Modelo generativo de consultas SQL a partir de consultas en español

Rubio López Zury Yael[1], Miranda Chávez Víctor Ulises[2], Nicolas Hernández Adair[3] y Pérez Sánchez Ives Lancelote[4]

[1] Instituto Politécnico Nacional, Escuela Superior de Cómputo, Unidad Profesional Adolfo López Mateos, Av. Juan de Dios Bátiz, Nueva Industrial Vallejo, Gustavo A. Madero, 07320 Ciudad de México, CDMX
zrubiol1700@alumno.ipn.mx

Resumen - **En este trabajo, se presenta una propuesta basada en reglas para abordar la traducción automática de consultas de lenguaje natural a consultas SQL. Basándonos en técnicas de procesamiento del lenguaje natural, nuestro modelo captura la estructura de las consultas en lenguaje natural, y las convierte en una serie de posibles consultas SQL equivalentes. El rendimiento obtenido por nuestro modelo utilizando un conjunto de datos basado en el conjunto "Spider", el cual fue traducido al español y limitado a consultas de nivel 1, fue del 25% bajo la métrica Top-N accuracy, lo que revela áreas de mejora, las cuales se describen en este trabajo.**

Palabras clave - **Inteligencia Artificial, Procesamiento de Lenguaje Natural, SQL, Bases de Datos**

1. Introducción

En este artículo se presenta el desarrollo de un modelo generativo de consultas SQL a partir de consultas en español, una iniciativa dirigida a facilitar el aprendizaje del lenguaje SQL, especialmente para aquellos que se adentran por primera vez en el campo de la programación y la gestión de bases de datos.

El modelo, denominado Ñ2SQL, surge como respuesta a la necesidad de proporcionar una herramienta accesible y eficaz para aquellos que buscan familiarizarse con el lenguaje SQL desde cero. El enfoque se centra en responder preguntas comunes en SQL, tales como SELECT FROM WHERE, con todas las especificaciones pertinentes, con el propósito de ofrecer una guía clara y directa para los usuarios.

Para el desarrollo de este modelo, se utilizó un conjunto de datos proveniente de Spider [1], que incluye una amplia variedad de consultas con diversas características. Se seleccionaron ejemplos que siguen una estructura básica comúnmente utilizada en consultas SQL, con la finalidad de proporcionar un marco de referencia claro y conciso para el aprendizaje del lenguaje SQL.

La utilidad principal de este trabajo radica en su capacidad para proporcionar un modelo que sirva como guía para personas que deseen aprender el lenguaje SQL desde cero. Para lograr este objetivo, nos enfocamos en blindar específicamente las consultas que se generan, asegurando que cumplan con las especificaciones requeridas y evitando posibles errores o malentendidos.

Para finalizar, se llevará a cabo una verificación del rendimiento del modelo utilizando una versión traducida y filtrada del conjunto de datos Spider. Esta evaluación permitirá validar la eficacia y la relevancia del modelo en la generación de consultas SQL correctas y relevantes, proporcionando así una perspectiva clara sobre su utilidad y aplicabilidad en el campo de la programación y la gestión de bases de datos.

2. Métodos

2.1 Recopilación del conjunto de datos

Para la elaboración del modelo, se recurrió al conjunto de datos proveniente de Spider, el cual presenta una amplia variedad de consultas con diversas características. Estas consultas incluyen una variedad de operadores, selectores y otras palabras clave propias del lenguaje SQL. Sin embargo, para simplificar el alcance del modelo, se extrajeron ejemplos que siguen una estructura básica comúnmente utilizada: SELECT, FROM y WHERE.

Este conjunto de datos seleccionado comprende consultas que incluyen un selector, el nombre de la base de datos y una condición que presenta el atributo, su valor y el operador correspondiente. Es importante destacar que las consultas no presentan anidamiento, lo que permite un enfoque más claro y directo en el desarrollo del modelo.

La utilidad del trabajo es proporcionar un modelo que sirva como guía para personas principiantes que deseen aprender el lenguaje SQL. Por lo tanto, se optó por centrarse en consultas sencillas que abordan los conceptos fundamentales del SQL de manera accesible para aquellos que se están iniciando en este campo.

Es relevante mencionar que, aunque las consultas originales están redactadas en inglés, se llevó a cabo una tarea de extracción, traducción y transformación para adaptarlas al contexto del modelo en español. Este proceso fue fundamental para garantizar la coherencia y la pertinencia de los datos utilizados en el desarrollo del modelo.

2.2 Ñ2SQL

En esta sección, se presenta la arquitectura Ñ2SQL, la cual pretende abordar el problema de la generación de consultas SQL a partir de consultas en español.

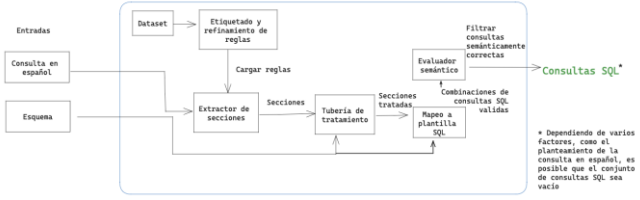


Fig 1. Diagrama de arquitectura Ñ2SQL

Generación de reglas. El primer paso consiste en generar las reglas para realizar la extracción de secciones, las cuales se obtienen haciendo uso del conjunto de datos Spider [1], el

cual se filtró para conservar las consultas objetivo, caracterizadas por tener la estructura SELECT $COLUMNAS FROM $TABLA WHERE $ATRIBUTO_CONDICIONAL $OPERADOR $VALOR_CONDICIONAL. Una vez filtrado se realizaron 2 tipos de anotaciones distintas de forma manual; una para identificar las secciones de la consulta en español que potencialmente contienen información correspondiente a una de las 3 secciones (COLUMNAS, TABLA, CONDICIÓN), mientras que el segundo tipo de anotación tiene el objetivo de detectar partes específicas dentro de las secciones que se identificaron en el punto anterior, siendo las más destacables las columnas individuales, y aquellas partes pertenecientes a la condición.

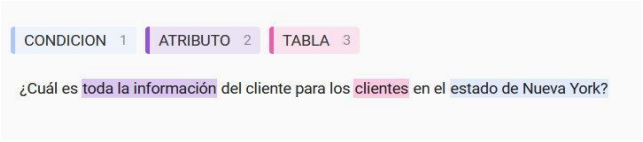


Fig 2. Ejemplo de anotación de sección

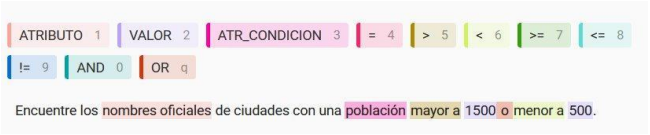


Fig 3. Ejemplo de anotación específica

Etiquetado y refinamiento de reglas. Una vez realizadas las anotaciones, se procedió a aislar el contexto a los lados del texto etiquetado con una ventana de una palabra para formar las reglas que permitirán identificar potenciales secciones relevantes. Solo en el caso de los operadores se utilizarán las anotaciones en vez de su contexto como parte de las reglas generadas.

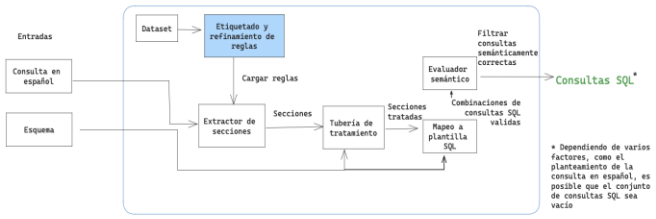


Fig 4. Sección de Etiquetado y refinamiento de reglas perteneciente al diagrama de arquitectura Ñ2SQL

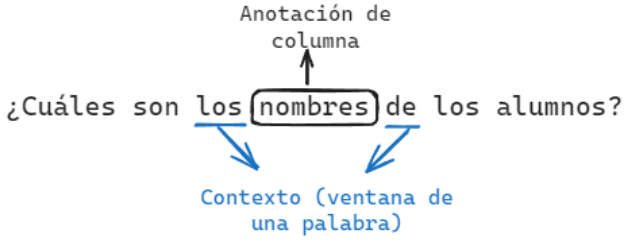


Fig 5. Ejemplo de obtención de contexto (azul) partiendo de una anotación de columna (recuadro negro)

Finalmente, se realizó un refinamiento de reglas apoyándonos del algoritmo de aprendizaje automático AQ [2], el cual se seleccionó ya que se adapta a nuestro enfoque (el modelo que devuelve es un conjunto de reglas lógicas en forma normal disyuntiva), permitiendo obtener conjuntos de reglas simplificados para utilizar en la siguiente etapa. Para lograr esto, se realizaron ajustes al algoritmo AQ para considerar una regla como candidata viable en cuanto supere 80% o más de cubrimiento de los elementos de la clase a aprender, esto con el fin de descartar reglas que no maximicen la generalización del modelo a casos no vistos. Cabe mencionar que el paso de refinamiento puede omitirse (o se puede optar por otro método para refinar las reglas), pero se sugiere realizar este paso si lo que se busca es aumentar las capacidades de generalización del modelo.

Extracción de secciones. Con base en los patrones identificados en la etapa de etiquetado y refinamiento de reglas, este bloque extraerá las secciones de texto de la consulta que considere relevantes para generar la consulta SQL (correspondientes a alguno de los bloques que conforman una consulta SQL básica, ya sea el nombre de la tabla, de los atributos, o la condición).

Siguiendo el ejemplo de la figura 3, suponiendo que sólo se conoce la consulta en español y la regla [contexto_izquierda = los][contexto_derecha = de] -> columna , el extractor buscará una sección de texto rodeada por el contexto definido en la regla, y extraerá la sección de texto, anotando como una potencial columna. Las extracciones realizadas son independientes unas de otras, por lo que si varias reglas cubren a la misma porción de texto, esto no tendrá mayor efecto y simplemente se extraerá anotando con la clase de la regla que también haya cubierto la misma sección.

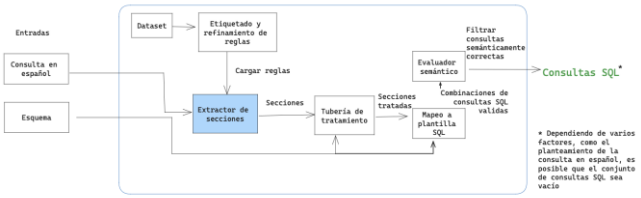
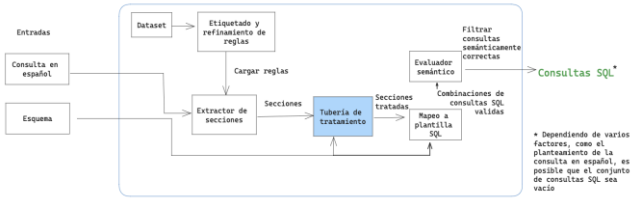


Fig 6. Extractor de secciones perteneciente al diagrama de arquitectura Ñ2SQL

Tubería de tratamiento. Las tuberías de tratamiento recibirán las secciones obtenidas en el punto anterior con sus respectivas clasificaciones, y tendrán el objetivo de limpiar las secciones de texto para transformarlas en la sección de la consulta que corresponda con la clasificación del texto. A continuación, se describen 3 potenciales estrategias para realizar esta limpieza



.

Fig 7. Sección de Tubería de tratamiento perteneciente al diagrama de arquitectura Ñ2SQL

Estrategia simple. Esta estrategia asume que los textos extraídos no necesitan ningún tratamiento adicional, y devolverá el texto recibido como el elemento de la consulta

SQL que se busca (dependiendo de la clasificación). En el caso de las condiciones, se hace uso de un tratamiento mínimo, en el cual se utilizan dos conjuntos de reglas, uno para extraer los operadores lógicos de una sección, y otro para extraer los atributos y valores condicionales; y una vez extraídas estas secciones, se forma una condición potencial sin dar ningún tratamiento adicional.

Estrategia de incrustaciones. Esta tubería está diseñada para llevar a cabo la evaluación semántica mediante el uso de incrustaciones de palabras, la cual, haciendo uso de un umbral de similitud, permite abordar el problema de la sinonimia, brindando flexibilidad a la hora de escribir consultas en español y permitiendo adaptarse a distintos contextos.

Esta implementación se fundamenta en la biblioteca SpaCy, utilizando concretamente el modelo preentrenado es_core_news_md. Este modelo, diseñado para el análisis de textos en español, facilita la ejecución de diversas tareas como la tokenización, el etiquetado gramatical, y además proporciona una serie de embeddings pre entrenados mediante el algoritmo FastText [3][4].

El funcionamiento de esta tubería se centra en tokenizar las palabras en una sección, e iterar sobre ellas, comparando su embedding con los embeddings de las palabras que corresponden a elementos de la base de datos que se proporcionó (nombres de columnas o de tablas). Esta misma estrategia se utiliza para obtener un atributo condicional, mientras que para obtener un valor condicional se utiliza el mismo enfoque planteado en la estrategia simple.

Mapeo a consultas SQL. En esta etapa se generan todas las combinaciones válidas de consultas posibles con base en las secciones tratadas resultantes de la etapa anterior (sin darle importancia a si las consultas generadas son semánticamente correctas, ya que esto se abordará más adelante).

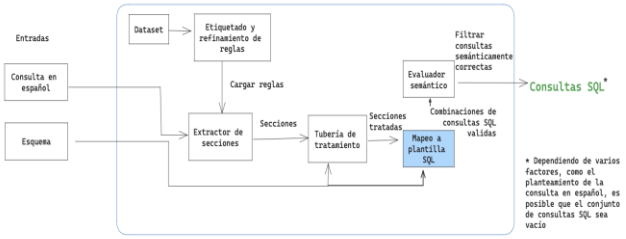


Fig 8. Sección de Mapeo a plantilla SQL perteneciente al diagrama de arquitectura Ñ2SQL

Evaluador semántico. Con la información del esquema de la base de datos y una potencial consulta SQL, se determinará si dicha consulta es correcta semánticamente, para lo cual se realizarán distintas acciones y verificaciones con el fin de verificar si la consulta tiene sentido semántico bajo el esquema proporcionado. Por ejemplo, si se tiene una consulta "SELECT alumno FROM edificio", pero el esquema no cuenta con una tabla "edificio", dicha consulta sería semánticamente incorrecta.

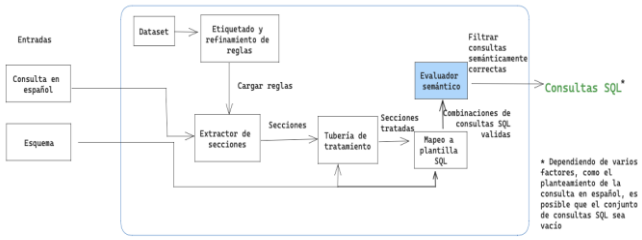


Fig 9. Sección de Evaluador semántico perteneciente al diagrama de arquitectura Ñ2SQL

3. Resultados

3.1 Resultados obtenidos

Para la obtención de resultados, se identificó la necesidad de realizar dos tipos de evaluaciones. La primera evaluación incluyó los datos obtenidos del conjunto de datos Spider y la segunda evaluación se llevó a cabo utilizando una instancia de datos específica diseñada para probar el modelo en un contexto más personalizado y práctico.

Para el primer tipo de evaluación, se recuperaron 166 objetos de tipo Database, con 674 consultas en lenguaje natural y consultas SQL. Los resultados fueron los siguientes:

Tabla 1. Resultados por experimento con los datos provenientes de SPIDER

Id experimento	Umbral de similitud	Umbral condición	No. consultas correctamente predichas	No. total de consultas a prueba	Top-N Accuracy
1	N/A	N/A	0	674	0%
2	0.8	0.8	98	674	14.54%
3	0.8	0.5	131	674	19.43%
4	0.8	0.1	173	674	25.81%

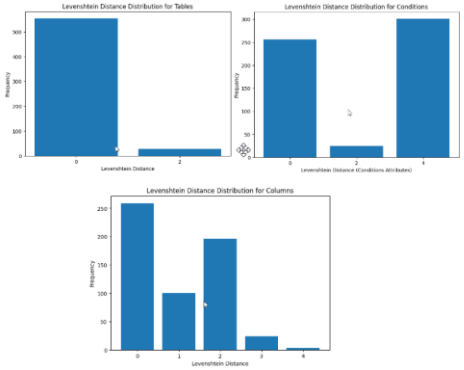


Fig 10. Distancia Levenshtein del experimento 2 tabla 1

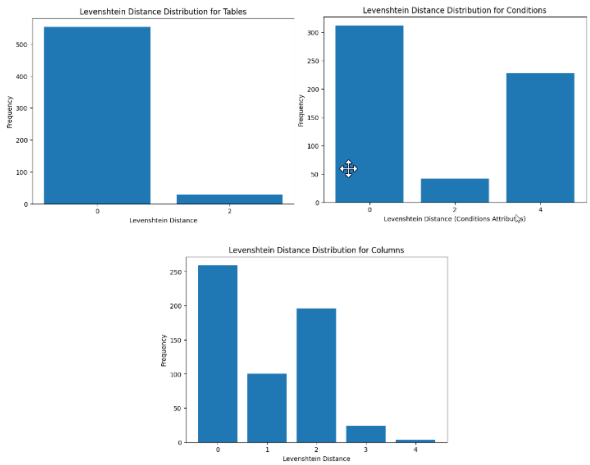


Fig 11. Distancia Levenshtein del experimento 3 tabla 1

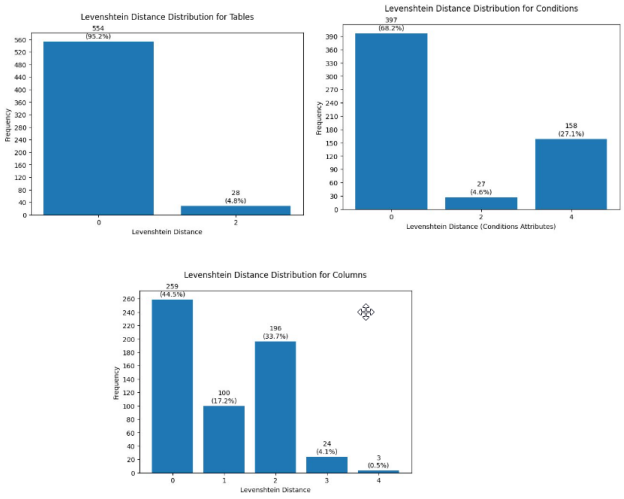


Fig 12. Distancia Levenshtein del experimento 4 tabla 1

Para el segundo tipo de evaluación las características de la instancia mencionada son las siguientes:

Base de datos	Consulta SQL tipo Query	Consulta en lenguaje natural
Nombre de la base: Database Tabla 1: Estudiantes Columnas: ● identificador ● nombre ● país ● edad Tabla 2: Cursos Columnas: ● identificador ● nombre ● profesor	SELECT nombre, identificador WHERE pais = 'Mexico'	"Muestra todos los nombres de los alumnos y sus identificadores que estudian en México"

Tabla 2. Detalles de los datos de entrada controlado para el rendimiento del modelo

En la siguiente tabla se detallan los resultados del segundo tipo de evaluación, en la que verificamos si la consulta SQL generada coincide correctamente con la consulta SQL esperada correspondiente a la consulta en lenguaje natural utilizada como entrada.

Tabla 3. Resultado para el experimento con datos controlados

Id experimento	Tubería	Umbral (si aplica)	La consulta SQL fue correcta
1	EmbeddingPipeline	0.8	Si

Ya calculados estos resultados, en el siguiente apartado analizaremos a detalle lo que nuestros hallazgos sugieren.

3.2 Análisis

Los resultados del experimento revelaron una baja exactitud en las consultas generadas, lo que sugiere una falta de comprensión del modelo sobre la información implícita en las consultas en lenguaje natural. Particularmente, se observó una dependencia crítica en la definición del umbral en la tubería de embedding, lo que destaca la importancia de ajustar este parámetro para mejorar el rendimiento del sistema. Estos hallazgos subrayan la necesidad de mejorar la capacidad del modelo para interpretar y procesar consultas con mayor variabilidad, proporcionando una base sólida para futuras investigaciones y mejoras en sistemas de procesamiento de lenguaje natural para la generación de consultas SQL.

4. Discusión

En vista de los resultados que podemos observar de acuerdo con los gráficos anteriores, estos nos indican claramente la necesidad de realizar mejoras significativas en el modelo y explorar nuevas estrategias para mejorar su desempeño.

Una de las líneas de trabajo futuro consiste en implementar nuevas tuberías de procesamiento de texto, y mejorar las ya existentes. Además, se buscará generar reglas más efectivas para la extracción de secciones de texto y la generación de consultas SQL. Esto implicará un análisis exhaustivo de las reglas existentes y la exploración de nuevas estrategias de generación de reglas que permitan capturar de manera más precisa la estructura y el significado de las consultas en español.

Por último, se llevarán a cabo experimentos mediante una búsqueda de cuadrícula de umbrales para optimizar la tubería de embeddings. La elección de los umbrales adecuados es crucial para obtener representaciones semánticas precisas de las palabras en el texto, lo que puede tener un impacto significativo en el rendimiento general del modelo.

En general, el trabajo futuro se centrará en explorar nuevas estrategias y técnicas para mejorar el rendimiento del modelo, desde la implementación de nuevas tuberías de procesamiento de texto, hasta la búsqueda de estrategias para refinar y generar reglas más efectivas. Además, se realizarán múltiples experimentos para optimizar la arquitectura actual y así mejorar la calidad de las predicciones del modelo.

Referencias

1. T. Yu et al., “Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task”. arXiv, el 2 de febrero de 2019. Consultado: el 16 de noviembre de 2023. [En línea]. Disponible en: http://arxiv.org/abs/1809.08887
2. G. Cervone, P. Franzese, y A. P. K. Keesee, “Algorithm quasi-optimal (AQ) learning”, WIREs Computational Stats, vol. 2, núm. 2, pp. 218–236, mar. 2010, doi: 10.1002/wics.78.
3. “Spanish · spaCy Models Documentation”, Spanish. Consultado: el 15 de mayo de 2024. [En línea]. Disponible en: https://spacy.io/models/es#es_core_news_md
4. “fastText”. Consultado: el 15 de mayo de 2024. [En línea]. Disponible en: https://fasttext.cc/index.html