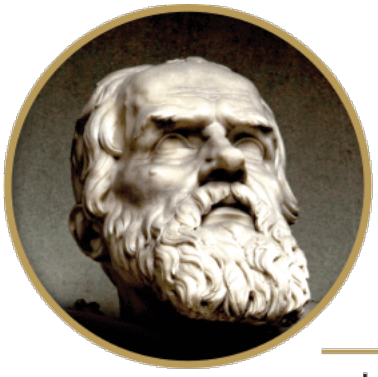
José Jacobo Del Valle Girón

Redes neuronales y transferencia de aprendizaje aplicado a la tarea de perfilamiento de autores de textos anónimos

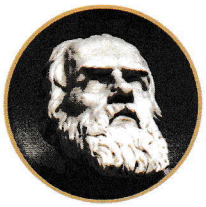


Galileo
UNIVERSIDAD
La Revolución en la Educación

Universidad Galileo

Facultad de Ingeniería de Sistemas,
Informática y Ciencias de la Computación

Guatemala, 2019



Galileo
UNIVERSIDAD
La Revolución en la Educación

Guatemala, 22 de julio de 2019

Señor
José Jacobo Del Valle Girón
Presente

Estimado señor Del Valle:

Tengo mucho gusto en informarle que después de tener a la vista la carta del Director de la Carrera quien, indica haber revisado el trabajo de Tesis, cuyo título es ***"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"*** y de haber obtenido el dictamen del asesor específico; el Ingeniero Ronald Israel López España, autorizo la publicación del mismo.

Aprovecho la oportunidad para felicitarlo por el magnífico trabajo realizado, el cual es de indiscutible beneficio para el desarrollo de las Ciencias de la Computación en Guatemala.

Atentamente,

FACULTAD DE INGENIERIA DE SISTEMAS,
INFORMATICA Y CIENCIAS DE LA COMPUTACION

Ing. Rodrigo Baessa
Decano

ril

Guatemala, 17 de julio de 2019

Ing. Rodrigo Baessa Iunge
Decano de FISICC
Universidad Galileo

Estimado Ingeniero,

Tengo el gusto de informarle que he tenido a la vista el trabajo de tesis titulado ***"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"*** elaborado por del estudiante **José Jacobo Del Valle Girón** quién se identifica con carnet número 10002079, el cual fue asesorado por del Ingeniero Ronald Israel López España colegiado número 5634, por tanto doy por *Aprobado* que se inicie Trámite Final de Tesis de la carrera de Ingeniería de Sistemas, Informática y Ciencias de la Computación

Atentamente,

FACULTAD DE INGENIERIA DE SISTEMAS,
INFORMATICA Y CIENCIAS DE LA COMPUTACION

Ing. Jack Trachtenberg
Director de Ingeniería de Sistemas,
Informática y Ciencias de la Computación

ril

Ciudad de Guatemala, 09 de julio de 2019.

Ingeniero
Rodrigo Baessa
Decano FISICC
Universidad Galileo
Presente.

Señor Decano:

Le informo que la tesis **"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"**, del estudiante José Jacobo Del Valle Girón, ha sido objeto de revisión gramatical y estilística, por lo que puede continuar con el trámite de graduación.

Atentamente.

Lic. Edgar Lizardo Porres Velásquez
Asesor Lingüístico
Universidad Galileo

Ronald Israel López España
Ingeniero de Sistemas Informática y Ciencias de Computación

Guatemala, 24 de junio de 2015

Ingeniero José Eduardo Suger Castillo
Decano
Facultad de Ingeniería en Sistemas, Informática y Ciencias de la Computación
Universidad Galileo

Estimado Ingeniero Suger,

Me dirijo a usted para informarle que he asesorado el trabajo de Tesis titulado "**REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS**" elaborado por el estudiante José Jacobo Del Valle Girón, carné 10002079.

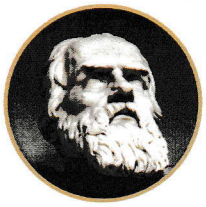
El estudiante elaboró un trabajo qué será de gran utilidad en el área de Ciencias de la Computación. Por ello me complace informarle que doy por **APROBADO** el contenido de este trabajo, el cual someto a su consideración.

Atentamente,

Ing. Ronald Israel López España

Ronald Israel López España
Ingeniero de Sistemas
Colegiado No. 5634

rilopez@galileo.edu



Guatemala, 3 de mayo de 2019

Señor
José Jacobo Del Valle Girón
Presente

Estimado señor Del Valle

Tengo mucho gusto en informarle que ha sido aprobado su punto de Tesis, previo a optar al diploma de **Ingeniero de Sistemas, Informática y Ciencias de la Computación** cuyo título es **"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"**.

Al mismo tiempo le informo que ha sido aprobada la designación del Ingeniero Ronald Israel López España para asesorar su trabajo de graduación.

Atentamente,

**FACULTAD DE INGENIERIA DE SISTEMAS,
INFORMATICA Y CIENCIAS DE LA COMPUTACION**

Ing. Rodrigo Baessa Iunge
Decano

ril

Guatemala, 3 de mayo de 2019

Ingeniero
Ronald Israel López España
Presente

Estimado Ingeniero López:

Tengo mucho gusto en informarle que ha sido aprobada su designación como Asesor del trabajo de Tesis del alumno José Jacobo Del Valle Girón, previo a optar al diploma de **Ingeniero de Sistemas, Informática y Ciencias de la Computación** cuyo título es **"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"**.

Para su información adjunto a la presente, copia de la carta de solicitud y respuesta al señor Del Valle.

Atentamente,

**FACULTAD DE INGENIERIA DE SISTEMAS,
INFORMATICA Y CIENCIAS DE LA COMPUTACION**

Ing. Rodrigo Baessa Iunge
Decano

ril

Guatemala, 8 de abril de 2019.

Ingeniero
Rodrigo Baessa
Decano FISICC
Universidad Galileo
Presente

Estimado Ing. Baessa:

Le saludo cordialmente y deseándole éxitos en sus labores diarias. El motivo de la presente es solicitarle la aprobación de punto de tesis previo a optar al título de ingeniero en sistemas, informática, y ciencias de la computación.

El punto de tesis es **"REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS"**, la cual cubre los siguientes temas:

1. Trasfondo del aprendizaje profundo (Deep Learning)
2. Transferencia de aprendizaje en redes neuronales en procesamiento de lenguajes naturales
3. Pre-entrenamiento de modelo de lenguaje
4. Entrenamiento de clasificadores usando transferencia de aprendizaje para la tarea de perfilamiento de autores

Asimismo, solicito la designación del Ing. Ronald Israel López España, colegiado No. 5634 como asesor del trabajo de tesis.

Agradeciendo su atención y en espera de una respuesta positiva me despido cordialmente,

Jacobo Del Valle
10002079
5765-3829
delvallejacobo@galileo.edu

Vo.Bo.

Guatemala, 8 de abril de 2019

Ingeniero Rodrigo Baessa Iunge
Decano de FISICC
Universidad Galileo

Estimado Ingeniero Baessa,

Me dirijo a usted para informarle que, a requerimiento de José Jacobo Del Valle Girón quien es estudiante de Ingeniería de Sistemas, Informática y Ciencias de la Computación, estaré asesorándole en la elaboración de su trabajo de tesis titulado "**REDES NEURONALES Y TRANSFERENCIA DE APRENDIZAJE APLICADO A LA TAREA DE PERFILAMIENTO DE AUTORES DE TEXTOS ANÓNIMOS**".

Atentamente,

Ing. Ronald Israel López España

Ronald Israel López España
Ingeniero de Sistemas
Colegiado No. 5634

Resumen

Redes neuronales y transferencia de aprendizaje aplicado a la tarea de perfilamiento de autores de textos anónimos

por José Jacobo DEL VALLE GIRÓN

En tiempos de la expansión del campo de *Deep Learning* (aprendizaje profundo), es sorprendente cómo las nuevas contribuciones hacen que sus áreas de aplicación sean cada vez más. Esto se debe al crecimiento exponencial de datos disponibles en esta época de la internet, que es relevante para el aprendizaje profundo, ya que para tener un modelo eficaz es necesario una cantidad masiva de datos. Algunas áreas no cuentan con la cantidad de datos que se necesitan, por lo que en la actualidad se utilizan aún técnicas de *Machine Learning* tradicional para obtener los resultados deseados. Una de estas áreas es el procesamiento de lenguaje natural (NLP por sus siglas en inglés).

Este trabajo presenta el perfilamiento de autores como una tarea que puede ser abordada con Deep Learning y aplicarse el concepto de *transfer learning* en el área de NLP de manera exitosa. Este concepto permite realizar tareas con menos datos de lo antes asumido.

Los resultados obtenidos con este proyecto alcanzan niveles del estado del arte de técnicas de Deep Learning y son comparables con resultados obtenidos al utilizar ambas técnicas tradicionales y de Deep Learning.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Tema principal

Redes neuronales y transferencia de aprendizaje aplicado a la tarea de perfilamiento de autores de textos anónimos.

1.2. Introducción

Los campos de *machine learning* (aprendizaje de máquinas o aprendizaje artificial) y *deep learning* (aprendizaje profundo) (LeCun, Bengio & Hinton, 2015) no son nuevos, pero se han vuelto gran sensación en los últimos años, cada vez ganando más popularidad. Los primeros conceptos desarrollados en el campo tienen décadas de antigüedad. La primer red neuronal — la base del modelo a utilizar en este trabajo y la base del campo de Deep Learning — fue planteada en los años 80 (Werbos, 1982). ¿Qué ha cambiado, entonces, en los últimos años, que ha generado una explosión en el área? La disponibilidad de los datos y la accesibilidad a poder computacional (Jordan & Mitchell, 2015). Con el surgimiento y la penetración cada vez más alta del internet se han abierto las puertas a cantidad de datos nunca antes vista.

La revolución de los datos ha llevado a la comunidad científica y a las industrias a recolectar grandes cantidades de datos e información y organizarlos de forma que se puedan interpretar y obtener valor de la misma.

Los sistemas de Deep Learning necesitan pasar por una fase de aprendizaje (comúnmente llamada fase de entrenamiento) y estos modelos tienen la característica de necesitar cantidades masivas de datos para generalizar bien lo aprendido. Esto, combinado con la explosión de datos, ha llevado a que las técnicas tengan un surgimiento en todas las áreas de software, mostrando mucha promesa en algunos campos en específico como Computer Vision (CV) (Hoo-Chang y col., 2016) y en Natural Language Processing (NLP).

Conforme ha avanzado el campo se ha determinado que para algunas tareas en específico no se cuenta con la cantidad necesaria para obtener un modelo competente. Esto ha llevado al desarrollo de una técnica llamada transfer learning, que permite utilizar conocimiento previamente obtenido al resolver una tarea general y aplicarlo a una tarea relacionada mediante proceso de ajuste. Esta fase requiere de una cantidad de datos considerablemente menor, comparado con tener que entrenar un modelo completamente desde cero.

La técnica de transfer learning ha permitido muchos avances en el área de CV específicamente (Hoo-Chang y col., 2016). Este trabajo intenta mostrar que se puede aplicar este concepto en un área distinta, puntualmente el campo de NLP y específicamente en el proceso de perfilamiento de autores.

1.3. Hipótesis

La aplicación de transfer learning en la tarea de perfilamiento de autores provee una ventaja en desempeño medible. Se desglosa en los siguientes

aspectos:

- **Aplicabilidad de deep learning en la tarea**. La hipótesis se basa en el hecho de que conceptos de deep learning sean aplicables a tareas del campo de NLP. En el campo de NLP se ha tenido mucha duda si deep learning tiene la capacidad de penetrar el campo con soluciones eficaces.
- **Transfer learning provee ventaja de desempeño en la tarea**. El hecho de mostrar que hay una ventaja, implica medir de forma empírica los resultados obtenidos por diferentes técnicas. Con este fin se utilizarán resultados publicados por investigadores de este campo y se usarán las mismas métricas para que los resultados sean directamente comparables.
- **Diferencia en cantidad de datos disponibles**. Se debe fundamentar que la cantidad de datos necesaria para el uso de deep learning es más elevado del que se necesita si se utiliza transfer learning.

1.4. Objetivo General

Aplicar conceptos del estado del arte a un problema que ha recibido poca atención en el área de Natural Language Processing (NLP), por sus siglas en inglés). Lo anterior con el propósito de mostrar la capacidad de las redes neuronales recurrentes (Recurrent Neural Networks (RNNs)) y *transfer learning* en la tarea específica.

1.4.1. Objetivos específicos

- Ser pionero en la aplicación de transfer learning en la tarea específica de perfilamiento de autores en el área de procesamiento de lenguaje natural.

- Exponer las ventajas y desventajas de utilizar esta tecnología en esta sub-tarea.
- Exponer como caso de uso la utilidad de esta aplicación en distintas áreas, como en el mercadeo.
- Contribuir a la academia con los resultados obtenidos.

1.5. Planteamiento del problema

En el área de NLP han existido técnicas poderosas por décadas que abusan de algunas suposiciones que no siempre reflejan la realidad y se basan en simples métodos estadísticos que asignan valores a características del texto (Edmundson, 1969; Kupiec, Pedersen & Chen, 1995). El campo también ha dependido de ingeniería de características en los datos, que significa que se realiza un esfuerzo activo y serio en analizar cada sub-tarea a fondo para determinar cuáles son los mejores determinantes de un resultado deseado, p.e. las características principales que determinan la categoría que se le debe asignar a un texto (Aggarwal & Zhai, 2012).

En campos como el mercadeo y las ciencias forenses hay una necesidad de identificar textos escritos a los cuales no se les ha atribuido un autor y obtener características de los mismos para poder catalogarlos de forma automática. Esto con el fin de poder, en el caso del mercadeo, orientar mejor la publicidad de ciertos productos a una audiencia apropiada (Aggarwal & Zhai, 2012); y, en el área forense, identificar a sospechosos vinculados a algún crimen a partir de un texto anónimo obtenido en el contexto del caso.

Existe fundamento lingüístico del hecho de identificar a una persona por su lenguaje, ya sea de manera escrita u oral (Coulthard, 2004; Louwerse, 2004). El

concepto de *idiolecto* y *sociolecto* nos permitirá realizar predicciones de manera precisa.

Después de analizar lo planteado anteriormente, se ve la necesidad de que un sistema automatizado sea capaz de no sólo extraer información de un texto anónimo de forma precisa y confiable, sino también hacerlo sin necesidad de un agente experto o realizar una ingeniería de características claves para cada posible parámetro que se quiera obtener de los textos.

A partir de esta conclusión, los requisitos para este proyecto son los siguientes:

- Permitir la categorización de textos a partir de diferentes características.
- Realizar el entrenamiento para obtener los datos de forma rápida y con demandas de hardware modestas.
- Una vez el entrenamiento esté completo, realizar predicciones sobre textos anónimos con los datos a obtener elegidos.
- Permitir la automatización de predicciones al ofrecer un servicio web fácil de consultar.

La figura 1.1 muestra la estructura del proyecto planteado, que llenaría los requisitos expuestos en esta sección. El flujo de la información tendría la forma ilustrada en la figura y las predicciones mostradas son ejemplos de categorizaciones.

1.6. Resumen

El proyecto deberá tener la flexibilidad de adaptarse a las necesidades del caso de uso específico que se le dé y a la vez tener la capacidad predictiva de un

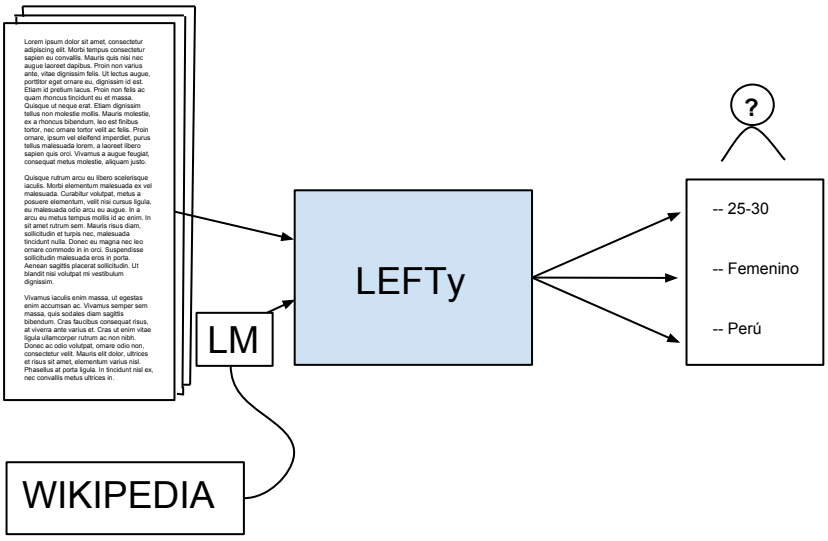


FIGURA 1.1: Estructura del proyecto planteado en este trabajo de investigación. La figura está diseñada para propósitos ilustrativos. *LM* denota el modelo de lenguaje que es pre-entrenado con texto de *Wikipedia* y es un paso previo.

modelo del estado del arte. El modelo utilizará tecnologías de Deep Learning, por ser la tecnología más prometedora en este campo.

Capítulo 2

Marco teórico

2.1. Trasfondo de Deep Learning

En este capítulo se planteará y explicará el fundamento teórico de las técnicas del campo de Deep Learning utilizados en este trabajo. Los fundamentos geométricos teóricos (Lei, Luo, Yau & Xianfeng Gu, 2018) del porqué de la funcionalidad básica de los modelos del campo, no se abordarán en esta tesis.

2.2. Aprendizaje supervisado

En el área de aprendizaje (profundo) artificial existen dos tipos básicos de problema. Esto se determina dependiendo de las características de los datos a utilizar para el entrenamiento del modelo (Schmidhuber, 2015).

Cuando los datos — ya sea información tabulada, imágenes, textos — no tienen ninguna categoría asociada o ningún valor a predecir y lo que se desea es obtener información no específica, es decir sin tener alguna referencia, se trata de un aprendizaje no supervisado.

Si los datos, por otro lado, tienen una clasificación — llamada etiqueta — asignada, que a futuro es el resultado a predecir, los métodos a utilizar son los del aprendizaje supervisado. Debido a que el problema a abordar en este trabajo es un problema de clasificación, el resto del fundamento teórico será basado en este tipo de contexto.

El aprendizaje supervisado puede entonces ser descrito como una función $f : X \rightarrow Y$ donde X representa los datos con los que se alimenta la función, es decir las características que utilizará el modelo para inferir, y Y el resultado asignado o a predecir.

Para ilustrar el proceso de Deep Learning se utilizará un caso individual de la función en donde y es el resultado deseado, $f(x) = \hat{y}$ es la función aplicada a un caso específico y $\hat{y}$ representa el resultado obtenido con la función que no necesariamente es el resultado deseado o esperado.

El objetivo. En concreto, buscamos una función f que sea la mejor candidata para predecir los resultados deseados. Si definimos una función de costo $L(\hat{y}, y)$ que representa, en un valor escalar, la diferencia cuantitativa entre la evaluación de una función f candidata y el resultado real y podemos concluir que el objetivo es encontrar una f^* que cumpla con:

$$f^* = \min_{f \in F} \frac{1}{N} \sum_{i=0}^{N} L(f(x_i), y_i)$$

Donde N es el número de instancias de los datos para entrenar el modelo; y F es un conjunto de funciones candidatas.

Definición de las funciones. La base de todo modelo de Deep Learning

es una red neuronal — cuyo comportamiento será definido en la siguiente sección así como otros detalles relevantes — y su comportamiento básico puede ser descrito de la siguiente forma:

$$f(x_i) = wx_i + b$$

Donde x_i es la instancia de datos i con propósitos de entreno o de predicción. Esto nos dice que w y b serán los parámetros a modificar de manera sistemática para encontrar la función f^*. Con fines de brevedad, la concatenación de w y b serán representados por θ.

Una posible función de pérdida en un problema de clasificación para una predicción obtenida, toma la forma del error de la entropía cruzada, es decir:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{j}^{C} y_j log\left(\hat{y_j}\right) \tag{2.1}$$

Donde C es el número de clasificaciones independientes posibles, $\mathbf{y}$ es un vector de dimensión C con solamente un único valor en 1 el cual indica la clasificación correcta y $\hat{\mathbf{y}}$ es el vector de predicción de dimensión C, que contiene las probabilidades para cada posible clasificación. De esto podemos concluir que $\sum \hat{\mathbf{y}} = 1$.

Si $C = 2$ se puede simplificar la expresión, utilizar solamente un resultado escalar

$$L(\hat{y}, y) = -ylog\left(\hat{y}\right) - (1-y)\, log\left(1-\hat{y}\right) \tag{2.2}$$

Al tener el error — la pérdida — para una predicción se puede expandir esta idea para obtener la pérdida a través de un conjunto de datos, que resultará

muy útil cuando se deba entrenar. Para obtener una aproximación de la pérdida sobre un conjunto de datos se puede utilizar el promedio sobre las pérdidas individuales de los datos evaluados:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i log\hat{y}_i + (1-y_i)log(1-\hat{y}_i)]$$

Optimización de la función de costo. Con una función a minimizar establecida y una cantidad N de datos sobre los que se debe encontrar una función f, candidata cada vez mejor, se recurre al método del descenso de gradiente. Este método nos permite utilizar propiedades básicas de las derivadas de las funciones y poco a poco avanzar hasta llegar a un valor mínimo de la función. Cada nueva función candidata, entonces, podrá ser derivada de la siguiente forma:

$$f_i(x_i) = \theta_i x_i$$

Donde

$$\theta_{i+1} = \theta_i - \gamma \nabla_\theta L(\hat{y}_i, y_i). \tag{2.3}$$

Este proceso de utilizar el descenso de gradiente a través de las instancias nos permite minimizar el error de la función, hasta deducir la función que muestra el menor error, lo que alcanza nuestro objetivo principal.

La tasa de aprendizaje. La velocidad de convergencia de este proceso dependerá en gran parte de γ, que representa la tasa de aprendizaje; específicamente es la ponderación que se le da al componente del gradiente cuando se propone la nueva función. Un γ muy alto arriesga una divergencia debido a que podría oscilar alrededor de un mínimo sin nunca converger en él. Un γ muy

bajo, por el otro lado, puede resultar en un aprendizaje muy lento, que puede llevar a un resultado no óptimo debido a limitaciones de recursos o tiempo. Este concepto representa mucho valor en este trabajo.

2.3. Redes neuronales

Las redes neuronales son el modelo base para el Deep Learning. Esto debido a que ha sido demostrado que son aproximadores universales (Hornik, Stinchcombe & White, 1989) — es decir que para cualquier función h se tiene la capacidad de encontrar un red neuronal $\hat{H}$ que aproxime a h con cierto grado de precisión — un error de ϵ — dado que se tenga la cantidad de unidades en la red y datos suficientes para entrenar. Formalmente se dice que

$$|\hat{H}(x) - h(x)| < \epsilon \tag{2.4}$$

Con ayuda del concepto de las redes neuronales, especificaremos más acerca de la función f que hasta ahora ha permanecido general, únicamente con la restricción de ser derivable. Hay diferentes tipos de redes neuronales en el campo y en este trabajo se abordarán únicamente las estándar y las recurrentes.

2.3.1. Redes neuronales estándar

Inspiración e intuición. Estas redes neuronales fueron basadas en comportamientos biológicos y reflejan un comportamiento similar a la comunicación de neuronas que se aprecia en la naturaleza. La entrada de datos en una neurona es procesada y alimentará a la siguiente neurona y así, sucesivamente, hasta haber recorrido toda la red. La redes neuronales no son secuencias directas y lineales

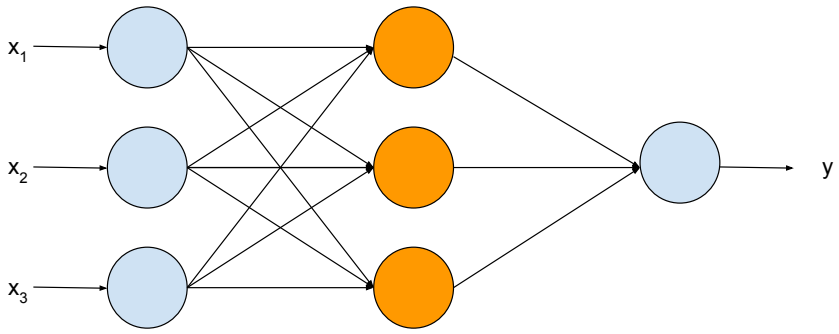


FIGURA 2.1: Una red neuronal estándar con una capa oculta que tiene tres unidades neuronales. La red está completamente conectada, tiene tres nodos de entrada y uno de salida.

de neuronas; las redes están divididas en capas, las que pueden contener más de una unidad. Una red completamente conectada, como la que se aprecia en la figura 2.1, conecta a todas las unidades de una capa con todas las unidades de la siguiente. Así como las neuronas en la naturaleza, las neuronas en una red neuronal tienen una unidad o célula principal, axones y su conexión es llamada sinapsis.

Propagación hacia adelante. En las redes neuronales esto es el proceso del flujo de la información a través de la red, y las funciones que la componen, hasta obtener un resultado. En el caso de una red neuronal típica, este proceso significa que la salida de una neurona en una capa alimenta parcialmente a las de la capa siguiente. La función a utilizar por cada unidad individual es una regresión lineal simple que puede ser descrita como $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$, es decir, una matriz de parámetros a multiplicarse con los datos de entrada a la neurona. Adicional a esto, se maneja una función de activación para el resultado de esa multiplicación, que es no lineal, a cada elemento resultante (p.e. tanh). De esto se concluye que para, por ejemplo, obtener el resultado de una neurona en la cuarta capa se

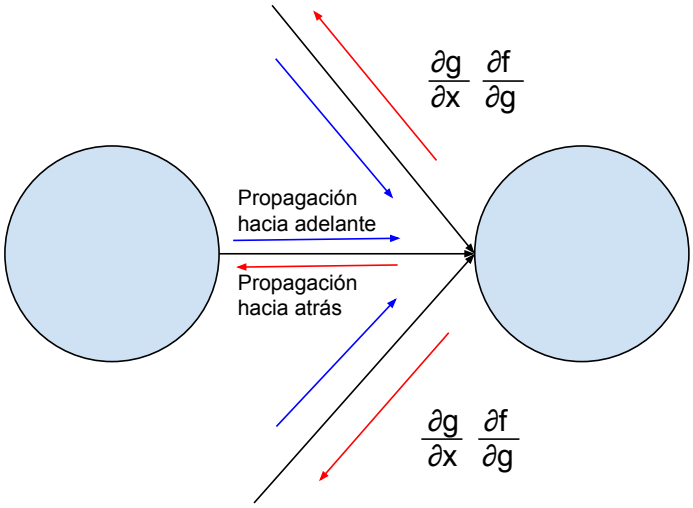


FIGURA 2.2: En azul se puede ver la dirección del flujo de información durante la propagación hacia adelante. En rojo se aprecia que la dirección se invierte para la propagación hacia atrás y que lleva los gradientes necesarios para propagar el error después del entrenamiento.

tiene que

$$f(\mathbf{x}) = \mathbf{W_4} \cdot \sigma(\mathbf{W_3} \cdot \sigma(\mathbf{W_2} \cdot \sigma(\mathbf{W_1} \cdot \mathbf{x}))) \tag{2.5}$$

Donde $\mathbf{W_4}$ es una matriz de parámetros de la cuarta capa, $\mathbf{W_3}$ es una matriz de parámetros de la tercera capa, y así sucesivamente, y $\mathbf{x}$ representa los datos de entrada.

Propagación hacia atrás (Rumelhart, Hinton & Williams, 1986). Ya que está presente un mecanismo para evaluar el conjunto de funciones que representa cada unidad de la red, debemos tener un mecanismo para optimizar los parámetros que definen cada función. La propagación hacia atrás se encarga de esto al utilizar el concepto previamente descrito como el descenso de gradiente. Para realizar la propagación hacia atrás se aplica la regla de la cadena que establece que $\frac{\partial f(g(x))}{\partial x} = \frac{\partial g}{\partial x}\frac{\partial f}{\partial g}$. Se debe notar que la función que se quiere derivar toma una forma similar a la que tenemos en la ecuación 2.5, con la adición de que $g(x)$ es

una función que anida aún más funciones. Para obtener la optimización de los parámetros podemos derivar en dirección hacia atrás propagando la mejora que se propone con el gradiente.

La actualización de la función de cada unidad neuronal se realiza de acuerdo a como fue descrito en la ecuación 2.3.

2.3.2. Redes neuronales recurrentes

Este tipo de redes tienen la peculiaridad que se alimentan no solamente de los resultados de las funciones de activación de las capas anteriores, sino también del resultado de la instancia previamente evaluada; para datos evaluados en en un tiempo t la definición sería $h_t = f_\theta(h_{t-1}, x_t)$. Este tipo de estructura de red neuronal es aplicado a conjuntos de datos secuenciales como es el procesamiento de textos — textos cuya representación consiste en una secuencia de palabras representadas de forma vectorial — y procesamiento de secuencias de señales.

Se debe aclarar para esta estructura de red neuronal también existen distintas extensiones. Estas surgen debido a la inhabilidad de las redes recurrentes típicas de conservar contexto en cadenas de texto largas. El subtipo relevante para este trabajo de investigación es la red Long Short Term Memory (LSTM) – memoria corta a largo plazo.

Esta extensión de las redes LSTM permite tener un estado memorizado por cada unidad de la red. Se definen parámetros, que son aprendidos por la red con el mismo mecanismo de propagación, que definen cuándo se debe guardar parte del contexto, qué parte en específico y finalmente cuándo se deben olvidar.

2.4. Generalización de una red neuronal

Una red neuronal aprende a elegir una función que minimice el costo de evaluar un conjunto de datos. Intuitivamente podemos ver que este proceso lleva a que la red aprenda características de los datos que está utilizando, y poco a poco aprenda a predecir categorías basado en este conjunto de datos de manera más precisa. Las palabras claves son *este conjunto de datos*, es decir se habla de que la red aprende sobre un conjunto limitado de datos y fuera de él no hay garantía de que sea competente. Para que el modelo pueda generalizar lo aprendido con estos datos, existen distintos mecanismos. En esta sección se explicarán dos de estos conceptos.

Cantidad de datos. La cantidad de datos utilizados para entrenar un modelo influye mucho en su capacidad de generalizar. Mientras más datos se tengan, mayor será la posibilidad de generalizar, esto con la condición que la data sea diversa y representativa del problema real. La explicación de esto se puede ilustrar llevando el concepto a sus extremos y con un ejemplo sencillo: se debe suponer que se quiere aprender a definir el conectivo lógico ***and***. Si se provee solamente un ejemplo de cómo funciona este operador, la red no sabrá qué hacer cuando los valores de entrada difieran de ese ejemplo. Por el otro lado si se le alimentan todas las combinaciones posibles, la red deberá ser capaz de aprender todo el contexto del problema.

Término de regularización. Esta técnica es muy esencial cuando se lidia con modelos de Deep Learning. Consisten en agregar un término de regularización a la función de predicción que se está optimizando. La función entonces tendrá la forma siguiente:

$$f^* = \min_{f \in F} \frac{1}{N} \sum_i L(f(x_i), y_i) + \sum_j \lambda(w_j^2)$$

Esto funciona ya que limita el crecimiento de los parámetros, que desenfrenado podría causar que el modelo dependa mucho de una característica de los datos y una dependencia exagerada puede llevar a falta de generalización. En otras palabras, mantiene un balance sobre la ponderación de los parámetros aprendidos. Adicionalmente se incluye el término λ que controla qué tanto afecta el término de regularización a la función de pérdida.

2.5. Proceso general al aplicar una red neuronal

Un proyecto de Deep Learning con redes neuronales lleva por lo general el mismo conjunto de pasos para llegar a un resultado cercano a lo óptimo. Los pasos a seguir son los siguientes:

- **Obtención de datos:** Dependiendo del problema a resolver, estos datos podrán tomar distintas formas y los métodos para obtenerlos podrán variar en gran manera. Para que una red neuronal pueda generalizar de forma exitosa lo aprendido durante la fase de entrenamiento, es importante tener una muestra representativa del escenario real del problema a resolver y tener una cantidad elevada de datos. Métodos comunes incluyen recolección y etiquetación manual, descarga de *corpora* de internet y *scraping* de la web.

- **Análisis y preparación de los datos:** Los datos obtenidos en el primer paso pueden llegar a tener características mínimas no deseadas, que pueden

añadir ruido a la representación que estos datos dan. Debido a esto es importante tratar los datos de manera que se eliminen datos que sesguen mucho los posibles modelos, datos faltantes, datos con formato inconsistente, e incluso considerar la posibildad de eliminar características completas. En esta fase también se separarán los datos en distintos segmentos que después serán útiles para determinar la eficacia del modelo resultante. Estos segmentos son los datos de *entrenamiento*, *validación* y *prueba*. La proporción de cada uno de estos puede variar y lo recomendado es que la distribución de cada uno de ellos sea la misma. De no ser posible esto, hacer que al menos los segmentos de validación y prueba tengan la misma distribución.

- **Diseño de modelo:** En este campo hay gran variedad de opciones, en especial si no se limitan estas opciones al Deep Learning ya que existen herramientas de distintos tipos para modelar un sistema. En el caso del Deep Learning, también se deben tomar decisiones importantes con respecto al modelo a utilizar. En concreto se deberá elegir el tipo de red neuronal como también su estructura. En esta fase se definirán de forma preliminar detalles como el número de capas en la red y el número de unidades por cada una de las capas.

- **Optimización de hiper-parámetros:** Esta fase estará muy relacionada a la siguiente, ya que se utilizan los mismo mecanismos para determinar los resultados preliminares. Para elegir los mejores hiper-parámetros para un modelo en específico se deberá evaluar los datos al variar sus valores y obtener resultados preliminares. Esto se deberá hacer al utilizar una matriz aleatoria para los distintos hiper-parámetros con el fin de optimizar el tiempo de ejecución de esta fase.

- **Entrenamiento del modelo:** En esta fase se optimizarán los parámetros de la red para que sea capaz de modelar el problema y realizar predicciones. En esta fase se aplicarán de forma iterativa las fases de propagación hacia adelante y hacia atrás, hasta llegar a una convergencia. Si se cuenta con tiempo limitado, se deberá elegir un resultado que sea suficientemente satisfactorio y se detendrá el entrenamiento en ese punto.
- **Validación de resultados:** Durante esta fase se hará uso de los datos segmentados para validación. Sobre este conjunto se aplicará el modelo en su modalidad de propagación hacia adelante únicamente, con lo que se obtienen predicciones. Estas se pueden comparar con las etiquetas asociadas respectivamente. El fin de evaluar el modelo en datos que no están presentes en el conjunto de entrenamiento, es determinar el poder de generalización del modelo.

Los pasos de entrenamiento y validación de resultados podrán repetirse las veces que sean necesarias para variar los parámetros y evaluar los tipos de errores para mejorar los resultados.

2.6. Definiciones varias

En esta sección se definen conceptos varios que son mencionados y utilizados en este trabajo. Las explicaciones varían en su profundidad dependiendo de la importancia y relevancia del concepto en el trabajo.

Época. En inglés llamada *epoch*, es el nombre que se le da al proceso de iterar sobre todas las instancias de entrenamiento, tanto en propagación hacia adelante como hacia atrás. Generalmente un modelo se itera sobre varias épocas a través de los datos, una única vista a cada instancia de datos puede no ser

suficiente para que el modelo aprenda las características que darán los mejores resultados.

Tipos de errores en clasificación. Cuando se clasifica datos en clases, existen dos tipos de errores posibles. Un falso positivo es un error tipo I, es decir se rechaza una hipótesis nula verdadera. Un error de tipo II, por el contrario, es un falso negativo, es decir no lograr rechazar una hipótesis nula falsa.

Desequilibrio de clases. Cuando en un problema de clasificación uno de los resultados es mucho más común que el resto se dice que hay un problema de desequilibrio de clases. Esto puede llevar a complicaciones en la generalización del modelo o en la evaluación del desempeño del mismo.

Matriz de confusión. En problemas de clasificación se tienen distintas medidas de las que la más común es la precisión del modelo. Esta medida puede no llegar a ser muy representativa del desempeño del modelo. Una situación en donde esto se podría manifestar es cuando los datos que representan el problema tienen un problema de desequilibrio de clases. Si la tasa de desequilibrio es demasiado alta, es posible predecir únicamente una clase con un modelo y obtener una precisión del modelo bastante elevada, a pesar de que el modelo no está prediciendo nada. Debido a esto, es importante tener en cuenta más herramientas de retroalimentación. Algunos ejemplos de ellas son:

- Exhaustividad
- Métrica de F_β
- Curvas ROC

	Referencia	
Predicción	0	1
0	**1507**	19
1	34	**1201**

CUADRO 2.1: Ejemplo de una matriz de confusión que muestra distintos tipos de errores. La tabla tiene únicamente propósitos ilustrativos. Los números resaltados representan la cantidad de predicciones correctas; las otras dos cifras serán los errores tipo I y tipo II.

Capítulo 3

Transferencia de aprendizaje

En este capítulo se cubren y se fundamentan las técnicas de transferencia de aprendizaje (transfer learning) usadas y las decisiones tomadas para el diseño del proyecto de clasificación de autores con redes neuronales recurrentes.

3.1. Transferencia de aprendizaje

Un concepto esencial para este trabajo y para toda el área de Deep Learning, es la capacidad de transferir aprendizaje adquirido previamente de un problema poco acotado y aprovechar este aprendizaje en problemas más acotados, al permitir así que las demandas de recursos como datos y tiempo de ejecución tengan más holgura. Se obtienen mejores resultados en menos tiempo al transferir aprendizaje de un modelo que cuando se inicializan los pesos de los parámetros de la red aleatoriamente (Erhan y col., 2010).

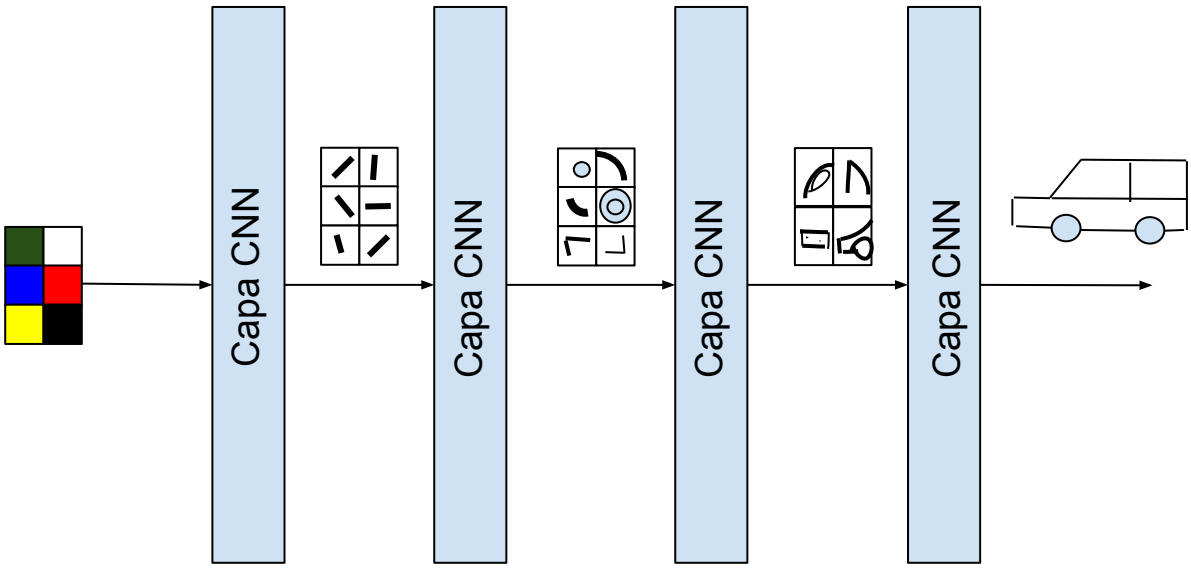


FIGURA 3.1: Ejemplo ilustrativo de lo que podría aprender una red profunda de visión artificial para la detección de vehículos.

3.2. Concepto de transferencia

Las redes neuronales capturan diferente información en sus capas (Yosinski, Clune, Bengio & Lipson, 2014; Zeiler & Fergus, 2014) y la forma en que difieren permite transferir conocimiento a través de redes entrenadas para diferentes tareas. En las primeras capas se aprende información bastante abstracta con respecto a la tarea, y conforme se avanza en la profundidad de la red se vuelve más específica la información que se va aprendiendo. Debido a esto se puede utilizar lo aprendido en las primeras capas para otra tarea del mismo dominio.

Ejemplo ilustrativo. En una red neuronal, cuya tarea es en el área de visión artificial las primeras capas comienzan a entender conceptos de orillas horizontales o verticales; las siguientes capas podrán entender qué son esquinas circunferencias; las siguientes capas podrán entender límites de objetos; etc.

La información en las primeras capas de una red neuronal de visión artificial puede llegar a ser muy útil, sin importar la tarea final o qué tan acotada esté. Por esta razón es posible tomar un modelo pre-entrenado en una tarea general, remover su capa final — la que determina el resultado final de esa tarea en específico — y continuar con el proceso de entrenamiento con datos de la tarea más acotada y permitir que las nuevas capas no entrenadas aprovechen los conceptos de las capas anteriores para que ellas tengan el poder de predecir al usar esa información y los resultados correctos.

3.3. Aplicación en CV

Para tener el contexto acerca de este concepto crucial para este trabajo debemos explorar otras áreas del Deep Learning, en particular el área de visión artificial (computer vision en inglés). Este campo fue el que llevó a la explosión de Deep Learning en el 2012, cuando una red denominada AlexNet dominó la competencia de clasificar el dataset ImageNet (Deng y col., 2009).

ImageNet. Este proyecto es una colección masiva de imágenes que están debidamente etiquetadas con cuadros encerrando los objetos detectados en una imagen. Debido a la cantidad exagerada de categorías y objetos encontrados en las imágenes y también la cantidad en sí de imágenes, se organiza un concurso de software todos los años para determinar cuál modelo es el mejor en visión artificial. Cuando en el 2012 una red de Deep Learning ganó el concurso se generó motivación en explorar este campo más a fondo.

Esta tarea llegó a ser el estándar para determinar si un modelo podía *ver* a nivel general. ¿Qué significa esto? La tarea de ver es una tarea muy basta y

bastante general. Hay muchas características por considerar, lo que aumenta la complejidad de este problema.

En CV existen muchas más tareas más acotadas que ver. Para estas sería muy útil utilizar información previamente aprendida con la finalidad de no necesitar muchos datos para la tarea más específica. Por su misma naturaleza, tendrá menos datos de entrenamiento.

3.4. Aplicación en NLP

Hasta el año 2018 no se había aplicado la transferencia de aprendizaje en el área de NLP, pero eso cambió después de trabajos pioneros (Devlin, Chang, Lee & Toutanova, 2018; Howard & Ruder, 2018; Peters y col., 2018) la técnica muestra promesa. Se podría decir que el momento ImageNet ha llegado a NLP y el progreso ha sido de crecimiento explosivo.

La tarea general en NLP para tener un modelo base ha sido la tarea del modelo de lenguaje (Howard & Ruder, 2018). Un modelo de lenguaje es el que intenta predecir la próxima palabra tomando en cuenta las palabras mandadas como argumento al modelo. En textos de temas específicos esto podrá resultar ser un tanto más fácil debido al vocabulario más limitado y por tener acotado el tema. Idealmente para tener una generalización poderosa y que puede aportar en tareas de muy distintos campos, se debe tener un modelo de lenguaje aprendido de un corpus de datos amplio y la cantidad de textos deberá ser masiva.

Un corpus que cumple con los requisitos mencionados sería la versión de Wikipedia en el lenguaje deseado. La enciclopedia virtual abarca muchos temas y tiene diversidad incomparable de vocabulario. Incluso deberá ser limitado a

una cantidad T de tokens. En futuros trabajos se deberá explorar la posibilidad de utilizar distintos corpora.

El modelo de lenguaje es entrenado al usar redes neuronales recurrentes en alguna variante que se considere prudente. Una red LSTM o una Quasi-recurrent Neural Network (QRNN) (Bradbury, Merity, Xiong & Socher, 2016) son ideales para propósitos de este trabajo y han mostrado los mejores resultados del estado del arte en modelos de lenguaje.

3.4.1. ULMFiT

Esta técnica expuesta originalmente por Howard y Ruder (2018), en su curso de Deep Learning, fue publicado para plantear el algoritmo de forma más formal y con estudios de ablación demostrar que cada aspecto de los pasos es esencial para obtener los mejores resultados. La técnica de ULMFiT se divide en dos fases (Howard & Ruder, 2018) las que se explicarán a continuación.

3.4.1.1. Afinación de modelo de lenguaje

Esta fase de la técnica consiste en tomar el modelo de lenguaje general y afinarlo para que esté orientado a la tarea final específica con la que se trabajará. Esto se logra al utilizar los datos de la tarea final para formar un texto sobre el que se terminará de entrenar. Para lograr este afinamiento de la mejor manera, se utilizan diferentes técnicas.

Afinamiento discriminatorio. Esta técnica llamada *fine-tuning*, en el artículo original, consta en usar una tasa de aprendizaje diferente a lo largo de las capas. Es decir, cada capa tiene asignada una tasa de aprendizaje distinta. Se define entonces que los parámetros serán divididos en $\theta^1, \ldots, \theta^L$ donde L es el

número de capas y θ^t es el conjunto de parámetros de la capa l. La actualización de los parámetros entonces, basado en la ecuación 2.3:

$$\theta_t^l = \theta_{t-1}^l - \gamma^l \nabla_\theta L(\theta) \tag{3.1}$$

Tasa de aprendizaje en triangulo inclinado. Llamado *slanted triangle learning rates* en el artículo original. Durante el recorrido a través de los datos se varía la tasa de aprendizaje tal, que su valor aumente durante las primeras *cut* instancias donde,

$$cut = \lfloor T \cdot cut\_frac \rfloor$$

cut_frac es la fracción de datos que queremos sean aprendidos con el crecimiento de la tasa de aprendizaje. Después de que este porcentaje es alcanzado, la tasa de aprendizaje deja de incrementar en valor y empieza a disminuir de forma menos acelerada hasta que se termina de iterar sobre los datos de entrenamiento.

3.4.1.2. Afinación de modelo a tarea de clasificación

Para afinar el modelo de la tarea de clasificación se deberá aplicar el mismo pre-procesamiento en cuanto a tokens de vocabulario. Tener el modelo preentrenado y afinado permite obtener resultados bastante precisos, sin necesidad de tener muchos datos de entrenamiento para la tarea específica.

Para el afinamiento de este modelo se aplican los conceptos desarrollados en la sección 3.4.1.1. Adicionalmente se utiliza una técnica que previene la pérdida de conocimiento.

Descongelamiento gradual. Llamado *gradual unfreezing* en el artículo original. Esta técnica es de las más importantes. El concepto de transferencia de aprendizaje se basa en no perder conocimiento aprendido desde la tarea general original. Al refinar un modelo completamente se toma gran riesgo de *olvidar* vasta información. Esta técnica propone congelar los pesos de todas las capas e ir descongelando una por una y después ajustar y entrenar todas las capas descongeladas. Iterando sobre esto hasta que ya no hay capas por descongelar y se termina de afinar el modelo completo.

3.4.2. Modelo de lenguaje bidireccional

Existe la posibilidad de volver el modelo de lenguaje en un modelo bidireccional (Howard & Ruder, 2018). Esto se puede hacer de forma manual, al revertir el orden de los textos en el corpus y entrenar de la misma forma. Al tener ambos modelos listos se pueden promediar los resultados de las predicciones de cada modelo para obtener la predicción final.

Esto ha sido denominado como modelo bidireccional poco profundo o superficial (Devlin y col., 2018). Esto no es debido porque no es un modelo de Deep Learning, sino que la representación en ambas direcciones no está presente en el modelo, sino se logra con el resultado de dos modelos unidireccionales independientes.

Capítulo 4

Datos utilizados

"On two occasions I have been asked, "Pray, Mr. Babbage, if you put into the machine wrong figures, will the right answers come out?"... I am not able rightly to apprehend the kind of confusion of ideas that could provoke such a question."

—Charles Babbage, Passages from the Life of a Philosopher

El concepto de *garbage in, garbage out* (basura para adentro, basura para afuera) es muy conocido e indica cómo un modelo, análisis, etc. es limitado en su capacidad por los datos que se usen para alimentarlo.

En proyectos de aprendizaje artificial se dice que el 80 % del tiempo se deberá usar para la búsqueda, la limpieza y el tratado de datos. Esta proporción no se mantiene en la mayoría de casos cuando se lidia con modelos de Deep Learning pero el principio se conserva: no se debe subestimar la importancia de los datos y su integridad.

CORRELATIVO	Edad	Género	País	Sueldo	Moneda
1	24	F	GT	3500	GTQ
2	145	M	Perú	100,500	Soles
3	35	N/A	.es	1.5	€
⋮		⋱			⋮
2304	18	F	GT	2.8k	Q

CUADRO 4.1: Ejemplo de datos tabulares que alimentarán a un modelo, ya sea de aprendizaje de máquina o de Deep Learning. En la tabla se ilustran algunos errores de datos (campo *edad*) y datos faltantes (en el campo *Género*). También se aprecian discrepancias en el formato de algunos campos (campo de *Moneda* y *Sueldo*) y campos completamente inservibles para un modelo (campo *CORRELATIVO*.

4.1. Pre-procesamiento de datos

4.1.1. Datos tabulares

La recolección de datos para la alimentación de modelos, por lo general es un proceso que involucra obtener información de distintas fuentes con distintos formatos y distintos detalles que esperar. El pre-procesamiento de datos se encarga de darle forma congruente a los datos para que puedan representar el problema de forma adecuada. Este proceso puede involucrar eliminar errores de formato, identificar datos atípicos y lidiar con ellos, identificar características valiosas para un modelo, etc. Estos pasos son más comunes cuando se lidia con datos tabulares, es decir datos que están separados por categorías o columnas y deben ser tratados cada una como variable.

4.1.2. Datos textuales

En NLP y en este trabajo se lidia con datos en forma de texto y no de forma tabular. Esto conlleva un conjunto de retos especiales a considerar; surgen conceptos nuevos a considerar. En NLP los pasos que generalmente se realizan para limpiar los datos son los siguientes:

- **Tokenizar.** Esta técnica de pre-procesamiento consiste en separar las cadenas de palabras — textos — en *tokens*. Esto con el propósito de que el modelo sea capaz de digerir las cadenas de palabras por segmentos bien definidos. El delimitador de los tokens en los datos de entrada podrá ser algo simple como un carácter de espacio (' '). También existen técnicas más elaboradas, por ejemplo: tokenización basado en la morfología de un lenguaje.
- **Limitar vocabulario.** Como consecuencia de tener cantidades grandes de datos para la alimentación del modelo se puede terminar con un vocabulario bastante numeroso. Este vocabulario está compuesto de todos los tokens únicos que se encontraron en los datos de entrada. Una técnica común para evitar utilizar tokens encontrados, muy pocas veces en la data, es limitar el número de tokens a usar. Se elige esa cantidad n de datos a usar al ordenar los tokens por su frecuencia y tomando los n más comunes.
- **Eliminar palabras con propósito gramático.** Muchas tareas de clasificación en NLP dependen mucho del contenido de los textos. Una técnica común para enfatizar el valor semántico de los textos es eliminar tokens cuyo propósito es únicamente gramatical. Se podrían nombrar artículos y preposiciónes como ejemplos.

- **Vectorizar.** Una técnica muy valiosa que consiste en representar las cadenas de caracteres en vectores en un espacio de d dimensión. Mientras más grande d, mejores las posibilidades de capturar más el significado de cada token o palabra. En modelos del estado del arte en NLP se utiliza una dimensionalidad generalmente de $d = 300$ o $d = 400$. Algunas técnicas de vectorización son capaces de capturar mucha información semántica, al grado de inferir información acerca de incluso tokens no antes vistos.

- **Zero padding.** Esta es una técnica usada para dar a los datos de entrada el mismo largo y permitir que el modelo no sufra por la variabilidad de longitud. Esta técnica es más común cuando se usan modelos de machine learning y no deep learning.

- **Tokens especiales.** Al aplicar técnicas como limitación de vocabulario o zero padding surge la necesidad de usar tokens especiales que le indican al modelo conceptos como un término no antes visto o el final de una secuencia. Estos tokens especiales generalmente son definidos por quien realiza el proceso de tokenización, aunque también se pueden incluir previo a esta etapa. Estos tokens especiales generalmente son representados en este estilo `<unk>` donde `unk` es generalmente una abreviación del concepto del token especial.

Todos estos pasos tienen alternativas o son completamente opcionales. Para algunas tareas de clasificación de texto tareas como la eliminación de palabras gramáticas son necesarios. Sin embargo, para una tarea como atribución de autor de un texto es necesario incluir información que pueda identificar al autor (Coulthard, 2004; Louwerse, 2004).

4.2. Fuente de los datos

LEFTy es un proyecto multi-tarea, que significa que las predicciones son de distintos conceptos. Esto generalmente lleva a utilizar datos distintos para cada tarea en específico. Gracias a la ventaja que la transferencia de aprendizaje provee — descrita en el capítulo 3 — la cantidad de datos necesaria para obtener buenos resultados no es excesiva y esto facilita la búsqueda de datos.

Las fuentes de los datos para cada subtarea son los siguientes:

- **Determinación de género y edad.** El corpus de datos que se usa para estas dos tareas fue el mismo. El corpus es llamado *SpanText* (Villegas, Ucelay, Errecalde & Cagnina, 2014) y contiene textos escritos por personas hispanohablantes, cada uno etiquetado con su género y edad. Para estos datos se usan solamente textos con una longitud mínima de 100 palabras.

- **Determinación de género y región en textos informales.** Adicional a las fuentes de datos mencionadas, se usan los datos de una competencia de perfilamiento de autores, en específico usando textos informales. La competencia es PAN en su edición del 2017 (https://pan.webis.de/clef17/pan17-web/author-profiling.html). Los datos han sido utilizados con el permiso de los organizadores de la competencia de PAN17 (Rangel, Rosso, Potthast & Stein, 2017a). Estos datos no han sido pre-procesados de ninguna forma y la única garantía que se tiene es que hay una cantidad mínima de tuits por autor y una cantidad mínima de autores por género y región. El conjunto de datos se encuentra dividido entre conjuntos de datos para entrenamiento y para prueba de desempeño.

- **Datos de pre-entrenamiento.** Para la fase de pre-entrenamiento del modelo de lenguaje se utilizan datos de *Wikipedia*. Se descarga un *dump* —

colección de datos — de noviembre del 2018. De estos datos se filtran entonces los textos que tenían como mínimo $m = 1000$ cantidad de palabras. Luego de esto, se seleccionaron los primeros 100 millones de *tokens*, esto para simular una distribución de artículos similar a *WikiText-103* (Merity, Xiong, Bradbury & Socher, 2016) que fue propuesta como muestra más realista de la distribución de textos de *Wikipedia*.

Capítulo 5

LEFTy: Implementación y aplicación

Language Efficient Text portray (LEFTy) es el nombre designado para referirse al trabajo actual. Esta solución propuesta emplea el concepto de *transfer learning* para permitir entrenar con gran capacidad tareas que no tienen muchos ejemplos etiquetados. Se propone este sistema para la tarea de perfilación de autores. Utiliza una RNN como base del modelo y las características base obtenidas fueron:

- **Edad.**
- **Género.**
- **Región de origen.**

5.1. Pre-entrenamiento de modelo de lenguaje

La fase de pre-entrenamiento en el contexto de NLP consiste en entrenar un tipo de modelo de lenguaje. En el artículo original de ULMFiT (Howard & Ruder, 2018), se utiliza un modelo estándar en donde se predice el siguiente token basado en una cadena de tokens. Bi-directional Encoder Representations from Transformers (BERT) (Devlin y col., 2018), por otro lado utiliza un Masked

Language Model (MLM) que consiste en predecir el 15 % de los tokens dado todo el contexto que los rodea. En este trabajo se utiliza un modelo estándar de lenguaje, es decir se predice el próximo token basado en los tokens anteriores como entrada.

5.1.1. Modelo de lenguaje de Wikipedia

El diseño base para este modelo de lenguaje es una red denominada Average-stochastic-gradient-descent Weight Dropped LSTM (AWD LSTM) (Merity, Keskar & Socher, 2017), la que emplea una modificación agresiva al método de regularización *dropout*. En el artículo se sugiere utilizar un concepto denominado *DropConnect* y difiere en *dropout* en que las funciones de activación no son las que toman el valor cero, sino los pesos. También se usan los conceptos de usar *dropout* en la capa de vectorización de palabras — esto no aporta a la regularización pero sí disminuye el tamaño de los vectores representantes de los vectores. Se instancian los distintos tipos de *dropout* con pesos asignados a cada uno. En el artículo se recomiendan usar ciertos pesos base y optimizar un hiperparámetro r_f únicamente le da escala a los pesos recomendados y definidos por ellos.

En el capítulo 4 se explica el pre-procesamiento que se le da a los datos de *Wikipedia*. Se detallará ese proceso a continuación.

Para realizar este procedimiento se utilizaron los recursos de *Google Colaboratory* (Colab), que ofrecen un ambiente de *Notebooks* de IPython y la habilidad de ejecutar comandos de nix.

Obtención de datos. El corpus de Wikipedia fue obtenido del sitio oficial (https://dumps.wikimedia.org/eswiki/). El corpus obtenido fue del mes de noviembre del año 2018. Estos archivos tienen una estructura específica y

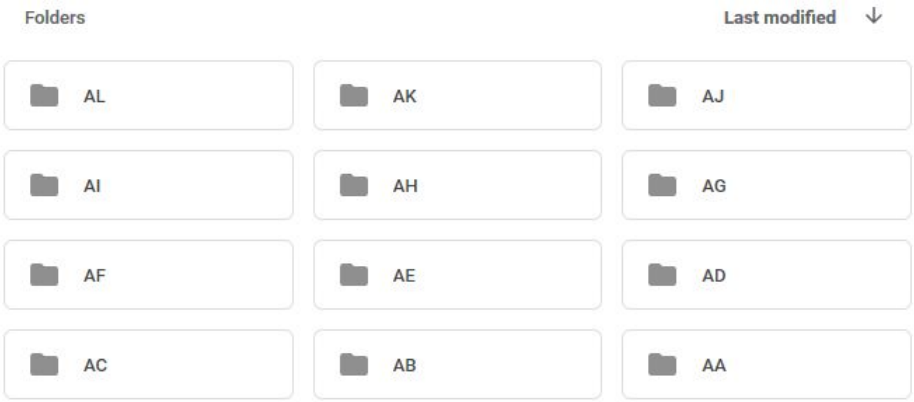


FIGURA 5.1: Estructura de datos resultante al extraer un archivo de *Wikipedia*.

Cadena de caracteres	**Resultado tokenizado**
Hola, buenos días.	`xxbos xxmaj hola , buenos días`
HOLA!!! qué bueno verte	`xxbox xxmaj hola xxrep 3 ! qué bueno verte`
Mis audífonos son Sennheiser	`xxbos xxmaj mis audífonos son xxunk`

CUADRO 5.1: Muestras de tokenización al utilzar *spaCy* y técnicas de *fastai*

la forma recomendada de extraer sus contenidos es utilizando *WikiExtractor* (https://github.com/attardi/wikiextractor). Esta herramienta permite realizar una extracción que filtra por un parámetro de mínimo de longitud del artículo. Se usó este parámetro para filtrar todos los artículos con menos de 10,000 caracteres.

Una vez finaliza la extracción del archivo — que demora cantidad no despreciable de horas — se procede a leer y filtrar los documentos. En el caso de este trabajo se ignoraron todos los documentos después de haber acumulado 100 millones de tokens en los textos procesados. Se conservaron 10 millones de tokens adicionales para la validación de resultados.

Tokenización. La herramienta utilizada para este proceso fue spaCy (https://spacy.io/). Esta herramienta tiene soporte para más de 34 idiomas, entre los

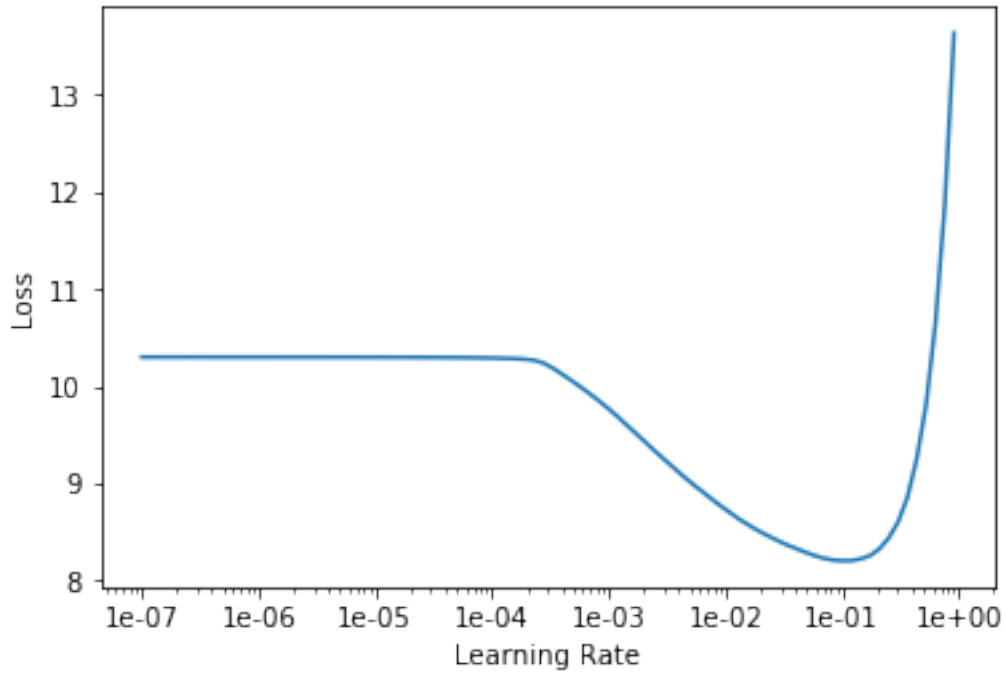


FIGURA 5.2: Ilustración de progreso de la función `lr_find` de *fast.ai*. Al momento de divergir el método deja de aumentar γ y devuelve los resultados.

que se incluye el español. Además de esta herramienta se emplean técnicas adicionales como codificar caracteres repetidos o codificar palabras en mayúsculas — ver tabla 5.1.

Definición de vocabulario. Unos tokens son codificados como `xxunk` (token desconocido), debido a la limitación de palabras únicas a codificar. El vocabulario fue limitado a 30,000 tokens únicos.

Entrenamiento. Luego tener los datos en el formato adecuado para entrenar el modelo, se aplican las técnicas descritas en el capítulo 3 sobre una red AWD LSTM. Previo a esto se encuentra un γ ideal y eso se hace aprovechando el método `lr_find` (figura 5.2) de la librería de *fastai* (https://github.com/fastai/fastai) que emplea el método descrito por Smith y Topin (2017). Se aumenta γ a medida

Modelo	Error en validación	*Perplexity*
QRNN	3.193912	24.2884
AWD LSTM	3.140521	**23.1038**

CUADRO 5.2: Resultados de los modelos entrenados. La métrica que se utiliza para comparar entre los modelos es *perplexity* que indica y representa qué tantas opciones se tienen para la siguiente palabra.

que se recorren los ejemplos a entrenar. Si en algún momento diverge el aprendizaje, se detiene el método y se imprime la gráfica. Se deberá elegir un γ que no tenga riesgo de divergir el aprendizaje y que tampoco disminuya la velocidad del proceso. La elección adecuada de este hiper-parámetro en esta etapa es crucial para obtener un modelo en un tiempo razonable, ya que este modelo es el más grande — en términos de parámetros y datos a procesar — que requerirá entrenamiento. Vale la pena enfatizar que este proceso se realiza una *única* vez y la definición de modelos clasificadores en un futuro podrá hacer uso del modelo entrenado originalmente.

5.1.1.1. Resultados

Para medir y reportar errores sobre Language Models (LMs) se utiliza la métrica de *perplexity* (Jurafsky & Martin, 2014), que es análoga a la entropía. La entropía representa la cantidad de información que se tiene; *perplexity* se puede ver como la cantidad de opciones que se tiene. ¿Qué significa esto? Que mientras menos opciones considere nuestro LM, más estable es y, por lo tanto, que modela mejor el lenguaje. Se calcula de la siguiente manera:

$$perplexity = e^{loss}$$

En nuestro caso, se utiliza el valor de pérdida de los datos de validación.

Modelo	***Perplexity***
RNN (Goldszmidt, 2018)	N/A (34 % prec.)
RNN (Ingham, 2018)	36.1473
Activation Mem. (Rae, Dyer, Dayan & Lillicrap, 2018)	29.2
Transformer-XL (Dai y col., 2019)	**18.3**
QRNN (propio)	24.2884
AWD LSTM (propio)	**23.1038**

CUADRO 5.3: Comparación de LMs con modelos del estado del arte en inglés y modelos de referencia para el español. Menor *perplexity* es mejor. Separamos el modelo *Transformer-XL* ya que utiliza otra arquitectura y está presente en la tabla solamente como referencia al mejor resultado, al momento de haber escrito este trabajo.

Se entrenaron dos modelos de RNN, uno fue basado directamente de la arquitectura y estrategias de AWD LSTM y el otro modelo basado en la arquitectura QRNN. Los resultados se muestran en la tabla 5.2. Sin embargo, esta tabla no cuenta la historia completa. El modelo QRNN demoró menos en su entrenamiento de forma no insignificante tomando 13 % menos en entrenarse con resultados muy comparables a los de la red AWD LSTM.

En la tabla 5.3 se aprecian distintos LMs que fueron entrenados en distintos datasets. El modelo Transformer-XL (Dai y col., 2019) utiliza la nueva tendencia de finales del 2018 y principios de 2019 de usar transformadores en lugar de RNNs, así como propone Google con BERT. El modelo de *Activation Memory* (Rae, Dyer, Dayan & Lillicrap, 2018) es de referencia que utiliza la arquitectura de una RNN simplificando un sub-conjunto de sus parámetros. Los otros dos modelos han sido modelos previamente entrenados con el propósito de ser usados para tareas clasificación en español con ULMFiT. Los datos presentados para los modelos propios fueron el valor de pérdida en un set de validación de 100 millones de tokens elegidos al azar de *Wikiepedia*.

Aunque hay argumentos en contra de usar *perplexity* para comparar LMs

de distintos lenguajes o que usan distintos vocabularios (Chen, Beeferman & Rosenfeld, 1998), se debe tener un marco de referencia. Los modelos con propósitos de usar ULMFiT fueron entrenados de una forma muy similar al propio y son las comparaciones más directas por ser LMs del idioma español.

5.1.1.2. Recursos utilizados para entrenamiento

Costo monetario. Para esta fase de entrenamiento se recurre a los servicios de *Google Cloud* (https://cloud.google.com) que son ofrecidos con un beneficio de 300 USD para utilizar durante el primer año. No es necesario ser estudiante o profesor para gozar de este beneficio. Teniendo estos recursos disponibles se optó por utilizar una instance de cómputo *n1-highmem-4* (https://cloud.google.com/compute/docs/machine-types) que cuenta con el siguiente hardware para el entrenamiento:

- Intel Xeon (Skylake) 4 vCPUs
- 26 GB de memoria (RAM)
- nVIDIA V100 GPU – 16 GB de memoria

Lo primordial cuando se trata de entrenamiento de RNNs, es la capacidad de cómputo de la GPU. La **V** en el modelo V100 indica que es de la última generación a la fecha de esta tesis y provee una ventaja significativa comparada con una K80 o P100.

Costo de recursos. El costo total resultante después de entrenar un modelo inicial y funcional llegó a $ 60.20 USD. Esto fue cubierto por los créditos iniciales ofrecidos por Google. También se debe considerar que este paso se debe realizar *una sola vez* para cada lenguaje. En caso de querer utilizar el modelo entrenado

en este trabajo, se podrá hacerlo y se podrá aplicar a otros problemas de clasificación.

Costo en tiempo. Para entrenar los modelos de lenguaje con una estructura AWD LSTM, una época se demora alrededor de una hora. Después de 4 épocas se aproximan los resultados al estado del arte y debe decidirse cuando detener el entrenamiento sin perder la generalidad del sistema.

5.1.2. Modelos clasificadores

Procesamiento de datos. El conjunto de datos principales que se utilizó fue el de PAN17 (Rangel, Rosso, Potthast & Stein, 2017b). Este conjunto de datos no es típico, ya que consta de tuits de un grupo de usuarios. Cada uno de estos tuits — 100 por usuario — tiene asociado su usuario que tiene las etiquetas de edad y región asignadas. Por lo tanto se puede abordar el problema de distintas formas; se puede tomar cada tuit individual y predecir el género y la región individualmente, sumando después las probabilidades para obtener un veredicto final para el autor o se pueden concatenar los tuits del autor y realizar una solo predicción sobre ese texto completo del autor. Para este trabajo se utilizó cada tuit individual con el fin de tener un alcance mayor sobre la tarea de perfilación de autores sobre textos informales.

Pre-procesamiento y tokenización. El procesamiento constó de normalizar los *tokens* especiales de los textos (ver tabla 5.4); en particular se normalizaron las menciones de usuario y los *hashtags*. Este proceso consistió en ordenar los *tokens* del vocabulario del modelo pre-entrenado de *Wikipedia* y buscar las instancias encontradas en los *tokens* de interés. Luego de haber hecho esto, anteponer un *token* especial que indique que se encontraba ahí originalmente, por ejemplo

Estado del texto	Resultado
Original	`#igualdaddegenero y #mediosdecomunicacion http://t.me/ssbM`
Sin normalizar	`xxunk xxunk y xxunk xxunk xxunk`
Normalizado	`xxhashtag igualdad enero y xxhashtag medios comunica xxurl`

CUADRO 5.4: Ejemplificación de normalización de textos informales obtenidos del dataset PAN17 para la tarea de perfilación de autores. El proceso de normalización permite extraer información de los *tokens* que son menciones de usuario y *hashtags*. En la tabla se aprecia un ejemplo real con drásticos cambios que depende del procesamiento.

`xxhashtag`. Se tokenizan los datos de la misma manera en que se tokenizaron los datos en la fase de modelo de lenguaje al utilizar *spaCy*.

Afinación de LM. Después de haber entrenado el modelo de lenguaje se procede a una fase de afinación (Howard & Ruder, 2018). Se emplean las técnicas descritas en el capítulo 3 para afinar el modelo. Este proceso se realiza con los datos de la tarea en específico. Se adapta el formato de los datos de clasificación a datos para entrenamiento de un LM y se afina el modelo para textos de este dominio específico. Sobre la métrica de *perplexity* o la precisión no se pone mayor énfasis, lo importante será que la estructura general de los textos y el vocabulario que se use, refleje los datos reales del problema específico.

Entrenamiento de los modelos clasificadores. Al tener el LM afinado se procede a entrenar un clasificador. Este proceso es descrito en la sección 2.5. A continuación se detallan las decisiones tomadas.

Entrenamiento del clasificador. Antes de comenzar con el entrenamiento se deben elegir los hiper-parámetros adecuados. Primero se eligió un peso para los parámetros de *dropout* (basado en pruebas empíricas) y después se eligió un γ óptimo basado en la herramienta de `lr_find` (figura 5.3). Los valores elegidos

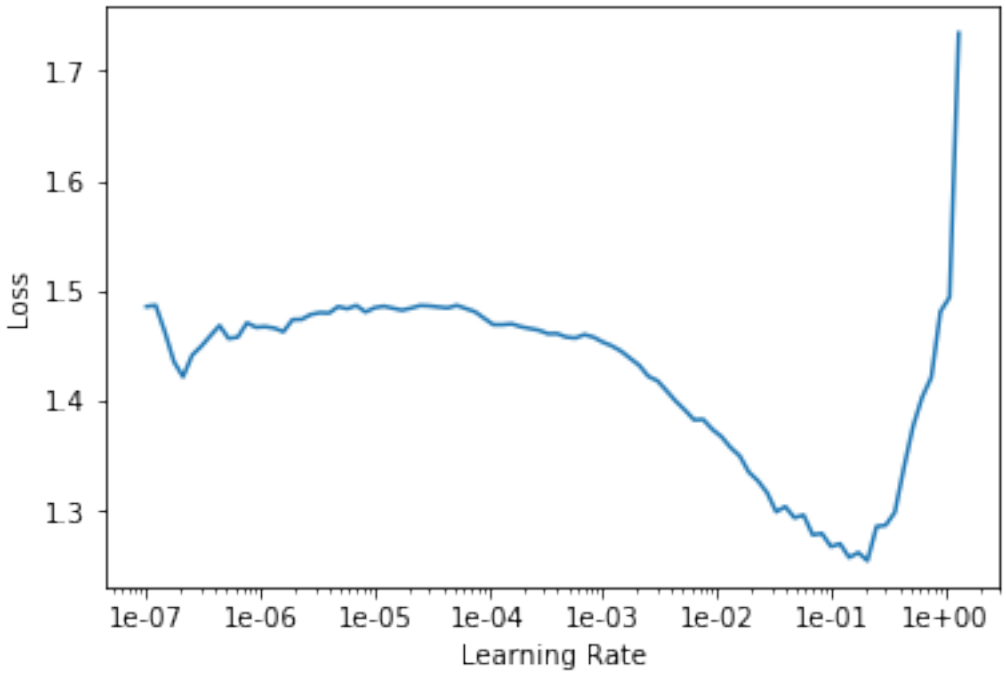


FIGURA 5.3: Evolución de la pérdida conforme se cambia la tasa de aprendizaje en el clasificador. Al igual que con el LM, se deberá elegir un valor óptimo para mayor eficiencia.

fueron `dropout = 0.3` y un γ inicial, con valor de `1e-2`. Se habla de un γ inicial ya que en el entrenamiento del modelo clasificador se aplica el concepto de usar un γ cíclico (Smith & Topin, 2017) abordado en el capítulo 3.

El entrenamiento de un clasificador con pocos datos no deberá tomar mucho tiempo y los resultados deberán ser satisfactorios aplicando los temas expuestos en este trabajo. En la figura 5.4 se puede apreciar cómo la evolución de pérdida avanza lentamente a través de las épocas y converge en un valor después de un número de épocas relativamente bajo. Sin embargo, es posible continuar con el entrenamiento hasta lidiar con *overfitting*. Este problema surge cuando se han afinado tanto los parámetros del modelo que el proceso de *back-propagation* empieza a ajustarse conforme solamente los datos de entrenamiento y no el problema en sí. La manera más fácil de detectar este fenómeno es con la

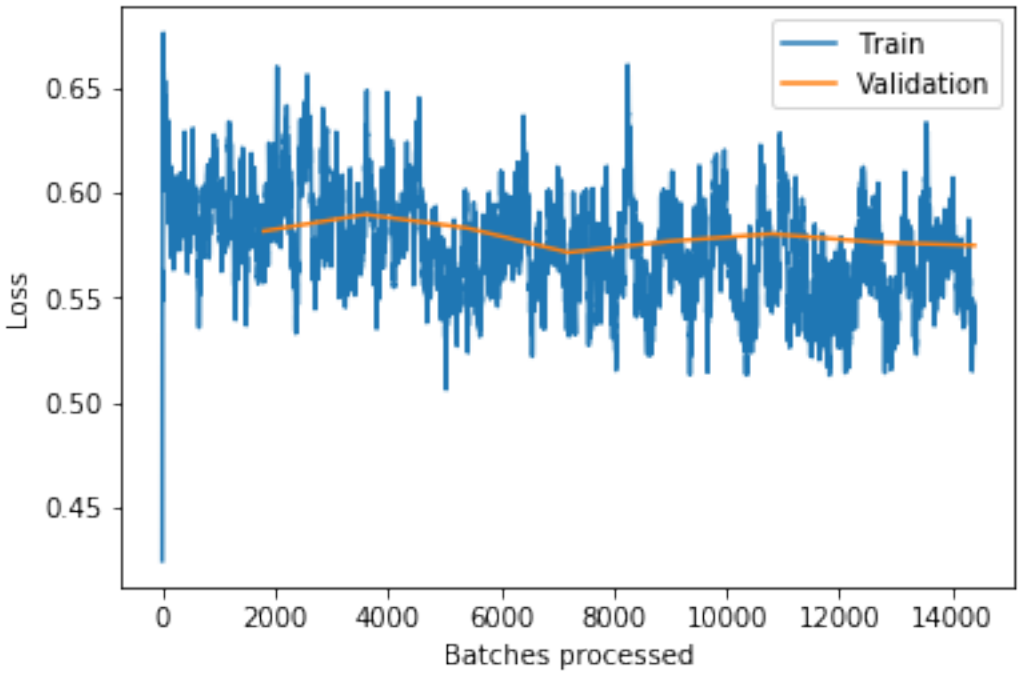


FIGURA 5.4: Avance de pérdida conforme avanzan las épocas de entrenamiento. Menor es mejor.

comparación de resultados sobre la función de pérdida de los datos de entrenamiento con los datos de evaluación. Si el valor de pérdida para ambos conjuntos de datos disminuye, significa que el entrenamiento se está realizando con éxito. Si en algún momento el valor de pérdida para los datos de entrenamiento continua disminuyendo pero al mismo tiempo el valor de pérdida para los datos de validación aumenta, significa que ya no estamos obteniendo resultados que puedan ser generalizados a datos no antes vistos.

5.2. Resultados

Se ha dicho que para RNNs y técnicas de Deep Learning no han tenido el mismo éxito que en otras tareas debido a la dificultad que un modelo de Deep Learning tiene al aprender la cantidad de parámetros inmensa con pocos datos

(Malmasi y col., 2016; Zampieri y col., 2017). Esta tarea es, por lo tanto, ideal para determinar la capacidad real la transferencia de aprendizaje en NLP y, sobre todo, cuando las instancias de aprendizaje son limitadas.

5.2.1. Predicción sobre textos informales

Los datos y resultados en referencia a predicción de región y género provienen del reporte de la competencia PAN17 (Rangel y col., 2017b) y proveen un contexto para la tarea del trazo de perfiles de autores. Debido a que los conjuntos de datos de pruebas son los mismos, las comparaciones se podrán hacer de forma directa. Se realiza una comparación con datos exclusivamente del idioma español. Los modelos a comparar son los siguientes:

- Term Frequency – Inverse Document Frequency (TF-IDF) + Convolutional Neural Network (CNN) (Schaetti, 2017)
- RNN (Kodiyan, Hardegger, Neuhaus & Cieliebak, 2017)
- AVG DNN (Franco-Salvador, Plotnikova, Pawar & Benajiba, 2017)
- CNN + RNN + Attention (Miura, Taniguchi, Taniguchi & Ohkuma, 2017)

Para estos resultados se utiliza la métrica usada en la competencia PAN17 la cuál es:

$$Acc_{cat} = \frac{Predicciones\ correctas}{Total\ de\ predicciones}$$

Donde *cat* es la categoría sobre la que se está prediciendo. Las predicciones son aplicadas sobre los datos de validación, que fueron segmentados por el equipo organizador de PAN17 y cuenta con 2,800 autores, cada uno asociado a 100 tuits.

	Referencia	
Predicción	Femenino	Masculino
Femenino	83110	49573
Masculino	56890	90427

CUADRO 5.5: Matriz de confusión de resultados sobre predicción de género en tuits individuales.

	Referencia	
Predicción	Femenino	Masculino
Femenino	963	283
Masculino	437	1117

CUADRO 5.6: Matriz de confusión de resultados de predicciones por autor.

5.2.1.1. Predicción de género

La predicción de género sobre un texto anónimo se efectuó basado en datos de tuits. La clasificación se llevó a cabo en dos niveles. Se efectuó una predicción a nivel de tuit individual y una predicción a nivel de autor.

En la tabla 5.5 se aprecian los resultados de las predicciones sobre tuits individuales. La evaluación de la métrica con estos datos sería entonces:

$$Acc_{gender\_idv} = \frac{173,537}{280,000} = 0.6198$$

La predicción sobre los autores es basada en las predicciones sobre tuits individuales. Esto resulta en una suma ponderada de las predicciones individuales descrita como

$$\hat{p}_a(\mathbf{X}) = \text{máx} \sum_{n=1}^{100} p(X_n),$$

donde $\mathbf{X}$ representa una matriz de probabilidades; las filas representan tuits individuales y las columnas las probabilidades de que el tuit pertenezca a la categoría de esa columna (p.e. el género femenino).

Modelo	Resultado
TF-IDF + CNN (Schaetti, 2017)	0.7150
RNN (Kodiyan y col., 2017)	0.7217
AVG DNN (Franco-Salvador y col., 2017)	**0.7721**
CNN + RNN + Attention (Miura y col., 2017)	**0.8118**
RNN + Transfer Learning (propio)	0.7429

CUADRO 5.7: Comparación de modelos de Deep Learning en la tarea de predicción de género — mayor es mejor. El modelo de Miura y col. (2017) se separa, ya que es un ensamble de modelos y no se puede hacer una comparación directa.

Dada la anterior, tenemos la tabla 5.6 y un resultado de

$$Acc_{gender} = 0.7429.$$

La sumatoria ponderada resulta en un sistema similar a un ensamble de modelos que toman prioridad según la confianza que tengan en su predicción. Debido a esto, el resultado aumenta en la métrica considerablemente.

Al comparar el modelo propio con los modelos de Deep Learning presentados en la competencia PAN17 (tabla 5.7), se puede apreciar que los resultados obtenidos son del estado del arte para RNN, siendo solamente superado por un AVG DNN y un ensamble de CNN, RNN y elementos de atención.

5.2.1.2. Predicción de región

La predicción de región sobre un texto anónimo se efectuó basado en datos de tuits. Al igual que en la clasificación de género, se efectúa una predicción a nivel de tuit individual y una predicción a nivel de autor.

De la misma manera que con las predicciones de género, en la tabla 5.8 se aprecian los resultados de las predicciones sobre tuits individuales. La evaluación de la métrica con estos datos — siendo esta solo indicativa — sería entonces

	Referencia						
Predicción	AR	CL	CO	MX	PE	ES	VZ
AR	22,610	3,013	3,147	2,220	3,783	2,558	2,188
CL	3,064	21,058	2,988	3,034	3,760	2,599	2,454
CO	3,435	2,993	20,007	4,401	4,251	2,746	3,696
MX	2,554	3,429	3,977	18,803	4,173	3,681	3,616
PE	2,992	3,780	3,544	3,542	17,812	2,552	2,605
ES	3,331	3,565	3,519	5,131	3,662	23,722	3,689
VZ	2,014	2,162	2,818	2,869	2,559	2,142	21,752

CUADRO 5.8: Matriz de confusión de resultados sobre predicción de región en tuits individuales.

	Referencia						
Predicción	AR	CL	CO	MX	PE	ES	VZ
AR	381	5	7	5	14	1	7
CL	2	370	2	3	4	1	1
CO	2	3	367	10	9	3	8
MX	3	5	4	364	9	3	9
PE	4	7	8	3	353	3	4
ES	5	5	8	13	4	388	17
VZ	3	5	3	2	7	1	354

CUADRO 5.9: Matriz de confusión de resultados de predicción de región por autor.

la siguiente:

$$Acc_{region\_idv} = \frac{145,764}{280,000} = 0.5206$$

La predicción sobre los autores basada en la suma ponderada tenemos la tabla 5.9 y un resultado de

$$Acc_{region} = 0.9207.$$

La sumatoria ponderada resulta en un resultado mucho mayor que el de las predicciones individuales. Esto puede deberse a dos cosas:

Modelo	Resultado
TF-IDF + CNN (Schaetti, 2017)	**0.9336**
RNN (Kodiyan y col., 2017)	0.9143
AVG DNN (Franco-Salvador y col., 2017)	0.9000
CNN + RNN(Att.) (Miura y col., 2017)	**0.9271**
RNN + Transfer Learning (propio)	0.9207

CUADRO 5.10: Comparación de modelos de Deep Learning en la tarea de predicción de región — mayor es mejor. El modelo de Miura y col. (2017) se separa, ya que es un ensamble de modelos y no es posible hacer una comparación directa.

- Muchos tuits son muy limitados en su número de caracteres, por lo que algunos no revelarán mucha información acerca del autor por sí solos.
- La cobertura de vocabulario con palabras regionales disminuye, por lo que algunos tuits individuales con palabras claves no proveen mucha información.

Nuevamente se comparan los resultados con los modelos de Deep Learning de PAN17 (tabla 5.10). Una combinación de TF-IDF y CNN (Schaetti, 2017) logró clasificar de la mejor manera; el modelo propio obtiene la segunda mejor precisión en modelos simples y la tercera al considerar el ensamble de modelos CNN y RNN de Miura y col. (2017).

5.2.2. Predicción sobre textos formales

Los experimentos se repiten sobre textos formales al usar los datos de SpanText (Villegas y col., 2014). Los datos de referencia y comparación se obtuvieron directamente de la publicación del conjunto de datos.

El corpus de *SpanText* está compuesto por un dos subconjuntos de datos:

- **Dataset balanceado**, es decir las distintas clases de textos que conforman el conjunto están repartidas equitativamente, es decir que p.e. por cada

Clase	Cantidad de textos	Longitud (palabras)
10smale	140	153
10sfemale	60	130
20smale	210	239
20sfemale	90	236
30smale	350	267
30sfemale	150	240

CUADRO 5.11: Distribución de datos usados para la tarea de clasificación de textos formales.

texto que sea de una persona de género masculino, hay un texto de autor femenino.

- **Dataset desbalanceado**. Este dataset se creó con el propósito de emular la distribución encontrada en los datos de la competencia PAN y así permitir una comparación más directa en términos de resultados (Villegas y col., 2014).

Los mejores resultados obtenidos por Villegas y col. fueron sobre el dataset desbalanceado, por lo que sobre esos se hará la comparación con el modelo propio. La distribución de los datos desbalanceados se muestra en la table 5.11.

5.2.2.1. Resultados

La tarea definida en el artículo de *SpanText* es la de la predicción ambos el género y la edad del autor del texto formal, esto significa que la precisión reportada es de la proporción de textos sobre los que se predijo ambas características. La cantidad de textos por clase es bastante limitada, así como la longitud promedio de los textos, que aumenta la dificultad de la tarea para modelos de Deep Learning.

Para la tarea de clasificación, Villegas y col. propusieron dos diferentes modelos de clasificación, junto con distintas formas de representar los tokens de

Modelo	**Resultado**
LIBLINEAR; TF-IDF + words (Villegas y col., 2014)	58.4
Naïve Bayes; boolean + words (Villegas y col., 2014)	61.2
RNN + Transfer Learning (propio)	**71.0**

CUADRO 5.12: Comparación de modelos en la tarea de clasificación de SpanText. El único modelo de Deep Learning es el propio. El resultado es la precisión de la predicción, por lo que un mayor resultado es mejor.

alimentación al modelo. Se eligieron los dos mejores modelos — uno por el tipo de modelo *LIBLINEAR* y otro para el tipo de modelo *Naïve Bayes* — para realizar la comparación con el modelo propio. Los resultados se pueden apreciar en la tabla 5.12.

Capítulo 6

Conclusiones

En este capítulo se presentan las conclusiones obtenidas a partir del trabajo de investigación y los resultados encontrados.

6.1. Conclusiones

El campo de Deep Learning ha mostrado mucha promesa en los últimos años, pero el área de NLP no ha mostrado tanta promesa sino hasta el último año. En este trabajo se busca fomentar y profundizar más la penetración que han tenido estas técnicas en el área, tanto en sectores industriales como académicos.

Para plantear las conclusiones de este trabajo, se deberá recordar los objetivos principales mencionados en un comienzo. Se pretendía plantear un sistema competitivo a niveles del estado del arte que no requiriera ingeniería de características sobre los textos. Un modelo que no requiriera expertiz en el campo de la lingüística para ser diseñado; un modelo que no necesitara tantos ejemplos de entrenamiento para generalizar y desempeñarse bien la tarea asignada. LEFTy cumple con estos objetivos de forma elegante y novedosa aplicando *transfer learning* a la tarea de perfilamiento de autores en el área de NLP.

Este concepto permite abrir muchas puertas y se ha demostrado que no sólo aplica para el área de visión artificial con muchas publicaciones en el último año de laboratorios grandes como OpenAI, fastai, Google, Deep Mind, etc. La promesa que trae esta técnica a NLP, permitiría aplicarse a una infinidad de tareas sin depender de muchos ejemplos, justo como se pudo apreciar con los resultados presentados sobre textos formales.

Sin embargo, *transfer learning* tiene compromisos que se deben tomar en cuenta, a pesar de todas las ventajas que provee. Una de ellas es no cambiar la estructura del modelo pre-entrenado de una manera muy significativa sin el riesgo de una pérdida catastrófica del aprendizaje general con el que se cuenta. Lo anterior lleva también a la conclusión que se debe tener mucho cuidado al afinar un modelo general a una tarea específica en NLP, ya que se tiene el mismo riesgo si se entrena de manera desenfrenada.

En este trabajo se reiteran algunas técnicas de afinamiento de modelos, tanto para modelos de lenguaje como para modelos clasificadores de una tarea final, reproduciendo de manera parcial los resultados presentados por Howard y Ruder.

Los resultados presentados en el capítulo 5 de este trabajo compiten con los modelos del estado del arte y con gente que se ha dedicado por años al estudio de NLP, lingüística y su intersección como ciencias. Esto demuestra que no se debe ser persona experta para aprovechar el poder del deep learning.

Se presentó un modelo de lenguaje del idioma español que puede ser utilizado para una cantidad ilimitada de tareas y se hizo público para que pueda ser aprovechado por la comunidad científica en esta área; este ha sido descargado más de 30 veces al momento de la publicación de este trabajo.

El modelo publicado también provee la ventaja de permitir una reducción de costos al entrenar modelos clasificadores futuros, ya que el mayor costo presentado en este trabajo fue el del entrenamiento del modelo de lenguaje general entrenado sobre datos de Wikipedia. Este hecho promete que el uso de Deep Learning no sea prohibitivo para compañías e individuos que no tienen acceso a poder computacional masivo o incluso ilimitado como algunas compañías, lo que se considera una contribución muy valiosa a la comunidad.

6.2. Trabajo futuro

Las estrategias, técnicas y tecnologías utilizadas en este trabajo muestra un balance entre obtener resultados del estado del arte y realizar la tarea sin sobrellevar costos prohibitivos. Debido a esto, algunas técnicas no fueron utilizadas debido a limitaciones económicas o temporales y una sección de trabajo futuro es necesaria.

Asimismo, debido a que el desarrollo en esta área tiene un ritmo muy acelerado, durante la elaboración de este trabajo se realizaron múltiples publicaciones con nuevos aportes a la comunidad científica que pueden ser de mucho valor para la aplicación específica descrita en este proyecto.

A continuación se plantea el trabajo futuro a considerar:

- La adición más simple que se podría hacer con el trabajo realizado es utilizar LSTMs bidireccionales. En este caso en particular significaría entrenar un modelo que evalúe los tuis con el orden inverso de las palabras y realizar un ensamble de los modelos resultantes.
- Radford y col. (2019) han mostrado que utilizar textos más diversos y en mayores cantidades para el entrenamiento de modelos de lenguaje puede

resultar en una ventaja adicional, no sólo en ese modelo en particular sino en las tareas específicas para las que después se utilizará el sistema pre-entrenado.

- El uso de una nueva estructura llamada transformador ha tomado un auge en el año 2018 y 2019 y ha presentado nuevos resultados del estado del arte. En algunas ocasiones ha superado por un margen elevado al resultado a vencer. Esto indica que puede aportar mucho valor usar una estructura de trasnformador en lugar de una de LSTM. Los transformadores proveen ventajas como la posibilidad de ser entrenados en modalidades paralelas. Sin embargo, muchos resultados han necesitado de mucho poder computacional por lo que puede llegar a ser prohibitivo su uso.
- En este trabajo se mostró que el pre-procesamiento de los textos antes de ser alimentados al modelo no es perfecto, por lo que se deben explorar formas de refinar esto. En particular, la normalización y conservación del aspecto semántico de los textos, especialmente los informales, requerirán de mayor esfuerzo y experimentación.
- Para este trabajo se clasificaron los tuits individualmente, sin embargo existía la posibilidad de utilizar concatenación de textos y evaluar sobre el texto resultante para cada autor. Se podría usar este abordamiento al problema y evaluar si se obtiene mejor desempeño. Esto tendría el propósito de mejorar la posición en una competencia y no de avanzar el campo en esta tarea en específico.
- Muchos avances se han realizado en los últimos años con respecto a modelos multi-tarea (multitask) — que son modelos que tienen la capacidad de realizar múltiples predicciones, cada una de una tarea independiente.

Tener solamente un modelo resultaría en menores tiempos de inferencia y menores requerimientos de memoria en el servidor donde se monte el sistema de predicción.

Glosario

aprendizaje profundo ver deep learning. (p. 1)

aprendizaje supervisado Proceso a tomar cuando se tiene un conjunto de datos y un resultado esperado asociado a cada uno de ellos. Uno de los dos tipos principales de problema en machine learning. (p. 8)

AWD LSTM Average-stochastic-gradient-descent Weight Dropped LSTM. (p. 36)

BERT Bi-directional Encoder Representations from Transformers. (p. 35)

CNN Convolutional Neural Network. (p. 46)

corpora Plural de *corpus*. Una conjunto de cuerpo de datos obtenidos generalmente con un fin académico. (p. 16)

CV Computer Vision. (p. 2)

deep learning Conjunto de técnicas que buscan resolver tareas a partir del aprendizaje de representación de datos. Área caracterizada por el uso de redes neuronales de arquitectura profunda. (p. 1)

entrenamiento Fase de aprendizaje de algún modelo. En esta fase se alimentan ejemplos con su debido resultado esperado y permite que con la retroalimentación de una función de perdida o de error el modelo mejore su desempeño. (p. 2)

idiolecto Conjunto de características que identifican la variante de lenguaje para un individuo. (p. 5)

ingeniería de características Proceso de elección de características esenciales para resolver un problema. Generalmente realizado por un experto del campo. (p. 4)

LEFTy Language Efficient Text portray. (p. 35)

LM Language Model. (p. 39)

LSTM Long Short Term Memory. (p. 14)

machine learning Es el campo que se encarga de estudiar algoritmos y modelos estadísticos y aplicarlos para resolver una tarea sin tener la necesidad de utilizar instrucciones explicitas relacionadas a la misma. (p. 1)

matriz de confusión Forma de presentar los resultados de un modelo clasificador. Permite encontrar distintos de error de forma muy eficiente. (p. 19)

MLM Masked Language Model. (p. 35)

Natural Language Processing Es el campo de las ciencias de computación e inteligencia artificial que maneja las interacciones entre computadoras y lenguajes naturales humanos, puntualmente en cómo procesar estos lenguajes. (p. 3)

NLP Natural Language Processing. (p. 2)

PAN Serie de eventos científicos y competencias que abordan tareas de naturaleza forense y de estilometría. (p. 33)

perfilamiento de autores La tarea de extraer caraceterísticas de un autor basado en un texto escrito por el mismo. (p. 2)

QRNN Quasi-recurrent Neural Network. (p. 25)

red neuronal Arquitectura que consta de unidades (neuronas) conectadas entre sí. Estas redes están ordenadas por capas, donde la capa de salida define a qué tipo de problema se pueden aplicar. Las redes neuronales se ha demostrado que son aproximadores universales. (p. 1)

RNN Recurrent Neural Network. (p. 3)

scraping El proceso de obtención de datos, generalmente de una página web, a partir de información desplegada por otro programa. (p. 16)

sociolecto Conjunto de características de una variante de lenguaje que comparte un subconjunto de hablantes relacionados por estatus social, etc. (p. 5)

SpanText Proyecto propuesto por investigadores argentinos que consta de un conjunto de textos formales catalogados por edad y género. (p. 33)

tasa de aprendizaje Ponderación que determina la magnitud en la que debe afectar el gradiente de una función de pérdida a los nuevos parámetros propuestos. Es responsable del ritmo y velocidad con la que se entrena una red neuronal. (p. 10)

TF-IDF Term Frequency – Inverse Document Frequency. (p. 46)

tokenización El proceso de separar texto plano en unidades discretas. Generalmente estas unidades representan palabras, raíces de palabras o caracteres. (p. 31)

transfer learning También conocido como transferencia de aprendizaje, se refiere al concepto de aprovechar conocimiento previamente adquirido en un modelo y aplicarlo en una tarea diferente pero relacionada — generalmente esta es una tarea más específica y/o acotada. (p. 3)

ULMFiT Universal Language Model Fine-tuning. Una técnica propuesta en el 2018 para poder afinar un modelo de lenguaje a una tarea diferente sin correr el riesgo de pérdida de conocimiento del modelo pre-entrenado. (p. 25)

vocabulario Conjunto de tokens que forman parte de los tokens conocidos por un modelo. (p. 31)

Bibliografía

Aggarwal, C. C. & Zhai, C. (2012). *Mining text data*. Springer Science & Business Media.

Bradbury, J., Merity, S., Xiong, C. & Socher, R. (2016). Quasi-Recurrent Neural Networks. *arXiv e-prints*, arXiv:1611.01576. arXiv: 1611.01576 [cs.NE]

Chen, S. F., Beeferman, D. & Rosenfeld, R. (1998). Evaluation metrics for language models. En *DARPA Broadcast News Transcription and Understanding Workshop* (pp. 275-280). Citeseer.

Coulthard, M. (2004). Author identification, idiolect, and linguistic uniqueness. *Applied linguistics*, *25*(4), 431-447.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V. & Salakhutdinov, R. (2019). Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv e-prints*, arXiv:1901.02860. arXiv: 1901.02860 [cs.LG]

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. En *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (pp. 248-255). Ieee.

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*.

Edmundson, H. P. (1969). New Methods in Automatic Extracting. *J. ACM*, *16*(2), 264-285. doi:10.1145/321510.321519

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P. & Bengio, S. (2010). Why Does Unsupervised Pre-training Help Deep Learning? *J. Mach. Learn. Res. 11*, 625-660. Recuperado desde http://dl.acm.org/citation.cfm?id=1756006.1756025

Franco-Salvador, M., Plotnikova, N., Pawar, N. & Benajiba, Y. (2017). Subword-based Deep Averaging Networks for Author Profiling in Social Media. En *CLEF (Working Notes)*.

Goldszmidt, G. (2018). fast.ai forums post. Recuperado el 31 de enero de 2019, desde https://forums.fast.ai/t/ulmfit-spanish/29715/4

Hoo-Chang, S., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... Summers, R. M. (2016). Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, *35*(5), 1285.

Hornik, K., Stinchcombe, M. & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, *2*(5), 359-366.

Howard, J. & Ruder, S. (2018). Universal language model fine-tuning for text classification. En *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Vol. 1, pp. 328-339).

Ingham, F. (2018). SpanishULMFiT Github. Recuperado el 31 de enero de 2019, desde https://github.com/fpingham/SpanishULMFit/blob/master/Wiki-Corpus.ipynb

Jordan, M. I. & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255-260.

Jurafsky, D. & Martin, J. H. (2014). *Speech and language processing*. Pearson London.

Kodiyan, D., Hardegger, F., Neuhaus, S. & Cieliebak, M. (2017). Author Profiling with bidirectional rnns using Attention with grus: Notebook for PAN

at CLEF 2017. En *CLEF 2017 Evaluation Labs and Workshop–Working Notes Papers, Dublin, Ireland, 11-14 September 2017* (Vol. 1866). RWTH Aachen.

Kupiec, J., Pedersen, J. & Chen, F. (1995). A trainable document summarizer. En *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 68-73). ACM.

LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

Lei, N., Luo, Z., Yau, S.-T. & Xianfeng Gu, D. (2018). Geometric Understanding of Deep Learning. *arXiv e-prints*, arXiv:1805.10451. arXiv: 1805.10451 [cs.LG]

Louwerse, M. M. (2004). Semantic variation in idiolect and sociolect: Corpus linguistic evidence from literary texts. *Computers and the Humanities*, *38*(2), 207-221.

Malmasi, S. & Dras, M. (2018). Native Language Identification With Classifier Stacking and Ensembles. *Computational Linguistics*, *44*(3), 403-446. Recuperado desde http://aclweb.org/anthology/J18-3003

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A. & Tiedemann, J. (2016). Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. En *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)* (pp. 1-14).

Merity, S., Keskar, N. S. & Socher, R. (2017). Regularizing and Optimizing LSTM Language Models. *arXiv preprint arXiv:1708.02182*.

Merity, S., Xiong, C., Bradbury, J. & Socher, R. (2016). Pointer Sentinel Mixture Models. *arXiv e-prints*, arXiv:1609.07843. arXiv: 1609.07843 [cs.CL]

Miura, Y., Taniguchi, T., Taniguchi, M. & Ohkuma, T. (2017). Author Profiling with Word+ Character Neural Attention Network. En *CLEF (Working Notes)*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. & Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv e-prints*, arXiv:1802.05365. arXiv: 1802.05365 [cs.CL]

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners.

Rae, J. W., Dyer, C., Dayan, P. & Lillicrap, T. P. (2018). Fast Parametric Learning with Activation Memorization. *arXiv e-prints*, arXiv:1803.10049. arXiv: 1803.10049 [cs.LG]

Rangel, F., Rosso, P., Potthast, M. & Stein, B. (2017a). Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. En *Cappellato L., Ferro N., Goeuriot L, Mandl T. (Eds.) CLEF 2017 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings (CEUR)* (Vol. 1866). Recuperado desde CEUR-WS.org

Rangel, F., Rosso, P., Potthast, M. & Stein, B. (2017b). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. *Working Notes Papers of the CLEF*.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533.

Schaetti, N. (2017). UniNE at CLEF 2017: TF-IDF and Deep-Learning for Author Profiling. En *CLEF (Working Notes)*.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, *61*, 85-117.

Sierra, S., Montes-y-Gómez, M., Solorio, T. & González, F. A. (2017). Convolutional Neural Networks for Author Profiling. *Working Notes of the CLEF*.

Smith, L. N. & Topin, N. (2017). Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates. *arXiv e-prints*, arXiv:1708.07120. arXiv: 1708.07120 [cs.LG]

Tan, L., Zampieri, M., Ljubešic, N. & Tiedemann, J. (2014). Merging Comparable Data Sources for the Discrimination of Similar Languages: The DSL Corpus Collection. En *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)* (pp. 11-15). Reykjavik, Iceland.

Villegas, M. P., Ucelay, M. J., Errecalde, M. L. & Cagnina, L. C. (2014). A Spanish text corpus for the author profiling task. En *XX Congreso Argentino de Ciencias de la Computación (CACIC 2014)*, Buenos Aires, Argentina.

Werbos, P. J. (1982). Applications of advances in nonlinear sensitivity analysis. En *System modeling and optimization* (pp. 762-770). Springer.

Yosinski, J., Clune, J., Bengio, Y. & Lipson, H. (2014). How transferable are features in deep neural networks? *arXiv e-prints*, arXiv:1411.1792. arXiv: 1411.1792 [cs.LG]

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J., ... Aepli, N. (2017). Findings of the VarDial Evaluation Campaign 2017. En *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)* (pp. 1-15). Valencia, Spain: Association for Computational Linguistics. Recuperado desde http://www.aclweb.org/anthology/W/W17/W17-1201

Zeiler, M. D. & Fergus, R. (2014). Visualizing and understanding convolutional networks. En *European conference on computer vision* (pp. 818-833). Springer.