Universidad de Murcia
Facultad de Informática

GRADO EN INGENIERÍA INFORMÁTICA

Aplicación de redes neuronales pre-entrenadas para diagnóstico de accidente cerebrovascular isquémico agudo mediante Transfer Learning

Trabajo Fin de Grado realizado por: Francisco José López Jiménez

Bajo la dirección de: D. Gregorio Bernabé García

Enero, 2021.

Esta página ha sido intencionadamente dejada en blanco

Declaración firmada sobre la originalidad del trabajo

D. Francisco José López Jiménez, con DNI 48694100H, estudiante del Grado Ingeniería Informática de la Universidad de Murcia, y autor del Trabajo Fin de Grado titulado "Aplicación de redes neuronales pre-entrenadas para diagnóstico de accidente cerebrovascular isquémico agudo mediante Transfer Learning". Declaro que: El trabajo presentado para su evaluación es original y ha sido desarrollado y elaborado por mí. Las fuentes consultadas y utilizadas han sido citadas de manera clara y sin violar ningún derecho de propiedad intelectual.

Murcia, a 21 de Enero de 2021

Esta página ha sido intencionadamente dejada en blanco

Índice

Índice de figuras

Índice de tablas

Resumen

Como consecuencia del incremento de la capacidad computacional y del incremento exponencial de los datos generados en la última década se ha producido un gran avance en el progreso y uso de la Inteligencia Artificial consecuencia de un gran incremento en la investigación, trabajo e inversión. Este incremento del uso de la inteligencia artificial se ha producido en diferentes ámbitos como el transporte, la industria, la sanidad, el entretenimiento, las finanzas…etcétera.

En este proyecto nos centraremos en un área que está ganando protagonismo en los últimos años, el procesado de lenguaje natural, también conocido por sus siglas en inglés NLP (*Natural Language Processing*). El NLP permite a los ordenadores entender, interpretar y trabajar con lenguaje humano.

En el ámbito clínico se genera una gran cantidad de texto desestructurado escrito en lenguaje natural, que queda en desuso en su gran mayoría. Queremos estudiar el rol que puede desempeñar el procesado de lenguaje natural en ayudar a profesionales de la salud a tomar decisiones clínicas, en concreto la red neuronal pre-entrenada BERT.

BERT se trata de una red pre-entrenada por Google para procesado de lenguaje natural que obtiene resultados del estado del arte en diferentes tareas de procesado de lenguaje natural. Esta red neuronal ha sido pre-entrenada con textos que contienen cientos de millones de palabras, y su principal característica distintiva es que es capaz de contextualizar una palabra bidireccionalmente con gran profundidad, para conocer o interpretar el significado de una palabra no solo se considera la palabra de manera aislada sin considerar el contexto como en otros modelos, ni siquiera el contexto que aparece a un lado u otro de la palabra, sino que se considera el contexto que aparece a ambos lados de la palabra con gran profundidad para determinar el significado de dicha palabra.

Una de las técnicas que ha ganado protagonismo en inteligencia artificial en los últimos años es el *transfer learning*, que consiste en aprovechar el conocimiento adquirido por un modelo en una tarea para otra tarea diferente.

Emplearemos BERT para nuestra tarea usando *transfer learning.* Agregamos una capa de nodos al final del modelo pre-entrenado que proporciona Google y ajustamos la red neuronal con los datos de nuestra tarea, de esta manera somos capaces de obtener resultados muy buenos gracias al conocimiento que ya tenía el modelo pre-entrenado sin hacer grandes cambios en la arquitectura del modelo.

Este trabajo fin de grado tiene como principal objetivo estudiar la aplicación de una red neuronal pre-entrenada, en este caso BERT, para un problema de clasificación de textos en el ámbito clínico mediante *transfer learning*, y comparar los resultados con otros algoritmos implementados por investigadores en esa misma tarea. Considerando que el modelo contextualiza con gran precisión es presumible que obtengamos buenos resultados, y como BERT es de código abierto podemos usarlo.

En concreto, nuestro propósito será identificar a pacientes con accidente cerebrovascular isquémico agudo (AIS) basándonos en texto desestructurado existente en informes radiológicos de resonancias magnéticas cerebrales, identificando si un texto se corresponde con una persona que tiene accidente cerebrovascular isquémico agudo (AIS) o no (NO-AIS), por lo que el cometido es una tarea de clasificación de textos en dos categorías.

Los textos a clasificar pertenecen a una base de datos que ya fue usada por Kim et al. (2019), cuando aplicaron diversos algoritmos para clasificación de textos y se compararon los resultados de los diferentes algoritmos. Cabe mencionar que están escritos en inglés y que contienen las descripciones y hallazgos que hicieron radiólogos a partir de resonancias magnéticas del cerebro. Los radiólogos llevaron a cabo esta tarea sin saber si el paciente tenía accidente cerebrovascular isquémico agudo o no.

Los algoritmos aplicados sobre los textos por los investigadores cuyo rendimiento fue testado previamente a este trabajo fueron regresión binaria logística, clasificador ingenuo Bayesiano, árbol de decisión único y máquinas de vector soporte (SVM). Nuestra implementación con BERT ha obtenido mejores resultados que el mejor algoritmo de los cuatro citados previamente, que era el árbol de decisión.

Por tanto, en este trabajo hemos visto que añadiendo una capa de nodos al final del modelo pre-entrenado de BERT y ajustándolo para hacer clasificación obtenemos resultados similares que el mejor de cuatro algoritmos probados por investigadores académicos sobre la misma base de datos y llevando a cabo la misma tarea, clasificar si un texto de informes radiológicos se corresponde con una persona que sufre accidente cerebrovascular isquémico agudo (AIS) o no. Demostramos que ésta red neuronal puede obtener resultados excelentes para procesado de lenguaje natural en el ámbito clínico empleando relativamente poco esfuerzo tanto a nivel de tiempo como a nivel computacional.

Extended Abstract

As a consequence of the increase in computational capacity and the exponential increase in the data generated in the last decade, there has been a great advance in the progress and use of Artificial Intelligence as a consequence of a great increase in research, work and investment. This increase in the use of artificial intelligence has occurred in different areas such as transport, industry, health, entertainment, finance ... and so on.

In this project we will focus on an area that is gaining prominence in recent years, natural language processing, also known by its acronym in English NLP (Natural Language Processing). NLP enables computers to understand, interpret, and work with human language.

In the clinical field, a large amount of unstructured text written in natural language is generated, which is largely unused. We want to study the role that natural language processing can play in helping healthcare professionals make clinical decisions, specifically the BERT pre-trained neural network.

BERT is a neural network pre-trained by Google for natural language processing that obtains state of the art results in different natural language processing tasks. This neural network has been pre-trained with texts that contain hundreds of millions of words, and its main distinguishing feature is that it is capable of contextualizing a word bidirectionally with great depth, to know or interpret the meaning of a word is not only considered the word in isolation without considering the context as in other models, not even the context that appears on one side or the other of the word, but the context that appears on both sides of the word is considered in great depth to determine the meaning of said word.

One of the techniques that has gained prominence in artificial intelligence in recent years is transfer learning, which consists of taking advantage of the knowledge acquired by a model in one task for a different task.

We will use BERT for our task using transfer learning. We add a layer of nodes at the end of the pre-trained model provided by Google and adjust the neural network with the data from our task, in this way we are able to obtain very good results thanks to the knowledge that the pre-trained model already had without making major changes in the architecture of the model.

The main objective of this final project is to study the application of a pre-trained neural network, in this case BERT, for a text classification problem in the clinical domain using transfer learning, and compare the results we obtain with other algorithms implemented by researchers in the same task. Considering that the model contextualizes with great precision we can assume that we will obtain good results, and as BERT is open source we can use it.

Specifically, our purpose will be to identify patients with acute ischemic stroke (AIS) based on existing unstructured text in radiological reports of brain magnetic resonance imaging, identifying whether a text corresponds to a person who has acute ischemic stroke (AIS) or not. (NO-AIS), so the task is a task of classifying texts into two categories.

The texts to be classified belong to a database that was already used by Kim et al. (2019), when they applied various algorithms for text classification and the results of the different algorithms were compared. It is worth mentioning that they are written in English and that they contain the descriptions and findings made by radiologists from magnetic resonance imaging of the brain.

Radiologists carried out this task without knowing whether the patient had acute ischemic stroke or not.

The algorithms applied on the texts by the researchers whose performance was tested prior to this work were logistic binary regression, naive Bayesian classifier, single decision tree, and support vector machines (SVM). Our implementation with BERT has obtained better results than the best algorithm of the four mentioned previously, which was the decision tree.

Therefore, in this work we have seen that by adding a layer of nodes to the end of the pre-trained BERT model and adjusting it for classification we obtain results as good or better than the best of four algorithms tested by academic researchers on the same database and the same task, classifying whether a radiological report text corresponds to a person suffering from acute ischemic stroke (AIS) or not. We show that this neural network can obtain excellent results for natural language processing in the clinical domain using relatively little effort both in time and computational resources.

1. Introducción

1.1 Motivación del proyecto

Como consecuencia del incremento de la capacidad computacional y del incremento exponencial de los datos generados [1] en la última década se ha producido un gran avance en el progreso y uso de la inteligencia artificial en forma de un gran incremento en la investigación, trabajo e inversión.

Este incremento del uso de la inteligencia artificial se ha producido en diferentes ámbitos como el transporte, la industria, la sanidad, el entretenimiento, las finanzas…etcétera [2]. Se estima que este incremento del uso sea aún más pronunciado en los próximos años [3] .

Una parte importante de la inteligencia artificial es el *machine learning*, dentro del cual se encuentra *deep learning*, que como veremos más adelante se tratan de redes neuronales con una arquitectura determinada. Estas redes neuronales han sido aplicadas a diversas áreas superando en muchos casos a los humanos [4]. Entre estas áreas se encuentran visión por ordenador, reconocimiento de voz, procesado de lenguaje natural, reconocimiento de audio.

En este proyecto nos centraremos en un área que está ganando protagonismo en los últimos años, el procesado de lenguaje natural, también conocido por sus siglas en inglés NLP (*Natural Language Processing*). El NLP permite a los ordenadores entender, interpretar y trabajar con lenguaje humano [5]. En nuestro caso trabajaremos con datos desestructurados expresados en lenguaje escrito y comprendido por humanos, aunque se podría utilizar también para interpretar lenguaje hablado.

Intentaremos estudiar la aplicación del NLP para el apoyo de decisiones clínicas (CDS), cuyo objetivo es “ayudar a los profesionales de la salud a tomar decisiones clínicas, tratar con datos médicos sobre pacientes o con el conocimiento de la medicina necesario para interpretar estos datos” [6].

Una de las técnicas que ha ganado protagonismo en inteligencia artificial en los últimos años es el *transfer learning*, que consiste en aprovechar el conocimiento adquirido por un modelo sobre una tarea para otra tarea diferente [7]. Intentaremos ver si esta técnica puede tener éxito en el ámbito clínico y por qué.

En este trabajo usaremos una red neuronal pre-entrenada por Google llamada BERT que se emplea para procesamiento de lenguaje natural escrito, y aprovecharemos ese conocimiento para intentar predecir si el paciente tiene accidente cerebrovascular isquémico agudo o no en base al texto que pertenece a un informe que describe lo que aparece en una resonancia magnética del cerebro. De esta manera estudiaremos la utilidad de esta red neuronal que se emplea en el buscador más usado del mundo [8] en el ámbito clínico, aplicando *transfer learning*.

1.2 Objetivo del proyecto

Este trabajo fin de grado tiene como principal objetivo estudiar la aplicación de una red neuronal pre-entrenada para un problema de clasificación de textos en el ámbito clínico mediante *transfer learning*, y comparar los resultados con otros algoritmos aplicados previamente en esa misma tarea.

En el ámbito clínico se genera una gran cantidad de texto desestructurado escrito en lenguaje natural, por lo que resulta interesante estudiar el rol que puede desempeñar el procesado de lenguaje natural en ayudar a profesionales de la salud a tomar decisiones clínicas, en concreto la red neuronal pre-entrenada BERT.

BERT se trata de una red pre-entrenada por Google para procesado de lenguaje natural que describiremos más en detalle posteriormente. Por tanto, el enfoque del desarrollo del proyecto y la base de datos usada han estado condicionados al uso de esta red neuronal.

Con respecto a la base de datos usada, hemos estado limitados tanto por la poca disponibilidad de bases de datos de textos en el ámbito clínico como porque los textos deben de ser de un tamaño determinado, pues BERT por su arquitectura no puede trabajar con inputs muy grandes. Los textos a clasificar pertenecen a una base de datos que ya fue usada por Kim et al. (2019) [9], cuando aplicaron diversos algoritmos para el diagnóstico de accidente cerebrovascular isquémico agudo y se compararon los resultados de los diferentes algoritmos. Veremos cuál es el resultado de BERT y cómo se compara a los otros algoritmos que se usaron en la investigación citada.

Por consiguiente, usaremos BERT para clasificar textos que describen lo que aparece en informes radiológicos del cerebro, teniendo cada uno de estos textos su correspondiente etiqueta que los clasifica en textos de una persona que tiene accidente cerebrovascular isquémico agudo (AIS) y textos que se corresponden con una persona que no tiene AIS.

Esta adaptación e implementación de BERT sería usada en el ámbito clínico por profesionales sanitarios, aunque nuestro principal objetivo en este trabajo es estudiar su implementación y resultados y ver si puede tener utilidad.

Una vez entrenada apropiadamente la red neuronal con los textos de los informes radiológicos el futuro uso de la misma sería darle un texto que describa lo que aparece en informes radiológicos del cerebro como los vistos en el ejemplo, y que la red neuronal predijera si se trata de una persona que sufre accidente cerebrovascular isquémico agudo o no.

En resumen, los objetivos que se persiguen en el trabajo son los siguientes:

- Estudiar el estado del arte del procesado de lenguaje natural en el ámbito clínico.
- Estudiar el estado del arte de las redes neuronales aplicadas al procesado de lenguaje natural en el ámbito clínico.
- Estudio de la red neuronal pre-entrenada BERT.
- Estudiar la aplicación de BERT para la resolución de un problema de aprendizaje supervisado de clasificación de textos en medicina mediante *transfer learning*.
- Comparar los resultados obtenidos por BERT en comparación a otros algoritmos implementados por investigadores previamente.

1.3 ¿Por qué aporta valor el procesado de lenguaje natural (NLP) al ámbito clínico?

La adopción del NLP en el ámbito clínico está creciendo debido a su potencial para buscar, analizar e interpretar grandes cantidades de datos de pacientes [10]. La madurez del desarrollo de métodos de NLP y los resultados del estado del arte obtenidos han llevado a un incremento de desarrollos exitosos de soluciones de procesado de lenguaje natural para investigaciones clínicas complejas, teniendo como consecuencia un incremento de la importancia atribuida a incorporar métodos de NLP en el ámbito clínico .

El uso de algoritmos médicos avanzados, *machine learning* en el ámbito clínico y el NLP tiene el potencial de aprovechar los conocimientos y conceptos enfrascados en grandes cantidades de datos que previamente tenían poca utilidad, debido a que no se contaba con las herramientas capaces de utilizar esos datos de forma valiosa. De hecho, un problema existente en el ámbito de la salud es que el 80% de la cantidad de datos en medicina es desestructurado y queda en desuso después de que sea creado, bien sea en formato de texto, imágenes, etcétera [11]. El NLP puede ayudar a poder hacer un uso eficaz de dicha cantidad de datos.

Las oportunidades y el potencial del procesado de lenguaje natural son enormes para la investigación en el ámbito clínico en general y para la investigación de salud mental en particular. A medida que los recursos de informática clínica se vuelven más grandes y más completos la posibilidad de determinar los resultados, los pronósticos, la efectividad y el daño causado está más cerca, requiriendo menos recursos de los que se necesitarían para realizar estudios de investigación primaria. Los datos que más valor inmediato ofrecen son los derivados de la historia electrónica clínica, lo que es comprensible teniendo en cuenta el número de pacientes sobre el que se han estado recogiendo, la amplitud del intervalo de tiempo en el que se han recogido, su uso en el ámbito clínico y la profundidad de la información clínica disponible en casos reales.

Es particularmente importante la capacidad del procesado de lenguaje natural para extraer información adicional no estructurada en estudios de investigación con muestras grandes, que frecuentemente se concentran en identificar tantos predictores (que son potenciales factores de confusión) de un resultado como sea posible. Estos predictores pueden incluir factores del 'macro-entorno' como circunstancias familiares, o factores a nivel individual como uso de tabaco. Éstos son especialmente importantes en el campo de la investigación mental, ya que puede haber ciertas reticencias a especificar condiciones que estigmaticen, como el uso ilícito de drogas cuando no son la razón primaria por la que se realiza el tratamiento.

Además, los códigos estructurados pueden no acomodarse a la incertidumbre de un diagnóstico y no permitir dejar constancia de información relevante a nivel clínico que apoya un diagnóstico, como por ejemplo el humor o los hábitos de sueño si no son la principal razón por la que se recibe tratamiento. Por tanto, el procesado de lenguaje natural mejora la identificación de casos a partir de registros de salud, y nos puede proporcionar un conjunto de datos más rico que el que obtendríamos usando sólo datos estructurados.

En la Figura 1 vemos 21 aplicaciones clínicas distribuidas en 7 tareas de NLP y podemos observar para qué se está usando actualmente el NLP en el ámbito clínico. Como podemos ver, la inmensa

mayoría de estudios se centran en usar procesado de lenguaje natural para hacer tareas de clasificación de textos.

	Classification	Clustering	Coreference resolution	Information extraction	Named entity recognition	Ranking	Word sense disambiguation	Total
Care improvement	8			2				10
Comparative effectiveness	1							1
Data management	2			1	1			4
Diagnosis	2							2
Efficiency	1							1
Enabling	7		2	4	14		3	30
Interactive NLP	1			1				2
Knowledge acquisition				1				1
Patient literacy						1		1
Pharmacovigilance	3			1				4
Phenotyping	13							13
Prognosis	13			3				16
Quality	2							2
Referral	1							1
Resource management	8							8
Risk prediction	1							1
Safety	4							4
Service improvement	1							1
Surveillance	5			1	1			7
Triage	1	2		1	1			5
Unclear	1							1
Total	**75**	**2**	**2**	**15**	**17**	**1**	**3**	

Figura 1. Tareas para las que se emplea el NLP en el ámbito clínico [12]

1.4 ¿Por qué BERT?

Primero entraremos a definir brevemente lo que es BERT, aunque lo veremos con más profundidad en apartados posteriores. BERT se trata de una red neuronal pre-entrenada por Google sobre grandes cantidades de texto en diferentes idiomas, cuya característica más destacada es la capacidad de contextualizar una palabra o frase en función de lo que aparezca a ambos lados de la misma con gran profundidad y precisión [13]. Sin embargo, en este apartado nos centraremos en ver por qué es importante BERT, qué tiene que lo hace especial.

En primer lugar, BERT ha sido pre-entrenado en una cantidad de texto enorme en múltiples idiomas. Por ejemplo, en inglés el modelo BERT-Base ha sido entrenado con los 800 millones de palabras existentes en BookCorpus, un conjunto de datos con más de 10.000 libros, mientras que en el caso del modelo BERT-Large que tiene mayor tamaño ha sido entrenado con la Wikipedia en inglés, que contiene más de 12.500 millones de palabras. Por tanto, tiene una gran representación de las palabras existentes en un idioma.

En segundo lugar, es de destacar que es capaz de contextualizar una palabra bidireccionalmente con gran profundidad. Para saber o interpretar el significado de una palabra no solo se considera la palabra de manera aislada sin considerar el contexto como en otros modelos, ni siquiera el contexto que aparece a un lado u otro de la palabra, sino que se considera el contexto que aparece a ambos lados de la palabra con gran profundidad para determinar el significado de dicha palabra. Por ejemplo, otros modelos devolverían la misma codificación para la palabra

'banco' en las frases de ejemplo que veremos a continuación, o considerarían solo el lado izquierdo o derecho de 'banco' para saber su significado, mientras que BERT devuelve una codificación diferente de manera acertada (ya que 'banco' significa cosas diferentes dependiendo del contexto) consecuencia de considerar el contexto a ambos lados.

Frases de ejemplo en las que la palabra 'banco' tiene significados diferentes dependiendo del contexto.

Frase 1: Voy a sacar dinero al banco.

Frase 2: El anciano se sentó en un banco porque estaba cansado.

Frase 3: Al cruzar el puente los niños vieron un banco de peces.

En tercer lugar, BERT es *open source*, por lo que podemos usarlo con facilidad.

Teniendo en cuenta que el modelo está pre-entrenado en una gran cantidad de datos podemos hacer *transfer learning* y aprovechar dicho conocimiento añadiendo una capa de *ouput* al final de BERT, de esta manera podemos alcanzar resultados del estado del arte para un amplio rango de tareas de procesado de lenguaje natural con diferentes conjuntos de datos sin hacer grandes cambios en la arquitectura del modelo. Considerando que el modelo contextualiza con gran precisión es presumible que obtengamos buenos resultados, y como es *open source* podemos usarlo [14].

Vamos a comparar los resultados obtenidos por BERT cuando fue publicado en 2019 con el estado del arte en varias tareas de procesado de lenguaje natural.

Empezamos viendo los resultados en GLUE, un *benchmark* de nueve tareas diferentes de entendimiento de las relaciones entre sentencias [15]. Estas tareas cubren un rango diverso de tamaño de conjuntos de datos, géneros de texto y grados de dificultad.

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$BERT_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Figura 2. Test de resultados en GLUE [13]

Como observamos arriba en la tabla, BERT supera en todos los conjuntos de datos y tareas los resultados previamente establecidos por otros modelos.

Procedemos a analizar los resultados en SQuAD v2.0, el conjunto de datos de la universidad de Stanford para hacer tareas en las cuales un sistema entrenado sobre texto debe de responder posteriormente preguntas relacionadas con lo leído. Se trata de una colección de 100.000 parejas de pregunta-respuesta en la cual dada la pregunta y un trozo de texto de la Wikipedia

que contiene la respuesta el modelo debe de acertar la respuesta correcta. Se diferencia del SQuAD v1.1 en que en la v2.0 es posible que no exista una respuesta corta en ese texto que se proporciona, haciendo el problema más semejante a la realidad.

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
$\text{BERT}_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

Figura 3. Test de resultados en SquAD v2.0 [13]

Observamos en la Figura 3 que BERT mejora todos los resultados previamente establecidos otra vez, y es capaz de responder con gran precisión a las preguntas formuladas sobre ese conjunto de datos.

2. Estado del arte

2.1 Redes neuronales

2.1.1 Origen redes neuronales

En 1943 el neurofisiólogo Warren McCulloch y el matemático Walter Pitts elaboraron un trabajo hablando sobre cómo las neuronas podrían funcionar. Crearon un modelo informático para redes neuronales basado en algoritmos que se llama lógica umbral, y abrió el camino para la investigación de redes neuronales en el ámbito biológico del cerebro, pero también en la aplicación de redes neuronales de cara a la inteligencia artificial [16].

En 1949 Donald Hebb escribió "*The Organization of Behavior*", en el que señalaba el hecho de que las uniones entre neuronas se fortalecían cada vez que se usaban, un concepto fundamental a la hora entender cómo funciona el aprendizaje de los humanos [17] .Hebb creó una hipótesis de aprendizaje basado en el mecanismo de plasticidad neuronal que ahora se conoce como aprendizaje de Hebb. Se trata de aprendizaje no supervisado que posteriormente evolucionó en modelos de potenciación a largo plazo.

En 1954 Farley y Clark fueron los primeros en usar máquinas computacionales, entonces llamadas "calculadoras" con la finalidad de simular una red de Hebb.

En 1958 Rosenblatt creó el perceptrón, un clasificador binario que con las entradas que recibía y tras la aplicación de un algoritmo generaba una predicción [18].

Widrow y Hoff desarrollaron a principios de los 60 el algoritmo LMS(Lest-Mean-Square), un proceso de aprendizaje que examina el valor antes de hacer ajustes en los pesos asociados a los nodos de las redes neuronales. Se basa en la idea de que mientras un perceptrón activo puede tener un gran error, uno puede ajustar los valores de los pesos para distribuirlo a lo largo de la red, o al menos a los perceptrones adyacentes, y llevaría posteriormente al desarrollo de las redes neuronales ADALINE y MADALINE y del algoritmo de *backpropagation* [19].

Dichas redes neuronales ADALINE y MADALINE fueron desarrolladas en ese periodo de tiempo por parte de los mismos Bernard Widrow y Marcian Hoff en Stanford. ADALINE se desarrolló para reconocer patrones binarios con el fin de predecir el siguiente bit si estaba leyendo una transmisión desde una línea telefónica. MADLINE fue la primera red neuronal que se aplicó a un problema real, usando un filtro adaptativo que elimina los ecos de las líneas telefónicas.

La primera red neuronal funcional de múltiples capas fue publicada por Ivakhnenko y Lapa en 1965.

Posteriormente la investigación sobre redes neuronales se estancó debido a una investigación sobre aprendizaje automático elaborada por Marvin Minsky y Seymour Papert en 1969. En dicha investigación descubrieron dos aspectos claves acerca de las máquinas computacionales que procesan las redes neuronales. El primer aspecto es que los perceptrones básicos eran incapaces de procesar el circuito de o-exclusivo, mientras que el segundo aspecto era que los ordenadores en ese momento no tenían suficiente poder de computación para poder realizar el trabajo requerido por una red neuronal.

En 1982 creció la inversión y la investigación en las redes neuronales de nuevo debido a que hubo una conferencia conjunta entre Japón y Estados Unidos sobre el tema en la que Japón anunció un esfuerzo de quinta generación en las redes neuronales y los Estados Unidos temían quedarse detrás.

En 1986 el problema era cómo extender el algoritmo de actualización descubierto en los 60 por Widrow y Hoff a múltiples capas. Rumelhart, Hinton and William popularizaron el algoritmo de *backpropagation*. Este algoritmo actualiza desde el final de la red neuronal hacía atrás los pesos de las conexiones entre diferentes nodos de los diferentes *layers* considerando la diferencia entre el resultado producido por la red neuronal y el resultado que debería haber tenido.

En 2010 Ciresan et al. Mostraron en 2010 que las GPUs hacían posible la aplicación de *backpropagation* para redes neuronales con varias capas. Entre 2009 y 2012 las redes neuronales artificiales empezaron ganando premios y acercándose a los resultados obtenidos por humanos en diversas tareas de *machine learning*.

2.1.2 Conceptos básicos sobre redes neuronales

Biológicamente, una neurona es una célula especializada y caracterizada por poseer un cuerpo celular, llamado soma; una cantidad indefinida de canales de entrada llamados dendritas que transmiten impulsos nerviosos hacia el soma celular, y un canal de salida llamado axón que entrega la información de la neurona desde el soma de una neurona hacía dendritas de otras neuronas, que reciben esta información como señal de entrada [20]. Como se puede inducir de esta definición, una neurona sola y aislada carece de razón de ser.

Como hemos visto previamente, en 1958 Frank Rosenblatt descubrió el perceptrón, que es un modelo simplificado de una neurona biológica.

Como podemos apreciar en la imagen de abajo, el perceptrón recibe un número indefinido de inputs x. Cada input se verá multiplicado por el peso correspondiente w que tenga ese input en la conexión con el perceptrón, dicho peso determina la importancia de este input en la entrada al perceptrón. El perceptrón efectúa un sumatorio de todos los inputs multiplicados por sus respectivos pesos y sobre eso aplica una función. Una vez aplicada la función eso será la salida hacia otras neuronas, salida que tendrá también su peso asociado.

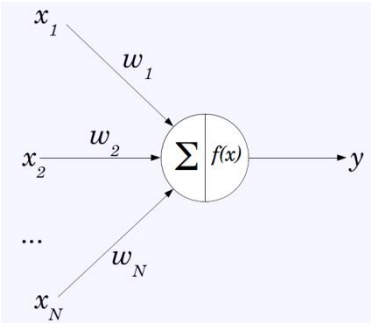

Figura 4. Perceptrón [21]

En notación matemática, la entrada de un perceptrón se representaría de la siguiente manera.

$$\sum_{i=1}^{m} w_i x_i,$$

Figura 5. Notación matemática Perceptrón.

Los perceptrones por si mismos sólo pueden resolver problemas linealmente separables, por lo que para resolver problemas más complejos se agrupan con otros perceptrones conectando las salidas de unos perceptrones con las entradas de otros, formando redes neuronales [22] .

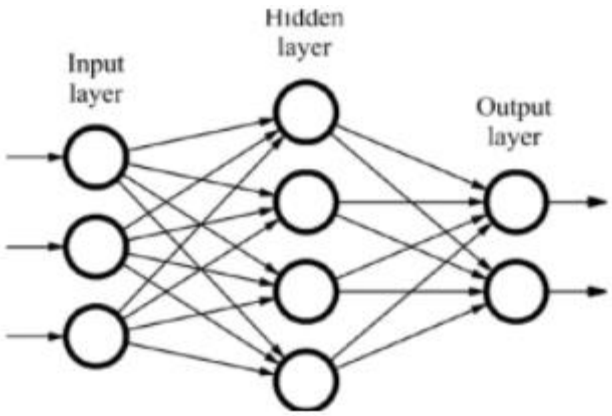

Figura 6. Red neuronal [23]

Layer

Un *layer*, también llamado 'capa' en español, se trata de un grupo de neuronas que se disponen gráficamente en una red neuronal alineados en forma de columna. Cada neurona se encuentra en un *layer* correspondiente y cada *layer* tiene su propósito [24].

El layer de entrada o *input layer* se encarga de recibir los inputs para la red neuronal, por lo que siempre debe de haber un *input layer* en una red neuronal. Una vez ha recibido los inputs la red neuronal hace una serie de cálculos a través de las neuronas que componen esta capa y los output de esta capa son transmitidos a la capa siguiente.

Un *hidden layer* es un *layer* que se encuentra entre el *input layer* y el *output layer* de la red neuronal, de ahí el nombre de 'hidden', cuya traducción al español es 'escondido'.

El *output layer* es responsable de producir el resultado final, por lo que debe de haber siempre un *output layer* en una red neuronal. Recibe los inputs de las capas predecesoras, hace los cálculos a través de las neuronas de este *layer* y da salida a los valores de output.

En una red neuronal siempre habrá un *input layer* y un *output layer*. Puede haber cero o más *hidden layers*.

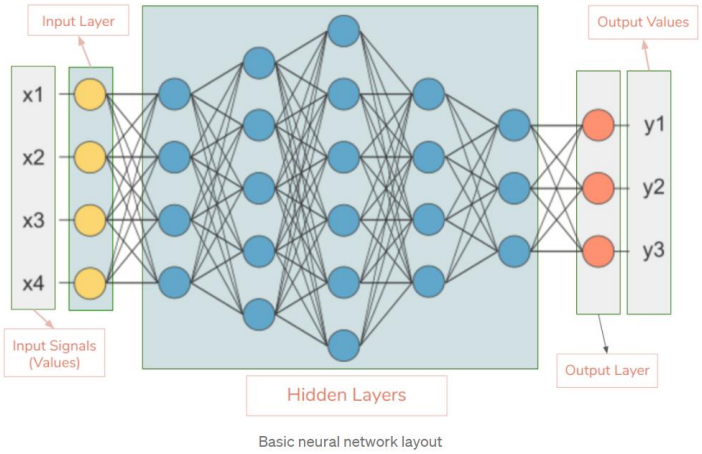

Figura 7. Red neuronal con input, hidden y output layers [25]

2.1.3 Definición de Deep Learning

El adjetivo “deep” en deep learning se refiere al número de capas que tiene una red neuronal. Una red neuronal que tiene más de tres capas, incluyendo la capa de input y la capa de output, se puede considerar un algoritmo de Deep Learning [26].

Por tanto, una red neuronal que tenga más de un hidden layer se considera ‘deep learning’.

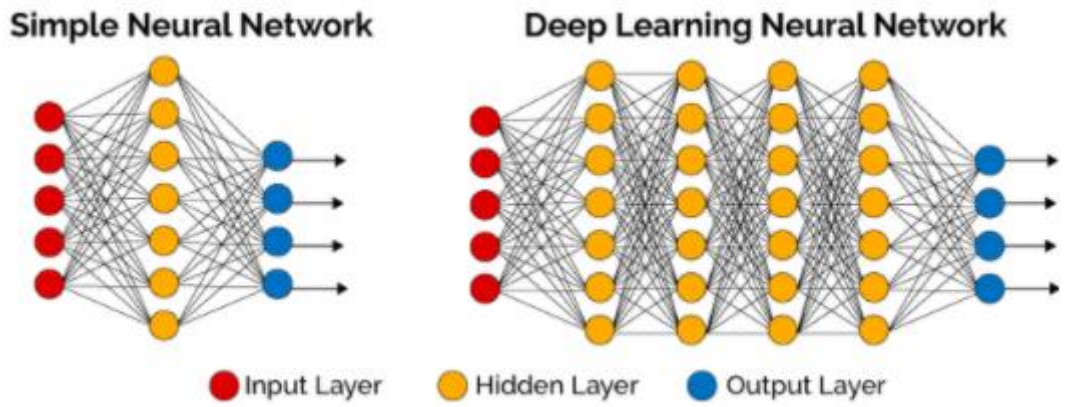

Figura 8. Imagen Deep learning [27].

2.1.4 Tipos de redes neuronales

A la hora de clasificar los tipos de redes neuronales, podemos dividir en función del número de capas, del tipo de conexiones entre las neuronas y del grado de esas conexiones entre las neuronas [28].

Según el número de capas:

- Redes neuronales monocapa: Se tratan de redes con una capa de entrada que no hace ningún cálculo y una capa de salida.
- Redes neuronales multicapa: Consisten en redes neuronales que tienen capas intermedias entre la entrada y la salida (capas ocultas). Pueden estar total o parcialmente conectadas.

Según el tipo de conexiones:

- Redes neuronales no recurrentes: La propagación de información se da en un sentido, sin ningún tipo de retroalimentación de información, por lo que no tienen memoria.
- Redes neuronales recurrentes: En estas redes neuronales sí existen retroalimentación de la información entre neuronas, bien sean neuronas de la misma capa, de distinta capa o incluso a la misma neurona. Por su retroalimentación poseen de memoria.

Según el grado de conexión:

- Redes neuronales totalmente conectadas: En este caso todas las neuronas de una capa se encuentran conectadas con las de la capa siguiente.
- Redes parcialmente conectadas: En este tipo de redes no hay total conexión entre las neuronas de diferentes capas.

Una vez vista la clasificación general de las redes neuronales, vamos a ver más en detalle los tipos de redes neuronales que tienen importancia de cara a este trabajo.

2.1.4.1 Red neuronal *feedforward* y MLP

Una red neuronal *feedforward*, en ocasiones llamada red neuronal tradicional, se trata de un tipo simple de red neuronal que contiene múltiples neuronas organizadas en *layers*, teniendo los nodos de diferentes *layers* conexiones entre ellos con pesos asociados [29]. La información se mueve solo en una dirección, desde el input hacia el output pasando por los *hidden layers*, en caso que existan. No hay ciclos o bucles en estas redes como sí existen por ejemplo en las RNN.

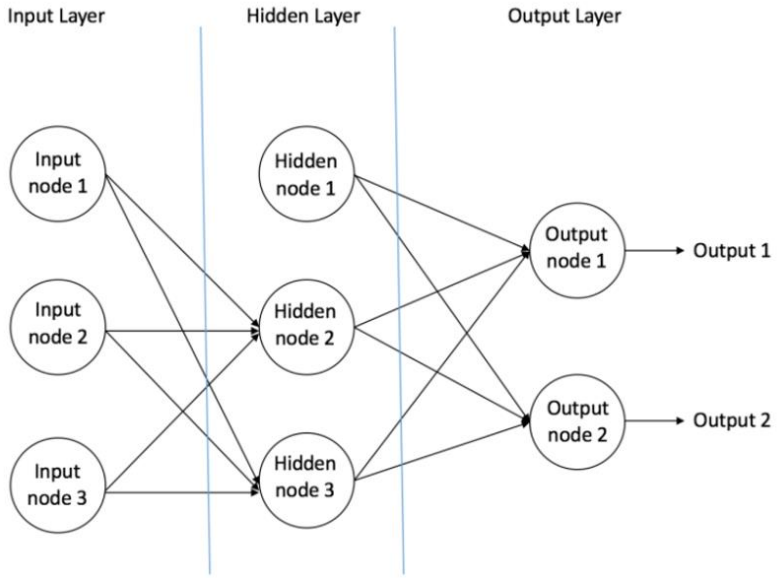

Figura 9. Red neuronal *feedforward*.

2.1.4.2 RNN

Las redes neuronales recurrentes (RNN) son un tipo de red neuronal en el que el output de un paso posterior se usan como input de un paso previo, mientras que en las redes neuronales tradicionales los inputs y outputs son independientes unos de otros. Por tanto, en una red neuronal se aprecian conexiones "hacia atrás" entre las neuronas [30].

Dado que la salida de una neurona en este tipo de redes puede depender de recibir en la entrada la salida de otra neurona previamente calculada, podemos decir que las redes neuronales recurrentes tienen cierta forma de memoria [31].

Las RNN son muy eficientes para tareas que involucren datos secuenciales. Esto es debido a que la "memoria" que tienen les permite recordar información que hayan recibido como entrada previamente, lo que hace que la red sea más precisa a la hora de predecir qué vendrá después, teniendo en cuenta lo que haya recibido previamente. Esto es diferente a las otras redes neuronales que no pueden recordar lo que han visto previamente, más allá de la repercusión que dichos ejemplos vistos previamente tuvieran en los pesos de la red neuronal. Conceptualmente, la RNN tiene como entradas el presente y el pasado reciente.

Dentro de las redes neuronales recurrentes destacan las Long-Short Term Memory (LSTM), un tipo de RNN cuya memoria es capaz de recordar información más lejana en el tiempo que en una RNN normal. Esto es debido a que la LSTM contiene su información en una memoria similar a la memoria de un ordenador, en la que una neurona de la LSTM puede leer, escribir y borrar

información de su memoria. Así, por ejemplo, nos permite producir con precisión la palabra "francés" en la frase "Crecí en Francia…Hablo un fluido _____", cosa que no sería posible en una RNN normal por la distancia entre "Francia" y el hueco establecido para la palabra a predecir.

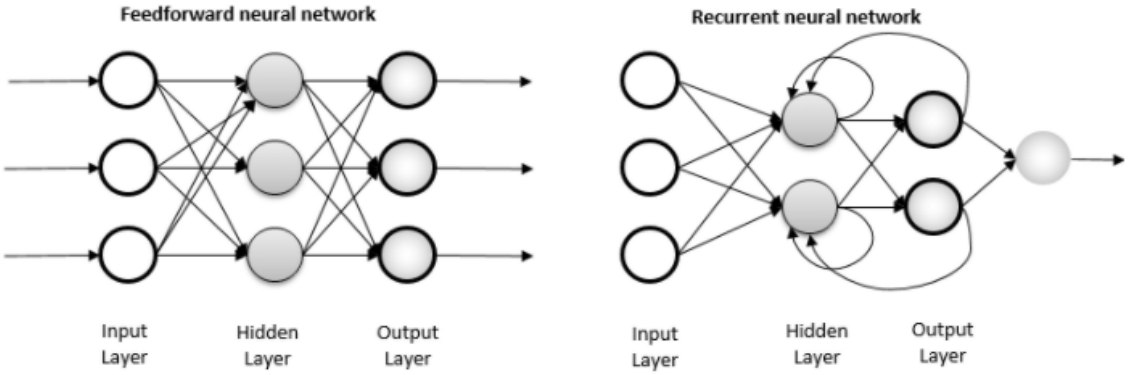

Figura 10. Red neuronal *feed-forward* y recurrente [32].

2.1.4.3 CNN

Las redes neuronales convolucionales, también llamadas CNN, se tratan de un tipo de red neuronal de *Deep learning*, *feed-forward* que son muy usadas en tareas de visión por computador [33]. Están compuestas por una capa de input, una capa de output y varias capas *hidden*. Las capas *hidden* están normalmente compuestas por una o varias capas convolucionales, de ahí el nombre de este tipo de red. A continuación de las capas convolucionales este tipo de red contiene normalmente unas capas MLP [34].

Las CNN son más fáciles de entrenar que otras redes neuronales regulares de *deep learning* y *feed-forward* y tienen muchos menos parámetros que estimar [35].

Aunque son generalmente usadas en visión por ordenador, se han aplicado recientemente con éxito a varias tareas de NLP con resultados prometedores.

Su uso en procesado de lenguaje natural se suele realizar de la siguiente manera que vemos a continuación. Contamos con una matriz que tiene una fila que contiene la representación en *word embeddings* de una palabra (codificación de una palabra en un vector numérico), y un filtro convolucional con la anchura de la matriz que puede tener distintas longitudes, aunque normalmente abarca de dos a cinco palabras, o lo que es lo mismo, de dos a cinco filas.

Dicho filtro se desplaza en una única dirección, de arriba hacia abajo, y en cada aplicación individual del filtro de convolución sobre unas filas produciendo uno de los números del vector columna que observamos en la derecha, que contiene el resultado de todas las aplicaciones del filtro. En el caso de la imagen de abajo podemos apreciar que el filtro ya ha recorrido la matriz desde el principio hasta el final, y que el resultado de su última aplicación es -0.4.

Por último, sobre el vector columna que aparece en la derecha y contiene el resultado de las aplicaciones del filtro convolucional lo más usual es que se lleve a cabo '*max pooling*' y nos quedemos solo con el valor más alto del vector columna que contiene el resultado de todas las aplicaciones del filtro, en este caso 0.7.

	Word embeddings			
cat	0.7	0.4	0.5	0.1
sitting	0.2	-0.1	0.1	0.3
there	-0.5	0.4	0.1	-0.2
or	-0.1	0.8	-0.3	**0.7**
here	-0.5	0.3	0.2	-0.4

Figura 11. Filtro convolucional desplazado [36].

Viendo un ejemplo de lo que sería una CNN muy simple en su aplicación para procesado de lenguaje natural en la imagen de abajo, podemos apreciar:

- Una capa convolucional, que funciona como hemos visto previamente. Consta de varios filtros convolucionales de distinto tamaño, dos filtros distintos de color rojo de 4 palabras, dos filtros distintos de color verde de 3 palabras y dos de color amarillo de 2 palabras. Cada filtro tiene su vector columna correspondiente resultado de cada una de las aplicaciones sobre la matriz, igual que el que hemos visto previamente.
- Un *layer* que hace *max pooling* que selecciona el valor más grande del vector columna de cada filtro.
- Un *layer* totalmente conectado, que en este caso predice valores para 2 clases.

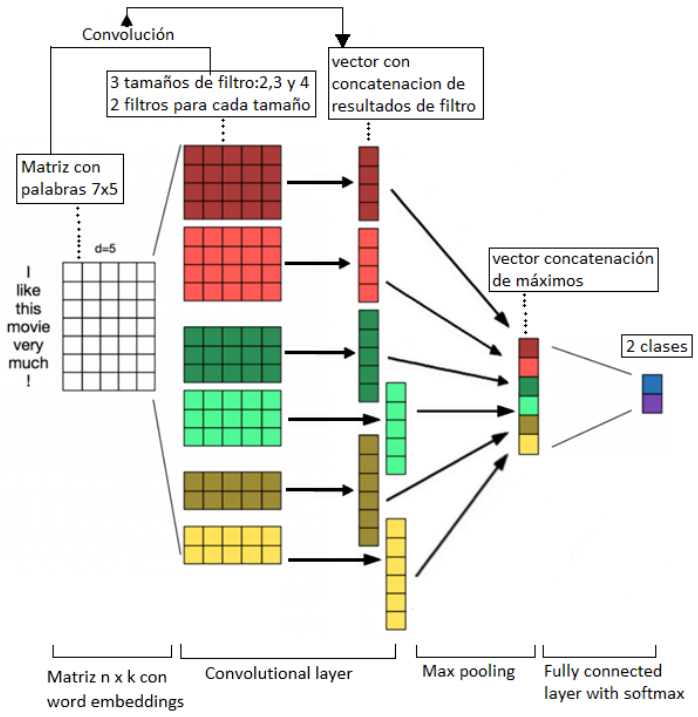

Figura 12. Red neuronal convolucional para NLP [37].

Normalmente las CNN para procesado de lenguaje natural tendrán cientos de filtros de cada tamaño, aunque la red neuronal que hemos visto es muy simple sirve para entender la idea.

2.2 Conjuntos de datos: Training set y Test Set

De todos los datos de los que disponemos, podemos hacer una partición de los mismos en *training set* (conjunto de entrenamiento) y *test set* (conjunto de test). El *training set* es la parte de los datos que se usa para entrenar y ajustar al modelo, en el caso de una red neuronal ajustar los pesos de cada capa que la compone. El modelo ve esta parte de los datos y aprende de ella [38].

Con respecto al *test set*, se trata de una parte de los datos reservada fuera del *training set*. Se usa para obtener una evaluación imparcial del modelo final que hemos obtenido después de entrenarlo con el *training set*, ya que una vez el modelo ha sido entrenado queremos ver cómo funciona con ejemplos no vistos previamente, tal como pasaría en el mundo real donde el modelo debe predecir con entradas no vistas previamente. Por tanto, nos sirve para intuir cómo de bien generaliza nuestro modelo.

2.3 Tipos de aprendizaje

2.3.1 Aprendizaje supervisado

El aprendizaje supervisado en un tipo de aprendizaje en el que cada uno de los datos de entrada contiene la solución deseada, llamada etiqueta. En este aprendizaje el objetivo es que el modelo aprenda a través de ver miles de ejemplos con el *output* correcto para un *input* determinado [39]. Dicho aprendizaje del modelo se produce a través de ir iterativamente ajustando el modelo con los *inputs* recibidos y las etiquetas asociadas a cada ejemplo de entrada.

Una vez el modelo ha sido entrenado y ajustado a través de ver las etiquetas correctas de unos *inputs* de ejemplo el escenario deseado es que el modelo fuera capaz de predecir con precisión *inputs* no vistos previamente.

2.3.2 Aprendizaje no supervisado

Mientras el aprendizaje supervisado parte de un conjunto de datos que está etiquetado y sabemos el valor objetivo para cada dato de entrada que conocemos, el aprendizaje no supervisado parte de datos no etiquetados previamente. Sobre ese conjunto de datos no etiquetados el algoritmo intentará clasificar la información por sí mismo.

Un ejemplo importante de aprendizaje no supervisado son los algoritmos de *clustering* o agrupamiento, que se dedican a agrupar el conjunto de datos de entrada en diferentes subgrupos compuestos con datos de entrada con características más o menos similares [40].

2.3.3 Aprendizaje por refuerzo

El aprendizaje por refuerzo o *reinforcement learning* se trata de un área en el cual un agente tendrá que explorar un espacio desconocido con la intención de determinar cuál es el conjunto de acciones a llevar a cabo, con el objetivo de crear la mejor estrategia para obtener la mayor recompensa en tiempo y forma. Las acciones que el agente debe hacer las aprenderá por sí mismo mediante prueba y error en función de las recompensas y penalizaciones obtenidas por cada una de estas acciones.

El aprendizaje por refuerzo se diferencia del aprendizaje supervisado en que no necesita etiquetas asociadas a los inputs de ejemplo [41].

2.4 Hiperparámetros.

Los hiperparámetros de una red neuronal son los valores de configuración utilizados durante el proceso de entrenamiento de la red neuronal, y se establecen previamente al entrenamiento de la misma. Los hiperparámetros que cuyos valores modificaremos en este trabajo son los siguientes:

- Epochs: Número de veces en el que todos los datos de entrenamiento han pasado por la red neuronal.

- Learning rate: Magnitud con la que la red neuronal actualiza sus parámetros en la dirección que considera que se disminuye el error.

- Batch size: Número de ejemplos que componen un lote que pasamos por la red neuronal y que consideraremos para hacer una actualización de los parámetros del modelo.

2.5 *Overfitting* vs *Underfitting*

El objetivo de un algoritmo de *machine learning* en aprendizaje supervisado es obtener una función que mapee los variables que se reciben como *input*(X) con la variable *output* (Y) lo más aproximada posible a la función objetivo que representan los datos totales del dominio del problema [42].

El aprendizaje que se lleva a cabo a partir del *training data* se denomina aprendizaje inductivo, ya que se deben aprender conceptos generales a partir de un conjunto de datos con ejemplos específicos. Por tanto, el objetivo de un algoritmo es generalizar bien en ejemplos que pertenecen al dominio del problema pero nunca han sido vistos previamente. Dando por supuesto que tenemos unos datos de buena calidad y tamaño suficiente, si el modelo está bien entrenado generalizará bien y predecirá con alto porcentaje de acierto esos ejemplos nunca vistos previamente.

A continuación procedemos a detallar las dos principales causas de mal rendimiento de los algoritmos de *machine learning*, *overfitting* y *underfitting*.

Overfitting, también llamado en español sobreajuste, es lo que sucede cuando el modelo se ajusta en demasía a las características específicas de los datos del training data, por ejemplo a detalles de los datos o al ruido de los mismos. En este caso, el modelo memoriza los datos de entrenamiento en lugar de aprender a generalizar a partir de una tendencia o conceptos que se encuentren en los datos de entrenamiento [43].

Teniendo en cuenta que los datos de los que se dispone para trabajar con ellos son un subconjunto de los datos del dominio y no la totalidad de los datos, ese sobreajuste trae consigo que el modelo no generalice bien, no obteniendo buenos resultados en ejemplos que no han sido vistos previamente.

Durante la fase de *overfitting* el porcentaje de acierto en los ejemplos de los datos disponibles de entrenamiento se incrementa, mientras que el acierto con ejemplos no vistos previamente desciende, debido a lo mencionado previamente.

Por otro lado, *underfitting* sucede cuando el modelo no se ajusta lo suficiente a los datos de entrenamiento, y por tanto no captura adecuadamente la estructura de los datos. Normalmente es consecuencia de un modelo demasiado simple, que no es capaz de procesar la complejidad del problema.

En este caso el modelo no ha aprendido lo suficiente de los datos de entrenamiento [44], por lo que arroja unos malos resultados en las métricas con esos datos, lo que hace fácil su detección.

Por supuesto, si el modelo no se ajusta y ha aprendido lo suficiente de los datos de entrenamiento que son la versión reducida que tenemos del dominio del problema, tendrá malos resultados en la generalización con ejemplos no vistos previamente.

Idealmente estamos interesados en obtener un modelo que no caiga ni en el *overfitting* ni en el *underfitting*, ya que como hemos visto en caso de *overfitting* el modelo no generaliza bien porque se ajusta demasiado a los datos de entrenamiento y en caso de *underfitting* porque se ajusta demasiado poco, y lo que buscamos es un modelo que generalice bien en ejemplos no vistos previamente.

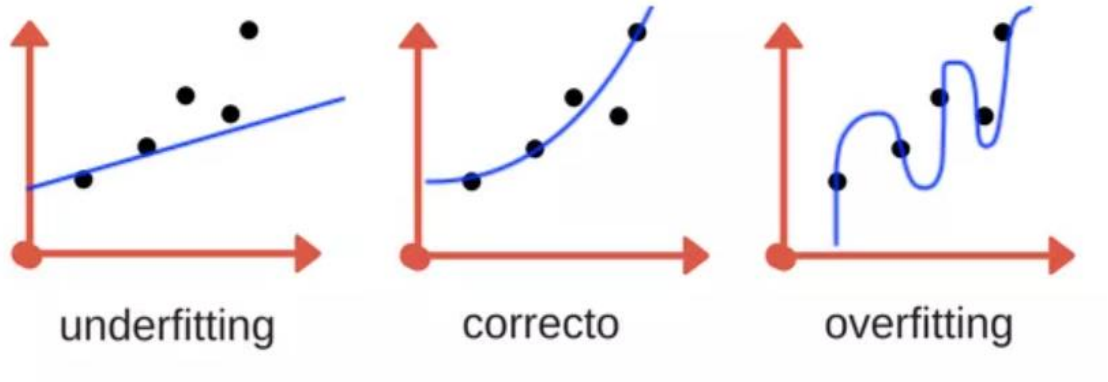

Figura 13. *Underfitting*, ajuste correcto y *overfitting* sobre los datos de entrenamiento [45]

2.6 Transfer learning

Transfer learning es un método de *machine learning* que consiste en reusar un modelo entrenado para una tarea como punto de partida para una segunda tarea [46].

Teniendo en cuenta que entrenar una red neuronal desde cero puede precisar de una gran cantidad de datos, de tiempo y de recursos computacionales, puede ser muy interesante usar, por ejemplo, una red neuronal pre-entrenada previamente y no tener que gastar esa gran cantidad de recursos para entrenar una red neuronal desde cero. Podemos descargar un modelo previamente entrenado sobre un gran conjunto de datos y usar sus parámetros aprendidos como punto de partida para continuar entrenándolo para otra tarea determinada con otro conjunto de datos diferente, normalmente más pequeño.

2.7 Tareas de procesado de lenguaje natural.

Vemos algunas tareas de procesado de lenguaje natural que se mencionan en el trabajo.

En primer lugar, una tarea de clasificación se trata de una tarea de aprendizaje supervisado, en la cual un sistema debe de predecir una categoría en base a un ejemplo de entrada, en nuestro caso un texto. Por ejemplo categorizar un email como ‘spam’ o ‘no spam’ [47].

Por otro lado, el reconocimiento de entidades nombradas (Name Entity Recognition), se trata de una tarea consistente en extraer y localizar información de un texto clasificando dicha información en categorías predefinidas previamente, como personas, lugares u organizaciones [48]. Un ejemplo sería extraer todos los medicamentos que aparezcan en un texto médico.

La extracción de relaciones consiste en extraer relaciones semánticas en un texto normalmente entre dos o más entidades, siendo posible que estas relaciones sean de diferentes tipos [49]. Por ejemplo, si tenemos la frase "Paris esta en Francia", podríamos obtener una relación "esta en" compuesta por Paris y Francia. Como hemos dicho previamente puede haber relaciones de muchos tipos, como por ejemplo "compuesta por", "fabricada por", "causada por", etcétera.

2.8 Métricas de evaluación

Las métricas de evaluación se usan para medir la calidad de un modelo de *machine learning*. Por este motivo es muy importante elegir una métrica que nos manifieste la bondad o no de nuestro modelo [50].

Entramos a definir primero qué son los siguientes términos:

TP: *True positive*. Ejemplos que el modelo predice como positivos y efectivamente lo son.

FN: *False Negative*. Ejemplos que el modelo predice como negativos pero en realidad son positivos.

TN: *True Negative*. Ejemplos que el modelo predice como negativos y efectivamente lo son.

FP: *False Positive*. Ejemplos que el modelo predice como positivos pero en realidad son negativos.

Estos valores se pueden representar gráficamente en una matriz de confusión.

		Clase predicha	
		Positiva	Negativa
Clase real	Positiva	TP	FN
	Negativa	FP	TN

Otras métricas son:

Accuracy: Ratio de predicciones correctas con respecto a todas las predicciones hechas.

$$Accuracy = \frac{\mathrm{TP+TN}}{\mathrm{TP+FP+FN+TN}}$$

Precisión: Calculamos el ratio de acierto entre las predicciones positivas del modelo. Es especialmente útil cuando nos interesa considerar el número de FP, como en la detección de *spam*.

$$Precisión = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

Recall o sensibilidad: Ratio que mide la proporción de positivos reales que el modelo ha acertado. Es útil cuando nos interesa considerar y medir los FN, como por ejemplo en lucha contra el cáncer, si se le dice a un enfermo que no tiene cáncer y sí lo tiene es peligroso.

$$Recall = \frac{TP}{TP + FN}$$

Especificidad: Ratio que mide la proporción de negativos reales que el modelo ha predicho correctamente

$$Especificidad = \frac{TN}{\mathrm{TN} + \mathrm{FP}}$$

F1-Score. Es la media armónica de la precisión y sensibilidad. Se utiliza en conjuntos de datos donde es importante considerar tanto los fallos tipo FN como los FP, ya que se usan tanto la precisión como sensibilidad y si alguno arroja un mal resultado F1 tendrá un mal resultado también [51].

$$F1 = \frac{2 * precision * recall}{\mathrm{precision} + \mathrm{recall}}$$

2.9 Validación cruzada de k iteraciones

A la hora de evaluar el desempeño de un modelo entrenamos el modelo con el training set y evaluamos la bondad del modelo según los resultados que obtengamos prediciendo los datos del test set. Sin embargo, esos resultados pueden no ser del todo representativos ya que son resultados para una división concreta del conjunto de datos total en training set y test set, y muy posiblemente si dividiéramos en training set y test set de manera diferente el modelo nos arrojaría resultados diferentes.

Para minimizar este efecto y obtener una medida de evaluación lo más representativa posible de la realidad usamos validación cruzada, un método estadístico usado para evaluar la bondad de un modelo de *machine learning* [52] .

En concreto, en este trabajo usaremos la validación cruzada de k iteraciones, consistente en este número de pasos.

1. Mezclar aleatoriamente los datos
2. Dividir los datos en k trozos
3. Para cada uno de esos trozos
   3.1 Se toma ese trozo de datos como test set
   3.2 Se toman los grupos restantes como training set
   3.3 Se entrena el modelo en el training set y se evalúa en el test set
   3.4 Se guardan los resultados obtenidos
4. Las métricas del algoritmo serán la media de todas las evaluaciones

De esta manera, obtenemos unos resultados más representativos que si sólo hubiéramos hecho una evaluación del modelo con una división concreta de los datos en training y test set, ya que hemos hecho una media de la evaluación del modelo usando varias maneras de dividir el conjunto de datos en su totalidad.

Vemos en la Figura 14 visualmente cómo sería la partición de datos de una validación cruzada de cinco iteraciones con sus correspondientes cinco trozos, en la que se haría el proceso descrito previamente.

Figura 14. Validación cruzada de cinco iteraciones [53]

2.10 Algunos algoritmos empleados en el procesado de lenguaje natural

La regresión logística consiste en usar un análisis de regresión para predecir el resultado de una variable categórica en varias categorías posibles, en caso de ser regresión binaria en dos categorías [54]. Este análisis se hace en función de varias variables independientes, intentando modelar la probabilidad de un evento en función de otros factores.

Un clasificador ingenuo Bayesiano se trata de un método probabilístico que se basa en el teorema de Bayes. Este clasificador asume que la presencia o ausencia de una característica no está relacionada con la presencia o ausencia de otra característica [55]. En el caso de usar este clasificador para predecir las etiquetas de un texto se calcula la probabilidad de cada etiqueta para un texto dado, y la etiqueta predicha será aquella que tenga una mayor probabilidad. En caso de que no sea posible calcular la probabilidad del texto en su conjunto se disecciona en palabras y se calcula la probabilidad de que cada una de las palabras de manera independiente pertenezca a esa categoría [56].

Un árbol de decisión se trata de un modelo en forma de árbol en el cual cada nodo es una decisión, y cada hoja representa una clase de output [57]. Un nodo puede tener cualquier número de hijos, aunque normalmente se tratan de árboles binarios con preguntas binarias. Los clasificadores basados en árboles son muy comunes debido a que son fáciles de entrenar y se puede analizar la estructura del árbol para obtener un mayor entendimiento del texto y poder intuir por qué se predijo una clase, al contrario de una red neuronal en la que no obtenemos ese tipo de información.

Support Vector Machine (SVM) se trata de un clasificador que dado un conjunto de entrenamiento aprende un hiperplano de clasificación en el espacio de características que tiene la máxima distancia de los ejemplos de entrenamiento [58]. Como consecuencia en las tareas de clasificación tiende generalizar mejor en datos no vistos previamente que otros algoritmos como los árboles de decisión.

2.11 Frameworks

Un *framework* es una pieza de software desarrollada por desarrolladores para construir aplicaciones. Usar un *framework* para desarrollar aplicaciones permite al desarrollador centrarse en la funcionalidad de alto nivel, ya que de la funcionalidad a bajo nivel se encarga el *framework* [59].

2.11.1 Frameworks en Deep Learning

Los dos *frameworks* más usados para *Deep Learning* en 2019 son PyTorch y TensorFlow. Los investigadores están abandonando TensorFlow en virtud de PyTorch [60]. En la industria, Tensorflow es el *framework* dominante aunque puede que no durante mucho tiempo.

Observamos el número de publicaciones que usaban cada *framework* en las conferencias de investigación de mayor prestigio.

Apreciamos como se incrementa hasta la holgada mayoría el número de artículos científicos que usaban Pytorch con respecto al total de artículos científicos que usaban o Pytorch o Tensorflow.

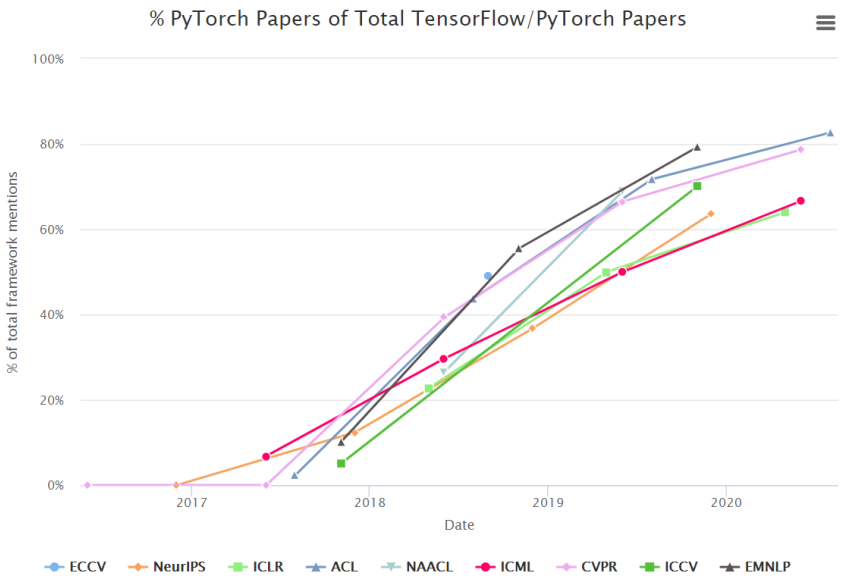

Figura 15. *Papers* que usan Pytorch con respecto a los que usan Tensorflow o Pytorch [61]

Observamos en la Figura 16 en línea continua el porcentaje de artículos científicos que usan Pytorch y Tensorflow con respecto al total de los artículos en las conferencias. Por lo tanto, se obtiene la misma conclusión anterior: Pytorch es mucho más usado en la comunidad investigadora.

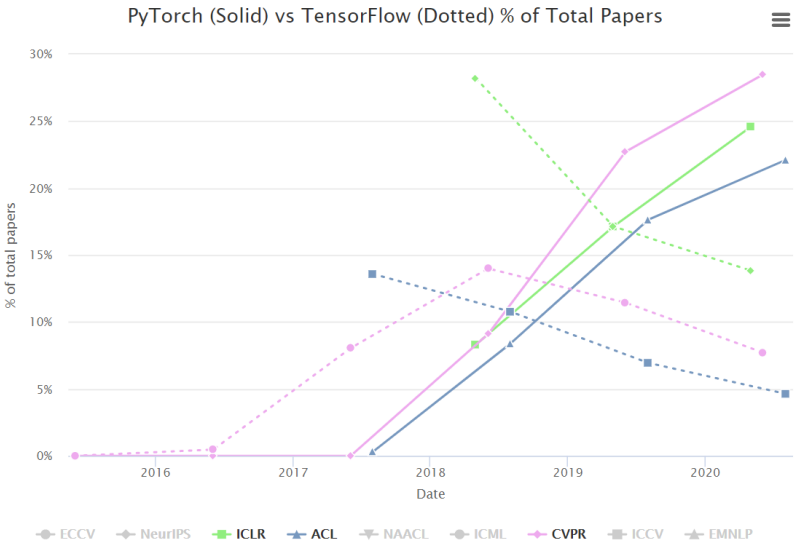

Figura 16. *Papers* que usan Pytorch o Tensorflow con respecto al total de *papers* [61]

2.11.2 PyTorch

Como hemos visto, PyTorch previamente era una minoría en la comunidad investigadora, mientras que actualmente es una holgada mayoría. El dominio es aún mayor en las conferencias de visión donde gana a Tensorflow en un factor de 2 a 1 y en las tareas de lenguaje donde se incrementa el dominio hasta un factor 3 a 1.

Los investigadores están usando PyTorch principalmente por tres razones:

- Simplicidad. Se integra muy bien con el ecosistema de Python, y se puede depurar con facilidad.
- Gran API
- Permite que se construyan arquitecturas complejas de manera sencilla [62].

2.12 Estado del arte del procesado de lenguaje natural (NLP)

El NLP (*Natural language processing*), en español 'procesado de lenguaje natural', se trata de un campo dentro de la lingüística, informática e inteligencia artificial que se concentra en las relaciones entre ordenadores y lenguaje natural humano, en particular en cómo programar los ordenadores para procesar y analizar cantidades enormes de datos en lenguaje natural [5].

El procesamiento del lenguaje natural tiene sus raíces en la década de 1950. En 1950, Alan Turing publicó un artículo titulado "Maquinaria e Inteligencia Informática" en el que proponía lo que ahora se llama "test de Turing" como una medida de inteligencia. Concretamente, el Test de Turing es un examen que examina la capacidad de una máquina de simular comportamiento humano en una conversación en lenguaje natural, por lo que se trata de una tarea que involucra interpretación automática y generación de lenguaje natural.

A grandes rasgos, podemos describir la historia del procesado del lenguaje natural de la siguiente manera:

- NLP simbólico. Engloba desde la década de los 50 hasta principios de la década de los 90. Consistía en dar una serie de reglas al ordenador para que el ordenador hiciera tareas de procesado de lenguaje natural aplicando esas reglas a los datos que se le proporcionaban.

- NLP estadístico. Engloba desde los 90 hasta principios de la década de 2010. Se caracteriza por la introducción de algoritmos de *machine learning* para procesamiento de lenguaje natural, potenciados por el incremento de la capacidad computacional y por la disminución del dominio de las teorías lingüistas de Chomsky.

- NLP neuronal. Tiene lugar desde el principio de la década de 2010 hasta el presente. En esta década se ha producido la generalización del uso de las redes neuronales para el procesado de lenguaje natural. Se vio influenciada por la ley de Moore que siguió jugando su papel incrementando la capacidad computacional de los ordenadores de manera exponencial, el incremento del uso de redes neuronales en otros campos con éxito y los sobresalientes resultados que mostraban estos algoritmos cuando se aplicaban a diferentes tareas de procesado de lenguaje natural.

Enfocándonos en el NLP neuronal, a lo largo de la presente década han cogido gran protagonismo los modelos pre-entrenados, en adelante PTM [63].

Con el incremento en el desarrollo del *deep learning*, el número de parámetros de las redes neuronales se ha incrementado en gran medida, con lo cual se necesita un conjunto de datos de gran tamaño para que no se produzca *overfitting* y el algoritmo no memorice los datos de entrenamiento que se le están pasando, en lugar de aprender de dichos datos para que el modelo generalice bien en el futuro.

Este *overfitting* podría tener solución si dichos modelos con gran número de parámetros fueran entrenados con conjuntos de datos de gran tamaño. Sin embargo, encontrar conjuntos de gran tamaño con etiquetas de output para cada ejemplo es muy complicado debido al elevado coste de anotación, puesto que cada texto debe de ser etiquetado por un humano.

Este problema de necesitar grandes bases de datos le da utilidad a los modelos pre-entrenados, ya que se pueden pre-entrenar sobre textos que son fáciles de encontrar como textos de gran tamaño que no estén etiquetados, por ejemplo la Wikipedia o una base de datos de libros. Estos modelos pueden obtener una representación del lenguaje con estos textos grandes y usar esa representación para otras tareas.

El pre-entrenamiento sobre textos grandes no etiquetados ha conseguido una mejora en los resultados cuando se han usado esos modelos pre-entrenados para diversas tareas de procesado de lenguaje natural.

Las ventajas de los modelos pre-entrenados son principalmente las siguientes:

1. Pre-entrenar en grandes cantidades de texto proporciona al modelo la capacidad de aprender representaciones universales del lenguaje, lo que luego puede ayudar de cara a las tareas concretas para las que se use ese modelo pre-entrenado en el futuro.

2. Pre-entrenar nos proporciona una mejor inicialización del modelo, lo que suele tener como consecuencia unos mejor resultados generalizados y acelera en gran medida la adaptación a una futura tarea de NLP.

3. Pre-entrenar puede ser visto como un tipo de regularización para evitar tener *overfitting* con

pocos datos.

4. Una vez dicho modelo esta pre-entrenado se puede usar y reutilizar para varias tareas, evitando entrenar el modelo desde cero, lo que sería una gran pérdida de tiempo y recursos computacionales.

La primera generación de modelos pre-entrenados se produjo en 2013-2014 con modelos como Skip-Gram y GloVe, y su principal virtud era ser capaces de hacer *word embeddings* (codificación de una palabra en un vector numérico), y que hace que una palabra o palabras con el mismo significado tengan una representación similar. Dicha representación se hará en vectores de valores reales en un espacio vectorial predefinido, y cada palabra se mapea a un vector [64].

Aunque estos modelos pre-entrenados pueden capturar el significado semántico de las palabras, no consideran el contexto de las mismas, y por tanto fallan en obtener conceptos de más alto nivel como estructuras sintácticas, roles semánticos o anáforas.

Sin embargo, la segunda generación de modelos pre-entrenados se concentra en *word embeddings* contextuales. Esta generación de modelos pre-entrenados tiene en sus máximos exponentes a CoVe, ELmo, GPT de OpenAI y el mismo BERT que usaremos en este trabajo. Como hemos dicho, esta generación es capaz de representar palabras en contexto y usar dicho conocimiento para varias tareas concretas de NLP posteriormente, por ejemplo tareas de clasificación de textos, responder a preguntas, etc. Con esta generación se pasa de solo inicializar la primera capa de nuestros modelos a pre-entrenar el modelo entero con representaciones jerárquicas [65].

2.13 Estado del arte de NLP en el ámbito clínico. Qué es, Problemas y soluciones

Una vez hemos visto previamente qué aporta el procesado de lenguaje natural en el ámbito clínico, en este apartado vamos a ampliar esa visión, además de ver los problemas que existen y algunas soluciones propuestas.

La narrativa clínica representa la forma principal de comunicación dentro del ámbito clínico, ya que al contrario de los elementos codificados estructurados proporciona un relato detallado y personalizado de la historia y evaluaciones del paciente, por lo que nos ofrece un mejor contexto para la toma de decisiones clínicas [12]. El procesado de lenguaje natural es un campo de la inteligencia artificial que estudia la manera en la que podemos automatizar el análisis y síntesis de esa información enfrascada en texto, y nos da la capacidad de obtener evidencias enterradas en textos clínicos. Tradicionalmente se tomaron aproximaciones basadas en las reglas para obtener esas evidencias, pero hacer las reglas requiere la participación de expertos clínicos para convertir su conocimiento en un conjunto de reglas que reflejen ese conocimiento.

Machine learning ha sido aclamado durante tiempo como la solución al problema de obtención de ese conocimiento, ya que se argumentaba que el tiempo necesario para anotar una base de datos era mucho menor que el tiempo necesario en obtener el conocimiento experto y hacer las reglas. Sin embargo, muchas veces el tiempo empleado en realizar la anotación de los datos es similar al que se habría empleado en la obtención del conocimiento experto y elaboración de las reglas.

No obstante, como hemos visto previamente, hay otros motivos por los cuales la importancia de los métodos de procesado de lenguaje natural en la informática relacionada con el ámbito clínico se ha ido incrementando a lo largo de los últimos años. Consecuencia de esto se han ido afrontando diversos problemas que tenía la comunidad, además de elaborar e implementar recomendaciones de cara a futuro. Por ejemplo, algunas recomendaciones estarían encaminadas a solucionar los problemas de la limitada colaboración existente entre instituciones, que tienen como consecuencia la falta de recursos compartidos y la falta de existencia de medidas de evaluación en tareas cruciales, como clasificación de conceptos médicos o información temporal.

Algunos de los problemas detectados son los siguientes:

Es complicada la disponibilidad de datos clínicos, dados los problemas de privacidad y la naturaleza sensible de los datos de la salud. Estos problemas de falta de disponibilidad de datos anotados manualmente pueden resultar en una falta de representatividad de los datos de entrenamiento, y como consecuencia los modelos de *machine learning* no tener los resultados que potencialmente podrían tener.

Para la aplicación de NLP se usaron conjuntos de datos pequeños, la inmensa mayoría de cientos o miles de ejemplos. De hecho, se usaron conjuntos de datos pequeños incluso cuando unos más grandes estaban disponibles, llegando a veces a usar un 0.002% de los datos disponibles y como mucho un 11.88%. Esta poca utilización de los datos es consecuencia del cuello de botella de la anotación encontrado en los algoritmos *machine learning* de aprendizaje supervisado que requiere que los datos de entrenamiento sean anotados.

Por otro lado, la anotación es susceptible de tener errores y requiere de una gran cantidad de trabajo, además de que puede ser específica para una tarea y no ser reutilizable en un futuro. Una manera de paliar este cuello de botella en el futuro es utilizando *transfer learning*, ya que aprovechamos el conocimiento aprendido por un modelo en otros conjuntos de datos más grandes que quizá se usaron para otras tareas para resolver una tarea que ahora no necesitaría de un conjunto de datos anotado de gran tamaño, ya que el modelo ya traería conocimiento aprendido previamente.

Además del pequeño volumen de los datos de entrenamiento, otro problema que puede afectar al rendimiento de los algoritmos de *machine learning* puede ser la procedencia de los datos. La estructura y estilo de las narrativas clínicas puede variar mucho entre instituciones, y cuando los datos vengan de un número pequeño de instituciones (como está sucediendo de momento) pueden no ser representativos. Esto puede llevar a que el modelo obtenga muy buenos resultados en los datos de entrenamiento pero no generalice bien en el futuro, puesto que los datos que se le inserten en el futuro pueden ser diferentes o estar expresados de una manera diferente a los datos vistos en la etapa de entrenamiento, teniendo un problema de representatividad de la muestra.

Por otra parte, el error de estos algoritmos puede parecer un apunte estadístico, pero en el mundo real los resultados del NLP deben de ser verificados por un médico antes de hacer recomendaciones [66]. Los profesionales sanitarios es más probable que confíen en modelos interpretables que les expliquen en base a qué criterios han tomado la decisión. También, una predicción con un enfoque probabilístico puede ser mejor que una predicción binaria. Este tipo

de cosas puede contribuir a que haya una discordancia entre los buenos resultados de estos modelos en investigaciones y su uso en el mundo real. Se debería de tener en consideración su encaje en el mundo real y su uso por parte de profesionales que no sean expertos en NLP.

Otro obstáculo son las diferencias de granularidad. Si el algoritmo evalúa unos textos o informes y el paciente tiene varios informes ¿Cómo se hace la evaluación del paciente?. Quizás sería una solución hacer post procesado con el output del modelo a nivel de un documento, o quizá no es posible.

Otro problema surge a la hora de desarrollar métodos para desarrollar datos compartibles en la comunidad. Los riesgos de privacidad son altos y convencer a ciertas asociaciones y autoridades no es fácil, más aún cuando consideramos que diferentes países y jurisdicciones tienen legislaciones diferentes. Incluso muchas veces quitar la identidad de los documentos puede no ser suficiente, ya que se podría argumentar que se puede inferir la identidad de la persona a partir de los datos expuestos en los informes o documentos clínicos.

A la hora de elegir entre diferentes modelos, por tanto, convendría considerar:

- Si el modelo es capaz de considerar el tiempo a la hora de hacer predicciones, no tomar cada input como un punto independiente del tiempo con el que hacer una predicción, sino considerar varios inputs en varios momentos diferentes.

- Interpretabilidad de los modelos. A la hora de elegir un modelo u otro se debería de considerar si el modelo explica el por qué tomo dicha decisión, esto es un problema típico y grande del ámbito clínico porque un profesional sanitario no puede o quiere arriesgarse a decir un diagnóstico sin saber por qué. El crecimiento del uso de técnicas de *deep learning* en el ámbito clínico incrementa este problema, ya que las redes neuronales no explican en base a qué tomaron decisiones.

Para solucionar dichos problemas se plantean algunas soluciones, como por ejemplo:

- Incrementar la disponibilidad de grandes conjuntos de datos compartibles.

- Cambiar los *workbenches* de evaluación. Los métodos de NLP se entrenan para un caso de uso específico y para un conjunto de datos específico, nadie sabe cómo se van a comportar para otro caso de uso y conjunto de datos diferentes.

- Elaboración de estándares para hacer informes. Es necesario que se asegure la transparencia y reproducibilidad de los métodos de procesado de lenguaje natural en el ámbito clínico. Para ello, se puede establecer un protocolo mínimo que deba seguir cualquier estudio de NLP en el ámbito clínico para facilitar que otro profesional pueda determinar si ese método puede ser aplicable y útil para un nuevo caso de uso o conjunto de datos y si sería muy costoso adaptarlo.

A modo de conclusión, los avances del NLP en el ámbito clínico han sido notorios a lo largo de los últimos años y han establecido resultados del estado del arte, especialmente los últimos años con redes neuronales. Sin embargo, hay varias aspectos que son mejorables como ya hemos visto y detallado previamente, solucionarlos podría hacer que se obtuvieran mejores resultados que los actuales y se incrementara el uso de estos modelos y su utilidad en el ámbito clínico,

que es de lo que se trata.

2.14 Estado del arte de Deep learning aplicado a NLP en el ámbito clínico

Las técnicas de *Deep Learning* han empezado a dominar el campo del procesado de lenguaje natural en el ámbito clínico por su simplicidad, procesado eficiente (asumiendo hardware paralelizado masivo y dedicado), y resultados del estado del arte en muchas tareas [67]. Al mismo tiempo, la amplia adopción de *electronic health records* (historias clínicas electrónicas) ha producido grandes cantidades de texto digital sobre los pacientes, mientras que la comunidad de informática aplicada a la medicina ha hecho un gran esfuerzo en hacer uso de los textos generados en el ámbito clínico a través del procesado de lenguaje natural NLP.

Veremos las técnicas de *deep learning* que están siendo aplicadas al ámbito clínico analizando los artículos e investigaciones hechas recientemente. En concreto se analiza el uso de redes neuronales recurrentes, redes neuronales convolucionales, redes neuronales *feed-forward*, y artículos que sólo usaban *word embeddings*, donde los artículos de este estilo sólo usan *word embeddings* sin redes neuronales, por ejemplo con un clasificador de *machine learning* más tradicional.

Vemos las arquitecturas de *deep learning* usadas en el ámbito clínico en los últimos años.

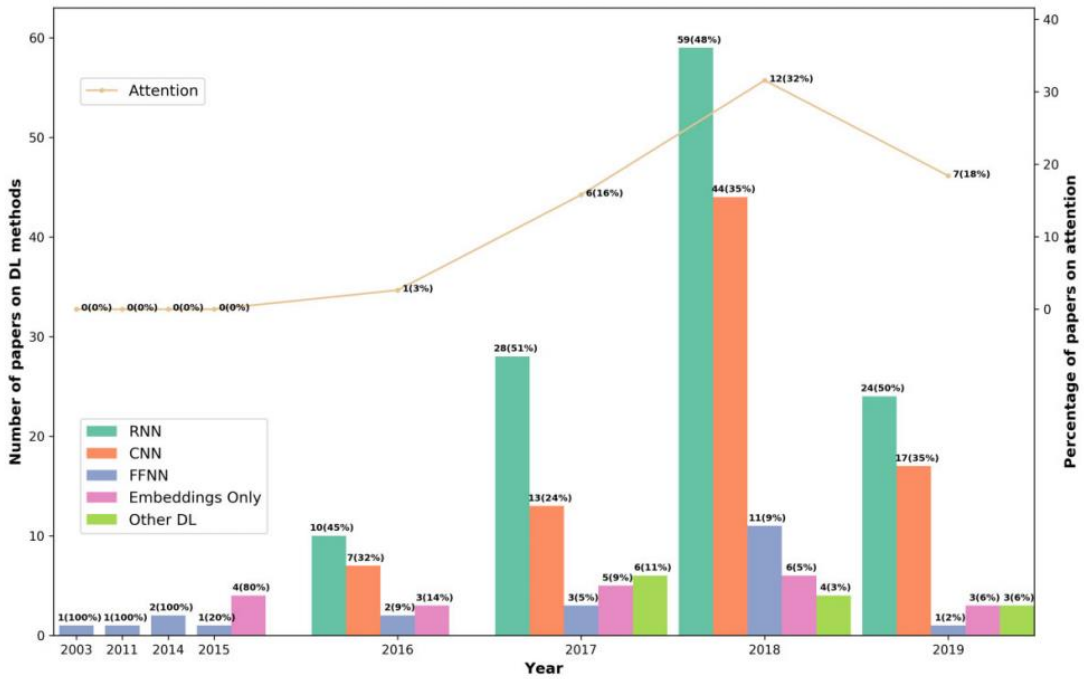

Figura 17. Arquitecturas de *deep learning* por año en el ámbito clínico [67]

Nota*. En la Figura 17 para el año 2019 sólo se consideró hasta Abril.

Como podemos ver, el número de publicaciones de *deep learning* se ha ido incrementando a lo largo de los años con gran celeridad, con incrementos de más del 200% hasta 2018.

Interesantemente, las redes neuronales tradicionales (*feed-forward*) también crecieron en porcentaje sobre el total de los trabajos de redes neuronales, aunque decrecen mucho en porcentaje a lo largo del año 2019. También se puede ver que los trabajos que usan sólo *word embeddings* fueron decreciendo en porcentaje a lo largo de los años después del interés inicial.

Aunque el procesado de lenguaje natural en el ámbito clínico tradicionalmente ha contado mucho con recursos de conocimiento médico específico, solo el 18% de los artículos científicos analizados utilizaron conocimiento médico externo, entendiendo por conocimiento médico el conocimiento que no se puede obtener exclusivamente a partir de los ejemplos de entrenamiento. Concretando aún más, solo el 5.7% de los artículos científicos analizados incorporaron ese conocimiento en la arquitectura de *deep learning*.

En cuanto a los tipos de problemas que se intentan solucionar, observamos en la Figura 18 las tareas clínicas y de procesado de lenguaje natural que se hicieron en 212 artículos científicos analizados. Se categorizaron las principales tareas de NLP en 4 tipos: Clasificación de textos (40.5%), reconocimiento de entidades nombradas (34.0%), Extracción de relaciones (13.7%), y Otros (10.8%).

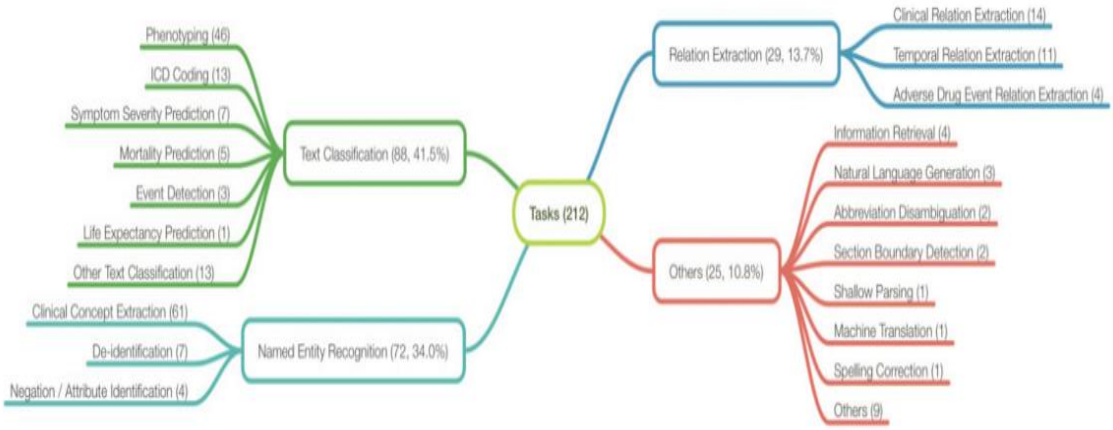

Figura 18. Tareas que se llevaron a cabo con *Deep learning* en ámbito clínico [67].

Entrando en mayor detalle, vemos que las tareas más comunes son:

- Extracción de conceptos, como por ejemplo extraer el problema, el test de laboratorio o el tratamiento. Se encuentra dentro de la categoría *Name Entity Recognition*
- Fenotipado, por ejemplo la amplia caracterización de las condiciones de los pacientes. Se encuentra dentro de la categoría de clasificación de textos.

A continuación vamos a ver el tipo de algoritmos de *deep learning* que se usan para cada una de las tareas. Se confirman las asociaciones hechas en la comunidad como que las CNN son las más comunes para hacer tareas de clasificación de textos y que RNN son las más comunes para reconocimiento de entidades nombradas y extracción de relaciones. Las CNN han dominado la clasificación de textos por la pronta existencia de exitosos métodos basados en CNN, mientras que a las RNN le paso algo parecido debido a que las LSTMs (un subtipo de RNN) se aplicaron con éxito pronto a tareas de reconocimiento de entidades nombradas.

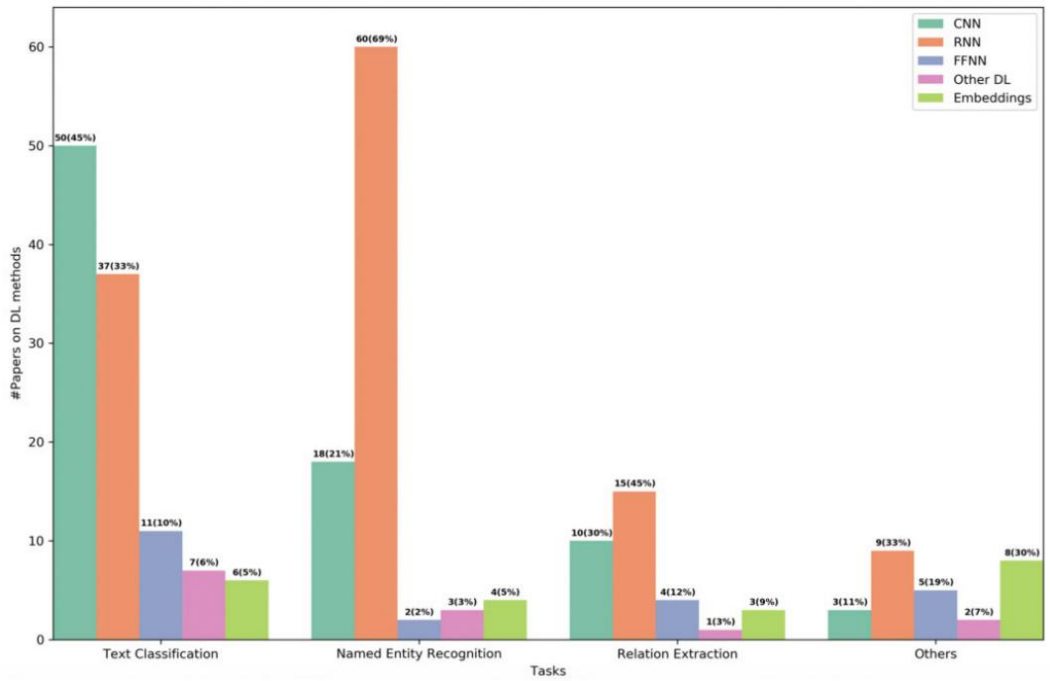

Figura 19. Arquitectura de *deep learning* para cada tarea en ámbito clínico [67]

A lo largo de los años ha decrecido el número de artículos que compara *deep learning* con *machine learning*, lo que puede ser debido al incremento de otros algoritmos de DL como modelos de referencia. De los 212 artículos analizados solo 108 (50.9%) comparan los resultados de algoritmos de *deep learning* con algoritmos de *machine learning*. En los 108 artículos que se comparan estos algoritmos, el 72% de los métodos de *deep learning* propuestos vencieron a métodos de *machine learning* tradicionales, que vencieron tan solo en un 11% de los artículos.

Los que primero adoptaron y trabajaron con NLP aplicado al ámbito clínico fueron los miembros de la comunidad de NLP, mucho antes que la comunidad informática en general o la médica, que fue la que mostró más reticencias a este tipo de métodos.

Como consecuencia del gran coste que hemos visto que tiene anotar datos clínicos y las preocupaciones relacionadas de la privacidad para compartir el training data, *transfer learning* es una estrategia interesante. Con el incremento de exitosos modelos pre-entrenados como BERT, se espera que el uso de *transfer learning* incremente en popularidad rápidamente.

Podemos concluir que las arquitecturas de *deep learning* incrementarán su papel de líder indiscutible en el procesado de lenguaje natural aplicado en el ámbito clínico durante los próximos años.

3. Aplicación de BERT para el diagnóstico de accidente cerebrovascular isquémico agudo mediante *Transfer Learning*

3.1 Investigaciones previas

En el trabajo de Kim et al. (2019) [9] se lleva a cabo una comparación de diversos algoritmos para resolver un problema de aprendizaje supervisado cuyo cometido es clasificar textos y hacer un diagnóstico en el ámbito clínico.

El objetivo del NLP en este trabajo es implementar algoritmos que permitan identificar a pacientes con accidente cerebrovascular isquémico agudo (AIS) basándose en texto desestructurado existente en los informes radiológicos de resonancias magnéticas cerebrales, identificando si un texto se corresponde con una persona que tiene accidente cerebrovascular isquémico agudo (AIS) o no (NO-AIS), por lo que el cometido es una tarea de clasificación de textos en dos categorías. También se comprueba cuál de los algoritmos probados obtiene mejores resultados en esta tarea.

Para contextualizar, merece la pena mencionar que el accidente cerebrovascular es una de las principales causas de muerte y morbilidad en todo el mundo. El registro médico contiene datos de laboratorio, información clínica y los códigos de diagnóstico de la Clasificación Internacional de Enfermedades (CIE). Estos códigos pueden indicar simplemente si un paciente ha sido admitido por un accidente cerebrovascular, pero a menudo no pueden distinguir con precisión si el paciente fue hospitalizado por síntomas agudos de accidente cerebrovascular u otros problemas derivados de un accidente cerebrovascular. Sin embargo, a través de diversas técnicas de imagen por resonancia magnética, podemos confirmar si el accidente cerebrovascular es isquémico o hemorrágico, y si es agudo o crónico.

Además, los informes sobre resonancias magnéticas rara vez se leen, y los datos no estructurados como los informes de texto y los datos de imágenes suelen contener información útil. Una manera de aprovechar la información que se encuentra en las descripciones de resonancias magnéticas es el procesado de lenguaje natural. Aunque se ha usado el procesado de lenguaje natural para el diagnóstico de otras enfermedades recientemente, antes de este estudio no se hizo ningún estudio usando NLP para identificar pacientes con accidente cerebrovascular isquémico agudo (AIS) a partir de informes radiológicos de resonancias magnéticas cerebrales.

Por otro lado, se está incrementando el número de resonancias magnéticas en los últimos años, por lo que leer texto en formato desestructurado y clasificarlo manualmente es difícil en un periodo de tiempo determinado. Esta manera de automatizarlo podría ser muy beneficiosa para clasificar grandes cantidades de informes automáticamente y con precisión.

Sobre los textos, cabe mencionar que están escritos en inglés y que contienen las descripciones y hallazgos que hicieron radiólogos a partir de resonancias magnéticas del cerebro. Los radiólogos llevaron a cabo esta tarea sin saber si el paciente tenía accidente cerebrovascular isquémico agudo o no.

Los textos que se corresponden a resonancias magnéticas cerebrales de personas que no tienen accidente cerebrovascular isquémico agudo se obtuvieron de pacientes a los que se les hizo una resonancia por otra enfermedad, como un tumor cerebral o una hemorragia intracraneal, u otros pacientes a los que se les hizo un chequeo médico o una evaluación por dolores de cabeza o mareos.

Todos los informes sobre resonancias magnéticas provienen de una única institución académica, durante un periodo de dos años.

Finalmente, se incluyeron 3204 resonancias magnéticas en el análisis final, de las que 432 (14.3%) estaban etiquetadas como AIS. En esta investigación se dividió el conjunto de datos en un 70% para el training set y un 30% para el test set.

Podemos ver en la Figura 20 dos textos de ejemplo describiendo lo que aparece en una resonancia magnética con su correspondiente etiqueta cada uno. La etiqueta 0 significa que esa persona no tiene accidente cerebrovascular isquémico agudo (NO-AIS) mientras que la etiqueta 1 significa que sí lo tiene (AIS).

Etiqueta	Texto
0	1. Diffusion restriction in corpus callosum (genu). ; Enhancement positive. -Early subacute infarct. rule out infection. rule out) lymphoma 2. Multiple old small infarcts in both cerebral deep white matter. 3. Focal old lacunar infarct in right thalamus. 4. magnetic resonance angiography; Focal stenosis of both proximal internal carotid arteries. 5. Diffuse vaso-occlusion of posterior cerebral artery. 6. Multifocal stenosis of right proximal middle cerebral artery (distal M1 & proximal M2).
1	Cystic encephalomalacia in left frontal lobe Diffusion restriction in right basal ganglia and right corona radiata - acute or subacute infarct Left frontal scalp hematoma Complete occlusion of right proximal middle cerebral artery

Figura 20. Textos de ejemplo con su correspondiente etiqueta de AIS o NO-AIS.

Los algoritmos aplicados sobre los textos cuyo rendimiento fue testado fueron regresión binaria logística, clasificador ingenuo Bayesiano, árbol de decisión único y máquinas de vectores de soporte (SVM) para los clasificadores binarios.

El rendimiento de estos algoritmos se evaluó usando la medida F1, que como hemos visto previamente se utiliza para combinar las medidas de precisión y sensibilidad en un sólo valor, lo que es práctico porque hace más fácil poder comparar el rendimiento combinado de la estas medidas entre varias soluciones.

Observamos en la Tabla 1 las métricas correspondientes a los cuatro algoritmos citados previamente con diferentes configuraciones de los mismos. Mencionar que las siglas BLR que aparecen en dicha tabla se corresponden con el algoritmo de regresión binaria logística, mientras que NBC se refieren a clasificador ingenuo Bayesiano, SDT al árbol de decisión único y SVM a las máquinas de vectores de soporte.

	TP	FP	FN	TN	Total	Sensibilidad(recall)	Especificidad	Precision	Accuracy	F1
BLR unigram	100	106	29	671	906	77.5	86.4	48.5	85.1	59.7
BLR tf-idf	102	103	27	674	906	79.1	86.7	49.8	85.7	61.1
BLR adding bigram	64	298	65	479	906	49.6	61.6	17.7	59.9	26.1
BLR adding bigram+tf-idf	60	297	69	480	906	46.5	61.8	16.8	59.6	24.7
NBC unigram	110	170	19	607	906	85.3	78.1	39.3	79.1	53.8
NBC tf-idf	112	170	17	607	906	86.8	78.1	39.7	79.4	54.5
NBC adding bigram	111	112	18	665	906	86.0	85.6	49.8	85.7	63.1
NBC adding bigram +tf-idf	116	118	13	659	906	89.9	84.8	49.6	85.5	63.9
SDT unigram	123	12	6	765	906	95.3	98.5	91.1	98.0	93.2
SDT tf-idf'	123	12	6	765	906	95.3	98.5	91.1	98.0	93.2
SDT adding bigram•	123	12	6	765	906	95.3	98.5	91.1	98.0	93.2
SDT adding bigram+tf-idf	123	12	6	765	906	95.3	98.5	91.1	98.0	93.2
SVM unigram	76	5	53	772	906	58.9	99.4	93.8	93.6	72.4
SVM tf-idf	80	5	49	772	906	62.0	99.4	94.1	94.0	74.8
SVM adding bigram	13	0	86	777	906	33.3	100.0	100.0	90.5	50.0
SVM adding bigram+tf-idf	50	1	79	776	906	38.8	99.9	98.0	91.2	55.6

Tabla 1. Métricas de los diferentes algoritmos clasificando textos en investigación previa [9]

Como podemos observar, los mejores resultados tanto en precisión como en la medida F1 los obtiene el árbol de decisión (SDT) con sus diversas configuraciones.

Por otro lado, datos estructurados como la edad no están incluidos en los datos analizados, aunque sí están disponibles. Agregar datos estructurados que pueden contener información valiosa a los desestructurados es muy posible que sea interesante en el futuro para mejorar el rendimiento de estos algoritmos. Sin embargo, en esta investigación hecha por Kim et al.(2019) [9] sólo se usa texto desestructurado sin identificación.

Hay varias limitaciones al estudio:

1. Los textos solo proviene de una institución, por lo que no es posible generalizar las conclusiones, aunque se podría probar el modelo con otras instituciones.
2. Solo se usan informes sobre resonancias magnéticas cerebrales convencionales completas. En la práctica clínica este tipo de resonancias pueden no hacerse dependiendo de la emergencia de la situación o del paciente. Por tanto, es posible en determinadas circunstancias hacer resonancias sólo de difusión que no tienen todas las características en el texto del accidente cerebrovascular isquémico agudo, ya que el informe solo incluye la descripción de la resonancia de difusión. Por esto, es importante ver las características de cada resonancia de cara al uso de algoritmos.

A modo de conclusión, en esta investigación se apreció que algoritmos de aprendizaje supervisado basados en procesado de lenguaje natural son útiles para la clasificación de informes sobre resonancias magnéticas hechas sobre el cerebro identificando los pacientes con AIS. El mejor algoritmo para este cometido fue un árbol de decisión.

3.2 Caso a tratar

3.2.1 Red utilizada (BERT). Descripción de BERT

BERT (*Bidirectional Encoder Representations from Transformers*) es una red neuronal pre-entrenada de representación del lenguaje creada por Google que salió a la luz a finales de 2018 [13].

Como hemos visto previamente, al contrario que otros modelos de representación de lenguaje, BERT ha sido pre-entrenado con el objetivo de obtener representaciones bidireccionales de manera profunda, es decir, una palabra no se considera de manera independiente sino que se considera el contexto tanto a izquierda como a derecha de la misma. Usa WordPiece embeddings (un estándar de Word embedding) con un vocabulario de 30.000 *tokens*, por lo que el vocabulario que tiene está fijado.

Hay dos pasos dentro de BERT, pre-entrenamiento y *fine-tuning*. Durante el pre-entrenamiento el modelo se entrena en datos que no están etiquetados para diferentes tareas [66]. Para *fine-tuning*, el modelo ya pre-entrenado de BERT se inicializa primero con los parámetros pre-entrenados y después se hace *fine-tuning* modificando todos los parámetros usando datos etiquetados para ajustar el modelo para una tarea concreta.

El proceso de pre-entreno de BERT se hace usando dos tareas de aprendizaje no supervisado:

- Tarea 1: *Masked* LM. Con el objetivo de obtener una representación bidireccional de manera profunda pudiendo considerar el contexto de una palabra a su izquierda y a su derecha, se enmascaran un 15% de las palabras al azar y la tarea consiste en predecir esas palabras que han sido enmascaradas.

Sin embargo, aunque esto permite obtener un modelo pre-entrenado bidireccional, la contrapartida es que se produce una discordancia entre la manera de hacer pre-entrenamiento y *fine tuning*, ya que el token [MASK] que se usa para enmascarar una palabra no aparece durante el *fine-tuning*. Para solucionar esto, no siempre se le puso el *token* [MASK] a las palabras que debían estar enmascaradas. Sobre el 15% de *tokens* elegidos aleatoriamente a los que les tocaría estar enmascarados y ser reemplazados con [MASK] se les hizo lo siguiente:

- A un 80% se le pone efectivamente el *token* [MASK]. Por ejemplo, la frase “mi perro es peludo”, si el *token* “peludo” fuera del 15% de los *tokens* elegidos aleatoriamente, podría ser “mi perro es [MASK]”.

- A un 10% se le pone un *token* aleatorio. Por ejemplo, la frase “mi perro es peludo” podría ser “mi perro es manzana”.

- Al restante 10% se le deja el *token* original. Por ejemplo, la frase “mi perro es peludo” sería “mi perro es peludo” sin ningún cambio.

La ventaja de este procedimiento es que el modelo no sabe qué palabras serán seleccionadas para ser predichas o han sido aleatoriamente reemplazadas, por lo que está forzado a mantener una representación contextual de cada *token*.

- Tarea 2: *Next Sentence Prediction*.

Muchas tareas de NLP como por ejemplo tareas de responder preguntas se basan en entender la relación entre dos sentencias. Para entrenar un modelo que entiende relaciones entre frases, dentro del pre-entrenamiento se lleva a cabo una tarea de predicción de la próxima sentencia

que puede ser generada de manera trivial con cualquier corpus de texto. Concretamente se escogen dos sentencias 'A' y 'B' para cada ejemplo de pre-entrenamiento, un 50% de las veces 'B' es la frase que efectivamente sigue a 'A' y un 50% no.

Una vez el obtenido el modelo pre-entrenado de BERT, podemos hacer *fine-tuning* tan solo añadiendo un capa de output al final del modelo y obtener resultados del estado del arte o cercanos al mismo para un amplio número de tareas como tareas de responder preguntas o clasificación de textos sin hacer grandes modificaciones a nivel de arquitectura. Por tanto, es sencillo ajustarlo para una tarea concreta pasándole los inputs y outputs de dicha tarea, ya que comparado con el proceso de pre-entrenamiento el *fine-tuning* es muy poco costoso

Una característica distintiva de BERT es que tiene una misma arquitectura para varias tareas. Hay una diferencia mínima entre la arquitectura de pre-entrenado y la arquitectura para una tarea concreta como podría ser clasificación de textos, lo que es conceptualmente simple y empíricamente poderoso.

Con su aparición, se demostró que representaciones pre-entrenadas reducen la necesidad de arquitecturas complicadas diseñadas específicamente para una tarea. Se obtuvieron mejores resultados que muchas arquitecturas diseñas específicamente para una tarea haciendo *fine-tuning* sobre el modelo pre-entrenado.

3.2.2 Enfermedad descrita en el estudio

El accidente cerebrovascular, también llamado ictus, es una enfermedad que ocurre cuando un vaso sanguíneo que suministra sangre al cerebro se obstruye o se rompe [69], provocando la consiguiente pérdida de la función neurológica en un área del cerebro [70].

A nivel global suceden aproximadamente 17 millones de accidentes cerebrovasculares anualmente [71], siendo la segunda causa principal de muerte después de enfermedades de arterias coronarias. En Estados Unidos mata a 130.000 personas al año y afecta a unas 800.000 personas, además de ser la tercera causa más común de minusvalía y reducir la movilidad en más de la mitad de las personas mayores de 65 años que lo superan.

Los accidentes cerebrovasculares más comunes son los isquémicos, con un 85% del total aproximadamente, mientras que el resto son hemorrágicos e incluyen a cerebrales y subaracnoideos.

3.2.3 Características de los textos

Como hemos visto previamente, los textos están escritos en inglés y contienen las descripciones y hallazgos que hicieron radiólogos a partir de resonancias magnéticas del cerebro. Los radiólogos llevaron a cabo esta tarea sin saber si el paciente tenía accidente cerebrovascular isquémico agudo o no.

Los textos que se corresponden a resonancias magnéticas cerebrales de personas que no tienen accidente cerebrovascular isquémico agudo se obtuvieron de pacientes a los que se les hizo una resonancia por otra enfermedad, como un tumor cerebral o una hemorragia intracraneal, u otros pacientes a los que se les hizo un chequeo médico o una evaluación por dolores de cabeza o mareos.

Todos los informes sobre resonancias magnéticas provienen de una única institución académica, durante un periodo de dos años. En el análisis se incluyeron textos de 3204 resonancias magnéticas, de las 432 (14.3%) estaban etiquetadas como 'AIS'.

Los textos que se corresponden a pacientes con AIS tienen una mayor cantidad de caracteres de textos que textos de pacientes sin AIS, con un valor mediano de número de caracteres de 551 y 309 respectivamente.

Podemos apreciar ese hecho de manera visual en la Figura 21.

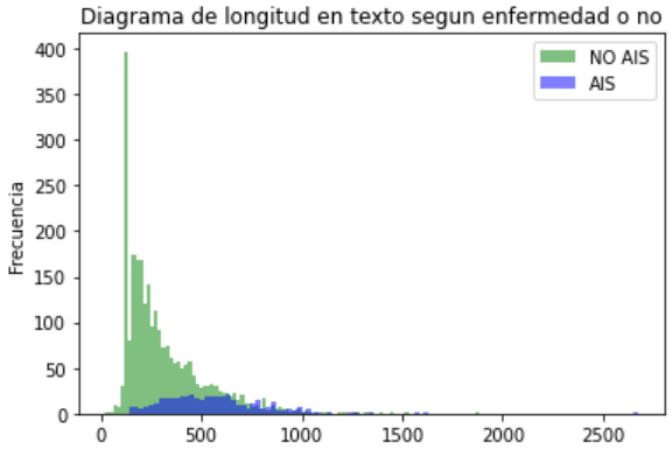

Figura 21. Longitud de los textos según enfermedad o no. Fuente: Elaboración propia

Observamos el mismo patrón si observamos el número de palabras de los textos segmentando por colores si esos pacientes tienen ictus isquémico agudo o no.

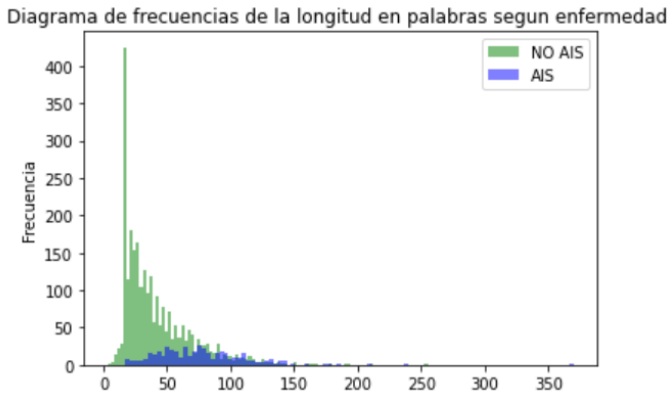

Figura 22. Longitud de las palabras en función de si hay enfermedad o no. Elaboración propia

3.2.4 Separación de textos en training set y test set

En nuestro caso, para llevar a cabo la implementación de BERT y poder comparar con los resultados previos vamos a hacer una separación de un 70% de datos para el training set y un 30% para el test set, de la misma manera que hicieron previamente Kim et al. (2019) [9].

3.2.5 Disponibilidad de los textos

Los textos utilizados en este trabajo pertenecen al trabajo de Kim et al. (2019) [9]. Están disponibles para ser descargados en formato .csv en la publicación de dicho trabajo en la organización científica PLoS One en la siguiente URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0212778#sec017

3.2.6 Tecnologías usadas

El trabajo ha sido realizado en usando el lenguaje Python, el lenguaje más usado para *machine learning* y el que tiene los *frameworks* más usados en esta comunidad de *Deep learning,* Pytorch *y* Tensorflow.

He trabajado en Google Colab, un servicio en la nube que nos proporciona Google gratuitamente al que podemos acceder con un navegador web y ejecutar código en Python sin instalar absolutamente nada en nuestro ordenador [72].

Sus grandes ventajas son:

- Posibilidad de usar una GPU gratis, muy útil teniendo en cuenta que tenemos que entrenar una red neuronal.
- Podemos usar datos que hayamos guardado en Google Drive, como es mi caso que es donde he guardado los textos que usamos.
- Se guarda en la nube, podemos acceder desde distintos dispositivos ya que se guarda en nuestra cuenta de Google.
- Podemos crear un número ilimitado de celdas donde ejecutar código, con lo que es muy fácil depurar.
- Posibilidad de crear archivos que ejecuten Python.

El modelo pre-entrenado que hemos usado ha sido usando la librería pytorch-pretrained-bert, que contiene diferentes modelos pre-entrenados de procesado de lenguaje natural (NLP) y nos proporciona implementaciones en Pytorch de BERT.

Concretamente, he usado el modelo pre-entrenado de BERT que nos proporciona la librería pytorch-pretrained-bert llamado BertForSequenceClassification que trae ya añadido el *output layer* de clasificación al final del modelo pre-entrenado, sólo tenemos que especificar el número de nodos de output que queremos en dicho *layer* final y ajustar el modelo a nuestra tarea concreta [73].

4. Evaluación

Usamos el modelo pre-entrenado de BERT-base, en concreto la versión '*uncased*' que no hace distinción entre mayúsculas y minúsculas en las palabras, lo que suele proporcionar mejor rendimiento. Dicho modelo se compone de 12 *layers* y 110 millones de parámetros, y fue pre-entrenado como dijimos previamente usando los 800 millones de palabras existentes en BookCorpus, una base de datos con más de 10.000 libros.

Hago el mismo proceso que hemos descrito previamente, uso el modelo pre-entrenado de BERT añadiéndole un *output layer* al final de dicho modelo pre-entrenado y entreno el modelo haciendo *finetuning,* ajustando los parámetros de la red neuronal, en especial los parámetros del *output layer*, para clasificación de textos en esta base de datos. Por tanto, usamos *transfer learning* aprovechando el conocimiento adquirido previamente por BERT y ajustándolo para nuestra tarea y conjunto de datos particular.

4.1 Resultados

A la hora de obtener resultados he usado validación cruzada de 4 iteraciones, por lo que en cada una de las iteraciones usaremos un 75% de los datos para entrenar el modelo y un trozo con un 25% de los datos para evaluar al mismo. De esta manera podemos evaluar el modelo con precisión y comparar con los resultados obtenidos previamente por Kim et al. (2019) [9] .

Los autores de BERT recomiendan probar con configuraciones que combinen los siguientes valores para cada uno de estos hiperparámetros:

- Tamaño de *batch*: 16, 32
- *Learning rate* (Adam): 5e-5, 3e-5, 2e-5
- Número de *epochs*: 2, 3, 4

Sin embargo, en este trabajo no hemos podido probar con un tamaño de batch de 16 o 32 debido a que las GPU que tenemos disponibles en Google Colab no son capaces de funcionar con ese tamaño de *batch*. Por este motivo, siempre he trabajado con un tamaño de *batch* de 8 aunque sí hemos podido probar con los diferentes valores de *learning rate* y de número de épocas recomendado.

Comentar que los resultados de entrenamiento e inferencia han sido obtenidos usando una GPU Nvidia Tesla T4 16GB GDDR6 con una frecuencia de 585Mhz.

Los resultados obtenidos con la aplicación de BERT usando las distintas configuraciones de hiperparámetros son los siguientes:

Hiperparámetros	Tiempo entrenamiento	Precision	Recall	Accuracy	F1
Batch=8; Learning rate=5,00E-5; Epochs=2	8:07			85,51%	
Batch=8; Learning rate=5,00E-5; Epochs=3	12:25			85,71%	
Batch=8; Learning rate=5,00E-5; Epochs=4	16:17			85,71%	
Batch=8; Learning rate=3,00E-5; Epochs=2	8:09			90,44%	
Batch=8; Learning rate=3,00E-5; Epochs=3	12:14			88,66%	
Batch=8; Learning rate=3,00E-5; Epochs=4	16:17			88,72%	
Batch=8; Learning rate=2,00E-5; Epochs=2	8:10	93,15%	84,62%	96,83%	88,19%
Batch=8; Learning rate=2,00E-5; Epochs=3	12:15	91,19%	88,64%	96,93%	89,09%
Batch=8; Learning rate=2,00E-5; Epochs=4	17:22	93,26%	90,88%	97,75%	91,82%

Tabla 2. Métricas de las distintas configuraciones de BERT.

Comentar con respecto a la Tabla 2 que aquellas configuraciones de hiperparámetros que tienen varios valores en blanco es porque en varias de las iteraciones de la validación cruzada predecían todos los ejemplos a negativo, con lo cual era imposible calcular varias medidas en esa iteración, como por ejemplo la precisión puesto que no hay ningún valor predicho a positivo y con ello tampoco podemos calcular la medida F1. Consecuencia de esto no podemos hacer la media de la precisión en las cuatro iteraciones ya que en varias de ellas no podemos calcularla. Sin embargo, la *accuracy* no se ve afectada en gran medida cuando calculamos la media de todas las iteraciones, ya que más del 80% del total de los ejemplos es negativo, con lo que si el modelo predice todo a negativo la *accuracy* no es muy baja en esa iteración.

Como podemos observar en la Tabla 2, con el *learning rate* mas grande no mejoramos prácticamente aunque incrementemos el número de épocas, lo que puede tener sentido ya que

con este *learning rate* el modelo aprende más rápido, por lo que es posible que aprenda en menos pasadas y se puede quedar en un conjunto de pesos final sub-óptimo. Esto puede ser el caso, ya que como hemos dicho en varias iteraciones predice todos los ejemplos a negativo.

Sin embargo, observamos que con el *learning rate* más bajo obtenemos los mejores resultados, y mejoramos incrementando el número de *epochs,* lo que tiene sentido pues si cambiamos los pesos del modelo con menos virulencia es de esperar que vaya mejorando al incrementar el número de *epochs*.

Comentar también que los el tiempo necesario para hacer el *fine-tuning* del modelo pre-entrenado con el mayor número de *epochs* es de tan solo 17:22 minutos, y nos da una *accuracy* de casi un 98%. Por otro lado, el tiempo que necesita BERT en promedio para hacer una predicción es de 35 centésimas de segundo, por lo que podría hacer predicciones de manera prácticamente inmediata en una futura aplicación en casos del mundo real.

Ahora comparamos los mejores resultados obtenidos por BERT con los resultados obtenidos previamente en la mejor configuración de cada uno de los cuatro algoritmos implementados en el trabajo de Kim et al. (2019) [9].

Algoritmo	*Precision*	*Recall*	*Accuracy*	F1
BERT	93,26%	90,88%	97,75%	91,82%
Árbol de Decisión (SDT)	91,10%	95,30%	98,00%	93,20%
Regresión binaria logística (BLR)	49,80%	79,10%	85,70%	61,10%
Clasificador ingenuo Bayesiano (NBC)	49,60%	89,90%	85,50%	63,90%
Máquinas de vectores de soporte (SVM)	94,10%	62,00%	94,00%	74,80%

Tabla 3. Comparación métricas de BERT y algoritmos implementados por Kim et al. (2019).

Vemos que BERT obtiene unos resultados muy similares al mejor algoritmo del trabajo previo, el árbol de decisión, y mucho mejores que los demás algoritmos. Para comparar con más detalle a BERT con el árbol de decisión vamos a ver las matrices de confusión obtenidas por BERT y por el árbol de decisión.

		Clase predicha	
		Positiva	Negativa
Clase real	Positiva	123	6
	Negativa	12	765

Tabla 4. Matriz de confusión usando árbol de decisión.

		Clase predicha	
		Positiva	Negativa
Clase real	Positiva	118	11
	Negativa	9	770

Tabla 5. Matriz de confusión usando BERT.*

*Nota: En la validación cruzada de 4 iteraciones de BERT hemos usado un 25% de ejemplos para el test, mientras que en el trabajo de Kim et al. (2019) usaron un 30%. Presentamos la matriz de confusión de BERT incrementando los resultados hasta armonizarlos con el número de ejemplos usados para hacer test en el trabajo previo para así poder comparar.

Entrando a analizar los resultados, podemos ver que en lo referente al número de falsos positivos con BERT obtenemos mejores resultados prediciendo aquellas personas que sí tienen la enfermedad, es decir, a las personas que les decimos que tienen la enfermedad es menos probable que no la tengan.

Sin embargo, también observamos que tenemos un mayor número de falsos negativos con respecto al árbol de decisión, es decir, si usamos BERT le diremos a más personas que no tienen la enfermedad cuando sí la tienen. Teniendo en consideración que se trata de una enfermedad de gravedad, es preferible tener un mayor número de falsos positivos y luego poder hacer una segunda prueba para confirmar, en lugar de tener un mayor número de falsos negativos y clasificar a una persona como que no tiene la enfermedad cuando sí la tiene, lo que podría terminar en consecuencias fatales teniendo en cuenta la gravedad de la enfermedad.

Por tanto, aunque la *accuracy* sea la misma en BERT y en el árbol de decisión, en estas circunstancias sería más interesante el árbol de decisión ya que es preferible tener una mejor puntuación en *recall* debido al menor número de falsos negativos a tener una mayor precisión con menos falsos positivos como en el caso de BERT para esta enfermedad concreta. Esta conclusión merece un par de consideraciones:

- Es posible que se puedan obtener resultados ligeramente mejores en BERT ajustando otras modificaciones que yo no he probado.

- Habría que considerar la facilidad de obtener los resultados, ya que implementar BERT y hacer *fine-tuning* es relativamente sencillo, lleva menos de 20 minutos en una GPU que tenemos a nuestra disposición de manera gratuita en Google Colab.

En conclusión, BERT obtiene resultados ligeramente peores al mejor algoritmo del trabajo previo, el árbol de decisión, y mucho mejores que los otros tres algoritmos para el diagnóstico de una enfermedad de la gravedad del accidente cerebrovascular isquémico agudo. Sin embargo, hay que considerar que BERT es sencillo de implementar y requiere poco tiempo de entrenamiento en una GPU de la que podemos disponer de manera gratuita. Dejo la puerta abierta a que sea posible mejorar ligeramente el rendimiento de BERT con otras modificaciones.

5. Conclusiones y vías futuras

A lo largo de este trabajo hemos visto por qué el procesado de lenguaje natural puede ser muy útil en el ámbito clínico y su potencial de crecimiento a futuro, además de los problemas que habrá que solucionar en ese ámbito. También hemos visto el crecimiento de la aplicación de redes neuronales y sus resultados, y la utilidad que pueden tener los modelos pre-entrenados y el *transfer learning* en general y la red neuronal pre-entrenada BERT en particular.

Hemos comparado los resultados obtenidos por BERT con los mejores resultados obtenidos por los varios algoritmos que implementaron varios investigadores previamente para la detección del accidente cerebrovascular isquémico agudo, usando textos que describían lo que aparecía en resonancias magnéticas del cerebro. En esta comparación hemos visto que BERT obtiene resultados ligeramente peores que el mejor algoritmo del trabajo previo, el árbol de decisión, especialmente considerando la gravedad de la enfermedad que estamos intentando diagnosticar donde los falsos negativos cobran especial importancia. Sin embargo, hay que considerar que BERT es sencillo de implementar y requiere poco tiempo de entrenamiento en una GPU de la que podemos disponer de manera gratuita, además de que es posible que se pueda mejorar ligeramente el rendimiento de BERT para este caso en concreto con mayores modificaciones.

Por otra parte, es de destacar que el tiempo que necesita BERT en promedio para hacer una predicción es de 0.035 segundos, o lo que es lo mismo, 35 centésimas de segundo, por lo que podría hacer predicciones de manera prácticamente inmediata en una futura aplicación en casos del mundo real.

Por tanto, concluimos que la red neuronal pre-entrenada BERT puede ser muy útil de cara a diversas tareas de procesado de lenguaje natural en el ámbito clínico, pues es capaz de obtener resultados excepcionales con poco esfuerzo tanto en términos de programación como de tiempo y recursos computacionales.

En cuanto a la elaboración del trabajo, he encontrado bastante dificultoso encontrar una base de datos que hiciera posible probar la implementación de BERT en el ámbito clínico. Esto es debido a que, como hemos analizado a lo largo del trabajo, es muy difícil obtener datos clínicos de un hospital o una serie de hospitales por problemas de privacidad principalmente, por lo que tuve que emplear decenas de horas y mirar decenas de artículos científicos en varias webs especializadas hasta encontrar una base de datos que pudiera usar.

Por otro lado, aunque soy consciente de la extensión del estado del arte, creía necesario justificar el por qué es importante el procesado del lenguaje natural en el ámbito clínico, por qué está creciendo en los últimos años y qué papel pueden ocupar las redes neuronales en general y los modelos pre-entrenados como BERT en particular usando *transfer learning*. Aunque también emplee una importante cantidad de tiempo en leer literatura científica sobre este tema, creí importante además de obtener buenos resultados en un caso real hacer una revisión de este ámbito en general.

De cara a futuros trabajos se podría usar BERT para otras tareas de procesado de lenguaje natural en el ámbito clínico. En caso de que se use BERT para una tarea de clasificación de textos, sea esta u otra, se podría intentar agregar datos estructurados como la edad, el peso o el sexo

a los textos en formato desestructurado para ver si mejoran los resultados. También se podría estudiar la aplicación de conocimiento médico al modelo, por ejemplo en forma de un post procesado a los resultados del modelo. También merecería la pena implementar esta misma tarea probando con tamaños de *batch* de 16 y de 32 que como hemos visto no ha sido posible debido a contar con tan solo una GPU de 16 GB de memoria.

Bibliografía

[1] Alok Aggarwal, "Resurgence of AI During 1983-2010". https://www.kdnuggets.com/2018/02/resurgence-ai-1983-2010.html .Accessed:2020-10-20

[2] "Infographic: How AI is Being Deployed Across Industries" https://www.roboticsbusinessreview.com/ai/infographic-how-ai-is-being-deployed-across-industries/ , Abril 2019. Accessed: 2020-10-23

[3] "Artificial Intelligence & Deep Learning | How it works and Expected Market" https://devisionx.com/technologies/deep-learning/ . Accessed: 2020-10-23

[4] "Deep Learning". En Wikipedia. https://en.wikipedia.org/wiki/Deep_learning .Accessed: 2020-10-25

[5] "Natural language processing". En Wikipedia. https://en.wikipedia.org/wiki/Natural_language_processing .Accessed: 2020-10-26

[6] D. Demner-Fushman, W. W.Chapman,C.J. McDonald. "What can natural language processing do for clinical decision support?" , Journal of Biomedical Informatics, Vol 42, Pages 760-772, October 2009.

[7] "Transfer learning". En Wikipedia. https://en.wikipedia.org/wiki/Transfer_learning .Accessed: 2020-10-27.

[8] "Google: cómo funciona BERT, la mayor actualización del algoritmo del motor de búsqueda más usado en el mundo." https://www.bbc.com/mundo/noticias-50223408 . Accessed: 2020-10-25

[9] C. Kim,V. Zhu,J. Obeid,L. Lenert. "Natural language processing and machine learning algorithm to identify brain MRI reports with acute ischemic stroke" PLoS One 2019; 14(2): e0212778. Feb 28

[10] S. Velupillaiab, H. Suominencd, M. Liakatae, A. Robertsa, A. D.Shahfg, K. Morleyah, D. Osbornij, J. Hayesij, R. Stewartak, J. Downsak, W. Chapmanl, R. Duttaak. "Using clinical Natural Language Processing for health outcomes research: Overview and actionable suggestions for future advances" Journal of Biomedical Informatics, Vol 88, Pages 11-19, December 2018.

[11] H. Kong. "Managing Unstructured Big Data in Healthcare System" Healthc Inform Res. 2019 Jan; 25(1): 1–2.

[12] I. Spasic, G. Nenadic. "Clinical Text Data in Machine Learning: Systematic Review". JMIR Med Inform. 2020 Mar; 8(3): e17984.

[13] Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"

[14] Jerry Wei. "BERT: Why it's been revolutionizing NLP" https://towardsdatascience.com/bert-why-its-been-revolutionizing-nlp-5d1bcae76a13 . Sep 2020. Accessed: 2020-10-25

[15] "GLUE Benchmark". https://gluebenchmark.com/ Accessed: 2020-10-28

[16] "Red neuronal artificial". En Wikipedia.https://es.wikipedia.org/wiki/Red_neuronal_artificial#Historia . Accessed: 2020-10-30

[17] "History: The 1940's to the 1970's" https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html . Accessed: 2020-10-30

[18] Fran Ramírez, "Historia de la IA: Frank Rosenblatt y el Mark I Perceptrón, el primer ordenador fabricado específicamente para crear redes neuronales en 1957" https://web.archive.org/web/20180722124753/https://data-speaks.luca-d3.com/2018/07/historia-de-la-ia-frank-rosenblatt-y-el.html , July 2020. Accessed: 2020-11-02.

[19] "Bernard Widrow". En Wikipedia. https://en.wikipedia.org/wiki/Bernard_Widrow . Accessed: 2020-11-03.

[20] "Neurona". En Wikipedia. https://es.wikipedia.org/wiki/Neurona Accessed: 2020-11-03.

[21] "Learning rule demonstration". En Wikipedia. https://lucidar.me/en/neural-networks/learning-rule-demonstration/ Accessed: 2020-11-04.

[22] "Perceptron". En Wikipedia.https://en.wikipedia.org/wiki/Perceptron .Accessed: 2020- 11-04.

[23] "Neural Network". https://databricks.com/glossary/neural-network .Accessed: 2020- 11-04.

[24] F. Malik, "What are hidden Layers" https://medium.com/fintechexplained/what-are-hidden-layers-4f54f7328263 .Mayo 2019. Accessed: 2020- 11-05.

[25] "Everything you need to know about Neural Networks" https://medium.com/ravenprotocol/everything-you-need-to-know-about-neural-networks-6fcc7a15cb4 . Dic 2017. Accessed: 2020- 11-06.

[26] Eda Kavlakoglu, "AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?" https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks , Mayo 2020. Accessed: 2020-11-07.

[27] "Hidden Layer Definition | Deep AI" https://deepai.org/machine-learning-glossary-and-terms/hidden-layer-machine learning#:~:text=In%20neural%20networks%2C%20a%20hidden,inputs%20entered%20into%20the%20network . Accessed: 2020-11-07.

[28] Soria, E., & Blanco, A. (2007). "Redes Neuronales Artificiales".

[29] "A Quick Introduction to Neural Networks". https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/ , Agosto 2016. Accessed: 2020-11-07.

[30] "Introduction to Recurrent Neural Network" https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/ , Oct 2018. Accessed: 2020-11-07.

[31]Jordi Torres. "Redes neuronales recurrentes. https://torres.ai/redes-neuronales-recurrentes/ . Sep 2019. Accessed: 2020-11-07.

[32] E.Pekel, Selin Kara. "A comprehensive review for artifical neural network application to public transportation". Sigma J Eng & Nat Sci 35 (1), 2017, Pag: 157-179

[33] Akshat Maheshwari, "Report on Text Classification using CNN, RNN & HAN" https://medium.com/jatana/report-on-text-classification-using-cnn-rnn-han-f0e887214d5f , Jul 2018. Accessed: 2020-11-11.

[34]"Convolutional neural network". En Wikipedia. https://en.wikipedia.org/wiki/Convolutional_neural_network . Accessed: 2020-11-11.

[35] "Types of artificial neural networks". En Wikipedia. https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#Convolutional Accessed: 2020-11-11.

[36] Machine Learning TV. (27 de Julio ,2018). "Simple Deep Neural Networks for Text Classification". Recuperado de https://www.youtube.com/watch?v=wNBaNhvL4pg&t=794s

[37] D. Britz. "Understanding Convolutional Neural Networks for NLP" http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/ .Nov 2015. Accessed: 2020-11-11.

[38] T. Shah ."About Train, Validation and Test Sets in Machine Learning" https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7 . ,Dic 2017. Accessed: 2020-11-12.

[39] Jordi Torres, "¿Qué es la inteligencia artificial?" https://torres.ai/que-es-la-inteligencia-artificial/ . May 2019. Accessed: 2020-11-12.

[40] V.Roman. "Aprendizaje No Supervisado en Machine Learning: Agrupación" https://medium.com/datos-y-ciencia/aprendizaje-no-supervisado-en-machine-learning-agrupaci%C3%B3n-bb8f25813edc Jun 2019. Accessed: 2020-11-13.

[41] "Reinforcement learning". En Wikipedia. https://en.wikipedia.org/wiki/Reinforcement_learning . Jun 2019. Accessed: 2020-11-13.

[42] J. Brownlee. "Overfitting and Underfitting With Machine Learning Algorithms", https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/ Mar 2016. Accessed: 2020-11-14.

[43] "Overfitting Underfitting". En Wikipedia. https://en.wikipedia.org/wiki/Overfitting#Underfitting . Accessed: 2020-11-14.

[44] A. Al-Masri. "What Are Overfitting and Underfitting in Machine Learning?" https://towardsdatascience.com/what-are-overfitting-and-underfitting-in-machine-learning-a96b30864690 , Mar 2019. Accessed: 2020-11-14.

[45] "Qué es overfitting y underfitting y cómo solucionarlo" https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/ , Dic 2017. Accessed: 2020-11-18.

[46]Jordi Torres, "Data Augmentation y Transfer Learning" https://torres.ai/data-augmentation-y-transfer-learning-en-keras-tensorflow/#62Transfer_Learning , Dic 2019. Accessed: 2020-11-18.

[47] Andrés Gonzalez, "Conceptos Básicos de machine learning" https://cleverdata.io/conceptos-basicos-machine-learning/#:~:text=Clasificaci%C3%B3n%20y%20regresi%C3%B3n%20(classification%20and,el%20anteriormente%20mencionado%20del%20spam. Accessed: 2020-12-14

[48] "Reconocimiento de entidades nombradas". En Wikipedia. https://es.wikipedia.org/wiki/Reconocimiento_de_entidades_nombradas . Accessed: 2020-12-15

[49] Andreas Herman, "Different ways of doing Relation Extraction from text" https://medium.com/@andreasherman/different-ways-of-doing-relation-extraction-from-text-7362b4c3169e , Mayo 2019. Accessed: 2020-12-15

[50] S. Kumar, "14 Popular Evaluation Metrics in Machine Learning" https://towardsdatascience.com/14-popular-evaluation-metrics-in-machine-learning-33d9826434e4 , Jun 2020. Accessed: 2020-11-18.

[51] Jose Martinez Heras, "Precision, Recall, F1, Accuracy en clasificación" https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/ , Oct 2020. Accessed: 2020-11-22.

[52] Jason Brownlee, "A Gentle Introduction to k-fold Cross-Validation" https://machinelearningmastery.com/k-fold-cross-validation/ , Mayo 2018. Accessed 2021-01-14

[53] Pallavi Pandey, "Cross-Validation in scikit-learn" https://machinelearninggeek.com/cross-validation-in-scikit-learn/ , Noviembre 2020. Accessed 2021-01-14

[54] "Regresión logística". En Wikipedia. https://es.wikipedia.org/wiki/Regresi%C3%B3n_log%C3%ADstica .Accessed: 2020-11-22.

[55] Jacob Avila, "Clasificador Naive Bayes" https://www.jacobsoft.com.mx/es_mx/clasificador-naive-bayes/ , Mayo 2020. Accessed: 2020-11-23.

[56] "Applying Multinomial Naive Bayes to NLP Problems" https://www.geeksforgeeks.org/applying-multinomial-naive-bayes-to-nlp-problems/ , Enero 2019. Accessed: 2020-11-23.

[57] T. Boros, S. D. Dumitrescu, S. Pipa. "Fast and Accurate Decision Trees for Natural Language Processing Tasks". In Proc. of Recent Advances in Natural Language Processing, pp 103-110.

[58] Yaoyong Li, Bontcheva K. & Cunningham H. (2009). Adapting SVM for Natural Language Learning: A Case Study Involving Information Extraction.Natural Language Engineering,15(2), 241-271. Cambridge University Press New York, NY, USA. [DOI:10.1017/S1351324908004968]

[59] V. Singh, "What is Frameworks? [Definition] Types of Frameworks" https://hackr.io/blog/what-is-frameworks , Abril 2020. Accessed: 2020-11-22.

[60] "The State of Machine Learning Frameworks in 2019" https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominates-industry/ , Oct 2019. Accessed: 2020-11-22.

[61] "PyTorch vs TensorFlow". http://horace.io/pytorch-vs-tensorflow/ . Accessed: 2020-11-24.

[62] “Comparison of AI Frameworks”. https://wiki.pathmind.com/comparison-frameworks-dl4j-tensorflow-pytorch#torch . Accessed: 2020-11-25.

[63] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang. “Pre-trained models for natural language processing: A survey”. arXiv preprint arXiv:2003.08271, 2020.

[64] J. Brownlee, “What Are Word Embeddings for Text?”https://machinelearningmastery.com/what-are-word-embeddings/ , Oct 2019. Accessed: 2020-11-25.
[65] “NLP's ImageNet moment has arrived” https://thegradient.pub/nlp-imagenet/ , Jul 2018. Accessed: 2020-11-25.

[66] S. Velupillai, H. Suominen, M. Liakata ,A. Roberts ,AD. Shah,K. Morley ,D. Osborn ,J. Hayes , R. Stewart , J. Downs , W. Chapman, R. Dutta “Using clinical Natural Language Processing for health outcomes research: Overview and actionable suggestions for future advances” J Biomed Inform. 2018 Dec; 88:11-19.

[67] S. Wu , K. Roberts , S. Datta , J. Du , Z. Ji , Y. Si , et al. “Deep learning in clinical natural language processing: a methodical review”. Journal of the American Medical Informatics Association. 2020; 27(3):457–70.

[68] S. Kothawade, “Breakdown The BERT In Pieces…” https://medium.com/subex-ai-labs/breakdown-the-bert-in-pieces-df46f60b65d8 , Abril 2020. Accessed: 2020-11-30.

[69] Ana Callejo Mora, “Ictus cerebral: Tratamientos, síntomas, causas e información” https://cuidateplus.marca.com/enfermedades/neurologicas/ictus.html, Enero 2019. Accesed: 2020-12-11

[70] Michael S Phipps, Carolyn A Cronin. “Management of acute ischemic stroke”, BMJ 2020; 368:l6983.

[71] Brandi R. French, Raja S. Boddepalli, Raghav Govindarajan, “Acute Ischemic Stroke: Current Status and Future Directions” Missouri Medicine, 2016 Nov-Dec; 113(6): 480–486.

[72] “¿Machine Learning en la Nube? Google Colaboratory con GPU!” https://www.aprendemachinelearning.com/machine-learning-en-la-nube-google-colaboratory-con-gpu/ , Feb 2019. Accessed: 2020-11-30.

[73]“PyTorch-Transformers |PyTorch” https://pytorch.org/hub/huggingface_pytorch transformers/ . Accessed: 2020-12-10.