SISTEMA DE PREGUNTAS Y REPUESTAS

Juan Pablo Herrera Musi
Maestría en Ciencia de Datos
ITAM, Ciudad de México, México
jphmusi@gmail.com

Daniela Pinto Veizaga
Maestría en Ciencia de Datos
ITAM, Ciudad de México, México
dpintove@itam.mx

20 de mayo de 2020

ABSTRACT

Motivados por la masiva cantidad de información disponible y la finita capacidad humana para procesar dicha información, en el presente trabajo empleamos métodos de procesamiento de lenguaje natural y aprendizaje profundo para desarrollar un sistema de preguntas y respuestas que, dado una pregunta, sea capaz de predecir la posición de la oración (dentro de un corpus o texto) que contiene el fragmento de la respuesta. Para la implementación del sistema, entrenamos cinco modelos (dos supervisados y tres no supervisados). Los resultados, en términos de accuracy, oscilan entre 0.30 y 0.60. Todos los modelos fueron entrenados con el apoyo de la base de datos *Standford Question Answering Data Set (SQuAD)*.

***P*alabras *C*laves:** información · sistema de preguntas y respuestas · comprehensión de lectura · SQuAD

1 Introducción

La Era de la información se caracteriza, entre otras cosas, por la masiva cantidad de información generada cotidianamente; de hecho, Reinsel et al. (2018), en su artículo *The Digitization of the World From Edge to Core*, estiman que, para el año 2025, la generación de datos e información crecerá a 175 Zettabytes . En este contexto, con recursos temporales finitos, tareas como leer y comprehender todo lo que se produce diaramente, resultan imposibles. Por ello, la demanda creciente por contar con herramientas computacionales que faciliten la obtención inmediata de información *relevante* y permitan la resolución de temas sociales, económicos, entre otros.[1]

Los sistemas de preguntas y respuestas *(QA-systems, por sus siglas en inglés)* son aproximaciones computacionales que intentan abordar la problemática antes mencionada mediante la desagregación del problema en dos partes esenciales: *i.* recolección de información y *ii.* comprehensión de lectura. Para efectos del presente trabajo, desarrollamos un sistema de preguntas y respuestas, basado primordialmente en dos tipos de modelos (uno supervisado y otro no supervisado) y entrenado con la base de datos *Standford Question Answering Data Set* (SQuAD), en su versión 1.0, con el fin de encontrar la(s) oracion(es) que contengan la(s) respuesta(s) a su(s) respectivo(s) par(es) de pregunta(s).

El presente trabajo[2] se estructura de la siguiente manera: en la sección 2 revisamos la literatura relacionada con los principales conceptos y avances en el ámbito de los *QA-systems*; luego, en la sección 3, introducimos y describimos las principales características de la base de datos empleada para el entrenamiento de nuestro modelo; posteriormente, en la sección 4 presentamos los modelos desarrollados, junto con sus componentes principales; en seguida, en la sección 5 exponemos los principales resultados; finalmente, en la sección 6 reflexionamos sobre nuestros principales hallazgos y resultados. Para mayor información relacionada con el código de nuestra implementación, remitirse al github asociado al presente proyecto: Question Answer_System.

[1] Prueba de ello, la reciente competencia de Kaggle denominada COVID-19 Open Research Dataset Challenge (CORD-19), donde piden a los concursantes, entre otras cosas resolver desarrollar herramientas de texto y minería que permitan a la comunidad médica obtener respuestas a preguntas altamente prioritarias en el contexto del novel COVID-19.

[2] Este proyecto se basa principalmente en la implementación desarrollada por Alvira Aswalin, disponible en su repositorio público: SQUAD.

2 Revisión de literatura

La comprehensión de lectura –entendida como la habilidad para leer un texto y responder una pregunta sobre lo leído– es una tarea desafiante para los humanos y, más aún, para las máquinas; es una tarea altamente demandante en el nivel de entendimiento del lenguaje natural y el funcionamiento del mundo. Para ejemplificar esta complejidad, Rajpurkar et al. (2016) indican que, por ejemplo, para responder, *¿what causes precipitation to fall?*, uno tendría que incurrir en un proceso mental del siguiente tipo: primero, localizar la parte relevante del pasaje *"precipitation ... falls under gravity*; luego, razonar que, en este contexto, *under* se refiere a una causa, no así a localización; y finalmente, determinar que la respuesta correcta es *gravity*. Definitivamente, ¡nada trivial!

Durante décadas, el campo de la inteligencia artificial ha intentado progresar en este sentido; sin embargo, los avances fueron casi nulos, debido, principalmente, a los escasos y deficientes insumos de entrenamiento: *datasets* de tamaños reducidos y con construcciones (aproximaciones) distantes de las características esenciales de la comprehensión de lectura. A pesar de lo anterior, investigadores y estudiosos del tema han visto en los *QA systems* una herramienta prometedora para proveer respuestas *directas* a preguntas diversas.

De esta manera, de acuerdo con la literatura relacionada al tema, dos son las grandes vertientes de *QA-systems*: los sistemas con dominio abierto y los sistemas con dominio cerrado; los primeros, ambiciosos desde su construcción, pretenden responder casi cualquier pregunta, con base en un gran bagaje de conocimiento previo; y, los segundos, abordan y responden preguntas acotadas a un dominio en específico. Además, según Mervin (2013) pueden ser clasificados dependiendo de cuál es su insumo principal: *i.* texto, *ii.* hechos, *iii.* la web, entre otros. Así, dependiendo de la complejidad de los insumos y el tipo de preguntas que intentan de responder (factoid questions, cloze questions, option questions, yes/no questions, what, why, when, who questions y choice questions.), las implementaciones y propuestas de resolución de estas tareas han sido diversas.

Además, múltiples investigadores han emprendido bastos proyectos para mejorar los insumos y colecciones de datos que faciliten el entrenamiento de diferentes QA-systems; entre algunas de estas iniciativas tenemos: Freebase (Bollacker et al. (2008)) o DBPedia (Auer et al. (2007)), otras como CNN/Daily Mail (Hermann et al. (2015)) y *The Children's Book Test(CBT)* (Hill et al. (2015)), por nombrar algunas. Una de las iniciativas más recientes, y con la cuál trabajamos en el presente trabajo, es la implementada por Rajpurkaret et al. (2016) y Rajpurkar et al. (2018), quienes presentaron el *Standford Question Answering Data Set* (SQuAD), en sus versiones 1 y 2, con más de $100,000$ pares de preguntas-respuestas. En particular, estas dos bases de datos constituyeron un hito para el avance en la construcción de los sistemas de preguntas y respuestas puesto que, a diferencia de los *datesets* anteriores, las respuestas a cada pregunta son segmentos de texto.

3 *Dataset: Standford Question Answering Data Set*

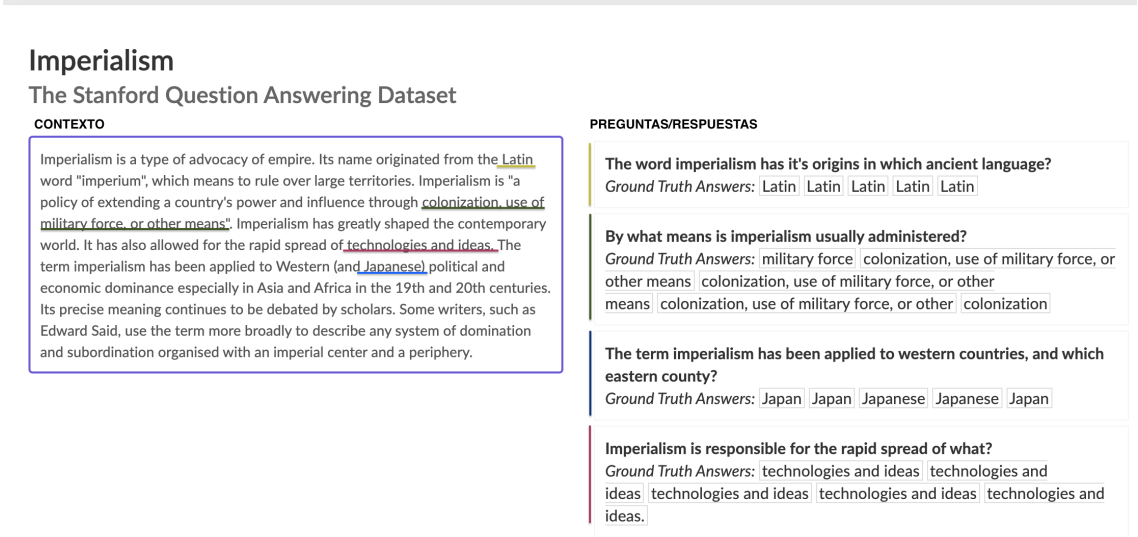


Figura 1: Estructura del SQUAD, v1

Como se mencionó en la sección anterior, para efectos del presente trabajo, trabajamos con el *Standford Question Answering Data Set* (SQUAD), en su versión 1.0, recabada del repositorio público de Swalin (2018). La base de datos en mención fue curada de la siguiente manera: con la ayuda del proyecto Nayuji's Wilkipedia, se extrajeron los 10,000 artículos con el pagerank más alto y, de este subconjunto, se muestrearon aleatoriamente 536 artículos. Cada artículo, a su vez, se desagregó en sus respectivos párrafos, siempre y cuando la longitud de los mismos fuera mayor a 500 caracteres. El resultado fue una base de datos con $23,215$ párrafos y $107,785$ pares de preguntas y respuestas, cuya particularidad es que, como se muestra en la *Figura 1*, la respuesta de cada pregunta es un segmento de texto del correspondiente a un pasaje de lectura.

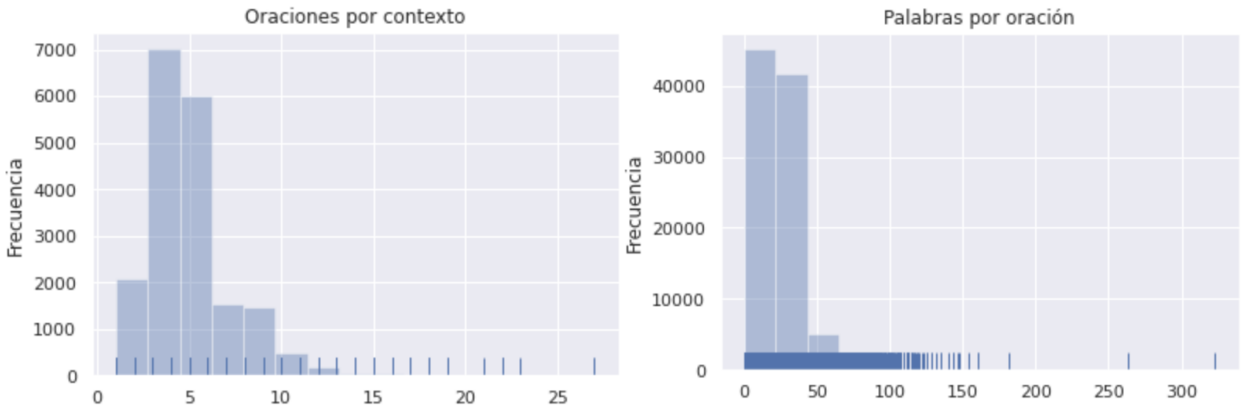
Una característica importante de esta base de datos, y que nos permite comprender de mejor manera la naturaleza del problema que pretendemos entender, es que 33 por ciento de las respuestas, contenidas en el *dataset SQuAD* v1.0, son frases sustantivas comunes; para mayor información, remitirise a la Figura 2.

Answer type	Percentage	Example
Date	8.9%	19 October 1512
Other Numeric	10.9%	12
Person	12.9%	Thomas Coke
Location	4.4%	Germany
Other Entity	15.3%	ABC Sports
Common Noun Phrase	31.8%	property damage
Adjective Phrase	3.9%	second-largest
Verb Phrase	5.5%	returned to Earth
Clause	3.7%	to avoid trivialization
Other	2.7%	quietly

Figura 2: Tipo de respuestas. Imagen recabada de Rajpurkaret et al. (2016)

En particular, la base de datos que empleamos en el presente trabajo cuenta con $87,636$ pares de preguntas/respuestas en el set de entrenamiento y $33,615$ pares en el set de valicación. Tanto el conjunto de entrenamiento, como el conjunto de validación cuenta con las siguientes seis columnas, dispuestas en formato anidado: text (respuesta a la pregunta), title, context (escrito con el contexto de la pregunta y la respuesta), question (pregunta), id y answer_start (información de la posición en la cuál comienza la respuesta).

Para caracterizar de mejor manera al conjunto de datos de entrenamiento, realizamos algunos conteos simples que nos arrojaron los siguientes datos: tenemos un total de $18,891$ contextos, los cuáles a su vez facilitan la obtención de $87,636$ pares de preguntas y respuestas; para complementar este corto análisis, empleamos la paquetería *text blob* para extraer las oraciones del campo "context" y contar el número de palabras y oraciones por contexto.



(a) Oraciones por contexto

(b) Palabras por oración

Figura 3: Análisis Exploratorio de los Datos

De la figura 3a identificamos que en promedio, existen 4.95 oraciones por contexto, con una desviación estándar de 2.32. Es decir, son contextos relativamente pequeños, puesto que 99 por ciento, tiene 12 o menos oraciones. Asimismo, conforme con la figura 3b, en promedio cada oración está compuesta por 23.95 palabras, con una desviación estándar de 12.13. En el extremo, existe al menos una oración con 323 palabras.

4 Descripción de los modelos propuestos

De manera general, desarrollamos dos métodos para abordar el problema: uno no supervisado, dónde se hacen predicciones sin entrenamiento previo ni variable *target*; otro supervisado, donde se usan las variables generadas (incluyendo una variable *target* para entrenar un modelo y hacer predicciones. La variable *target* fue generada a partir de la indexación de las oraciones de cada contexto, y la predicción del modelo corresponde a la posición de la oración que contiene la respuesta dentro del contexto. En el caso del método no supervisado, la variable *target* es empleada únicamente para medir el desempeño del modelo.

4.1 Preprocesamiento y transformaciones

Previo a la implementación de ambos métodos, realizamos una serie de transformaciones y preprocesamiento: embedding de palabras, embedding de oraciones, cálculos de distancias cosenos y euclidianas, creación de una variable target, análisis de dependencia de las palabras y transformaciones de tipo one hot encoding.

4.1.1 Embedding de palabras

Primero, a manera de preprocesamiento y exploración, creamos una representación matricial de nuestras palabras, dado el contexto en el que aparecen en nuestro corpus. Para ello, primero construímos nuestro corpus, *tokenizando*, primero las oraciones y luego las palabras que conforman el "context" en nuestro data set; luego entrenamos dos modelos *word2Vec* con la paquetería *gensim*, uno con la arquitectura Continuos Bag of Words(CBOW) y con Skip-gram a fin de mapear cada palabra a un vector que capture su significado. Con el primero, se asigna la representación de la palabra actual usando una "ventana" de palabras alrededor del target, es decir, se predijo la palabra actual, dado un contexto; con el segundo, se asigna, a partir de la palabra actual, la "ventana" de palabras contexto alrededor. Para mayor detalle, remítase a la Figura 4.

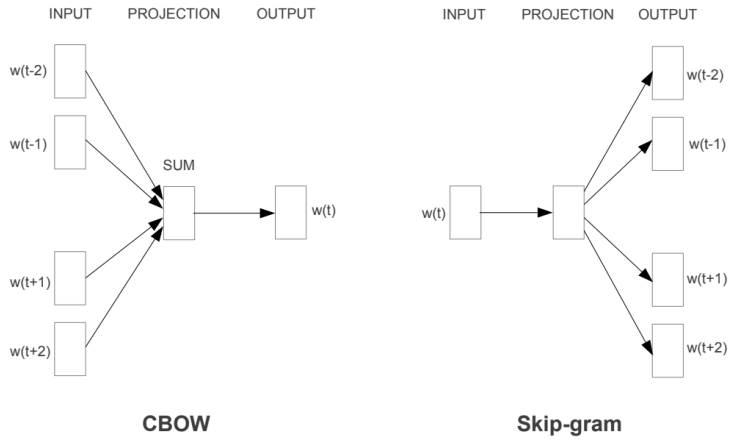


Figura 4: Comparación entre CBOW y Skipgram. Imágen tomada de Mikolov et al. (2013)

A pesar de que ambos modelos fueron entrenados con el mismo corpus y los mismos parámetros –vectores de palabras con dimensión 100 y una ventana de 5 palabras–, las similitudes coseno computadas entre la media de los vectores peso de una determinada palabra y los vectores de cada palabra del corpus fueron muy distintas. Por ejemplo, de acuerdo con la Tabla 1, las cinco palabras más similares a la palabra "building" varían, según se trate de uno modelo u otro.

Tabla 1: Similitudes coseno: Top cinco palabras similares a la palabra *building*.

Word2Vec			
Skipgram		CBOW	
Palabra	Similitud	Palabra	Similitud
construction	0.820	centre	0.855
constructing	0.799	campus	0.820
constructed	0.794	station	0.812
buildings	0.792	site	0.809
housed	0.790	shopping	0.804

4.1.2 Embedding de oraciones

Luego, intentamos adoptar un enfoque distinto al previo, mapeando oraciones completas en un espacio vectorial. Lo anterior, con el fin de retener algunas propiedades interesantes, como los *features* de los embeddings de las palabras que conforman cada oración. En específico, para esta tarea, empleamos *InferSent*[3], un embedding de oraciones que fue entrenado con el dataset *Standford Natural Language Inference* para resolver tareas de inferencia del siguiente tipo:

$$premisa \implies hipótesis \tag{1}$$

Donde se busca encontrar la relación direccional entre una premisa (oración 1) y una hipótesis (oración 2).

Para ello, primero *tokenizamos* cada oración dentro del conjunto de entrenamiento y el conjunto de validación, obteniendo dos listas de oraciones de tamaños: $93,576$, para el conjunto de entrenamiento; y $10,458$ tokens, para el conjunto de validación. Luego construimos el vocabulario de vectores de palabras para los datos de entrenamiento y usamos el mismo para los datos de validación. Finalmente, obtenemos cuatro representaciones matriciales de las oraciones contenidas en los siguientes corpus: i) los datos de entrenamiento; ii) las preguntas de los datos de entrenamiento; iii) los datos de validación; iv) las preguntas en los datos de validación. La salida de cada una de estás implementaciones es una matriz con n vectores (equivalentes al número de oraciones contenidas en cada uno de los corpus antes mencionados), cada vector con una dimensión de tamaño $4,096$.

Con estas representaciones semánticas de las oraciones, a manera de exploración, calculamos las distancias (euclidiana y coseno) entre cada una de las oraciones/preguntas y las oraciones contenidas en el contexto.

Contexto:

[0] Architecturally, the school has a Catholic Character. **[1]** Atop the main building's gold dome is a golden statue of the Virgin Mary. **[2]** Immediately in front of the main building and facing it, is a copper statue of Christ with arms upraised with the legend "venite ad me omnes". **[3]** Next to the main building is the Basilica of the Sacred Heart. **[4]** Immediately behind the Basilica is the Grotto, a Marian place of prayer and reflection. **[5]** It is a replica of the Grotto at Lourdes, France where the Virgin Mary reputedly appeared **to Saint Bernadette Soubirous in 1858.** **[6]** At the end of the main drive (and in a direct line that connects through 3 statues and the gold dome), is a simple, modern stone statue of Mary. **[7]** As at most other universities, Notre Dame's students run a number of news media outlets. **[8]** The nine students run outlets include three newspapers, both a radio and television station, and several magazines and journals. **[9]** Begun as a one-page journal in September 1876, the scholastic magazine is issued twice monthly and claims to be the oldest continuous collegiate publication in the United States.

Pregunta:

¿To whom did Virgin Mary allegedly appear in 1858 in Lourdes France?

Respuesta:

To Saint Bernadette Soubirous in 1858.

Distancias Coseno:

Oración	Distancia
0	0.515
1	0.560
2	0.491
3	0.539
4	0.511
5	0.388
6	0.515
7	0.506
8	0.525
9	0.382

Figura 5: Distancias coseno entre oraciones que forman parte del contexto y la oración/pregunta. Elaboración propia.

De la figura 5, observamos que, en efecto, tras generar el encoder de las oraciones en nuestros dos corpus: contexto y preguntas, la distancia entre la oración/pregunta y la oración que forma parte del contexto y contiene la respuesta, es una de las distancias más pequeñas: 0.388.

Finalmente, con la generación de nuestras representaciones semánticas de las oraciones, también podemos visualizar la importancia las palabras, dada una oración. Por ejemplo, para la oración: *many of his observations were included as improvements in the 1981 lds edition of the book of mormon*, la importancia de las palabras es la mostrada en la Figura 6, dónde se establece que dada esa oración, la palabra más importante es *mormón*. Además notamos que las palabras con menos importancia corresponden a artículos y preposiciones, que sirven para unir otras palabras.

[3]InferSent aprovecha las incrustaciones de palabras existentes, Glove y fastText, para construir incrustaciones de oraciones.

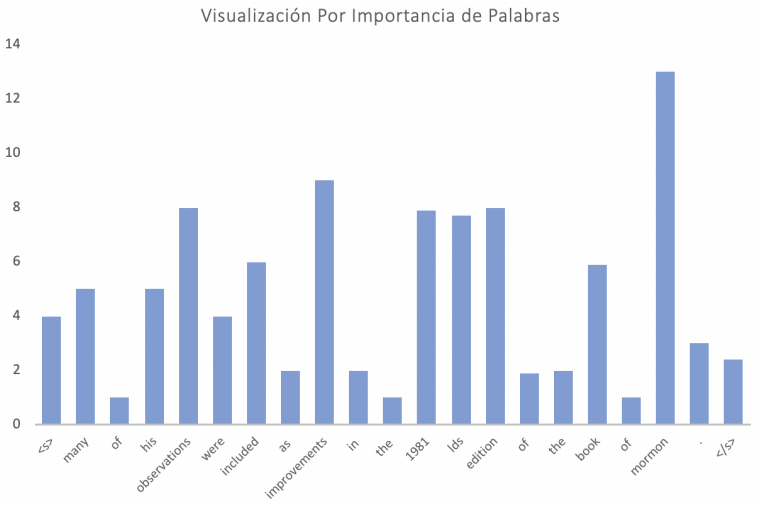


Figura 6: Importancia de las palabras, dada la oración *"many of his observations were included as improvements in the 1981 lds edition of the book of mormon"*. Gráfica generadada con la ayuda de Infersent.

4.1.3 Variable *target*

Una vez generados nuestros cuatro *encoders*, ordenamos los datos en forma de listas anidadas: i) lista anidada con los *embeddings* de las oraciones del contexto; ii) lista con las oraciones tokenizadas del contexto; iii) lista con los *embeddings* de las preguntas.

Con estas listas, identificamos la oración, dentro de cada contexto, en la cuál se encuentra la respuesta literal a cada pregunta; luego, la indexamos, de acuerdo con su posición en el contexto. Por ejemplo, en el ejemplo descrito en la Figura 5, el valor de la variable target, dada la pregunta, sería 5. De esta manera, en principio, tenemos tantas etiquetas a predecir como número de oraciones dentro de un contexto. Se encontrarón 981 preguntas para las que su respuesta no estaba contenida literalemnte dentro de alguna de las oraciones del contexto, estos casos se omitieron en el análisis por estar fuera de los dominios del problema definido.

Observando la Figura 7, identificamos que en realidad, hay únicamente 20 etiquetas; es decir, las respuestas a las distintas preguntas son normalmente respondidas dentro de las primeras 20 oraciones –a pesar de que en realidad hay contextos con más de 20 oraciones. Aún más relevante, notamos que la mayoría de las respuestas son respondidas dentro de las primeras 8 oraciones del contexto.

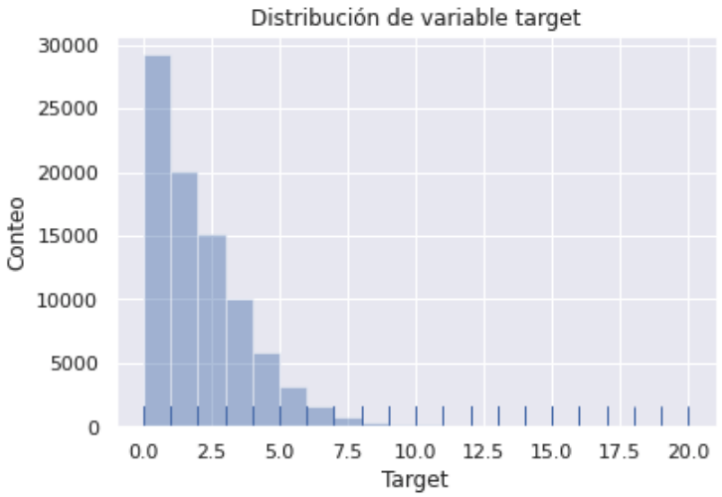


Figura 7: Distribución de la variable target en los distintos contextos del set de entrenamiento. Elaboración propia.

Considerando estos hallazgos, junto con lo comentado a propósito de la Figura **??** –en específico, que 99 por ciento de los contextos están conformados por 12 o menos oraciones–, decidimos restringir el tamaño de nuestros contextos a 10 oraciones. Esto, con el fin de simplificar nuestras predicciones, acotando el espacio de predicciones a únicamente 10 etiquetas.

4.1.4 Creación de atributos de distancia

Para cada pregunta dentro de los datos se crearon los atributos de distancia entre la pregunta y cada una de las oraciones que componen el contexto de la pregunta. Se calculó la distancia euclidiana y la distancia coseno, se almacenaron en listas anidadas cuya posición indica las oraciones que se compararon.

4.1.5 Análisis de raíces con *spacy*

La interpretación de textos es difícil para los humanos, más aún para las máquinas; en este sentido, suponiendo que:

> Es muy probable que las preguntas y sus respectivas oraciones/respuestas contengan las mismas raíces (verbos).

Empleamos la paquetería *spacy*, para: i) encontrar las raíces de las oraciones de los contexto y las raíces de las oraciones/preguntas; ii) generar un vector que contenga el número de raíces en cada índice; y, iii) construir una matriz con las posiciones en las que coinciden las raíces de la pregunta y las oraciones del contexto.

En particular, la paquetería *spacy* nos provee una estructura sintáctica de las oraciones, para resolver posibles ambigüedades estructurales. Esto lo logra mediante un algoritmo que *parseo* que permite encontrar las relaciones dirigidas entre las palabras, identificando, como se muestra el la Figura 8 las siguientes partes de una oración: sujeto, objeto, atributos, raíces y sus dependencias[4].

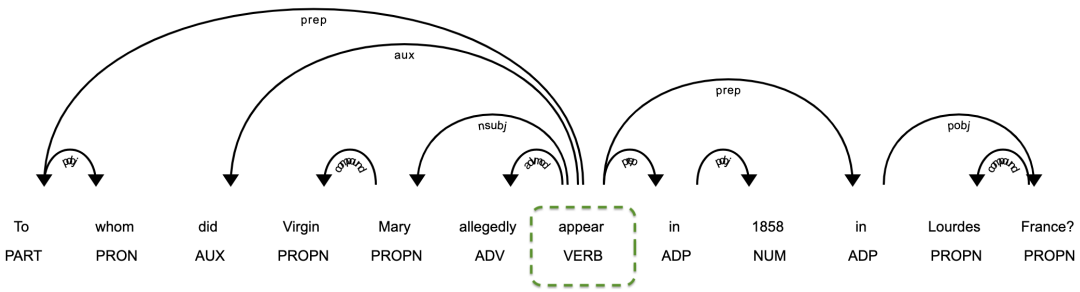


Figura 8: Arbol de dependencia de la oración/pregunta: *"To whom did Virgin Mary allegedly appear in 1858 in Lourdes France?"*

Por construcción, las preguntas en nuestro dataset solo tienen una raíz, mientras que las oraciones del contexto pueden tener una o más raíces. Para verificar lo anterior, remítase a la Figura 9, dónde se presentan dos *dependancy trees*, de dos oraciones que corresponden al contexto dónde se encuentra la respuesta a la pregunta: *to whom did Virgin Mary allegedly appear in 1858 in Lourdes France?* Como se puede observar en esta figura, el segundo *dependancy tree* tiene cuatro raíces o verbos: *is, replica, at and appeared.*

Para esta pregunta en particular, el objetivo es encontrar la raíz de la pregunta y compararla con la raíz o raíces de cada una de las oraciones o tokens del contexto. Recordando el supuesto inicial, podemos verificar que dadas las representaciones de ambas oraciones, la raíz de la pregunta y de la oración que contiene la respuesta es la misma. Es importante mencionar que esta regla no siempre se cumple, además puede haber muchas oraciones dentro del contexto con esta característica, sin embargo aumenta la posibilidad de que la oración sea la que responde la pregunta.

4.2 Modelo no supervisado

Para la implementación del modelo no supervisado, calculamos las distancia euclideana y coseno para detectar las oraciones (de los contextos) que tengan la menor distancia de las respectivas preguntas. Los resultados de estas implementaciones, como podrán ser corroborados en la sección 5, son bastante buenos, sin embargo, no consideran las

[4]La estructura de dependencias y etiquetas que emplea spacy son obtenidas de un inventorio fijo de relaciones gramaticales

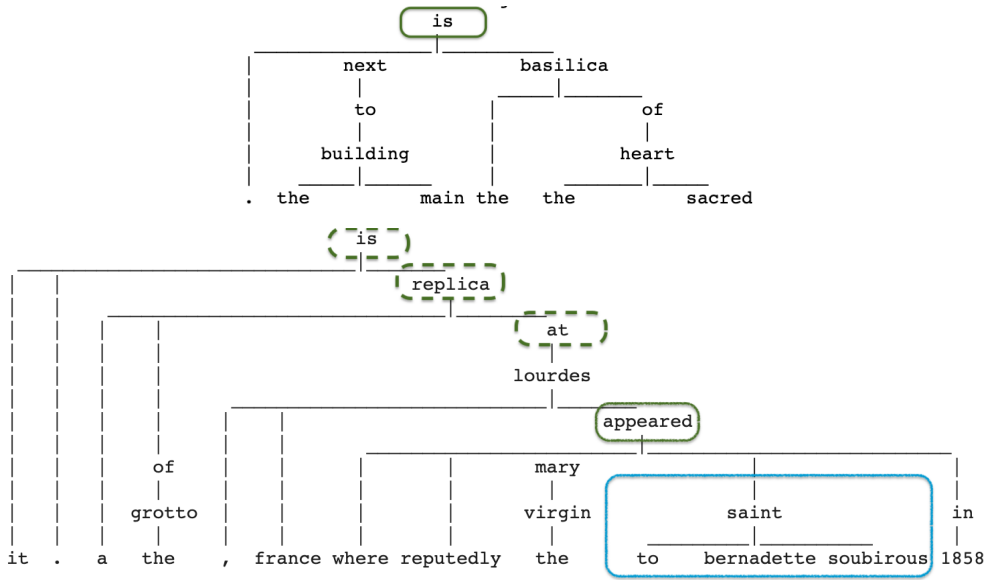


Figura 9: Arbol de dependencia: oraciones/contexto relacionadas con la pregunta: *"To whom did Virgin Mary allegedly appear in 1858 in Lourdes France?"*

variables, descritas en los apartados 4.1.3, 4.1.4, 4.1.5. A priori, creíamos que quizá estas variables podrían aportar y contribuir de forma positiva en nuestras predicciones.

4.3 Modelo Supervisado

Partiendo de lo implementado y desarrollado en las secciones 4.1.3, 4.1.4, 4.1.5, se propusieron tres modelos de clasificación que asignan a cada pregunta la posición de la oración *target*. Se probaron los modelos: random forest, árbol de clasificación y adaboost. Para cada uno de ellos se probaron diferentes sets de hiperparámetros. Se consideraron tres variables para hacer la predicción: la distancia euclidiana y la distancia coseno entre las oraciones y la pregunta y el indicador de que la oración y la pregunta comparten la misma raíz. Estas variables fueron transformadas a formato one hot encodding. El resultado fue una matriz de *features* de dimensión $(86655, 27)$.

El entrenamiento de los modelos se hizo utilizando: i) la paquetería *scikit-learn* para python; ii) un *magic loop*, creado por nosotros, que iteró por los diferentes modelos; y iii) un *grid search* que probó los hiperparámetros. Además se utilizó *cross validation* con $k = 5$. El tiempo total de entrenamiento de los modelos fue menor a 1 hora.

5 Experimentos y Resultados

Tras la implementación de todos los modelos –supervisados o no– identificamos que, independientemente de su naturaleza, los modelos son bastante buenos, como se podrá verificar en la Tabla 2. Además, considerando cómo métrica los *accuracies* de nuestros modelos, no existe diferencias significativas entre el rendimiento de los modelos supervisados contra los no supervisados. Sin embargo, comparando los modelos implementados, dentro de los modelos no supervisados y los modelos supervisados, los desempeños son muy diversos.

En el caso de los modelos no supervisados, el desempeño del modelo que consideró la distancia coseno fue superior en *accuracy* que el modelo que consideró la distancia euclidiana: $accuracy$ en test de 0.60 versus 0.37. Esto puede deberse principalmente a que la distancia euclidiana no considera los ángulos entre los vectores y la distancia coseno sí.

En el caso de los modelos supervisados, el modelo con el mejor desempeño en términos de $accuracy$ fue el modelo Random Forest. En el otro extremo, el modelo con el peor rendimiento en términos de $accuracy$ fue el modelo AdaBoost, cuyo rendimiento fue incluso inferior al del modelo no supervisado que consideró únicamente la distancia euclidiana.

Tabla 2: Desempeño de los distintos modelos: Accuracies

Modelos y Resultados					
No Supervisados			Supervisados		
Modelo	Acc.-Training	Acc.-Test	Modelo	Acc.-Training	Acc.-Test
Distancia Coseno	0.5996	0.600	Random Forest	0.628	0.599
Distancia Euclidiana	0.36966	0.370	Adaboost	0.337	—
			Decision Tree	0.524	—

Ahora bien, como se observa en la Tabla 3, existe una relación inversa entre el desempeño en $accuracy$ (de los modelos supervisados implementados) y su rendimiento en términos de tiempo de entrenamiento: mientras que el modelo Adaboost requirió únicamente 3 segundos para entrenarse, el modelo Random Forest requirió un total de de 804 segundos para entrenarse.

Tabla 3: Desempeño de los distintos modelos

Modelos Supervisados: Tiempos	
Modelo	Tiempo-Training
Random Forest	804 s.
Adaboost	3 s.
Decision Tree	114 s.

Tras revisar estos resultados, valdría la pena cuestionarse sobre el costo de oportunidad que implicó desarrollar los modelos supervisados, con la generación de los features adicionales, puesto que su desempeño en $accuracy$ no es particularmente superior al de los modelos no supervisados.

6 Conclusiones

Para evaluar el desempeño de nuestros modelos, es necesario estratificar nuestro(s) objeto(s) de comparación: los humanos y el resto de modelos implementados al SQuAD dataset. En efecto, comparado con la capacidad humana, nuestro modelo, y el resto de los modelos que se han implementado a propósito del presente dateset, dista mucho de satisfactorio. Como fue señalado por Rajpurkar et al. (2016), el desempeño de los humanos es uniforme y consistentemente mejor que cualquier modelo implementado hasta el momento (con un accuracy por encima de 86 por ciento), sin importar el tipo de pregunta y respuesta (sean respuestas relacionadas con fechas, objetos, entidades, etc.). Sin embargo, comparado con otros modelos, muchísimo más elaborados y con mayor preprocesamiento en los *features/inputs* de los modelos, el desempeño de nuestros modelos es aceptable. De hecho, los creadores del dataset SQuAD (en sus múltiples versiones) comentan en su paper que para su mejor modelo (un modelo con accuracy de 79.3 por ciento, crearon 180 millones de features, la mayoría variables lexicalizadas o producto de árboles de dependencia.

Partiendo de la última observación de los creadores del dataset SQuAD, dejamos pendiente las siguientes tareas:

- incorporar un mayor detalle y análisis en la parte del tree dependancy parsing, que permita un mejor entendimiento de las relaciones implícitas de contexto dentro de cada una de las oraciones;
- extender del presente trabajo, para la predicción de las respuestas exactas;
- probar otros modelos de aprendizaje supervisado;
- realizar una limpieza previa de la información, como por ejemplo, remover stopwords, etc.

References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. volume 6, pages 722–735.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Hermann, K. M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *NIPS*.

Hill, F., Bordes, A., Chopra, S., and Weston, J. (2015). The goldilocks principle: Reading children's books with explicit memory representations.

Mervin, R. (2013). An overview of question answering system. 1.

Mikolov, T., Le, Q., and Sutskever, I. (2013). Exploiting similarities among languages for machine translation.

Rajpurkar, P., Jia, R., and Liang, P. (2018). Know what you don't know: Unanswerable questions for squad. *ArXiv*, abs/1806.03822.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*.

Reinsel, D., Gantz, J., and Rydning, J. (2018). The Digitization of the World From Edge to Core. *World Economic Forum*.

Swalin, A. (2018). SQuAD.