Universidad de Deusto

Fundamentos del Procesamiento del Lenguaje Natural

Entrega Final del Proyecto

HudaAI

Estudiantes:	Unai Olaizola Osa Diego López Aroca
DNIs:	73032586L 72842984Y
Emails:	unai.olaizola.osa@opendeusto.es diego.lopez.a@opendeusto.es
Repositorio GitHub:	NLP-Group-Project

11 de enero de 2026

Índice general

Capítulo 1

Propuesta de Proyecto

1.1 Abstract

A diferencia de la búsqueda léxica, que se limita a recuperar pasajes únicamente cuando contienen una coincidencia exacta de palabras, la búsqueda semántica es un tipo de retrieval que busca captar el significado contextual de los términos. De este modo, una consulta como "paraíso" también podría devolver referencias a "jardines" o "moradas eternas". Este reto se intensifica en el caso del Corán y del árabe clásico: el texto sagrado está profundamente marcado por metáforas, alusiones y un lenguaje altamente polisémico, mientras que el árabe presenta una compleja morfología y variaciones léxicas que hacen especialmente difícil la desambiguación y la correspondencia de significados. Por ello, mediante este trabajo pretendemos desarrollar un buscador semántico que, a partir de modelos de representación lingüística entrenados en árabe clásico y apoyados en recursos en inglés, sea capaz de identificar equivalencias contextuales más allá de las coincidencias de palabras o sinónimos. El objetivo es capturar la riqueza interpretativa del Corán y superar las limitaciones de la búsqueda léxica tradicional, ofreciendo resultados más coherentes con el trasfondo religioso, histórico y lingüístico del texto.

1.2 Motivación

Nuestro objetivo con este proyecto es emplear diferentes técnicas del procesamiento del lenguaje natural en conjunto alrededor de una idea que consideramos original.

Siempre desde el más profundo respeto, ya que vamos a tratar con un documento religioso como es el Corán, deseamos analizar ciertos aspectos y embarcarnos en los desafíos que conlleva el análisis de este documento.

1.2.1 Objetivos y herramientas

He aquí las tareas y herramientas que tenemos pensadas emplear:

- Contamos con la suerte de haberlo encontrado en un formato correcto y además en varios idiomas. Por lo que hemos decidido que vamos a trabajar con el texto en árabe e inglés, para poder realizar un más profundo estudio y comparación.
- Al ser el árabe un idioma tan extendido y estudiado, ofrece varias librerías y herramientas que facilitarán ciertos aspectos de nuestro proyecto en gran parte. Entre ellas ya hemos usado e indagado profundamente en CAMeL Tools [Obe+20].
- Por otra parte, consideramos que la "tokenización" en arábe resultará en un experimento interesante ya que la caracterización árabe deriva en reglas diferentes.
- Otra de las tareas que pensamos realizar es desarrollar un buscador basado en retrieval semantico para devolver el pasaje del Corán más similar o más fiel a un concepto o idea introducida. Algunos de estos conceptos abstractos podrían ser: "adoración", "paraíso", "prohibiciones", "Yihad", …

1.3 Originalidad y búsqueda semántica (semantic retrieval)

Nuestro proyecto incorpora un aspecto poco explorado en PLN aplicado al Corán: la búsqueda semántica. Según la revisión en **Arabic natural language processing for Qur'anic research: a systematic review** [Bas+23], la mayoría de los trabajos sobre el Corán se centran en herramientas de concordancia, análisis léxico y búsquedas exactas, y destacan la falta de recursos anotados para árabe clásico. Apenas se mencionan aplicaciones de retrieval semántico en textos religiosos, lo que refuerza la originalidad de nuestra propuesta de ajustar modelos multilingües para desarrollar un buscador conceptual del Corán.

1.3.1 Métodos de evaluación para la búsqueda semántica

Para evaluar los resultados obtenidos con nuestra implementación de un buscador semántico, hemos barajado varias opciones después de haber investigado varias publicaciones al respecto. Los siguientes serían los que meditamos emplear, o al menos, tener en cuenta:

- On-topic rate: Métrica que evalúa la importancia de la búsqueda dada una query. Un alto valor en esta métrica significa que está funcionando bien, mientras que un valor bajo muestra que se necesita una mejora. [Zhe+24]
- Normalized Discounted Cumulative Gain (nDCG): Mide la eficacia de la clasificación en todos los resultados de búsqueda, teniendo en cuenta la relevancia ponderada de cada documento. Un valor alto indica una clasificación de relevancia adecuada para los documentos. [Mah+24]
- Cosine Similarity: este método básico de la estadística va a ser empleado para comparar la similitud entre los embeddings de diferentes palabras. Un valor cercano a 1 mostraría que las palabras son similares y -1 que son antónimas por así decirlo.

1.4 Estado del arte / Related Work

Estos últimos años han visto surgir diferentes iniciativas relacionando el tratamiento del Corán con técnicas del procesamiento de lenguaje natural. En este breve apartado, vamos a anotar los más importantes e influyentes proyectos que pueden estar relacionados y servir de inspiración para nuestra tarea en esta asignatura.

1.4.1 Proyectos y materiales destacados

- **A A New Semantic Search Approach For The Holy Quran Based On Discourse Analysis And Advanced Word Representation Models** [LD24]. Ha sido seguramente el paper que más nos ha inspirado para recolectar ideas para el proyecto. Publicado por Samira Lagrini y Amina Debbah en el año 2024, profundiza en la investigación de la búsqueda semántica en árabe, también con el Sagrado Corán. Haciendo fuerte hincapié en la dificultad que el Corán tiene debido a las alusiones y referencias religiosas a veces incluso incomprensibles para los humanos. Además, en la investigación se usan varios de los conceptos con los que vamos a tratar, entre otros: el modelo "FastText", la métrica de similitud "Cosine Similarity" entre otros.
- **Al-Bayan: An Arabic Question Answering System for the Holy Quran:** [Abd+14] presenta un sistema de preguntas y respuestas en árabe enfocado en el Corán. Su objetivo es permitir consultas semánticas precisas, superando la dificultad de interpretar referencias religiosas complejas. El sistema combina técnicas de procesamiento de lenguaje natural y modelos de representación de palabras para mejorar la recuperación de información relevante del texto coránico.
- **Embedding search for quranic texts based on large language models:** [Alq24] Este estudio publicado por Mohammed Alqarni desde la Universidad de Jeddah, Arabia Saudita, también profundiza en la búsqueda semántica empleando textos Coránicos. Lo que lo diferencia y relaciona con nuestro proyecto es que emplea embeddings comunes y embeddings derivados de LLMs para comparar los resultados.
- **Quranic Conversations: Developing a Semantic Search tool for the Quran using Arabic NLP Techniques:** [SSA23] Este paper publicado en el año 2023 también explica la implementación de una búsqueda semántica con modelos pre-entrenados y métodos de evaluación como la similitud

de coseno. Por otra parte, emplean 30 *tafsirs* (comentarios sobre los significados de los versículos, contexto histórico y sabiduría). Convirtiendo estos en tensores, emplean la búsqueda semántica para luego buscar el siguiente *tafsir* más similar para devolver el *ayah* (pasaje) correspondiente.

- **Detecting semantic-based similarity between verses of the quran with doc2vec:** [AAA21] se emplea la similaridad de coseno para medir la similitud de versos del Corán representados mediante el modelo *Doc2Vec*. Este modelo es una extensión del modelo *Word2Vec*, representa documentos completos en un espacio vectorial. Paper publicado por los autores Alshammeri, Menwa and Atwell, Eric and ammar Alsalka, Mhd en el año 2021.

Capítulo 2

Procesamiento y Análisis de Datos

2.1 Introducción

Antes de empezar con la documentación correspondiente al trabajo hecho para esta segunda entrega, hemos de aclarar que ya habíamos comenzado con el análisis de nuestros datos en la primera entrega. Por lo tanto, para esta segunda hemos añadido contenido al notebook donde se analizan los datos y hemos empezado con tareas que se requerirán en las futuras entregas. No obstante, en este documento, nos limitaremos a explicar nuestro código relacionado al análisis de datos de nuestro proyecto *HudaAI*.

2.2 Carga del dataset del Corán

Para realizar el análisis del texto sagrado, pensamos en importar el texto en el formato más amigable para el análisis. Para nuestra suerte, encontramos el texto en un formato el cual no requería de mucho retoque. Aún así, creamos un script llamado *prepare_data.py* para normalizar, limpiar y preparar los datos para el análisis. En este, hacemos uso de la librería *camel_tools* la cual ofrece herramientas para procesar texto árabe. En este archivo importamos el texto y lo modificamos para normalizarlo, pero debido a que tratamos con el Corán en inglés y en árabe, el proceso es diferente.

- La normalización del Corán en árabe elimina los diacríticos llamados (ḥarakāt), para cortar las marcas de vocalización y símbolos fonéticos. De esta manera, los diacríticos no interferirán en el análisis.
- En cuanto a la normalización en inglés, nos hemos limitado a eliminar mediante expresiones regulares cualquier caracter que no sean las letras del alfabeto inglés y saltos de espacio.

Por finalizar, dividimos el dataset en las columnas de 'Capítulo', 'Versículo' y 'Texto' (para ambos idiomas), quedándonos con el dataset en el siguiente formato:

	Capítulo	Versículo	Texto
0	1.0	1.0	بسم الله الرحمن الرحيم
1	1.0	2.0	الحمد لله رب العالمين
2	1.0	3.0	الرحمن الرحيم
3	1.0	4.0	مالك يوم الدين
4	1.0	5.0	اياك نعبد واياك نستعين

Figura 2.1: Dataset limpio del Corán en árabe

2.3 Estadísticas básicas

Empezamos el análisis de los datos extrayendo datos básicos como el número de versos, el número de capítulos, obteniendo los siguientes resultados:

- Número de versos del Corán (tanto en árabe como en ingés): 6236 versos en total.
- Número de capítulos del Corán: 114 capítulos.
- Número total de palabras: 82627 palabras.
- Número promedio de palabras por verso: 13 palabras.
- Distribución de longitud de los versos en árabe:

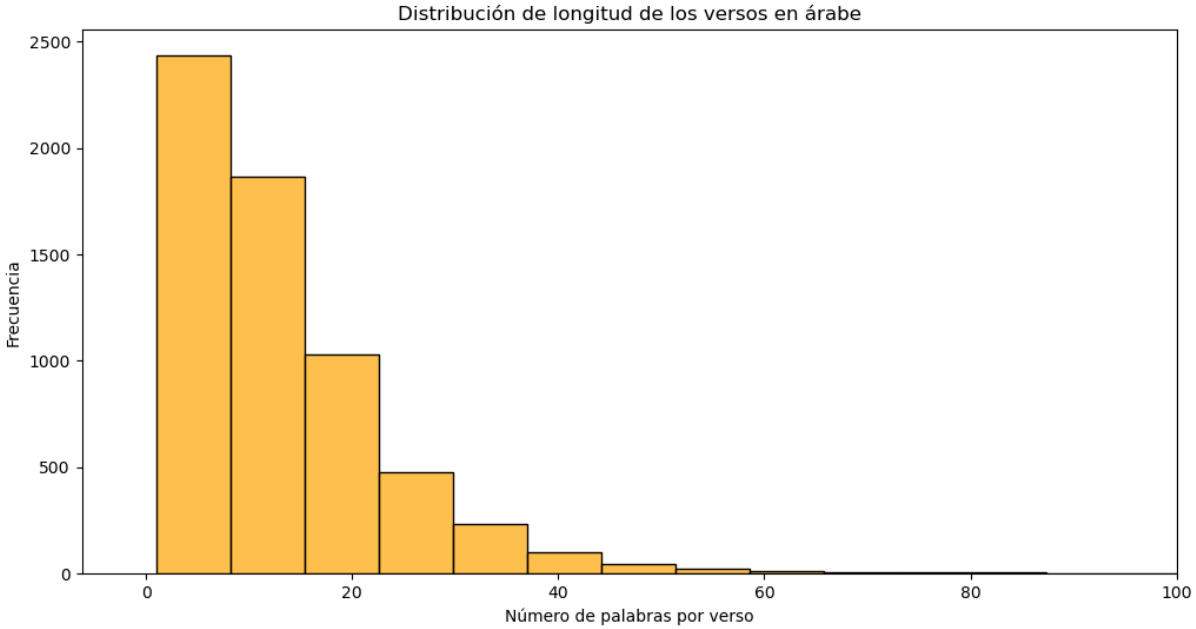

Figura 2.2: Gráfico de la distribución de longitud de los versos

- Longitud de versos del Corán:

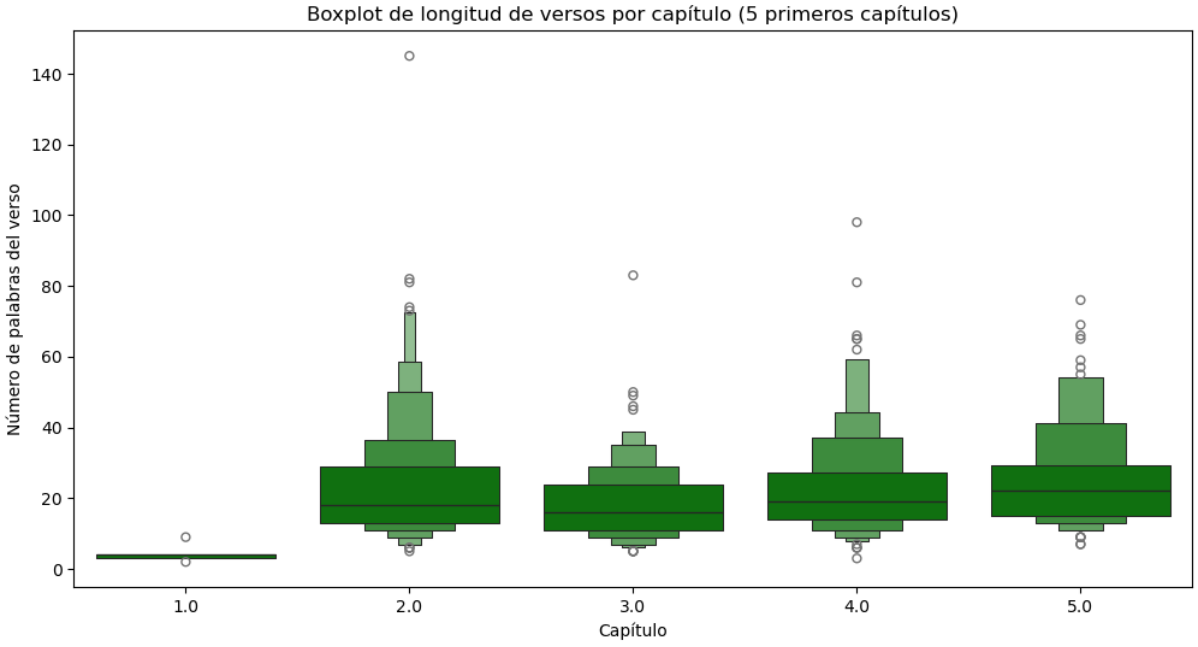

Figura 2.3: Boxplot de longitud de versos

Luego probamos con la librería camel_tools que nos proporciona herramientas de tokenización, análisis y eliminación de ambigüedades en palabras. En esta parte empezamos eliminando las stop words en árabe mediante la librería nltk y tokenizando gracias al tokenizer simple_word_tokenize.

A continuación, usamos el BERTUnfactoredDisambiguator para desambiguar las palabras, extrayendo el lema y una glosa aproximada en inglés.

Hasta aquí, obtenemos:

- No stop words.
- Tokens normalizados y lematizados.
- Tokens desambiguados.
- Un significado aproximado del token en inglés.

Posteriormente realizamos POS Tagging para analizar la distribución gramatical del vocabulario.

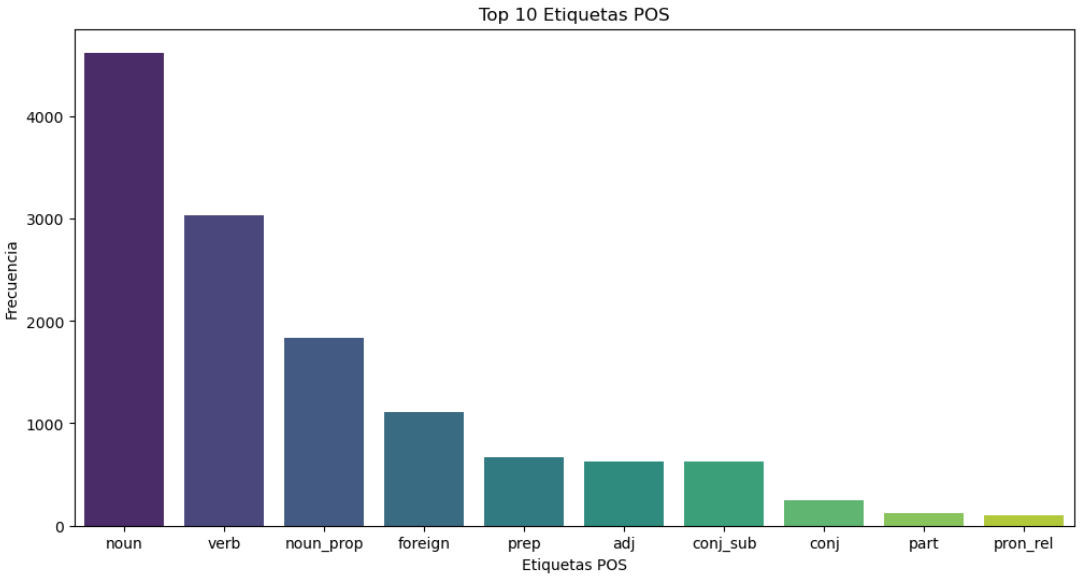

Figura 2.4: POS Tagging del Corán

También analizamos la frecuencia de tokens usando la clase Counter.

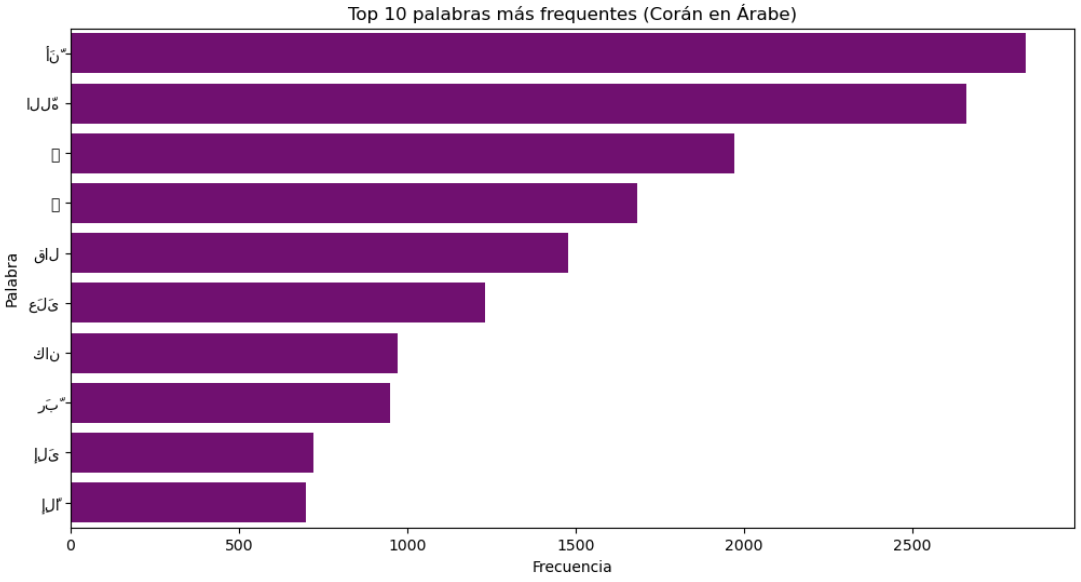

Figura 2.5: Top 10 palabras más frecuentes Corán (árabe)

Tradujimos estas palabras usando las glosas en inglés:

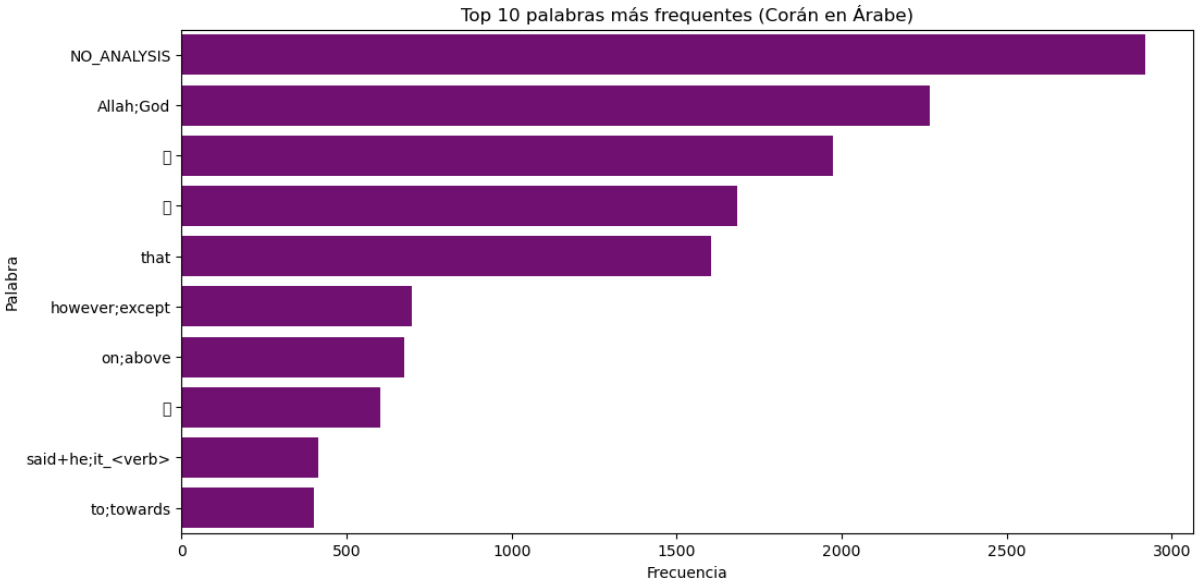

Figura 2.6: Top 10 palabras más frecuentes Corán (árabe - traducción inglés)

Tras limpiar traducciones incorrectas y stop words:

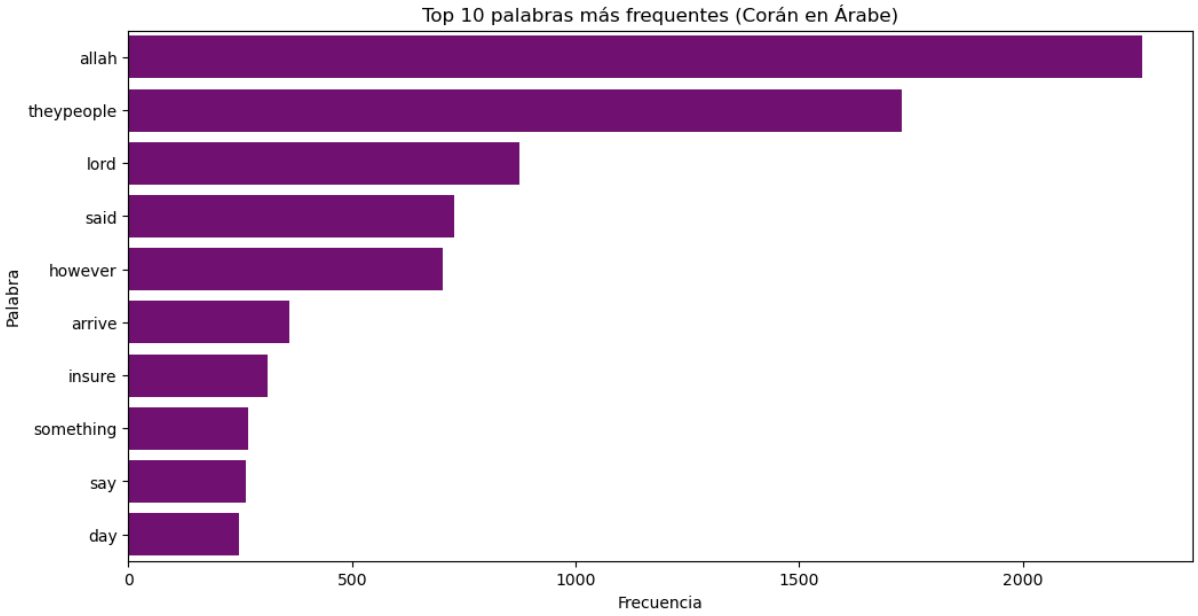

Figura 2.7: Top 10 palabras más frecuentes Corán (árabe - traducción inglés)

También analizamos la frecuencia en inglés:

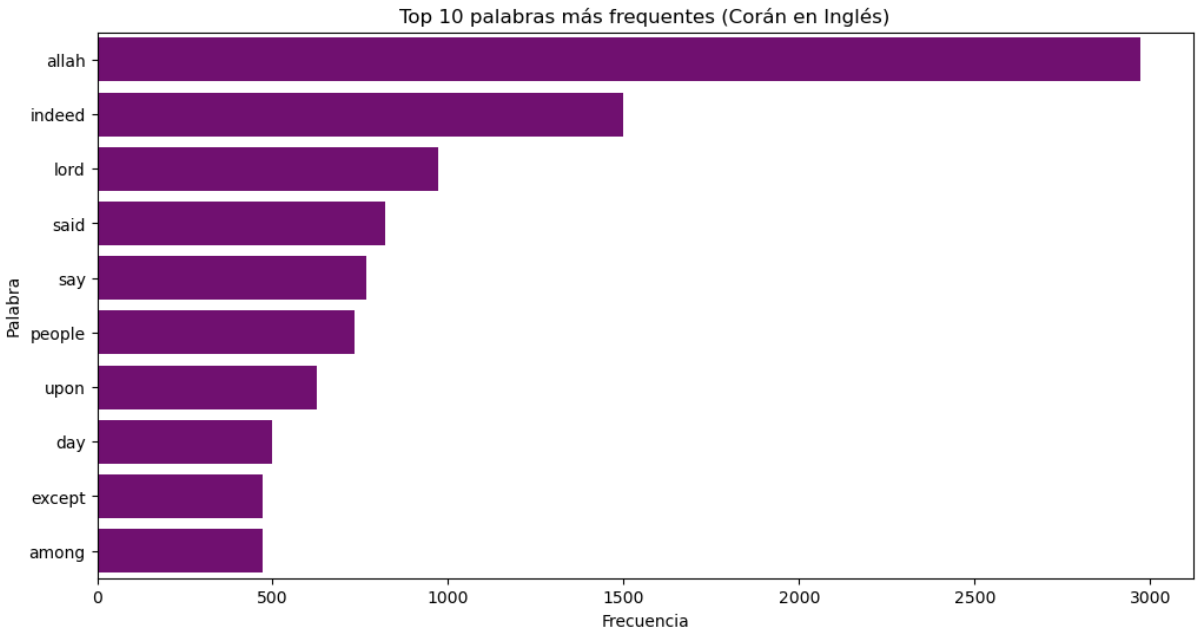

Figura 2.8: Top 10 palabras más frecuentes Corán (inglés)

2.4 Métodos de representación - TF-IDF

Para calcular TF-IDF usamos TfidfVectorizer de scikit-learn y analizamos los términos más relevantes.

2.4.1 TOP-10

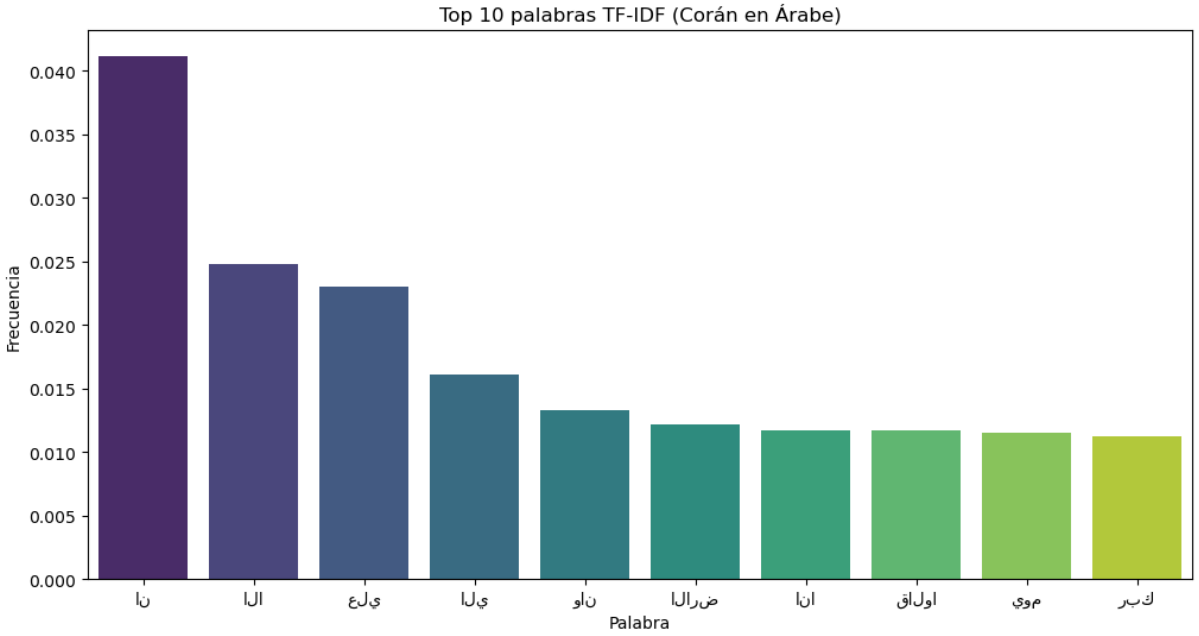

Figura 2.9: Top 10 palabras TF-IDF (Corán en Árabe)

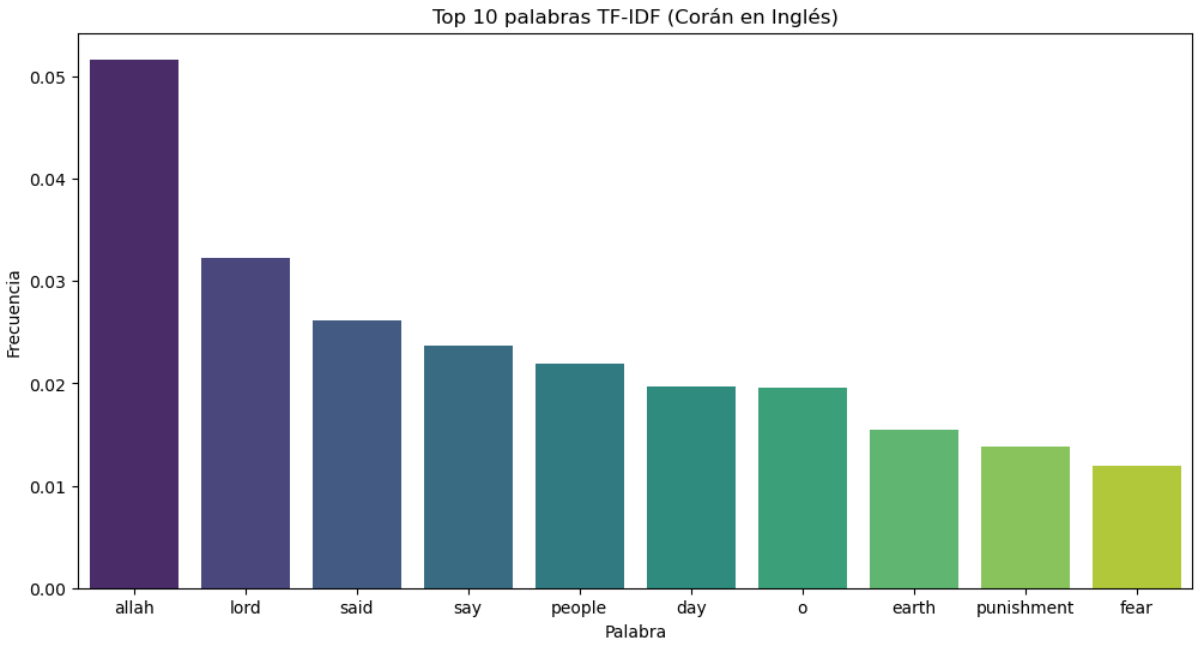

Figura 2.10: Top 10 palabras TF-IDF (Corán en Inglés)

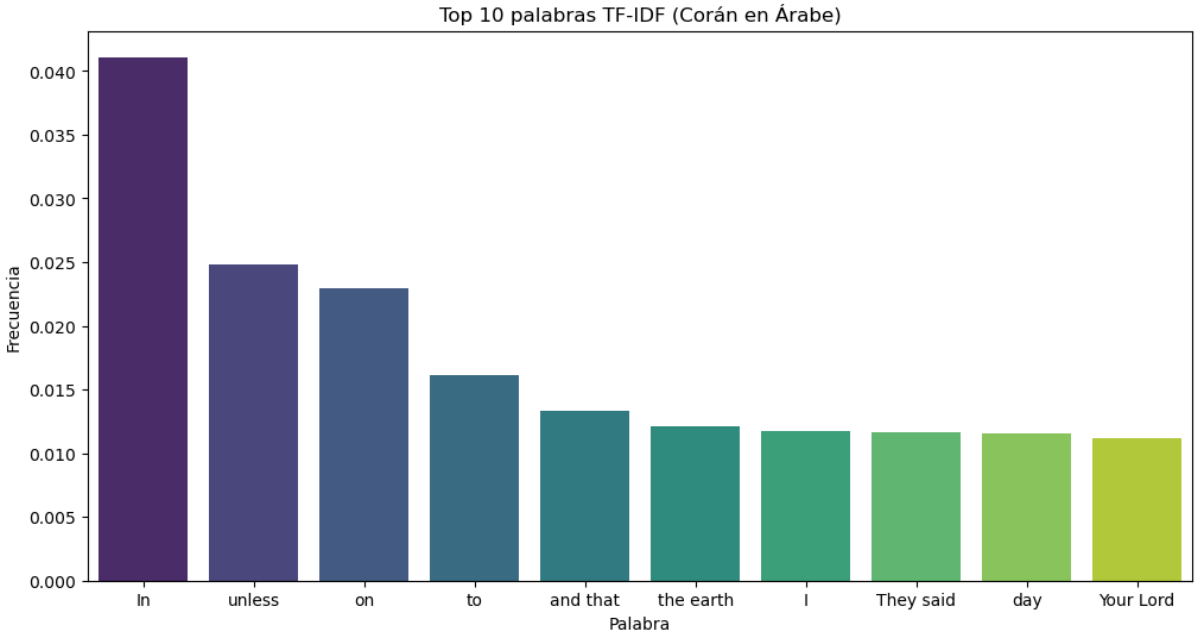

Figura 2.11: Top 10 palabras TF-IDF (Corán en Árabe)

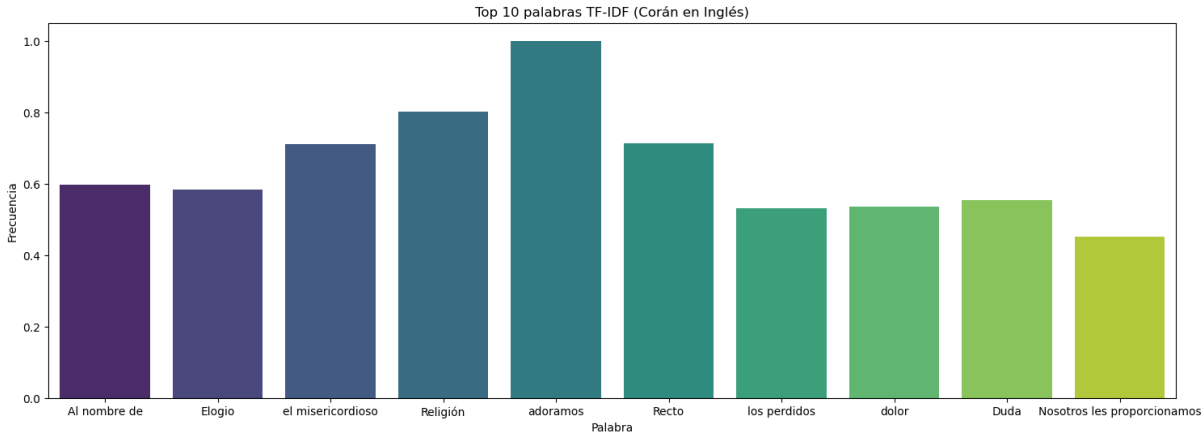

Figura 2.12: Top 10 palabras TF-IDF por documentos (Corán en Inglés)

2.5 Representación mediante Word Embeddings

Explicamos la creación de embeddings contextuales y no contextuales.

2.5.1 Embeddings contextuales - SentenceTransformers y BERT

Se emplearon embeddings basados en SentenceTransformers y BERT para tareas comparativas.

2.5.2 Embeddings no-contextuales - FastText

Los embeddings FastText se usaron para la futura tarea de búsqueda semántica.

```
df_ft["arab_embeddings"] = df_ft["text"].apply(
    lambda x: ft.get_sentence_vector(x)
)
```

2.6 Reutilización de funcionalidades a cara de futuras entregas

De cara a las siguientes entregas, hemos estructurado el proyecto para facilitar mejoras futuras. El código se encuentra organizado en distintos Jupyter Notebooks dentro de nuestro repositorio de **Github**.

Capítulo 3

Primera Iteración del Problema

3.1 Búsqueda Semántica

La búsqueda semántica fue nuestra primera idea a abordar en este proyecto para la asignatura. Para documentar esta subtarea, empleamos el mismo concepto arbitrario del *Jupyter Notebook* correspondiente: الجنة (paraíso), el cual usaremos para conseguir los pasajes más similares (siendo estos los que hablen de divinidad, cielo, ... etc) y los más disimilares (devolviendo los versículos que reciten la maldad o el infierno). Cabe recalcar que como todo el proceso anterior y futuro, vamos a trabajar con el Corán tanto en árabe como en inglés para realizar un análisis más entendible y comparable.

El primer paso para realizar el buscador sería pasar a embeddings los versículos del Libro Sagrado. Hemos empleado 2 modelos diferentes, uno siendo *Sentence-Transformer* (de tipo contextual) y el otro siendo un modelo de *Fasttext* (de tipo no-contextual). Siendo el siguiente paso comparar, en nuestro caso, la métrica de similitud del coseno del embedding de nuestro concepto original con todos los demás. Obteniendo así un valor entre [-1, 1] indicando los extremos una disimilitud/similitud total y el 0 una relación inexistente. A continuación, documentaremos la creación y comparación entre los dos modelos empleados para esta tarea:

3.1.1 Sentence-Transformer

Para construir primero el modelo de embeddings que considera el contexto de las palabras que forman las frases, vamos a usar el modelo llamado *distiluse-base-multilingual-cased-v2*. Modelo que ha sido demostrado que rinde eficientemente en varios idiomas y que vamos a utilizar para crear los embeddings contextuales árabes e ingleses. Obteniendo los siguientes resultados para los 10 embeddings contextuales más similares dado el concepto de "paraíso". El formato del dataframe siendo: línea del donde se encuentra el capítulo, número del capítulo, número del versículo, texto y el valor de similitud de coseno. (El orden del formato puede cambiar respecto al idioma).

Top 10 Sentence-Transformers:

	text	cos_similarity
0	1|1|0.471993	بسم الله الرحمن الرحيم
5475	73|1|0.427254	بسم الله الرحمن الرحيم يا ايها المزمل
7	2|1|0.420491	بسم الله الرحمن الرحيم الم
293	3|1|0.410380	بسم الله الرحمن الرحيم الم
294	3|2|0.404759	الله لا اله الا هو الحي القيوم
5377	70|3|0.404346	من الله ذي المعارج
4133	40|1|0.402066	بسم الله الرحمن الرحيم حم
1	1|2|0.399989	الحمد لله رب العالمين
5495	74|1|0.397255	بسم الله الرحمن الرحيم يا ايها المدثر
5909	85|1|0.390075	بسم الله الرحمن الرحيم والسماء ذات البروج

Figura 3.1: Los 10 capítulos árabes más similares obtenidos mediante el *SentenceTransformer*

Top 10 Sentence-Transformers:

	text	cos_similarity
6022	89|30|And enter My Paradise."	0.418759
5752	79|41|Then indeed, Paradise will be [his] refuge.	0.413461
1580	11|108|And as for those who were [destined to ...	0.411291
4046	38|77|[Allah] said, "Then get out of Paradise,...	0.382775
5177	62|1|Whatever is in the heavens and whatever i...	0.379114
3494	31|26|To Allah belongs whatever is in the heav...	0.371977
4131	39|74|And they will say, "Praise to Allah, who...	0.371575
2658	22|64|To Him belongs what is in the heavens an...	0.362967
5126	59|1|Whatever is in the heavens and whatever i...	0.361466
4523	46|14|Those are the companions of Paradise, ab...	0.357667

Figura 3.2: Los 10 capítulos ingleses más similares obtenidos mediante el *SentenceTransformer*

Como podemos apreciar, no hay ningún capítulo que se repita en ambos resultados, siendo esto debido a que el contenido semántico de los capítulos varía considerablemente entre idiomas. No obstante, podemos apreciar que el rango de valores de similitud de coseno de los 10 embeddings más similares se mantiene parecido entre idiomas, en un rango próximo a [0.35-0.47], no siendo especialmente relevante. Por otra parte, podemos apreciar que los capítulos con mayor valor de similitud devuelven pasajes donde se tratan o incluso se citan conceptos relacionados con el paraíso.

3.1.2 Fasttext

A la hora de crear los embeddings no-contextuales, nos hemos decantado por *fastText* por recomendación propia de nuestro profesorado, que al final, ha resultado ser mejor opción que el modelo explicado anteriormente. También aplicando la similitud de coseno como métrica de evaluación, hemos seguido el mismo protocolo para la creación del modelo, exceptuando diferentes hiperparámetros propios de *fastText*. Mientras que para el *SentenceTransformer* no hemos tenido que añadir ningún hiperparámetro, para los embeddings no-contextuales hemos probado diferentes combinaciones, quedándonos finalmente con la siguiente configuración:

- **model=”skipgram”**: es uno de los dos modelos que ofrece *fastText*, especialmente diseñado para predecir el contexto de la palabra deseada.
- **dim=300**: referenciando el tamaño que tendrá cada uno de los embeddings (1x300).
- **epoch=10**: hemos decidido entrenarlo por 10 epochs en total.
- **minn=3**: hace referencia al número de sub-palabras mínimas, en este caso N-gramas de 3 caracteres.
- **maxn=6**: N-gramas de 6 caracteres serán aceptados como máximo. Haciendo como total el rango de entre 3 y 6 caracteres los N-gramas que se van a aceptar.

Top 10 FastText:

	text	cos_similarity
5815	81|16|0.930583	الجوار الكنس
4986	56|8|0.888708	فاصحاب الميمنه ما اصحاب الميمنه
4987	56|9|0.887276	واصحاب المشامه ما اصحاب المشامه
3917	37|130|0.867364	سلام علي ال ياسين
3927	37|140|0.866841	اذ ابق الي الفلك المشحون
5019	56|41|0.860035	واصحاب الشمال ما اصحاب الشمال
5752	79|41|0.859282	فان الجنه هي الماوي
5750	79|39|0.852354	فان الجحيم هي الماوي
3107	26|176|0.845801	كذب اصحاب الايكه المرسلين
4886	54|41|0.840484	ولقد جاء ال فرعون النذر

Figura 3.3: Los 10 capítulos árabes más similares obtenidos mediante el *fastText*

```
Top 10 FastText:
                                                    text  cos_similarity
2683  23|11|Who will inherit al-Firdaus. They will a...        0.860665
4019  38|50|Gardens of perpetual residence, whose do...        0.827688
1323  9|89|Allah has prepared for them gardens benea...        0.826501
2617  22|23|Indeed, Allah will admit those who belie...        0.820044
549   4|57|But those who believe and do righteous de...        0.816862
1931  16|31|Gardens of perpetual residence, which th...        0.816276
428   3|136|Those - their reward is forgiveness from...        0.813250
3397  29|58|And those who have believed and done rig...        0.813130
1772  14|23|And those who believed and did righteous...        0.813026
4025  38|56|Hell, which they will [enter to] burn, a...        0.811526
```

Figura 3.4: Los 10 capítulos ingléses más similares obtenidos mediante el *fastText*

Como en el anterior caso, podemos ver que los capítulos no coinciden al cambiar de idioma, seguramente por la razón previamente mencionada. Aún así, podemos ver una clara eficacia y un mayor valor de similitud de coseno.

3.1.3 Resultados de comparación

Después de haber obtenido los resultados de los embeddings más similares habiendo introducido el concepto arbitrario de ”paraíso”, nos hemos decantado por los embeddings no-contextuales. Por una parte, por un mayor valor de similitud, obteniendo mayoritariamente un valor el doble de alto que con los embeddings contextuales, no bajando de 0.8 en el top 10. Por otra parte, hemos podido apreciar que el *SentenceTransformer* ha devuelto como pasajes más fieles aquellos donde la misma palabra de ”paraíso” o ”cielo” (sinónimo principal) se mencionan. Mientras que los embeddings no-contextuales han ido más allá, devolviendo versículos donde el concepto de ”paraíso” se transforma mediante metáforas y recursos literarios. Donde podríamos citar versículos como el número 50 del capítulo 38 donde se referencia al paraíso como un jardín de residencia perpetua.

> *”...gardens of perpetual residence whose doors will be opened to them...”* (Capítulo 38, versículo 50, línea número 4019 en nuestro archivo .txt)

Es interesante mencionar que, intuitivamente, podriamos pensar que SentenceTransformer deberia funcionar mejor en estos casos donde las metaforas y los dobles significados son tan frecuentes. Sin embargo, encontramos justo lo contrario. Por eso, la explicación rápida y simple a esto es que, además de que el Corán tiene estructuras muy repetitivas, los vectores creados por FastText alrededor de la palabra ”paradise” están bastante cerca de otros términos como:

- ”garden”
- ”eternal”
- ”righteous”
- ”mercy”
- ”reward”
- ”heaven”

Por eso, que FastText esté funcionando “mejor” en nuestro caso no contradice la teoría que los fundamenta, simplemente refleja cómo son nuestros datos (el Corán) y cómo funcionan ambos modelos en la práctica real. FastText, al basarse en medias de palabras, capta perfectamente esta repetición de vocabulario mientras que SentenceTransformer no tiene automáticamente embeddings cercanos a la palabra “paradise”, porque el modelo no sabe teológicamente que están relacionados.

3.1.4 Visualización de los resultados

En este sub-apartado vamos a buscar representar gráficamente nuestro espacio multidimensional de embeddings donde se visualizarán los embeddings relacionados con el concepto arbitrario anterior que todavía mantendremos. No obstante, realizaremos una serie de cambios para ir más allá: ahora, además de los pasajes representados más similares representados mediante embeddings, representaremos también los embeddings más disimilares. Siendo estos los que tienen el valor de similitud de coseno más negativo, intentando buscar términos opuestos a la de "paraíso", donde tendríamos que encontrar pasajes relacionados a algún cierto de cástigo o menciones al infierno. Para este caso, visualizaremos los 5 embeddings más similares y 5 más disimilares.

Visualización en 3 dimensiones

Partiendo de que el tamaño original de nuestros embeddings es de 300 dimensiones, vamos a realizar una reducción de dimensionalidad mediante el uso de *PCA*, quedándonos finalmente con 3. Mediante el uso de la librería *Plotly*, hemos creado una representación de nuestros embeddings deseados en 3 dimensiones que luce tal que así:

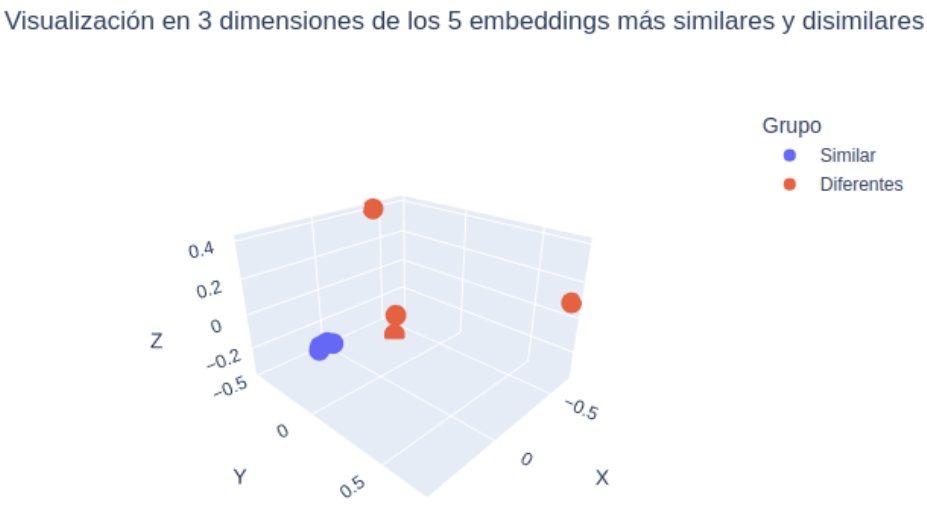

Figura 3.5: Representación de nuestros embeddings en un espacio tridimensional usando *PCA*

Aunque no se cuenta con la interactividad que ofrece *plotly* en el documento, en el *Jupyter Notebook* correspondiente se cuenta con la opción de arrastrar y configurar el espacio. No obstante, podemos ver una clara agrupación de los embeddings más fieles al concepto arbitrario en azul, pareciendo que estén uno encima del otro, mientras que los más disimilares están más alejados y dispersos tanto de los embeddings más representativos como de ellos mismos (los puntos rojos). Debemos objetar que la interactividad en el código ofrece un *tooltip* que se despliega encima del embedding representado mediante el punto ofreciendo sus coordenadas y su contenido textual.

Visualización en 2 dimensiones

También consideramos visualizar los 5 embeddings más similares y disimilares en un espacio bidimensional. A diferencia del anterior plot, usamos el reductor de dimensionalidad *UMAP* por recomendación de nuestro profesorado ya que suele funcionar mejor que el *Principal Component Analysis* para los embeddings de alta dimensionalidad. Como comentario adicional, vamos a seguir usando este segundo método para el resto del proyecto.

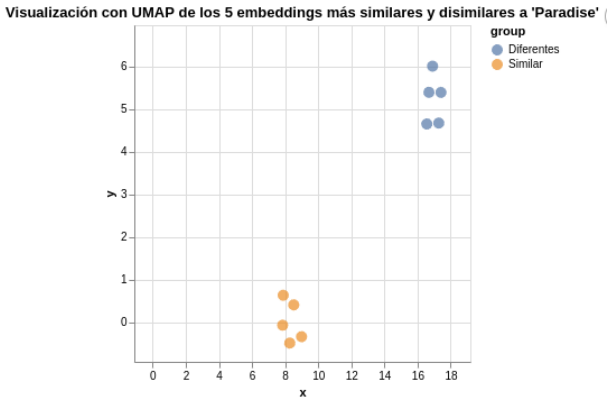

Figura 3.6: Representación de nuestros embeddings en un espacio bidimensional usando *UMAP*

Usando el mismo concepto arbitrario, podemos ver una clara agrupación de los embeddings similares, y en el caso de esta visualización en 2D, también de los embeddings más disimilares. A diferencia que en el gráfico anterior, donde sólo los similares estaban agrupados y los menos representativos se encontraban más dispersos entre sí. Esto se debe a la priorización de cada reductor de dimensionalidad; ya que *UMAP* prioriza una estructura más local, mientras que el *PCA* una más global. No obstante, se sigue manteniendo una distancia considerable entre ambos grupos en ambas visualizaciones.

3.2 Clustering de los capítulos del Corán

Con este segundo experimento hemos buscado agrupar los 114 capítulos del Corán en diferentes grupos donde el tema que los reúna sea similar entre éstos, y difiera de los otros capítulos agrupados. Para realizar esta segunda tarea a cabo, reusaremos lo construido anteriormente, siendo esto los embeddings no-contextuales obtenidos mediante el módulo *fastText*, y el reductor de dimensionalidad *UMAP*. Como algoritmo agrupador, nos hemos decantado por *HDBSCAN*. Para realizar a cabo la tarea, empezamos por cargar los modelos de embeddings y agrupando estos por capítulo. Resultando en un *dataframe* conteniendo todos los embeddings de cada capítulo juntos en una misma fila, siendo el identificador el número entero del capítulo. Obteniendo como esperábamos la dimensionalidad deseada, 114 filas y 2 columnas.

```
Dimensiones de nuestro df con los embeddings agrupados por capitulos: (114, 2)
   capitulo                                    arab_embeddings
0         1  [0.005841809, -0.016853591, -0.12730436, -0.04...
1         2  [0.011628852, 0.016536925, -0.118335046, -0.03...
2         3  [0.011735144, 0.015093205, -0.120900355, -0.03...
3         4  [0.0070681977, 0.012807678, -0.13287722, -0.03...
4         5  [0.012550958, 0.01641551, -0.1185168, -0.03757...
```

Figura 3.7: Resultado del *dataframe* con el que trabajaremos (ejemplo árabe)

Después, procederíamos a ”stackear” los embeddings y normalizarlos, y reduciendo la dimensionalidad con *UMAP* con la similitud de coseno como métrica de similitud. Por la parte de agrupación, hemos escogido que el tamaño mínimo de cada cluster sea de 3 capítulos y usaremos la distancia euclidiana para reflejar la distancia entre clusters. Una vez obtenidos los dos componentes, realizaremos las predicciones que asignen a cada capítulo el número de cluster al que considere correspondiente. Obteniendo para el ejemplo árabe 15 en total y 11 en inglés, no obstante, se asignarán al cluster con identificador ”-1” todos aquellos capítulos que no han conseguido ser agrupados para nuestro caso. Obteniendo los siguientes resultados:

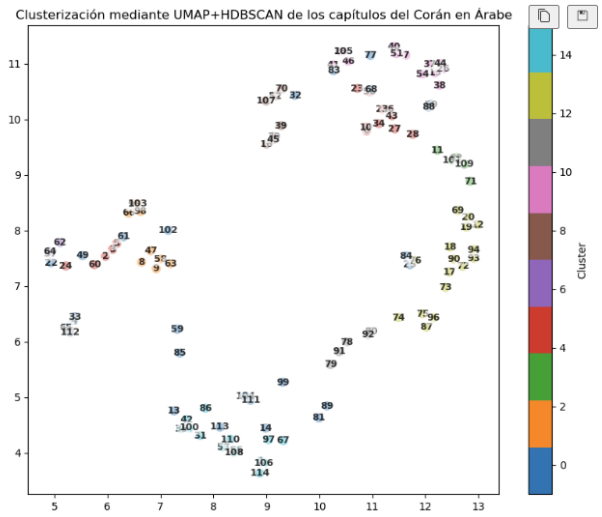

Figura 3.8: Clusterización árabe con *UMAP* y *HDBSCAN*: total de 15 clusters.

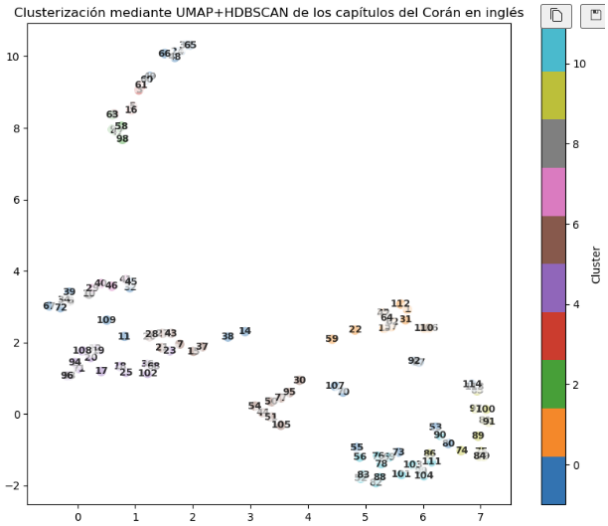

Figura 3.9: Clusterización inglés con *UMAP* y *HDBSCAN*: total de 11 clusters.

Aunque en el *Noteboook* correspondiente hayamos realizado un análisis más profundo es esencial recalcar si han habido capítulos que no hayan cambiado de cluster respecto al cambio de idioma. Ya que como podemos apreciar en los 2 plots anteriores, la distribución es totalmente diferente aunque hayamos conseguido un número total de cluster similar.

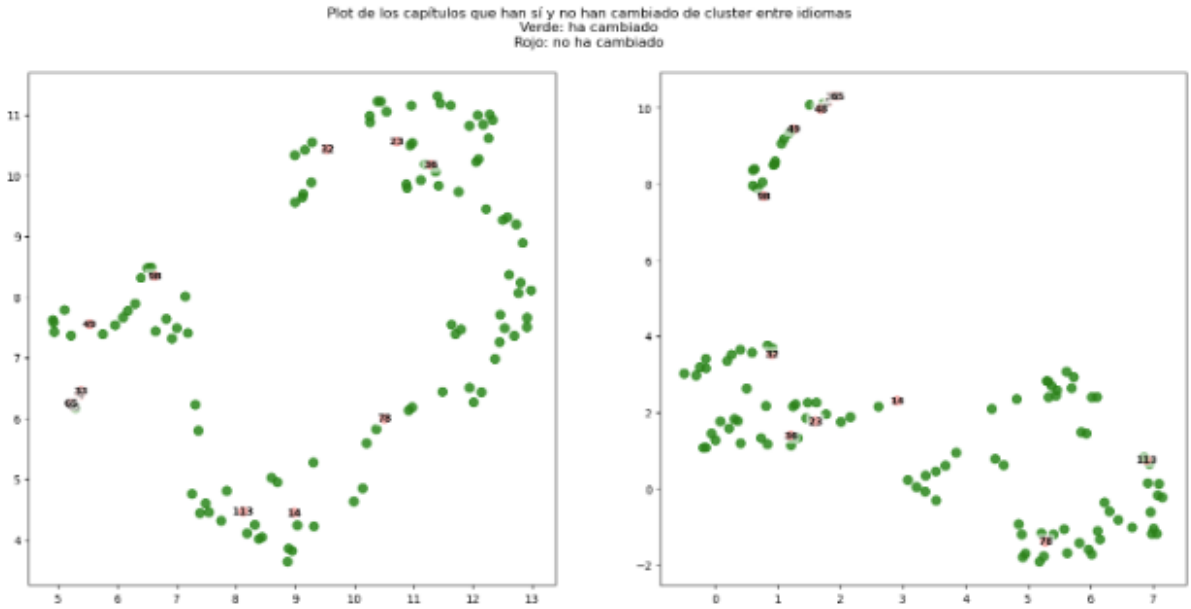

Figura 3.10: Comparación lado a lado de los capítulos que han cambiado de agrupación respecto al idioma (Verde: ha cambiado; Rojo: no ha cambiado).

Como puede observarse, únicamente 10 de los 114 capítulos no han cambiado de clúster al realizar la agrupación en distintos idiomas. Se trata de los capítulos **33, 65, 98, 113, 14, 78, 32, 23 y 36**. Esto sugiere que los embeddings de los capítulos que no han variado de cluster son probablemente muy parecidos tanto en inglés como en árabe debido a una gran coherencia semántica entre ambos idiomas. Es decir, su contenido es tan característico que mantiene la misma posición en el espacio vectorial independientemente de la lengua. Tomemos por ejemplo el capítulo 113:

113|1 *Say, I seek refuge in the Lord of daybreak.*
113|2 *From the evil of what He has created.*
113|3 *And from the evil of the darkness when it settles.*
113|4 *And from the evil of those who blow on knots.*
113|5 *And from the evil of an envier when he envies.*

Donde se trata un tema muy concreto: la protección contra la maldad y la envidia. Capítulo que contiene términos como ”Lord of daybreak” que son muy distintivos. Además, los capítulos que han cambiado de cluster respecto al idioma comparten un espacio de embeddings distinto, es decir, los embeddings árabes e ingleses no están alineados entre sí. Resultando de esta manera en un cambio de distribución semántica, alterando como hemos podido ver en la visualización comparativa de los espacios de clusters (Fig. 3.8 y Fig. 3.9), una diferencia de posiciones clara y evidente.

3.3 Generador de Topics

En esta sección, teníamos como objetivo crear unos ”topics” o ”títulos” representativos de los diferentes capítulos del Corán. Como los versículos de cada capitulo son bastante cortos y muy limitados en contenido, decidimos seguir con BERTopic. Aprovecharemos el modelo de embeddings fasttext y el modelo UMAP. También aprovechamos el agrupamiento que hicimos previamente en *HDBSCAN* para aumentar el tamaño de los documentos que se analizarán en BERTopic y así poder crear un título representativo por cluster.

BERTopic es muy interesante para esta tarea, ya que nos ofrece una estructura y metodología organizada y directa para poder abordar esta propuesta. Aquí abajo dejo una imagen que ilustra los pasos que hemos tomado para realizar correctamente BERTopic.

Figura 3.11: Este esquema detalla específicamente los pasos tomados en BERTopic

3.3.1 Primeras representaciones

Para empezar, agrupamos los versículos en capítulos y los stackeamos en una matriz de tamaño (114 capítulos, 300 dimensiones). Seguido, creamos una lista de palabras demasiado frecuentes en los textos sagrados para no sesgar los resultados.

- 'الله', Allah
- 'محمد', Muhammad
- 'رب', Lord
- 'الله', God (igual que Allah)

- 'قرآن', Quran
- 'اسلام', Islam (sin hamza para evitar duplicados)
- 'الاسلام', Islam (con artículo)
- 'ربكم', your Lord
- 'ربه', his Lord
- 'ربهم', their Lord
- 'يارب', oh Lord

Después, hacemos un CountVectorizer para obtener la importancia de las palabras y un ClassTfidfVectorizer para eliminar las stopwords y reducir las palabras más frecuentes. Aquí usamos bm-25. Finalmente, juntamos todo esto con UMAP, HDBSCAN y fasttext como modelo y hacemos unos plots para ver que palabras hemos obtenido por cada topic.

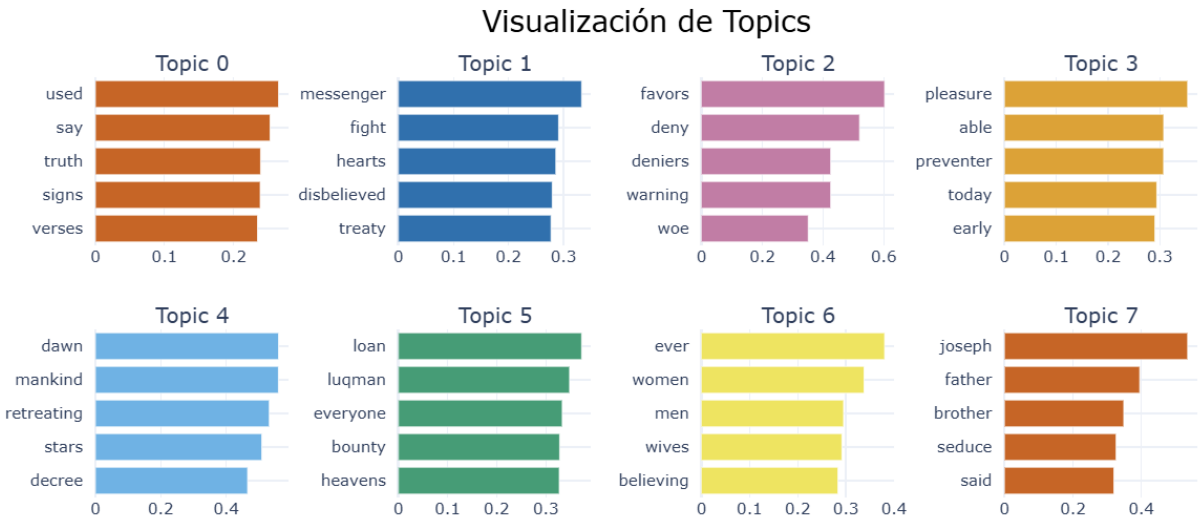

Figura 3.12: Visualización de palabras más representativas por topic

3.3.2 Rerankers

Para mejorar la representación de los topic con palabras más variadas y fieles al topic hemos aplicado un par de rerankers. Entre los rerankers que usamos contamos con KeyBERTInspired y con MaximalMarginalRelevance.

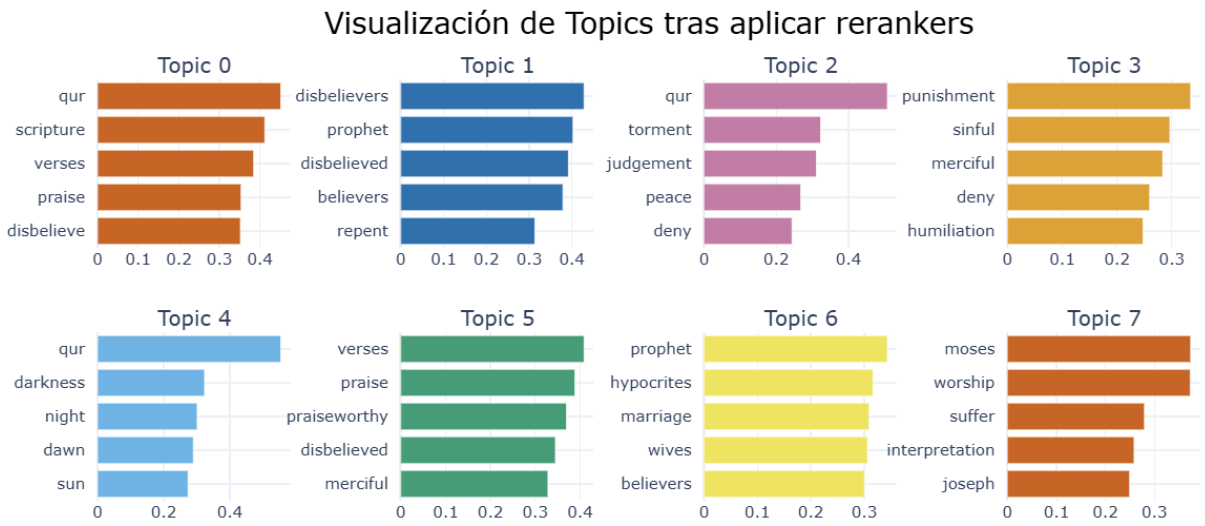

Figura 3.13: Visualización de palabras más representativas por topic con rerankers

3.3.3 Generador de títulos

Para la generación de títulos usamos el modelo *flan-t5-base* de google. Para hacerlo funcionar correctamente, hemos hecho un prompt few-shot.

```
Topic 0:
  Original: -1_made_created_merciful_earth
  Reranker: -1_qur_worship_deity_solomon
  Nuevo: -1_allah___

Topic 1:
  Original: 0_used_say_truth_signs
  Reranker: 0_qur_scripture_verses_praise
  Nuevo: 0_Qur an___

Topic 2:
  Original: 1_messenger_fight_hearts_disbelieved
  Reranker: 1_disbelievers_prophet_disbelieved_believers
  Nuevo: 1_allah___

Topic 3:
  Original: 2_favors_deny_deniers_warning
  Reranker: 2_qur_torment_judgement_peace
  Nuevo: 2_zaqqum___

Topic 4:
  Original: 3_pleasure_able_preventer_today
  Reranker: 3_punishment_sinful_merciful_deny
  Nuevo: 3_punishment___
```

Figura 3.14: Topics creados por *flan-t5-base*

Por eso hemos decidido usar un modelo en local de ollama, más específicamente *gemma3:4b*. Los resultados fueron los siguientes para los topics en ingles:

- 1. Qur'an Worship of the Deity
- 2. Qur'an Verses of Praise and Belief
- 3. Disbelievers' Disbelief and Believers' Faith
- 4. Qur'an Torment, Judgement, and Divine Peace
- 5. Qur'an Darkness, Night, and Dawn's Arrival
- 6. Verses of Praise for the Praiseworthy
- 7. Prophet, Hypocrites, and Marriages in Faith
- 8. Moses' Worship and Suffering Interpretation

- 9. Praiseworthy Scriptures of Divine Creation
- 10. Angels, Worshippers, and Moses' Worship
- 11. Disbeliever, Sinner, and Purification to Paradise
- 12. Believers, Prophets, and Christian Scriptures
- 13. Qur'an Prayer, Pharaoh, and Satan's Ways
- 14. Resurrection, Qur'an, and Soul's Reality
- 15. Moses, Pharaoh, and Believers' Worship

Como tarea final, hemos querido comparar la similitud de estos títulos generados tanto para los topics en árabe como para los ingleses mediante fasttext. Lo que obtenemos como respuesta es algo no muy concluyente, solo que algunos títulos son tan generales que acaparan la mayoría de topics y por eso se relacionan con muchos de ellos, como podremos ver aquí:

	ID topic ingles	Topic ingles	ID mejor match arabe	Mejor match traducido arabe	Similarity Score
1	1	Qur'an Verses of Praise and Belief	7	Infidelity and the evils of disbelievers	0.956880
5	5	Verses of Praise for the Praiseworthy	16	Forgive criminals in the sleeves	0.920711
10	10	Disbeliever, Sinner, and Purification to Paradise	7	Infidelity and the evils of disbelievers	0.915299
8	8	Praiseworthy Scriptures of Divine Creation	0	Creation of universe with fire	0.907977
0	0	Qur'an Worship of the Deity	11	Remembrance of God and the kingdom of ants	0.903431
6	6	Prophet, Hypocrites, and Marriages in Faith	5	Infidels and their salvation in paradise	0.899342
7	7	Moses' Worship and Suffering Interpretation	14	Angels and their praise of Muslims	0.890297
12	12	Qur'an Prayer, Pharaoh, and Satan's Ways	11	Remembrance of God and the kingdom of ants	0.888657
4	4	Qur'an Darkness, Night, and Dawn's Arrival	4	Royal power and divine judgment	0.882766
13	13	Resurrection, Qur'an, and Soul's Reality	4	Royal power and divine judgment	0.867560
14	14	Moses, Pharaoh, and Believers' Worship	7	Infidelity and the evils of disbelievers	0.856236
3	3	Qur'an Torment, Judgement, and Divine Peace	4	Royal power and divine judgment	0.853879
2	2	Disbelievers' Disbelief and Believers' Faith	7	Infidelity and the evils of disbelievers	0.852073
11	11	Believers, Prophets, and Christian Scriptures	7	Infidelity and the evils of disbelievers	0.849122
9	9	Angels, Worshippers, and Moses' Worship	14	Angels and their praise of Muslims	0.807861

Figura 3.15: Cosine similarity de los títulos generados por topic

Capítulo 4

Entrega final - RNN y Transformers

4.1 RNNs

Top 10 Sentence-Transformers:

	text	cos_similarity
0	1|1|0.471993	بسم الله الرحمن الرحيم
5475	73|1|0.427254	بسم الله الرحمن الرحيم يا ايها المزمل
7	2|1|0.420491	بسم الله الرحمن الرحيم الم
293	3|1|0.410380	بسم الله الرحمن الرحيم الم
294	3|2|0.404759	الله لا اله الا هو الحي القيوم
5377	70|3|0.404346	من الله ذي المعارج
4133	40|1|0.402066	بسم الله الرحمن الرحيم حم
1	1|2|0.399989	الحمد لله رب العالمين
5495	74|1|0.397255	بسم الله الرحمن الرحيم يا ايها المدثر
5909	85|1|0.390075	بسم الله الرحمن الرحيم والسماء ذات البروج

Figura 4.1: Los 10 capítulos árabes más similares obtenidos mediante el *SentenceTransformer*

4.2 Transformers

Bibliografía

[Abd+14] Heba Abdelnasser **andothers**. «Al-Bayan: An Arabic question answering system for the Holy Quran». **in***Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing*: Association for Computational Linguistics, 2014, **pages** 57–64.

[Obe+20] Ossama Obeid **andothers**. «CAMeL tools: An open source python toolkit for Arabic natural language processing». **in***Proceedings of the twelfth language resources and evaluation conference*: 2020, **pages** 7022–7032.

[AAA21] Menwa Alshammeri, Eric Atwell **and** Mhd ammar Alsalka. «Detecting semantic-based similarity between verses of the quran with doc2vec». **in***Procedia Computer Science*: 189 (2021), **pages** 351–358.

[Bas+23] Muhammad Huzaifa Bashir **andothers**. «Arabic natural language processing for Qur'anic research: a systematic review». **in***Artificial Intelligence Review*: 56.7 (2023), **pages** 6801–6854.

[SSA23] Yasser Shohoud, Maged Shoman **and** Sarah Abdelazim. «Quranic Conversations: Developing a Semantic Search tool for the Quran using Arabic NLP Techniques». **in***arXiv preprint arXiv:2311.05120*: (2023).

[Alq24] Mohammed A Alqarni. «Embedding search for quranic texts based on large language models.» **in***Int. Arab J. Inf. Technol.*: 21.2 (2024), **pages** 243–256.

[LD24] Samira Lagrini **and** Amina Debbah. «A New Semantic Search Approach For The Holy Quran Based On Discourse Analysis And Advanced Word Representation Models». **in***International Journal of Computing and Digital Systems*: 17.1 (2024), **pages** 1–14.

[Mah+24] Ali Mahboub **andothers**. «Evaluation of Semantic Search and its Role in Retrieval-Augmented-Generation (RAG) for Arabic Language». **in***arXiv preprint arXiv:2403.18350*: (2024). Preprint.

[Zhe+24] Chujie Zheng **andothers**. «Semantic Search Evaluation». **in***arXiv preprint arXiv:2410.21549*: (2024). Preprint.