Práctica Final
Predicción del género de libros

Inteligencia Artificial en las Organizaciones

Grupo 83-1

Miguel Gutiérrez Pérez
100383537@alumnos.uc3m.es

Mario Lozano Cortés
100383511@alumnos.uc3m.es

Alba Reinders Sánchez
100383444@alumnos.uc3m.es

Alejandro Valverde Mahou
100383383@alumnos.uc3m.es

GitHub: ***InteligenciaArtificialOrganizaciones***

16 de diciembre de 2020

Índice

1. Introducción

El objetivo de esta práctica consiste en abordar una clasificación sobre resúmenes de libros para determinar su género literario. Las razones que llevan a la elección de este problema tienen que ver con que actualmente cualquier persona con la dedicación suficiente puede escribir un libro sin la necesidad del patrocinio de una editorial, lo que conlleva una **explosión en el número de nuevos libros generados**. Por consiguiente, las librerías y bibliotecas necesitan catalogar una gran cantidad de escritos, lo cual, les lleva a necesitar de métodos de clasificación automática. Por ello, se plantea el uso de resúmenes y meta-datos de los libros puesto que la tarea de clasificación **debe poder realizarse con el menor número de datos posible**, puesto que no todos los libros que llegan a estas entidades disponen de todos los datos completos.

La primera cuestión imprescindible que surge al conocer el problema propuesto es qué técnica de Inteligencia Artificial emplear. Dado que se realiza un análisis sobre diferentes textos, la opción evidente es la **Minería de Texto**, la cual es una técnica de minería de datos que busca extraer **información útil y relevante de documentos de texto** de diferentes fuentes diferentes, como puede ser páginas web, correos electrónicos, periódicos o redes sociales. Para ello, se hace una identificación de patrones en los datos, como puede ser la repetición de palabras o conjuntos de palabras, estructuras sintácticas que se repitan a lo largo de los datos, etc. Esta minería de texto tiene numerosas aplicaciones, y en esta práctica se va a desarrollar una clasificación en función de unas categorías que serán definidas gracias a la elección de un conjunto de datos apropiado.

A continuación se ofrece un **esquema del funcionamiento de la tarea propuesta** en donde un libro sin catalogación llega a alguna a de estas entidades que necesitan catalogar su género a partir de la información más reducida posible, generalmente título y argumento. Inicialmente se plantea la distinción de un único género, sin embargo, **es bien sabido que un escrito no tiene por qué adscribirse a un único género** y por lo tanto se debe considerar como futura **ampliación** catalogar tantos como sea posible.

Title: Animal Farm

Plot: *Animal Farm is a novel about a group of animals who take control of the farm they live on. The animals get fed up of their master, Farmer Jones, so they kick him out. Once they are free of the tyrant Jones, life on the farm is good for a while and there is hope for a happier future of less work, better education and more food. However, trouble brews as the pigs, Napoleon and Snowball, fight for the hearts and minds of the other animals on the farm. Napoleon seizes power by force and ends up exploiting the animals just as Farmer Jones had done. The novel ends with the pigs behaving and even dressing like the humans the animals tried to get rid of in the first place.*

⟹ **Genre:** Political satire

Figura 1: Esquema de la tarea

2. Contexto de la práctica

La minería de datos, y más en concreto, la Minería de Texto, tal y como se ha comentado en trabajos pasados [1], es un área que está sufriendo una gran revolución gracias a las redes neuronales. Esta revolución es similar a la que sufrieron las redes convolucionales desde la aparición de las arquitecturas de Deep Learning y permite enfoques novedosos como el desarrollado en esta práctica gracias a la técnica de *embeddings*.

La aplicación de *embeddings* supone por lo tanto un punto de partida muy interesante para numerosas tareas de *text-mining*, así, se encuentran *paper* académicos que muestran las ventajas y mejora de resultados obtenidas por estas técnicas. En este contexto surge el estudio de Alex Dekhtyar y Vivian Fong sobre la catalogación de requisitos de software aplicando *embeddings* y comparando los resultados con los obtenidos con otras técnicas como Naïve Bayes [2]. El objetivo de dicho estudio era realizar clasificación sobre requisitos relacionados y no relacionados con asuntos de seguridad. Usando para ello las arquitecturas de *word embedding word2vect desarrolladas* por Google [3] se puede obtener una representación semántica muy acertada de las palabras. La arquitectura propuesta es por lo tanto muy similar a una red neuronal convolucional de procesamiento de imágenes.

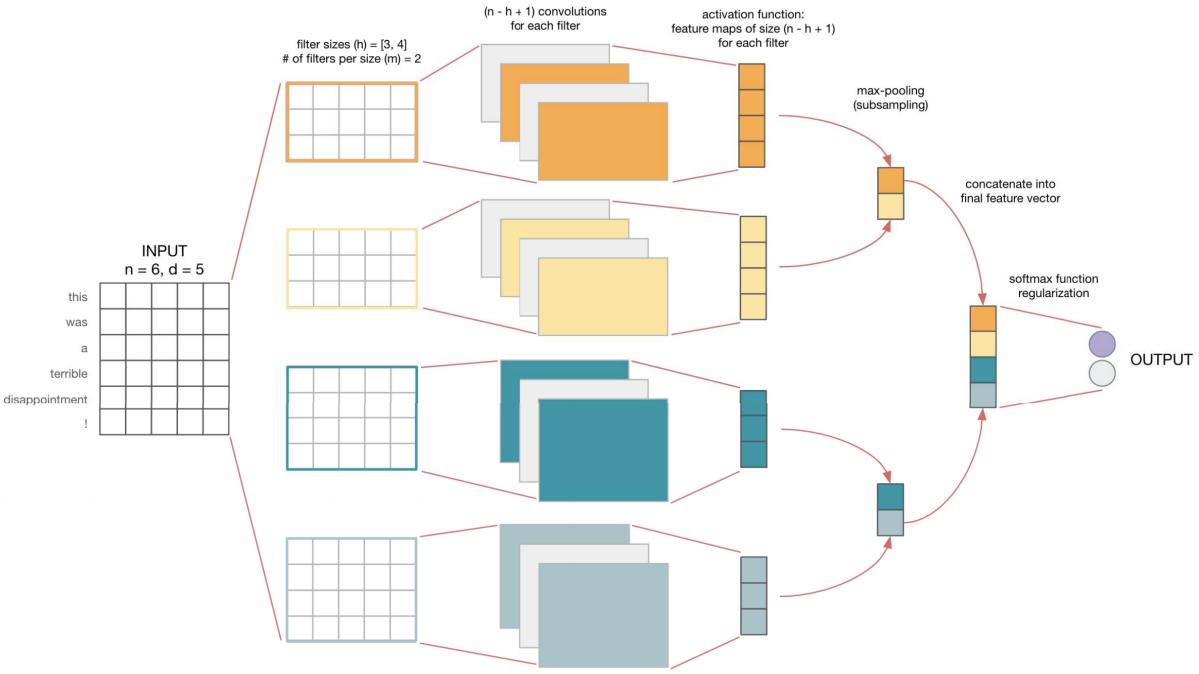

Figura 2: Esquema de la red utilizada

La arquitectura propuesta muestra un conjunto de convoluciones que hace posible aplicar numerosos filtros de varios tamaños de forma que se representan así los distintos n-gramas de cualquier frase. Posteriormente se aplican mapas de características y se acaba con una clasificación binaria de los requisitos.

Los resultados obtenidos por los autores demuestran que la inclusión de *embeddings* hace que se mejore la precisión (*precision*) sin sacrificar la sensibilidad (*recall*), obteniendo un F1-Score del 7 %. **Por todo ello resulta muy interesante aplicar técnicas similares a las propuestas por los autores a la clasificación de géneros de libros propuesta en este documento**.

3. Conjunto de datos

En esta sección se describe el **conjunto de datos** utilizado en su forma original, así como todos los cambios que se hacen como parte del **preprocesado** junto con las razones que han llevado a su realización.

3.1. Estructura original

El conjunto de datos ha sido obtenido de **Kaggle** [4], y dispone de un total de **54283 libros y 12 columna** con información sobre ellos. En la Figura X se puede apreciar visualmente la forma de las instancias de este conjunto de datos.

Author(s)	Description	Edition	Format	ISBN	No. Pages	Avg. Rating	No. Ratings	No. Reviews	Title	Genres	Cover Image
William Golding	At the dawn of the next world war, a plane crashes on an uncharted island, stranding a group of...	Penguin Great Books of the 20th Century	Paperback	9.78E+12	182	3.66	1840595	30634	Lord of the Flies	Classics|Fiction|Young Adult|Academic|School|Literature	Link

Figura 3: Instancia del conjunto de datos original

A continuación se explica el **contenido** de cada columna en más detalle:

- **Author(s):** Cadena de caracteres que indica el o los autores del libro. En caso de ser más de uno, cada autor aparece separado por '|'.
- **Description:** Cadena de caracteres que indica el resumen o descripción del libro.
- **Edition:** Cadena de caracteres que indica la edición del libro.
- **Format:** Cadena de caracteres que indica el formato del libro, como por ejemplo *hard-cover* o *paperback*.
- **ISBN:** Valor numérico que indica el ISBN del libro. Debido a que se utiliza notación científica para su representación, no se puede leer ni usar correctamente, pues faltan números.
- **No. Pages:** Valor numérico que indica el número de páginas que tiene el libro.
- **Avg. Rating:** Valor numérico que indica la valoración media que los usuarios han dado al libro.
- **No. Ratings:** Valor numérico que indica la cantidad de valoraciones de usuario que ha recibido el libro.
- **No. Reviews:** Valor numérico que indica la cantidad de críticas que ha recibido el libro.
- **Title:** Cadena de caracteres que indica el título del libro.
- **Genres:** Cadena de caracteres que indica los géneros a los que pertenece el libro. Los diferentes géneros aparecen separados por '|'.
- **Cover Image:** Imagen que muestra la portada del libro. Se indica un enlace que lleva hasta dicha imagen, aunque por motivos de espacio no se ha incluido en la figura de arriba.

En un primer momento se consideró usar el conjunto de datos **CMU Book Summary Dataset** [5], pero la manera en la que estaban dispuestos los géneros, así como el resto de atributos, daba lugar a un **procesado más complejo** para obtener sus valores.

Además, se consideró que los géneros que se utilizaban no eran muy acertados, como por ejemplo *Speculative fiction* o *Postmodernism*. Posteriormente se encontró el conjunto de datos que se ha descrito arriba, el cual tiene muchas más instancias (54,283 frente a 16,559), proporciona mucha más información (tiene más atributos) y, lo más importante, es mucho más sencillo obtener los valores de sus instancias.

3.2. Preprocesado

En toda tarea de Minería de Texto existe algún tipo de **preprocesado**, ya sea obtener sólo los valores que se necesiten, eliminar instancias con errores, vectorizar el texto, etc. Esta vez no va a ser diferente, e incluso se puede afirmar que ha sido una **parte importante del trabajo**, y que ha ocupado una cantidad considerable de tiempo. A continuación se detallan todas las **modificaciones** que se han llevado a cabo:

- **Selección de las columnas útiles:** De las 12 columnas que tiene el conjunto de datos seleccionado, solo son necesarias dos, *description* y *genres*, que contenían, respectivamente, el resumen o descripción del libro y los géneros que se le atribuyen. Por ello, solo solo se mantendrán dichas columnas eliminando el resto.
- **Eliminación de instancias con descripciones en un lenguaje diferente al ingles:** Los resúmenes del conjunto de datos se encuentran escritos no solo en inglés, si no también en árabe, italiano, chino, coreano, japonés, portugués...Se ha decidido por razones evidentes, mantener solo aquellos libros cuyos resúmenes estén en ingles. Para la detección del idioma se ha utilizado la librería ***langdetect*** [6] de *Python*, y aquellas instancias donde se detectaba un idioma diferente al inglés se han borrado. Gracias a la gran cantidad de instancias que hay, la eliminación de las instancias no afecta de manera significativa.
- **Corrección de formatos incorrectos:** Algunos de los resúmenes del conjunto de datos contenían errores de formato como saltos de línea (de diferentes tipos), tabulaciones o espacios en medio de los resúmenes, así que se sustituyeron por un único espacio en blanco.
- **Eliminación de caracteres no útiles:** Se han eliminado de los resúmenes del conjunto de datos caracteres que no proporcionaban ningún tipo de información, como comillas, guiones, corchetes, paréntesis, puntos, exclamaciones, interrogaciones, etc.
- **Eliminación de términos no útiles:** Se han eliminado de los resúmenes del conjunto de datos términos que no proporcionaban ningún tipo de información útil. Para ello se ha utilizado una lista de ***stopwords***, reutilizada de la segunda práctica de la asignatura, pero a la cual se han añadido términos que se han ido encontrado al realizar pruebas. Evidentemente, no se han incluido todos los términos sin información útil posibles, pero dada la gran cantidad de términos que puede haber, creemos que la cantidad de ***stopwords*** obtenida es suficiente.
- **Cambios requeridos por la aproximación usada:** La aproximación que se utilizada en este trabajo requiere que los resúmenes esté en minúsculas. Al contrario que ocurría en la segunda práctica, no es necesario dividir los resúmenes en términos, por lo que no se utilizara la cadena de caracteres que contiene el resumen de manera directa.
- **Eliminación de géneros no útiles:** Después de analizar un poco más a fondo los diferentes géneros que se usan, se ha observado la falta de un criterio claro para elegir que géneros usar, dando lugar a géneros de países, como *spain*, al mismo tiempo que el género *spanish literature*. También se encontraron géneros muy concretos y sin mucho sentido, como *Amazon* o *Apple*. Esto llevó a que fuera necesario revisar manualmente todos los

géneros, los cuales eran en torno a 850, y eliminar aquello que se consideraron no útiles. Al finalizar esta revisión quedaron 625 géneros.

Una vez aplicado todo esto, los datos ya están preparados para su uso. Sin embargo, queda una última cosa. La cadena de caracteres que contiene los géneros se separara según el caracter '|' en los diferentes géneros para su posterior uso. Como en este trabajo se va explorar una clasificación **multi-clase** con el primer género, y una clasificación ***multi-label*** con todos los géneros, para la primera solo se utilizará el primer género, y para la segunda se usarán todos.

4. Conceptos teóricos

A la hora de elegir un método de Minería de Texto se consideran diversas opciones tales como utilizar la herramienta *Weka* de la *Universidad de Waitako*. Sin embargo, dado que esta herramienta ya ha sido utilizada a lo largo de la asignatura y, además, presenta ciertas limitaciones en cuanto a la flexibilidad y capacidad de toma de decisiones de diseño en los modelos, se decide aplicar **Word Embedding** en Redes de Neuronas con la **biblioteca de código abierto *Tensorflow***. De esta manera se usa un método diferente al usado en clase, lo cual permite experimentar y aprender tecnologías nuevas.

No obstante, antes de iniciar la codificación del modelo, dado que se trata de una técnica nueva, es necesario tener claros los conceptos teóricos involucrados. Una Red Neuronal únicamente procesa números, lo que implica que es necesario realizar una transformación. Representar las palabras como vectores es importante, ya que los modelos de IA no 'entienden' las palabras, y no pueden realizar cálculos ni aprendizaje sobre ellas. Por este motivo, es necesario realizar una **vectorización**, que no es más que transformar estas palabras en números, agrupados en forma de vector. A continuación se exponen las diferentes técnicas de vectorización consideradas.

4.1. Codificación con valor único

Una primera aproximación podría ser asignar un único número a cada una de las palabras consideradas en la vectorización. Por ejemplo, con la frase 'hace un espléndido día' se obtiene un vector como el que sigue:

Palabra	Valor
hace	1
un	2
espléndido	3
día	4

Tabla 1: Codificación con un único valor

Este enfoque presenta una serie de consideraciones importantes:

- El valor de cada palabra se decide de manera arbitraria.
- No se obtiene una representación fiel de la distancia entre palabras, lo cual es un hecho que es importante en el desarrollo de esta práctica.

Por ello, se descarta el enfoque aquí propuesto por no ser eficiente en la tarea a realizar.

4.2. One-hot encoding

La codificación *one-hot* consiste en convertir cada palabra en un vector con tantas posiciones como palabras se tengan, con un 1 en la posición que se corresponda a la palabra considerada y un 0 en caso contrario.

A continuación se muestra un ejemplo de la codificación con el pequeño set de palabras utilizado en la sección anterior.

	hace	**un**	**espléndido**	**día**
hace	1	0	0	0
un	0	1	0	0
espléndido	0	0	1	0
día	0	0	0	1

Tabla 2: One-hot encoding

El principal problema de esta técnica de vectorización es que la distancia entre las palabras es la misma, lo cual impide de nuevo disponer de una representación realista de este hecho esencial en el enfoque que se propone.

4.3. Embedding

Está técnica sirve para representar aquellas **palabras que son semánticamente parecidas con una codificación similar**. Lo que hace esta técnica tan atractiva es que no es necesario especificar esta similitud de forma manual.

Un *embedding* es un vector denso de números reales, donde su longitud viene determinada por parámetro. En lugar de determinar los pesos a mano, se tratan como **parámetros que pueden ser entrenados**, como si de pesos de Redes de Neuronas densas se trataran. Por eso mismo, esta técnica funciona especialmente bien con las Redes de Neuronas, que incorpora la modificación de estos pesos a la fase de entrenamiento de la red.

La dimensionalidad del *embedding* debe ser proporcional a la cantidad de datos disponibles, ya que cuanto más grande sea el vector, más detalles podrá obtener de cada palabra, pero requiere de más datos para ser entrenado. La siguiente tabla muestra un ejemplo de la codificación con el pequeño conjunto de palabras utilizado en la sección anterior.

hace	2.32	7.35
un	0.89	4.23
espléndido	0.75	8.65
día	2.65	3.00

Tabla 3: Embedding

Por lo tanto, esta técnica resulta de gran utilidad a la hora de conseguir representar la distancia semántica entre las palabras que forman un texto. Siendo este hecho fundamental para la tarea propuesta **se decide apostar por esta codificación para construir el modelo de *Text Mining***.

5. Clasificación

Normalmente un libro se encuadra en más de un género literario, por ello, de manera inicial se decide enfocar el problema en la clasificación de un solo género para posteriormente, poder realizar una ampliación del algoritmo que prediga todos los géneros posibles de un escrito.

Dado que la técnica que se desea utilizar es prácticamente exclusiva para los modelos de Redes de Neuronas, este es el acercamiento que se decide implementar.

5.1. Arquitectura del modelo

Una vez se cargan los datos preprocesados se realiza una aleatorización de los mismos para evitar posibles sesgos, se vectorizan las salidas y se dividen los datos en conjunto de entrenamiento (70 %) y en conjunto de test (30 %). Los modelos considerados siguen todos la misma estructura de Red de Neuronas:

- **Capa de vectorización del texto**: Transforma cadenas de caracteres a índices de vocabulario, el vocabulario se crea a partir de la frecuencia de valores individuales. Se modifican los siguientes parámetros:
  - **Número máximo de tokens**: Representa el tamaño del vocabulario a usar.
  - **Longitud de la secuencia de salida**: número de índices de palabras que se pasa a la capa siguiente.
- **Capa de *embedding***: A partir del vocabulario busca el vector de *embedding* para cada índice de palabras. Estos vectores se aprenden según el modelo se entrena.
- **Capa de agrupación promedio global**: Devuelve un vector de salida de longitud fija para cada ejemplo haciendo la media sobre la dimensión de la secuencia. Se utiliza para permitir a la red usar los datos del *embedding*.
- **Capa densa**: Neuronas completamente conectadas. Según el modelo puede tener una o varias capas y el número de neuronas puede variar por capas.
- **Capa de salida**: Capa densa que tiene tantas neuronas como géneros distintos haya.

De forma gráfica se puede presentar la estructura como se muestra en la Figura X.

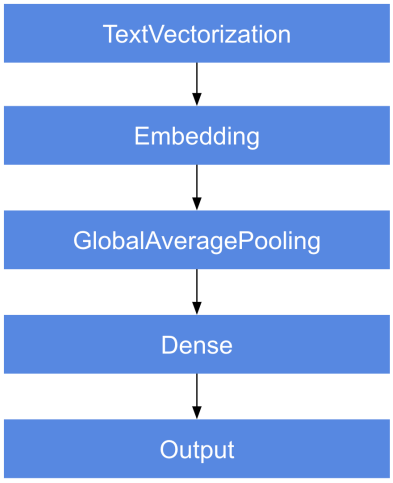

Figura 4: Arquitectura del modelo

5.2. Clasificación de 1 género

La idea de usar un sólo género a la hora de clasificar un libro es útil si este género es el principal del mismo, sin embargo, el *conjunto de datos* original no hace una diferenciación sobre esto y establece todos los géneros de un libro con la misma importancia. Por ello, se ha decidido guardar el primer género de cada libro, aunque esto **podría suponer un sesgo** sobre el modelo, el cual se debe tener en cuenta.

Por ejemplo, si un libro pertenece a los géneros Ciencia ficción, Aventura y Fantasía, al entrenar únicamente con el primer género puede que la salida la red lo clasifique como Fantasía y lo trate como un error al desconocer que también pertenece a este género. Por ello, es de suma importancia realizar la ampliación multi-género.

Al realizar las transformaciones oportunas seleccionando el primer género **se obtienen un total de 194 géneros diferentes**. Se trata por lo tanto de un problema de clasificación multi-clase ya que se tienen 194 clases y cada instancia tiene una sola etiqueta.

Para poder ser utilizadas por la red, las salidas se vectorizan usando *one-hot encoding* para representar a estas clases, donde cada posición es un género distinto. Un 0 significa que no pertenece a ese género y un 1 que sí pertenece. Por lo tanto, la forma que tiene el vector de una salida cualquiera es: [0, 0, . . . , 0, 1, . . . , 0]. El cual tiene tamaño 194 y un sólo 1.

Otro aspecto importante que hay que tener en cuenta al tratarse de un problema multi-clase es la función de coste. Se usa **entropía cruzada categórica** (*Categorical Cross-Entropy*) porque se desea entrenar a la red para sacar como salida la probabilidad para todas las clases sobre cada uno de los ejemplos. Esta función de coste es la más utilizada en problemas de clasificación multi-clase, e indica el porcentaje de veces que el modelo otorga el valor más alto de creencia a la clase acertada.

La Figura X muestra un ejemplo de la codificación *one-hot* en la vectorización de los géneros de la novela '1984'.

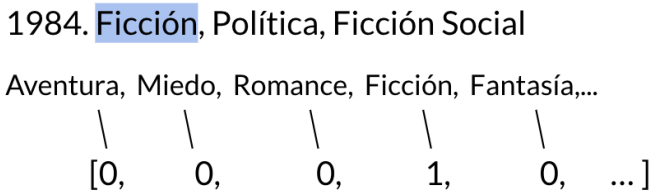

Figura 5: One-hot encoding de 1 género

5.3. Clasificación multi-género

Por el sesgo comentado en el apartado anterior al elegir solo el primer género, se ha decidido implementar también la clasificación de todos los géneros. Sin embargo, esto **aumenta significativamente la dificultad** del problema, pues para un mismo texto hay diferentes géneros, además de que no todos los libros pertenecen a la misma cantidad de géneros.

Al separar todos géneros se obtienen **625 géneros diferentes**, que es una cantidad tres veces mayor que en el problema de un único género. Se trata de un problema *multi-label*, ya que cada instancia puede pertenecer a una o más etiquetas.

La vectorización de la salida ocurre de la misma manera que en caso de un único género, salvo que ahora este vector tendrá una longitud de 625, y tendrá tantos unos como cantidad de géneros a los que el libro pertenezca. En la Figura X se muestra la codificación *one-hot* en la vectorización de la salida para un libro que pertenece a los géneros: ficción, aventura y fantasía.

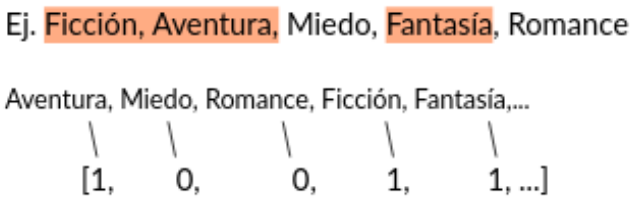

Figura 6: One-hot encoding multi-género

Por último, esta vez como función de coste se utiliza **entropía cruzada binaria** (Binary Cross-Entropy), debido a que funciona de manera independiente para cada clase, o en este caso, género.

6. Experimentación y análisis

Con el objetivo de obtener los mejores resultados, es necesario realizar una experimentación que modifique los distintos hiper-parámetros del modelo. Dado que la tecnología usada está ligada con las Redes de Neuronas, se ha probado exclusivamente con este tipo de modelos. Los parámetros que se han decidido modificar son:

- **Batch Size**: Hace referencia al tamaño de los lotes, es decir, cada cuantos individuos se actualizan los pesos de la red. Un tamaño mayor lleva a un menor tiempo de ejecución, y menos especificidad sobre los ejemplos.
- **N Grams**: Número máximo de palabras que puede tener un *token*.
- **Sequence Length**: Número de índices de palabras que recibe la capa de *embedding*.
- **Vocab Size**: Tamaño total del vocabulario a usar. El vocabulario está compuesto de las palabras más comunes en los textos.
- **Embedding Dimension**: Tamaño de los vectores de *embedding*.
- **Topology**: Topología de las capas densas de la red. En caso de que incluyan una capa de *dropout*, se representa con una **D** acompañado de su valor.

Para los resultados obtenidos, el número de ciclos se calcula para que tenga el valor óptimo para cada uno de los experimentos para evitar el sobre aprendizaje.

La métrica que se decide usar es *Categorical Accuracy*, dado que se tiene más de una clase sobre la que clasificar.

6.1. Modelo 1 género

Este primer acercamiento se hace exclusivamente con el primer género de cada libro. A través de este acercamiento se tienen **194 clases**, por tanto, la última capa del modelo posee **194 neuronas**.

Los experimentos que se han realizado para este acercamiento se han ido creando progresivamente según se buscaba mejorar los resultados. Se van modificando los hiper-parámetros con el fin de conseguir el mayor *accuracy* en test.

A continuación se muestra una tabla que resume esta experimentación y los resultados obtenidos, Figura X:

	batch size	n grams	sequence length	vocab size	embedding dim	topology	train accuracy	test accuracy
Exp. 1	32	None	100	10000	16	(128)	0,6493	0,4746
Exp. 2	32	**5**	100	10000	16	(128)	0,6503	0,4734
Exp. 3	32	5	100	10000	16	**(128 D0,4)**	0,6407	0,4555
Exp. 4	32	5	100	10000	**64**	**(128 D0,3; 256 D0,3)**	0,5962	0,4697
Exp. 5	32	5	100	10000	**128**	(128 D0,3; 256 D0,3)	0,641	0,482
Exp. 6	32	5	**50**	10000	128	(128 D0,3; 256 D0,3)	0,5702	0,4525
Exp. 7	32	5	**150**	10000	128	(128 D0,3; 256 D0,3)	0,6185	0,4813
Exp. 8	32	5	150	**8000**	128	(128 D0,3; 256 D0,3)	0,7318	0,4713
Exp. 9	32	5	150	**12000**	128	(128 D0,3; 256 D0,3)	0,6385	0,4882
Exp. 10	32	5	150	**14000**	128	(128 D0,3; 256 D0,3)	0,5802	0,4832
Exp. 11	32	5	**200**	**12000**	128	(128 D0,3; 256 D0,3)	0,9072	0,473

Figura 7: Resumen experimentación y resultados del modelo 1 género

Como se puede apreciar en la tabla de la Figura X, los resultados son bastante aceptables, con valores de *accuracy* en test que se encuentran entre el **47 %** y **48 %**, siendo el **Exp. 9** el que mejores resultados genera con un **48,82 %**. Aun así, se puede ver que los resultados obtenidos no varían demasiado según los hiper-parámetros. Los análisis que se pueden realizar sobre estos hiper-parámetros son:

- Aumentar la complejidad de la red genera mejores resultados.
- Si se intenta disminuir el *batch size*, el tiempo de computo crece exponencialmente, así que no se ha modificado. Se ha decidido no aumentarlo por encima de 32 porque, como dijo Yann LeCun, padre fundador de las redes convolucionales, *'Training with large minibatches is bad for your health. More importantly, it's bad for your test error. Friends dont let friends use minibatches larger than 32'*(https://twitter.com/ylecun/status/989610208497360896?lang
- Usar *ngrams* es útil. Se decide usar tamaño 5, aunque en la exploración del vocabulario, el tamaño máximo de *ngram* es 4, y la gran mayoría de ellos mantiene el tamaño 1. Se usa para permitir *ngrams* de cualquier tamaño.
- Se prueban diferentes valores de *sequence length*, y los mejores resultados se encuentran con valores de 150.
- El tamaño del vocabulario que se ha probado que mejor funciona es 12000.
- Aumentar el tamaño del *embedding* mejora los resultados, pero también aumenta considerablemente el tiempo de ejecución. Por eso no se aumenta por encima de 128.

A pesar de que los resultados son mejorables, se considera un buen acercamiento a este problema, asumiendo que es una simplificación del problema real. También hay que tener en cuenta que este problema tiene un gran sesgo que puede deberse a que tan solo se coge un único género, por lo que la red puede que esté clasificando un género del libro realmente, pero no es conocido por el modelo, y por tanto lo trata como un error.

Por ejemplo, si un libro pertenece a **Aventura** y **Ciencia ficción**, y el modelo lo clasifica como **Ciencia ficción**, se considerará un error.

6.2. Modelo multi-género

La experimentación que se lleva a cabo para este segundo acercamiento se hace con todos los géneros que puede llegar a tener un libro. Con esto se pretende ampliar el problema, e intentar determinar todos los posibles géneros de un libro, acercándose más a un problema real. Pero hay que tener en cuenta que se tienen **625 clases** por lo que la última capa del modelo tiene **625 neuronas**, lo que aumenta la complejidad del acercamiento. Además como se trata de un problema multi-etiqueta, su dificultad es todavía mayor.

El proceso de creación de los experimentos se basa en ir modificando la configuración de los hiper-parámetros ligeramente para encontrar aquella con la que se obtienen mejores resultados. Se van subiendo o bajando estos valores para intentar alcanzar un *accuracy* en test lo más alto posible.

En la Figura X se muestra una tabla resumen de todos los experimentos realizados junto con sus resultados:

	batch size	n grams	sequence length	vocab size	embedding dim	topology	train accuracy	test accuracy
Exp. 1	32	None	100	10000	16	(128)	0,2445	0,2245
Exp. 2	32	None	100	10000	16	**(128; 256)**	0,2463	0,2264
Exp. 3	32	None	100	10000	16	**(128; 256; 512)**	0,2157	0,2241
Exp. 4	32	**2**	100	10000	16	(128)	0,2498	0,2277
Exp. 5	32	**5**	100	10000	16	(128)	0,2459	0,2267
Exp. 6	32	**2**	100	10000	**64**	(128)	0,2354	0,2229
Exp. 7	32	2	100	**15000**	**16**	(128)	0,2502	0,2261
Exp. 8	32	2	**50**	**10000**	16	(128)	0,2445	0,2242
Exp. 9	32	2	**150**	10000	16	(128)	0,247	0,2274
Exp. 10	32	2	150	**15000**	16	(128)	0,2509	0,227
Exp. 11	32	2	150	**8000**	16	(128)	0,2441	0,2246
Exp. 12	32	2	150	**5000**	16	(128)	0,2359	0,2204
Exp. 13	32	2	**200**	**8000**	16	(128)	0,2426	0,2241
Exp. 14	32	2	**250**	8000	16	(128)	0,2462	0,2231
Exp. 15	**16**	2	**100**	**10000**	16	(128)	0,2401	0,2314
Exp. 16	**8**	2	100	10000	16	(128)	0,2296	0,2122

Figura 8: Resumen experimentación y resultados del modelo multi-género

Las conclusiones a las que se llegan son:

- Aumentar la complejidad de la topología no mejora los resultados.
- Utilizar *2 grams* es positivo y genera mejores resultados, aunque inspeccionando el vocabulario, se comprueba que no hay muchos *ngrams* que sean de tamaño 2.
- El tamaño de vocabulario de 10000 palabras es el más adecuado en este caso.
- El tamaño del vector de *embedding* de 16 posiciones es el que mejor funciona.
- El número de índices que recibe la capa de *embedding* se prueba con distintos valores pero se obtienen mejores resultados con 100.
- El *batch size* se modifica en los últimos experimentos consiguiendo una mejora de resultados con un tamaño de 16.

Por tanto se concluye con que el mejor experimento es el **Exp. 15**, con un *accuracy* en test de **23,14 %**. Este valor no es especialmente alto pero, dada la complejidad del problema, se puede considerar como un éxito.

6.3. Caso de uso

7. Conclusiones

Respecto a la asignatura en sí, este trabajo ha servido para observar en primera persona la realización de un **tarea *real***, donde ha habido que buscar diversos conjuntos de datos, analizarlos, y elegir el que mejor se adecuaba a las necesidades del problema. Después, ha sido necesario un **preprocesado importante**, donde se han encontrado problemas con una complejidad mayor que los que aparecen en el resto de prácticas de la asignatura (y en otras en general).

Por último se comprende que **es posible no obtener datos muy buenos**, pues, de nuevo, es un problema complicado de resolver. Todo lo comentado muestra lo diferente que son los problemas del **mundo académico** a los del **mundo real** o problemas que no están tan acotados.

Poniendo la atención sobre la rama de la Inteligencia Artificial utilizada, la **minería de datos**, comentar que es un ámbito, no solo **complejo**, si no además muy **variado**. Se ha optado por la aproximación realizada por el hecho de incluir variedad en la aproximaciones utilizadas en la asignatura, así como para usar una técnica que sea mas personalizable que las vistas en clase.

Sin embargo, esto no quiere decir que algunas técnicas mas clásicas o simples funcionarán peor. Si el alcance de este trabajo fuera mayor y se dispusiera de mas tiempo, sería casi imperativo **probar otras técnicas** (vistas y no vistas en clase) y realizar una comparación entre ellas de manera exhaustiva.

Referencias

[1] M. Gutierrez, M. Lozano, A. Reinders, and A. Valverde, "Análisis de reseñas de tripadvisor," p. 11, 2020.

[2] A. Dekhtyar and V. Fong, "Re data challenge: Requirements identification with word2vec and tensorflow," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp. 484–489, IEEE, 2017.

[3] T. M. et al., "world2vec." `https://code.google.com/archive/p/word2vec/`, July 2013.

[4] N. Santhanam, "Goodreads' best books ever." `https://www.kaggle.com/meetnaren/goodreadsbest-books/activity`, Dec. 2018.

[5] D. Bamman and N. Smith, "Cmu book summary dataset." `https://www.cs.cmu.edu/~dbamman/booksummaries.html`, 2013.

[6] N. Shuyo, "langdetect 1.0.8." `https://pypi.org/project/langdetect/`, Mar. 2020.