiento para modelos de lenguaje propuesta por Howard and Ruder (2018). La técnica aprende un modelo de lenguaje (LM) sobre un corpus de texto no etiquetado de gran tamaño. Luego se puede transferir el conocimiento aprendido por el LM para resolver diversas tareas.

El método es "universal" en varios sentidos. El corpus para entrenar el LM no tiene que estar relacionado con la tarea que se desea resolver, funciona para diversas tareas con distintas clases, tamaño de corpus y documento y para cada una de estas tareas no es necesario cambiar la arquitectura del modelo, calcular nuevas características o reetiquetar clases.

La arquitectura básica de ULMFiT consiste de varias capas de AWD-LSTM, commo se muestra en la figura 5.6. A esta base, dependiendo si se está entrenando el LM o si se están haciendo los últimos ajustes para el modelo final, se agregan distintas capas al final.

Figura 5.6: Los tres pasos del entrenamiento de LSTM. Tomado de Howard and Ruder (2018).



El entrenamiento de ULMFiT consiste en tres pasos: primero se entrena sobre un corpus grande para aprender el LM, luego se entrena de la misma forma sobre el corpus objetivo, finalmente se entrena para resolver la tarea deseada. Estos tres pasos

se explican a continuación a mayor detalle.

5.4.1. Pre-entrenamiento del LM

Para que el modelo tenga una noción de la estructura del lenguaje con el que va a trabajar primero se entrena un LM que prediga la siguiente palabra dadas las n palabras anteriores que ha visto. Para esto no es necesario un corpus etiquetado, sólo uno de gran tamaño que esté en el mismo idioma. Normalmente *Wikipedia* es una buena opción.

5.4.2. Entrenamiento del LM

Una vez entrenado el LM sobre el corpus grande se procede a reentrenarlo utilizando el corpus objetivo. Esta es la parte donde se transfiere el conocimiento. Aunque el corpus objetivo sea pequeño el modelo utiliza la información aprendida durante el pre-entrenamiento para agilizar el entrenamiento.

Se utilizan dos técnicas novedosa para este segundo entrenamiento:

Discriminative fine-tuning utiliza un velocidad de aprendizaje distinta para cada capa del modelo. En lugar de utilizar un solo valor *theta* para actualizar los pesos se usa un vector de valores $\overline{\theta}$ con longitud igual a la profundidad del modelo.

Slanted triangular learning rates es una técnica para variar la velocidad de aprendizaje de forma lineal, primero se incrementa rápidamente para después decrementarla más lentamente.

5.4.3. Entrenamiento de la tarea objetivo

Tanto en Howard and Ruder (2018) como este trabajo se busca entrenar un clasificador. Para esto se agregan dos capas al final del LM entrenado previamente, una capa oculta con un ReLU de activación y luego un *softmax* para predecir las probabilidades de las clases.

Para entrenar el clasificador se proponen otras dos técnicas novedosas.

Concat pooling busca recuperar la mayor cantidad de información del recorrido del texto. Por esta razón, una vez alcanzado el final del documento en el tiempo T, se concatenan los resultados del max y min pooling de todos los tiempos anteriores a T (tantos como sea posible en el GPU).

Gradual unfreezing fija los pesos de todas menos las últimas capas agregadas. Esto se hace para evitar alterar drásticamente todo lo aprendido en las dos etapas anteriores. Una vez entrenada la red por una época se libera la siguiente capa y se vuelve a entrenar una sola época. Este proceso se repite hasta liberar todas las capas de la red.

BPTT for Text Classification guarda el estado final del conjunto de datos anterior para inicializar los valores de memoria del nuevo entrenamiento.

Bidirectional language model entrena dos LM cada uno recorriendo el texto en sentidos opuestos. Se usan todas las técnicas anteriores para entrenar dos modelos finales y sus resultados se promedian.

Capítulo 6

Metodología

Antes de hablar de los distintos entrenamientos que se hicieron y sus diferencias es conveniente hablar de sus similitudes. Más que un preprocesamiento de los datos se llevó a cabo una selección de estos para homogeneizarlos, es decir se eliminaron textos poco representativos y de tamaño excesivo. Sólo se consideraron los textos para los que se hubiese logrado extraer al menos 10 documentos de su sitio fuente, además tenían que tener una longitud mayor a 500 caracteres y menor a 10,000. La distribución del corpus resultante se muestra en la tabla 6.1. Esto fue para evitar textos muy cortos, que difícilmente pueden considerarse como artículos, y muy largos, que resultan demasiado extensos y salen mucho de la media.

Una vez más, como se quería capturar la mayor cantidad de información, estas restricciones son muy laxas y no eliminan todos los documentos problemáticos. Se identificaron dos tipos de documentos que aún abundaban tras homogeneizar. Primero, los documentos repetidos, puede que la misma noticia esté presente en el mismo sitio pero con direcciones URL distintas, aún peor, en algunos sitios de noticias falsas un mismo fragmento de texto es reciclado en distintos artículos. El segundo caso identificado fue los fragmentos de texto no formal que lograron colarse, por ejemplo, texto de la forma "...Subscribe Now..." aparecía al inicio o final de todas las páginas de ciertos sitios.

Ambos casos se atendieron indirectamente durante la creación de los corpus de entrenamiento y prueba, en lugar de intentar eliminarlos directamente. La limpieza

del corpus no era una prioridad ya que lograrlo es muy costoso en tiempo y, al querer tener un sistema que trabaje en cualquier documento del mundo real, es deseable trabajar directamente con los textos obtenidos al utilizar las simples reglas generales de limpieza que se describieron anteriormente. Además es posible que los fragmentos no formales puedan servir como indicadores y los textos repetidos como aumento de datos, así que, mientras el desempeño final sea bueno su inclusión es bienvenida.

La validación del entrenamiento se hizo con mucho cuidado. Si se tienen documentos y patrones repetidos tanto en el conjunto de entrenamiento como en el de pruebas, se distorsiona el valor real de desempeño del modelo. Es necesario que los conjuntos de entrenamiento y prueba sean disjuntos para que el desempeño del modelo se pueda medir con confianza. Para evitar esto se agruparon todos los documentos provenientes del mismo sitio juntos al generar los conjuntos de entrenamiento y prueba, de esta forma si el clasificador memoriza patrones que sean distintivos de un sitio en particular en lugar de buscar características más generales la puntuación de validación será mucho menor que la de entrenamiento. Para combatir la alta dispersión de tipos de texto todas las validaciones se hicieron utilizando validación cruzada en cinco partes, además, debido al carácter estocástico del método ULMFiT, se realizó cada evaluación dos veces para mitigar el efecto que la inicialización de los pesos y el orden en el que se alimentan los datos el entrenar puedan tener en los resultados.

Las pruebas se hicieron en tres partes. Primero se utilizó el corpus de *Validación* para validar la eficiencia de utilizar ULMFiT como clasificador de noticias. Segundo se entrenó un clasificador de veracidad de hechos reportados con ambos corpus de noticias de ciencia. Finalmente se entrenó un clasificador para distinguir automáticamente entre una noticia de ciencia confiable y una no confiable. A continuación se elabora sobre cada uno de estos entrenamientos.

En la primera prueba, como ya se explicó anteriormente, se intentó replicar el clasificador de Baly et al. (2018a), un predictor de tres clases (HIGH, MIXED y LOW) que representan la veracidad de hechos reportados. Para esto simplemente se entrenó ULMFiT con el corpus *Validación* que, como ya se describió antes, extrajo los datos con un "crawler" y las etiquetas de los sitios reportados por los autores. Se

esperaba obtener un resultado similar o mejor al reportado por los autores para así justificar la utilización de este método en lugar de los clasificadores tradicionales. Sin importar el resultado (ya sea que supere o sea peor que su contra parte tradicional) este servirá como estándar para analizar el resto de los resultados en contexto.

La siguiente prueba consiste en resolver el mismo problema de la primera pero utilizando los corpus de ciencia (*Pro-Science* y *Conspiracy-Pseudoscience*), se enteraron modelos desde cero para predecir la veracidad de hechos reportados. Esto sirve para poner el corpus en contexto. ¿Qué tan difícil de aprender son estos corpus comparados con el de *Validación*?

Finalmente se entrenó un clasificador de dos clases, ciencia confiable y no confiable. Para este se consideraron como confiables todos los documentos del corpus *Pro-Science* y como no confiable todos los del corpus *Conspiracy-Pseudoscience*.
Todos los modelos se evaluaron utilizando la medida de exactitud. Esta medida trabaja con predicciones discretas, por ejemplo un sitio de la última prueba sólo puede pertenecer a la clase *pro ciencia* o a la clase *consp-pseudo*, pero en realidad la salida *softmax* de todos los modelos predice probabilidades de pertenecer a cada una de las clases objetivo. Por defecto se asume que la clase a la que pertenece una predicción es la que tenga la mayor probabilidad, pero esto no siempre es lo ideal. Por ejemplo, si para una entrada e un modelo M da una predicción de la forma $M(e) = (0.499, 0.501)$ no es realmente confiable decir que $e \subset M$. Para contrarestar esto se puede agregar un intervalo para el cual la predicción se asigna una tercera clase *desconocido*. También puede que, debido a la forma del conjunto de entrenamiento, el modelo pueda predecir con extrema confianza una clase pero le sea difícil identificar a la otra, por ejemplo $e_0 \in C_0, e_1 \in C_1 \implies M(e_0) = (0.98, 0.02), M(e_1) = (0.4, 0.6)$. Para esto se puede mover el umbral para el cual se considera que una predicción pertenece a una clase (para el ejemplo anterior podría ser conveniente una confianza mayor a 0.7 para pertenecer a C_0 y asignarle el resto a C_1).

Para evaluar si era conveniente mover el umbral o agregar la tercera clase *desconocido* se hizo un barrido con un sólo umbral y luego otro con dos umbrales (mandando a la tercera clase todos los valores entre ambos umbrales), para cada paso del ba-

rrido se calcularon otras tres nuevas métricas la precisión, exhaustividad y el valor F1 y se buscó maximizar este último. Las tres métricas se calcularon para cada clase por separado, (*pro ciencia* y *consp-pseudo*), y se obtuvo un consenso entre ambos valores, este es un promedio tomando en cuenta el desequilibrio de clases (con una representación de aproximadamente dos a uno a favor de *consp-pseudo*).

Esto se logró primero obteniendo las probabilidades predichas para los cinco conjuntos de valuación de la validación cruzada para los dos modelos entrenados. Luego se asignaron las clases correspondientes para cada paso de los barridos. Con esto se calcularon las nueve métricas que se muestran en las tablas 7.4 y 7.5.

$$exactitud = \frac{VP+VN}{VP+VN+FP+FN}$$
$$precisión = \frac{VP}{VP+FP}$$
$$exhaustividad = \frac{VP}{VP+FN}$$
$$F1 = 2\frac{precisión*exhaustividad}{precisión+exhaustividad}$$

Con VP verdaderos positivos, VN verdaderos negativos, FP falsos positivos y FN falsos negativos

Tabla 6.1: Comparación de la distribución de los corpus *Pro-Science* y *Conspiracy-Pseudoscience* tras la reducción

Etiqueta	**Total documentos**	**Con reducción**
Conspiracy-Pseudoscience	17,936	12,323
Pro-science	8,297	6,020

Capítulo 7

Resultados

El primer entrenamiento que se realizó fue con el corpus de *Validación*. Esto fue para poder comparar el método ULMFiT con los métodos tradicionales en la tareas de identificación de la veracidad de los hechos reportados.

En la tabla 7.1 se muestran los resultados de la validación cruzada de ambos entrenamientos realizados. Para cada uno de los cinco conjuntos de entrenamiento de la validación cruzada (del 0 al 4) se reporta el número de iteraciones que se hicieron antes de ser detenidos por el *early stopping* y la exactitud que se alcanzó para ese conjunto de datos. Abajo se hace un promedio global de los cinco conjuntos de entrenamiento y debajo de estos se calcula un promedio de ambos experimentos como valor final a reportar.

Esta estructura descrita para la tabla 7.1 también se utiliza para las tablas 7.2 y 7.3.

En el segundo experimento se volvió a predecir la veracidad de los hechos reportados pero ahora utilizando los corpus de noticias *Pro-Science* y *Conspiracy-Pseudoscience*. Con esto se cuenta con un punto de comparación para el corpus foco de este trabajo. Los resultados se presentan en la tabla 7.2. Es evidente que ULMFiT tiene un desempeño bastante peor con estos corpus comparado con el de *Valicación*.

El último experimento que se realizó fue la identificación de sitios a favor y en contra de la ciencia. Los valores de exactitud son mucho mayores a los entrenamientos anteriores porque en este caso sólo se están prediciendo dos clases distintas (pro y

Tabla 7.1: Resultados para corpus de *Validación*

Conjunto de validación	**Entrenamiento 1**		**Entrenamiento 2**	
	Iteraciones	Exactitud	Iteraciones	Exactitud
0	14	0.5840	4	0.5856
1	7	0.6269	10	0.6246
2	8	0.6522	5	0.6399
3	6	0.5688	5	0.6409
4	9	0.6838	9	0.6730
Promedio		0.62314		0.63279
		0.627965		

Tabla 7.2: Resultados de validación cruzada para corpus de ciencia.

Conjunto de validación	**Entrenamiento 1**		**Entrenamiento 2**	
	Iteraciones	Exactitud	Iteraciones	Exactitud
0	8	0.4467	9	0.4985
1	9	0.4771	11	0.5463
2	4	0.5767	12	0.5777
3	6	0.6632	4	0.6485
4	6	0.4212	4	0.3855
Promedio		0.51698		0.5313
		0.52414		

con), a diferencia de los otros donde se predecían tres (HIGH, MIXED y LOW). No obstante un 0.84 de exactitud es muy bueno indicando que se cometió un error en poco más del 15 % de las predicciones.

Tabla 7.3: Resultados de validación cruzada para tendencia científica.

Conjunto de validación	**Entrenamiento 1**		**Entrenamiento 2**	
	Iteraciones	Exactitud	Iteraciones	Exactitud
0	8	0.8724	6	0.7974
1	5	0.8588	10	0.8182
2	5	0.8866	6	0.8550
2	10	0.9095	9	0.9114
4	4	0.7567	9	0.7935
promedio		0.8568		0.8351
		0.84595		

Para estimar con mayor detalle la eficiencia de los modelos entrenados como identificadores de artículos a favor y en contra de la ciencia se analizó la confianza exacta con la que predice cada uno de los ejemplos de los conjuntos de prueba. Luego se hizo un barrido para encontrar el umbral óptimo. Para evaluar esto se utilizaron las tres métricas de precisión, exhaustividad y F1, poniendo especial atención es esta última ya que toma en cuenta las dos anteriores. El tamaño de paso del barrido se fijó después de haber hecho pruebas con distintos valores y encontrar que un tamaño menor a 0.05 no ofrecía una mejora notable.

Debido a la enorme cantidad de datos que generó este barrido se promediaron los cinco resultados de cada modelo de la validación cruzada y se muestran en tablas separadas (7.4 y 7.5) los resultados de los dos entrenamientos realizados. Además se omiten por completo los resultados del barrido con dos umbrales (para la tercera clase *desconocido*) debido a que en todos los experimentos el valor óptimo se alcanzaba al minimizar el umbral para la clase nueva, indicando así que no era necesaria.

Se muestran los valores obtenidos para cada paso del barrido. La primera columna representa el valor de corte (el umbral), luego siguen tres triadas: F1, Precisión y Exhaustividad. Cada una de estas triadas tiene el valor obtenido para la clase en contra de la ciencia (con) a favor de la ciencia (pro) y, al inicio, el promedio de ambos.

Primero es importante ver los resultados del entrenamiento de comparación (7.1) que, como ya se dijo anteriormente, sirve para poner en contexto los resultados y analizar la eficiencia del método. Se obtuvieron 0.62314 y 0.63279 de exactitud como promedio de la validación cruzada de ambos entrenamientos, estos a su vez se promedian a **0.6279** para compararse con el **0.6435** reportado por Baly et al. (2018a). Cabe repetir que los datos con los que se entrenó este modelo no son los utilizados en el trabajo original por lo que su comparación no se puede hacer ciegamente. Tomando todo esto en cuenta se puede concluir que al analizar los mismos sitios de noticias con las mismas etiquetas se obtuvo un resultado que, aunque por debajo de los resultados originales, es suficientemente cercano como para validar la utilización de ULMFiT

Tabla 7.4: Resultados del barrido para el primero modelo.

Valor de corte	F1	con	pro	Precisión	con	pro	Exhaust	con	pro
0.05	0.67932	0.87041	0.58606	0.87807	0.78333	0.92431	0.61693	0.98077	0.43937
0.10	0.73129	0.87814	0.65962	0.84892	0.81197	0.86696	0.67769	0.95772	0.54103
0.15	0.76001	0.88248	0.70025	0.83655	0.83302	0.83827	0.71800	0.94003	0.60963
0.20	0.77898	0.88527	0.72711	0.82871	0.85104	0.81781	0.74922	0.92461	0.66362
0.25	0.79261	0.88706	0.74651	0.82369	0.86681	0.80264	0.77444	0.91098	0.70781
0.30	0.80392	**0.88811**	0.76283	0.82007	0.88173	0.78997	0.79709	0.89767	0.74801
0.35	0.80552	0.88510	0.76667	0.81105	0.89017	0.77244	0.80815	0.88339	0.77143
0.40	0.80632	0.88136	0.76969	0.80261	0.89865	0.75574	0.81869	0.86821	0.79452
0.45	**0.80704**	0.87830	0.77227	0.79681	0.90617	0.74345	0.82756	0.85579	0.81379
0.50	0.80407	0.87279	0.77053	0.78799	0.91140	0.72776	0.83266	0.84102	0.82857
0.55	0.80183	0.86689	0.77008	0.77938	0.91900	0.71124	0.84041	0.82382	0.84850
0.60	0.80033	0.86136	0.77055	0.77278	0.92787	0.69709	0.84919	0.80703	0.86977
0.65	0.79390	0.85246	0.76531	0.76320	0.93344	0.68011	0.85226	0.78780	0.88372
0.70	0.78200	0.83788	0.75473	0.74900	0.93832	0.65661	0.85214	0.76053	0.89684
0.75	0.77306	0.82539	0.74752	0.73855	0.94546	0.63757	0.85473	0.73611	0.91262
0.80	0.75974	0.80832	0.73603	0.72584	0.94978	0.61655	0.85256	0.70771	0.92326
0.85	0.74418	0.78713	0.72322	0.71242	0.95552	0.59377	0.85009	0.67395	0.93605
0.90	0.72225	0.75685	0.70536	0.69532	0.96032	0.56599	0.84355	0.63021	0.94767
0.95	0.68708	0.70594	0.67787	0.67135	0.96751	0.52681	0.83116	0.56246	0.96229

como predictor de noticias. Para reafirmar esta hipótesis y ponerla en mayor contexto se hicieron otros entrenamientos. En la tabla 7.6 se muesntran los entrenamientos que se realizaron para el corpus de *Validación* con diversos métodos:

- Base mayoritaria - Se predicen todos los documentos con la etiqueta más abundante.
- SVM bigramas - Se entrenó una SVM usando los vectores de conteo de bigramas de palabras.
- SVM 141 - Se entrenó una SVM utilizando las 141 características propuestas por Horne et al. (2018b).

La primera cosa que salta a la vista es que la exactitud de la SVM con bigramas y la base mayoritaria son idénticas, esto no es casualidad, en efecto la SVM de bigramas

Tabla 7.5: Resultados del barrido para el segundo modelo.

Valor de corte	F1	con	pro	Precisión	con	pro	Exhaust	con	pro
0.05	0.69284	0.87382	0.60452	0.88011	0.78619	0.92595	0.62549	0.98370	0.45066
0.10	0.75351	0.88812	0.68782	0.87373	0.81837	0.90075	0.69311	0.97104	0.55748
0.15	0.77711	0.89333	0.72039	0.86475	0.83548	0.87904	0.72578	0.96001	0.61146
0.20	0.78758	0.89418	0.73555	0.85324	0.84612	0.85672	0.74480	0.94824	0.64551
0.25	0.79950	0.89633	0.75224	0.84747	0.85778	0.84243	0.76499	0.93867	0.68023
0.30	0.80617	0.89685	0.76192	0.84108	0.86655	0.82865	0.77918	0.92950	0.70581
0.35	0.81586	**0.89862**	0.77546	0.83675	0.87880	0.81623	0.79834	0.91952	0.73920
0.40	0.81774	0.89683	0.77914	0.82862	0.88622	0.80051	0.80852	0.90799	0.75997
0.45	**0.81900**	0.89459	0.78210	0.82026	0.89439	0.78408	0.81896	0.89525	0.78173
0.50	0.81675	0.89012	0.78094	0.80989	0.90050	0.76566	0.82564	0.88056	0.79884
0.55	0.81177	0.88325	0.77689	0.79726	0.90595	0.74421	0.83051	0.86239	0.81495
0.60	0.80604	0.87621	0.77179	0.78624	0.91006	0.72581	0.83320	0.84575	0.82708
0.65	0.80080	0.86838	0.76782	0.77591	0.91593	0.70757	0.83780	0.82676	0.84319
0.70	0.79216	0.85770	0.76017	0.76256	0.92129	0.68509	0.84020	0.80380	0.85797
0.75	0.78058	0.84374	0.74975	0.74821	0.92533	0.66177	0.83981	0.77742	0.87027
0.80	0.76798	0.82672	0.73931	0.73428	0.93150	0.63803	0.84051	0.74618	0.88654
0.85	0.75641	0.80918	0.73066	0.72268	0.94012	0.61656	0.84311	0.71428	0.90598
0.90	0.73723	0.77992	0.71639	0.70709	0.95110	0.58801	0.84328	0.66681	0.92940
0.95	0.69904	0.72283	0.68744	0.68070	0.96140	0.54370	0.83255	0.58850	0.95166

terminó prediciendo todos los artículos como la clase mayoritaria. En segundo lugar el entrenamiento con SVM 141, que pretende replicar el método reportado por Baly et al. (2018a), obtuvo una exactitud muy por debajo de la reportada por los autores. Se tienen dos hipótesis para explicar este resultado. La diferencia puede estar en la selección y limpieza del corpus, tanto en la extracción con el *crawler* como en el tratamiento que se le pudo dar antes de ser trasformados a las 141 características. La segunda hipótesis es que el entrenamiento que se hizo no es el mismo utilizado por Baly et al. (2018a), puede que los hiperparámetros de la SVM tengan un peso imperante en el entrenamiento o que falte algún paso clave que no fue descrito en el artículo.

La segunda prueba fue completamente análoga a la primera pero utilizando los datos de noticias de ciencia. Con estos se obtuvo un valor promedio de exactitud

de 0.52414, más bajo que el 0.6279 de la prueba de comparación con una diferencia notable mayor a 0.1. Ambos corpus se construyeron de la misma manera, usando el mismo *crawler* para extraerlo, selección y división de los datos, y se está resolviendo la misma tarea de clasificación, con las mismas tres clases, usando el mismo modelo. Entonces, se puede concluir que las noticias de ciencia son un corpus más complicada que las noticias políticas, por lo menos para identificar si los hechos reportados en ellas son confiables, que es la tarea para la cual ambos modelos fueron entrenados. Esto no necesariamente implica que en general el corpus de ciencia es más complejo que el de tendencia política, puede que simplemente los textos políticos sean mucho más evidentes al hacer citas falsas que los de divulgación de ciencia.

Finalmente en las tablas 7.3, 7.4 y 7.5 se analiza a mucho mayor detalle los resultados del entrenamiento principal. Además en la tabla 7.6 se compara la exactitud obtenida con ULMFiT con los tres métodos base que se usaron para comparar el corpus de *Validación*. Los resultados son muy similares a los de *Validación* con la SVM con bigramas igualando la base mayoritaria y la SVM 141 desempeñándose por debajo de los otros métodos.

Los resultados de exactitud son bastante buenos con un **0.84595** para dos clases, esto indica que, en promedio, los modelos aciertan a la clase correcta el 85 % de las veces. Esto está bastante por encima del azar y compite con otros sistemas automáticos de noticias como Janze and Risius (2017) donde reportan una exactitud de **0.8087**. Como las dos clases *pro-science* y *consp-pseudo* no están balanceadas, con una discrepancia mayor a 2 a 1, en las tablas 7.4 y 7.5 se hace un análisis de los modelos por clase calculando su precisión, exhaustividad y F1 las cuales luego se promedian de forma ponderada $PromPond = V_0 * \frac{V_1}{C_0+C_1} + V_1 \frac{V_0}{C_0+C_1}$ donde V_n es el valor para la clase n y C_n el número de ejemplos para la clase n. Con esto se logra corregir la desproporción de ejemplos de cada clase y tener una idea más clara del desempeño global de los modelos. Conjunto a esto se hace un barrido del umbral de clasificación tanto para buscar un valor óptimo como para entender más a fondo el desempeño de los modelos. El barrido con dos umbrales se realizó pero, además de ser extremadamente largo y difícil de representar de forma legible en tablas, en todos

los casos se maximizaban el valor F1 cuando el umbral era lo más pequeño posible, indicado que su inclusión era inútil.

Al final ambos modelos resultaron mucho mejores identificando las noticias falsas y obtuvieron en todos los pasos del barrido un valor F1 mayor que las verdaderas. Además, parecen ser bastante robustos; la exhaustividad (el porcentaje de aciertos al predecir una clase) se mantiene alta (arriba de 0.55 para las noticias falsas y 0.4 para las verdaderas) incluso en los extremos del barrido, con un umbral de 0.95 y 0.05, indicando que casi el 50 % de los artículos se clasifican correctamente con una confianza superior al 0.95.

El umbral ideal para una clasificación balanceada de ambas clases parece ser en 0.45, es decir con una ligera tendencia a predecir a favor de *pro-science*, con un valor F1 promedio de *0.81302* identificando en promedio 0.87552 de las noticias pseudocientíficas y 0.79776 de las *pro-science*. Para maximizar la identificación de pseudoscience el umbral ideal parece estar entre 0.3 y 0.35 con un valor F1 de **0.89862** precisión de 0.880265 y exhaustividad de 0.908595 (esto es promediando los valores de los umbrales máximos de cada modelo).

Tabla 7.6: Comparación exactitud.

Método	Exactitud pro-con	Exactitud *Validación*
Base mayoritaria	0.67175	0.59929
SVM bigramas	0.67175	0.59929
SVM 141	0.65392	0.58263
ULMFiT	**0.84595**	**0.627965**

Capítulo 8

Conclusiones

Intentó resolver la tarea de clasificación de noticias utilizando el modelo ULMFiT, una arquitectura de redes recurrentes junto con varias técnicas de optimización y regularización, el cual ha dado muy buenos resultados en otros campos de procesamiento de texto. Al compararlo con los métodos tradicionales (extracción manual de características junto con SVM) se obtuvieron resultados muy similares requiriendo una mínima intervención humana.

El proceso de entrenamiento del modelo ULMFiT es más general que los métodos tradicionales. Los corpus utilizados se construyeron casi directamente del sitio fuente, sólo aplicando unas cuantas reglas simples de limpieza, este texto a su vez se alimentó directamente al modelo. Los resultados de comparación en la tabla 7.6 sugieren que este corpus crudo no se desempeña tan bien en otros métodos. Todo esto facilita su utilización así como posibles mejoras o mantenimiento.

Aunque no es claro por qué los intentos de reproducir los resultados de Baly et al. (2018a) dieron exactitudes muy bajas, se puede concluir que ULMFiT tiene un desempeño que compite con los métodos en el estado del arte. Además este método supera en eficiencia a la todos los clasificadores binarios de noticias conocidos.

Este trabajo es innovador, ya que las herramientas similares que se han fabricado anteriormente, en el campo de las noticias y la veracidad, nunca se han enfocado en el tema de ciencia.

Se logró obtener un modelo final al que se le puede alimentar el texto casi crudo

de un sitio de Internet y logra identificar si la información científica del sitio no es confiable en un 88 % de los casos. Este predictor puede servir como un apoyo al público en general, actuando como un primer frente contra la diseminación de pseudociencia en Internet.

Además todos estos resultados se obtuvieron utilizando ULMFiT tal cual es provisto por los autores, es muy posible que con un ajuste de hiperparámetros se podrían obtener todavía mejores predicciones.

Los tres corpus de noticias que se utilizaron en este trabajo: *Pro-Science*, *Conspiracy-Pseudoscience* y *Validación*, junto con todo el código utilizado, se han puesto a disposición del público para que se pueda tanto reproducir todos los resultados de este trabajo como utilizarse en trabajos similares de detección de noticias falsas.

Bibliografía

R. Baly, G. Karadzhov, D. Alexandrov, J. R. Glass, and P. Nakov. Predicting factuality of reporting and bias of news media sources. *CoRR*, abs/1810.01765, 2018a. URL http://arxiv.org/abs/1810.01765.

R. Baly, G. Karadzhov, D. Alexandrov, J. R. Glass, and P. Nakov. Predicting factuality of reporting and bias of news media sources, 2018b. URL https://github.com/ramybaly/News-Media-Reliability.

A. Blake. A new study suggests fake news might have won donald trump the 2016 election, 2018. URL https://www.washingtonpost.com/news/the-fix/wp/2018/04/03/a-new-study-suggests-fake-news-might-have-won-donald-trump-the-2016-election/?noredirect=onutm_term=.52b2510b25c4/.

J. Bohannon. I fooled millions into thinking chocolate helps weight loss. here's how, 2015. URL https://io9.gizmodo.com/i-fooled-millions-into-thinking-chocolate-helps-weight-1707251800.

Y. Chen, N. Conroy, and V. Rubin. News in an online world: The need for an "automatic crap detector". volume 6-10, 10 2015.

N. Conroy, V. Rubin, and Y. Chen. Automatic deception detection: Methods for finding fake news. 10 2015.

B. C. Csáji. Approximation with artificial neural networks. *Faculty of Sciences; Eötvös Loránd University, Hungary*, 2001.

K. Douglas. Secrets and lies: The psychology of conspiracy theories, 2017. URL `http://sciencenordic.com/secrets-and-lies-psychology-conspiracy-theories/`.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*.

J. Gottfried and E. Shearer. News use across social media platforms 2016, 2016. URL `http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/`.

S. Hochreiter and J. Schmidhuber. Long short-term memory. 1997.

B. Horne, W. Dron, S. Khedr, and S. Adali. Assessing the news landscape: A multi-module toolkit for evaluating the credibility of news. pages 235–238, 04 2018a. doi: 10.1145/3184558.3186987.

B. D. Horne, W. Dron, S. Khedr, and S. Adali. Sampling the news producers: A large news and feature data set for the study of the complex media landscape. *CoRR*, abs/1803.10124, 2018b. URL `http://arxiv.org/abs/1803.10124`.

J. Howard and S. Ruder. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146, 2018. URL `http://arxiv.org/abs/1801.06146`.

W. Howell. Digital wildfires in a hyperconnected world., 2013.

C. Janze and M. Risius. Automatic detection of fake news on social media platforms. 07 2017.

E. Lobato, J. Mendoza, V. Sims, and M. Chin. Examining the relationship between conspiracy theories, paranormal beliefs, and pseudoscience acceptance among a university population. *Applied Cognitive Psychology*, 28, 09 2014. doi: 10.1002/acp.3042.

D. M. J. Lazer, M. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, M. Schudson, S. Sloman, C. Sunstein, E. A. Thorson, D. J. Watts, and J. L. Zittrain. The science of fake news. *Science*, 359:1094–1096, 03 2018. doi: 10.1126/science.aao2998.

MediabiasFactcheck.com. Media bias factcheck. URL `https://mediabiasfactcheck.com/`.

S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182, 2017. URL `http://arxiv.org/abs/1708.02182`.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. 2013.

A. Mitchell and D. Page. State of the news media, 2015. URL `http://www.journalism.org/2015/04/29/state-of-the-news-media-2015/`.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

J. P. Moore. The dangers of denying hiv. *Nature*, 459, 2009. URL `https://doi.org/10.1038/459168a`.

C. Olah. Understanding lstm networks, 2015. URL `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

G.-A. H. S. G. E. J. J. M. Posadas-Durán, Juan-Pabloa. Detection of fake news in a new corpus for the spanish language. *Journal of Intelligent Fuzzy Systems*, 36, 5 2019. doi: 10.3233/JIFS-179034.

V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea. Automatic detection of fake news. 08 2017.

V. Rubin, Y. Chen, and N. Conroy. Deception detection for news: Three types of fakes. 11 2015.

A. Shvets. A method of automatic detection of pseudoscientific publications. pages 533–539, 2015.

T. Wilner. We can probably measure media bias. but do we want to?, 2018. URL `https://www.cjr.org/innovations/measure-media-bias-partisan.php`.

J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017. URL `http://arxiv.org/abs/1703.10593`.