ISSN: 2174-7245
Año 2025, Volumen 15, Número 1.doi: 10.7203/Normas.v15i1.32190
Publicado: 2025. Enviado: 2025-10-28. Aceptado: 2025-11-2.

Extracción automática de concordancias del CORDE: una propuesta para la obtención de datos

Automatic extraction of concordances from CORDE: a proposal for data collection

Elia Puertas Ribés
Universitat Jaume I (UJI)

Abstract

Within the framework of corpus linguistics, the Spanish language has numerous reference databases for diachronic and synchronic studies. These platforms are essential reference tools for the analysis of linguistic phenomena using real-world data, as they compile texts from the origins of Spanish to the present day. From a technical perspective, the rise of new corpora has fostered critical reflection on some of the limitations of these applications. In diachrony, for example, conventional corpora, composed of a computerized database and a search engine, sometimes lack the ability to download concordances, which hinders research and slows down both the organization and analysis of the evidence obtained during the search. Manually extracting information from these tools is therefore tedious, as researchers must copy and store the data one by one, without automated assistance, resulting in excessive time consumption and a greater risk of human error that affects the accuracy of the studies. Faced with this challenge, this paper presents a proposal for the automated extraction of data from the Diachronic Corpus of Spanish (CORDE) of the Royal Spanish Academy. To do so, we used the tools offered by computational linguistics to integrate Selenium and Python into the analysis of a reference database, CORDE, from which we aim to obtain concordances through web scraping, a technique that searches, downloads, and processes web content with minimal manual intervention. Finally, it is worth mentioning that the script described in this paper is available to researchers on GitHub.

Keywords: computational linguistics, web scraping, data extraction, corpus linguistics, CORDE.

Resumen

En el marco de la lingüística de corpus, la lengua española cuenta con numerosas bases de datos textuales de referencia para el estudio diacrónico y sincrónico. Estas plataformas son herramientas de consulta obligatoria para el análisis de fenómenos lingüísticos a partir de datos reales, ya que recopilan textos desde los orígenes del castellano hasta la actualidad. No obstante, desde una perspectiva técnica, el auge de nuevos corpus ha fomentado una reflexión crítica sobre algunas de las limitaciones de estas aplicaciones. En diacronía, por ejemplo, los corpus convencionales, compuestos por una base de datos informatizada y un mecanismo de búsqueda, carecen en algunos casos de la posibilidad de descargar las concordancias, lo que dificulta la labor investigadora y ralentiza tanto la organización como el estudio de los testimonios obtenidos en la consulta. La extracción manual de información en estas herramientas representa, pues, un trabajo tedioso, dado que los investigadores deben copiar y almacenar los datos uno a uno, sin asistencia automatizada, con un consumo excesivo de tiempo y un riesgo mayor de errores humanos que afectan a la precisión de los estudios. Ante ese reto, en este trabajo se presenta una propuesta para la extracción automatizada de datos del Corpus Diacrónico del Español (CORDE), de la Real Academia Española. Para ello, nos hemos servido de las herramientas que ofrece la lingüística computacional para integrar Selenium y Python en la extracción de información de una base de datos de referencia como es el CORDE, de la que se pretenden obtener las concordancias por medio del web scraping, técnica que busca, descarga y procesa contenidos de una página con escasa intervención manual. Por último, cabe mencionar que el script que se describe en este trabajo se pone a disposición de los investigadores en GitHub.

Palabras clave: lingüística computacional, web scraping, extracción de datos, lingüística de corpus, CORDE.

Citar como: Puertas Ribés, Elia (2025). Extracción automática de concordancias del corde: una propuesta para la obtención de datos. Normas, 15(1), 1-11, doi: 10.7203/Normas.v15i1.32190.

1. Introducción[1]

Los investigadores en lengua española se benefician desde hace ya un tiempo de numerosas bases de datos de referencia, tanto para los estudios sincrónicos como para los diacrónicos. Estos corpus lingüísticos se establecen, por consiguiente, como una herramienta de consulta obligatoria para el análisis —cuantitativo y cualitativo— de cualquier fenómeno lingüístico que parta de datos reales, dado que atesoran una cantidad enorme de textos que abarcan desde los orígenes del español hasta la actualidad.

Ahora bien, el auge de nuevos corpus lingüísticos, con un nivel de desarrollo técnico cada vez mayor, ha fomentado, a su vez, una reflexión crítica sobre las limitaciones que se encuentran en aquellos corpus más convencionales (Enrique Arias, 2012; Vaamonde, 2015; Blas Arroyo y Puertas Ribés 2014; etc.). En el plano de la diacronía, por ejemplo, el Corpus Diacrónico del Español (CORDE), de la Real Academia Española, constituye una base de datos de referencia y de consulta obligatoria, aunque carece, desafortunadamente, de la posibilidad de descargar las concordancias recuperadas tras la búsqueda, de manera que ralentiza la labor investigadora en el proceso de organización y estudio de los testimonios obtenidos en la consulta (Vaamonde, 2015). Por consiguiente, la extracción manual de información en este tipo de plataformas representa un trabajo tedioso, pues los investigadores deben copiar y estructurar los datos de uno en uno, sin asistencia automatizada, lo que supone, en ocasiones, un consumo excesivo de tiempo (Torruella y Llisterri, 1999: 4).

Ante ese reto, en este trabajo presentamos la propuesta de un *script* para la descarga automática de datos del CORDE. Para ello, nos servimos del Procesamiento del Lenguaje Natural (PLN), un campo que, según Jurafsky y Martin (2021), ha experimentado grandes avances gracias a modelos computacionales que han facilitado el desarrollo de herramientas avanzadas para la recuperación, análisis y visualización de grandes volúmenes de textos. En nuestro caso, para la extracción automática de datos del CORDE, se recurre concretamente al *web scraping*, una técnica que busca, descarga y procesa contenidos de una página sin prácticamente intervención manual (Salve Cuenca, 2025: 304).

Ahora bien, es importante tener presente que la obtención y el acceso a los datos es diferente en función de la configuración del sitio web. Por eso, en páginas estáticas —cuyos archivos HTML, CSS y JavaScript ya están generados en el servidor—, la recuperación de datos por medio del *web scraping* resulta relativamente sencilla, ya que es posible extraer la información utilizando directamente herramientas diseñadas específicamente para ello, como BeautifulSoup, una librería de Python especializada en el análisis de documentos HTML y XML.

En cambio, en sitios dinámicos —como es el caso de muchas bases de datos textuales convencionales—, el contenido se genera o modifica en el servidor en respuesta a interacciones del usuario: por ejemplo, hacer clic en botones, enviar formularios, etc. Por ello, la configuración del *backend* es más compleja en las páginas dinámicas que en las estáticas y, en consecuencia, el uso de librerías como BeautifulSoup no es suficiente para la extracción

[1]El presente estudio forma parte del proyecto de investigación «Componentes socioestilísticos, idiolectales y discursivos en la variación y el cambio lingüístico en español: contribuciones desde la sociolingüística histórica» (2022-2026), financiado por el Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación y por FEDER Una manera de hacer Europa (Ref. PID2021-122597NB-I00) así como por el Subprograma estatal de Movilidad, del Plan Estatal de Investigación Científica, Técnica y de Innovación (PRX23/00019). Además, quisiera agradecer a Juan Pablo Herrera su colaboración en la creación y revisión del código diseñado.

de información. En estas situaciones es preciso simular la navegación real por la página web para que el servidor recupere los datos solicitados de las búsquedas. En el caso que nos ocupa, esa labor se pone en práctica por medio de Selenium, un *framework* de automatización de navegadores que permite controlar de forma remota un sitio web —emulando clics, rellenando formularios, navegación compleja, etc.— y recuperar el DOM resultante, esto es, la estructura completa de la página con todos los elementos HTML y CSS ya actualizados tras la consulta.

De este modo, con un *script* escrito en Python, el uso de Selenium facilita localizar y extraer etiquetas, atributos o bloques de texto concretos, descargar archivos vinculados y, en definitiva, automatizar todo el recorrido que un investigador haría manualmente. En el presente trabajo, su aplicación se lleva a cabo en una web dinámica de la Real Academia Española, el Corpus Diacrónico del Español (CORDE), que, a diferencia de otras bases de datos textuales de esta misma institución —que sí facilitan la descarga masiva de datos—, no presenta este servicio de exportación de concordancias. Así pues, mediante la automatización de este proceso con Selenium, es posible recuperar los testimonios de manera rápida y sistemática; una tarea que, en definitiva, optimiza la obtención de los textos en un formato más cómodo y manejable, así como la descarga de los resultados a un formato estructurado (.xlsx). De esta forma, no solo se ahorra tiempo, sino que garantiza que cada búsqueda se registre y almacene de manera uniforme, minimizando errores humanos y facilitando el posterior tratamiento estadístico o lingüístico de los datos.

Este artículo se estructura del siguiente modo: en el segundo apartado se ofrece una breve revisión de las características técnicas relacionadas con la exportación y el almacenamiento de datos de los corpus diacrónicos. Seguidamente, en el apartado 3, se describen los fundamentos técnicos y la metodología necesarios para el diseño del código propuesto. A continuación, en el apartado cuarto se presenta la propuesta del *script* y se explican los archivos de los que consta; por último, en § 5, se sintetizan las principales implicaciones metodológicas y los retos futuros.

2. Exportación y descarga de datos en corpus diacrónicos

En la actualidad, cada vez son más los grupos de investigación que aúnan esfuerzos por diseñar bases de datos óptimas para el estudio filológico aprovechando las ventajas que proporcionan las Humanidades Digitales y la Lingüística Computacional, de ahí la proliferación de estudios que centran su atención en el diseño y explotación de estos recursos digitales: lematización y anotación, presentación de diferentes ediciones del texto, la adición de los metadatos, las posibilidades de búsqueda avanzada (*regex*, CQL, etc.) (Vaamonde, 2015; Díaz Bravo, 2018; Calderón, 2019; Calderón y Vaamonde, 2020; etc.).

En términos metodológicos, en el presente trabajo nos centraremos únicamente en la funcionalidad de exportación de datos. Para ello, es interesante describir someramente las posibilidades que ofrecen los distintos corpus diacrónicos de acceso abierto a los usuarios a la hora de organizar y almacenar los datos en sus dispositivos. Antes de nada, huelga decir que la mayoría se aprovecha ya del potencial que ofrecen las nuevas tecnologías para satisfacer y agilizar la extracción de información. Por eso, ante la imposibilidad de enumerarlos todos, mencionaremos solamente algunos de ellos.

Por ejemplo, entre las bases de datos que ofrecen la funcionalidad de exportación en formato estructurado destacamos, en primer lugar, aquellos sistemas que posibilitan almacenar y descargar los datos de forma más sencilla y con un acceso más directo al contenido lingüístico, como el formato de texto plano (TXT), que constituye la opción más simple, ya que presenta únicamente el contenido lingüístico sin anotaciones ni metadatos[2]. En cambio, existen otros tipos de codificación más complejos, que clasificaremos aquí en dos categorías: en el polo opuesto a los primeros (TXT), se encuentran los corpus que preservan los datos mediante estructuras jerárquicas (XML, TEI, JSON, etc.) con el propósito de facilitar la recuperación de datos a través de distintos niveles de etiquetado lingüístico (y editorial), la incorporación de nuevas anotaciones, así como la conservación de metadatos —o información contextual del texto—, como son: *autor, fecha, fuente*, etc. En un nivel intermedio, se hallan los formatos delimitados por tabulaciones (TSV) o por comas (CSV), que permiten almacenar y visualizar la información bien en hojas de cálculo, como Excel, bien por medio de bibliotecas de análisis de datos, como Pandas en Python. Por último, algunas plataformas permiten además descargar los resultados en PDF, aunque, en este caso, se trata de un formato que únicamente permite la visualización de los datos

En este sentido, algunos de los corpus del español en diacronía que cuentan con los formatos de exportación que hemos comentado son los siguientes:

Corpus	Texto plano (TXT)	Formatos tabulares (CSV, TSV, etc.)	Estructura jerárquica (XML, TEI, etc.)	Otros formatos (PDF, JPG, etc.)
Corpus de textos de Inmediatez Comunicativa (INCOM)	X	X	X	
Post Scriptum (P.S.)	X	X		
Corpus Hispánico y Americano en la Red: Textos Antiguos (CHARTA)	X		X	
Corpus Histórico del Español Norteño (CORHEN)		X		X
Oralia Diacrónica del Español (ODE)	X			
Old Spanish Textual Archive (OSTA)			X	
Corpus Léxico de Inventarios (CorLexIn)				X

Tabla 1. Clasificación de corpus diacrónicos en español según el sistema de exportación (elaboración propia)

De forma similar, las bases de datos de la Real Academia Española han ido incorporando las mismas funcionalidades en herramientas más recientes, como el CORPES XXI. Sin embargo, el CORDE constituye un proyecto cerrado, que no permite la modificación de sus utilidades, por lo que dificulta la organización y el análisis de las concordancias obtenidas en las consultas. Este corpus, que se convierte para cualquier filólogo en una herramienta de referencia y de consulta obligatoria, fue creado a finales de la década de los noventa, y es considerado el primer corpus histórico del español de acceso abierto (Rojo, 2010, 2016). Por consiguiente, teniendo en cuenta los diferentes tipos de exportación de datos, consideramos que el formato tabular es el más adecuado para el diseño del código que se presenta en § 4, dado que tratamos de almacenar las concordancias que proporciona el CORDE en filas y columnas con el objetivo de facilitar su posterior análisis.

En este contexto, la siguiente sección presenta los fundamentos técnicos y las herramientas

[2]Algunos corpus, pese a facilitar la exportación en TXT, presentan un etiquetado estructural, como ocurre en ODE.

específicas que se han empleado para automatizar y controlar procesos de extracción, transformación, así como los criterios que guiaron la implementación del código.

3. Fundamentos técnicos y metodología

Para la exportación efectiva de los resultados es imprescindible analizar previamente la infraestructura digital en la que se pretende poner en funcionamiento el código para comprender cómo está configurada y qué técnicas computacionales son óptimas para la ejecución del *script*. Así pues, la arquitectura del CORDE, como la de cualquier sitio web, se conforma por el *frontend,* interfaz con la que interactúa el usuario y que determina la forma en que se presentan y visualizan los contenidos, y el *backend*, que comprende la programación y la gestión de datos en el servidor, de modo que permite desarrollar la aplicación, procesar las solicitudes de los usuarios y recuperar la información almacenada de manera eficiente (Celi Párraga *et alii*, 2023: 6). Estos conceptos, a su vez, están estrechamente relacionados con el tipo de sitio web: una página es estática, cuando no se modifica su apariencia ni su contenido tras las acciones del usuario, mientras que aquellas consideradas como dinámicas, como el CORDE, se caracterizan por no proporcionar toda su información de manera inmediata en el *backend*, sino que van actualizando el contenido del sitio web en función de la consulta del internauta.

Para la obtención de datos, en el caso de las páginas estáticas, se emplea el *web scraping* para rastrear y descargar datos no estructurados (o semiestructurados) en un formato estructurado mediante el uso de *bots* que automatizan por completo el proceso con el fin de evitar la necesidad de una intervención manual detallada (Hernández *et alii*, 2015; Martínez *et alii*, 2019). Así pues, 'scrapear' un sitio web ofrece la posibilidad de obtener información, que, en términos de tiempo, se considera una forma más eficiente que la extracción manual (Acevedo-Castiblanco *et alii*, 2023). En cambio, en las páginas dinámicas, la configuración del *backend* condiciona el método de acceso a la información. Por tanto, la metodología del *web scraping* no podría realizarse automáticamente solo con librerías de Python, como BeautifulSoup, de modo que, además de usar Python como lenguaje de programación, se emplea el componente de WebDriver de Selenium para emular las acciones interactivas que podría hacer un usuario (acceder al CORDE, hacer clic en los botones, recuperar las concordancias, etc.). A continuación, el código que se propone en el siguiente apartado puede obtenerse en GitHub[3], una librería de código abierto que permite no solo descargarlo, sino también aportar modificaciones que ayuden a mejorar su funcionamiento.

4. Propuesta de extracción automática

El código que se presenta a continuación permite descargar las concordancias obtenidas en el CORDE tras una consulta lingüística. Para una correcta ejecución del *script*, los archivos necesarios para su implementación deben almacenarse en el directorio de trabajo en el que se encuentra la terminal del ordenador. Para ello, el código que se propone en este trabajo puede consultarse y descargarse gratuitamente en formato ZIP desde GitHub.

En los siguientes subapartados se describe el funcionamiento del código, que consta de cuatro archivos: las instrucciones de uso (*README.md*) (§ 4.1.); los requisitos previos antes de su

[3]https://github.com/eliapuertas/Extraccion_de_concordanciasCORDE.git

ejecución (*requirments.txt*) (§ 4.2.); las indicaciones que se visualizan en la terminal si hay algún error en el proceso (*debug.py*) (§ 4.3.) y el código ejecutable a través de un comando en la terminal (*corde_scraping.py*) (§ 4.4.).

4.1. *Instrucciones de uso: README.md*

En primer lugar, el código consta de unas instrucciones que sirven de guía al usuario para conocer qué pasos deben seguirse a la hora de ejecutar el *script*. Por ello, el archivo *README.md* es un documento que proporciona información al usuario no solo sobre cómo usarlo, sino también sobre cuáles son los requisitos necesarios, así como los pasos a seguir para la instalación de las dependencias —esto es, los paquetes y los programas imprescindibles para un funcionamiento adecuado— y los navegadores compatibles (Chrome, Firefox y Edge) para su aplicación.

4.2. *Requisitos previos a la ejecución: requirements.txt*

Así pues, el usuario que acceda a la información del documento *README.me* podrá copiar en la terminal el primer comando que aparece para instalar automáticamente los paquetes que se requieren.

````
Además, necesitaremos un intérprete de python instalado es
recomendable utilizar una versión igual o superior a la `3.11`

Una vez que tengamos nuestro intérprete ejecutaremos el
siguiente código para instalar los paquetes necesarios:

```bash
# Es importante que estemos situados en este directorio (el
que contiene los scripts)
pip install -r requirements.txt
```
````

Imagen 1. Comprando para la instalación de los requisitos

El documento *requirements.txt*, pues, proporciona un listado con las dependencias (o paquetes, herramientas, etc.) que no pertenecen a Python, pero que se requieren para su funcionamiento, al igual que las versiones recomendadas. No obstante, con el comando que se muestra en la Imagen 1, el usuario no necesitará abrir —ni entender en qué consiste— este segundo archivo, que tiene apariencia de lista:

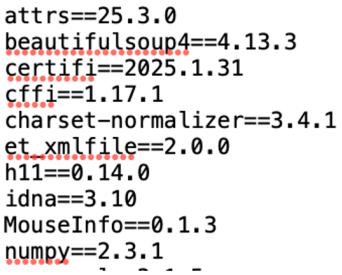

Imagen 2. Listado de dependencias en requirements.txt

4.3. *Archivo para encontrar errores en el código: debug.py*

Además, el código cuenta con un archivo (*debug.py*) que se ejecuta automáticamente en la terminal para seguir el proceso de implementación del *script* principal (*corde_sccraper.py*) con el objetivo de identificar y analizar posibles errores durante la ejecución del código. Su funcionamiento se ilustra en la Imagen 3, en la que el usuario puede conocer en tiempo real en qué etapa del proceso se encuentra la ejecución del código:

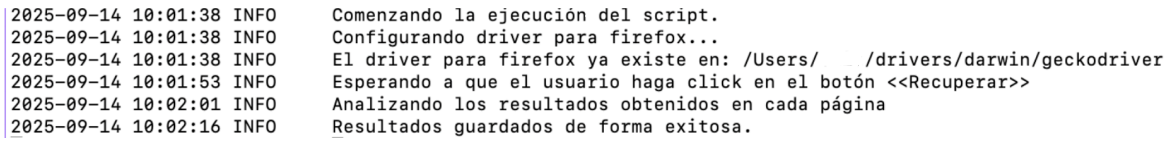

Imagen 3. Ejemplificación de la ejecución del debug.py

4.4. *Archivo ejecutable: corde_scraper.py*

Finalmente, el archivo *corde_scraper.py* facilita la obtención de las concordancias del CORDE. Para ello, es imprescindible ejecutar en la terminal el siguiente comando que se proporciona en las instrucciones (*README.md*):

Imagen 4. Comando para ejecutar automáticamente el código

De este modo, el código se pone en funcionamiento, por lo que se abrirá de forma automática el explorador instalado (por ejemplo, Firefox) con la interfaz de consulta del CORDE, en la que el investigador realizará su consulta.

La configuración del código, por medio de diferentes funciones, posibilita iniciar el navegador web usando Selenium para emular la interacción que un internauta haría de forma manual. Por consiguiente, se abre la URL del CORDE y se visualiza su interfaz, en la que el código permanecerá un máximo de cinco minutos en funcionamiento para que el usuario realice la búsqueda, pulse el botón de *Buscar* y, en la página siguiente, el de *Recuperar*.

Seguidamente, el código analiza el contenido de la plataforma sin intervención manual, por lo que exporta los datos pasando automáticamente desde la primera hasta la última página de los resultados obtenidos. En esta fase, se emplean expresiones regulares (*regex*) para localizar y extraer patrones específicos de texto dentro de la página web, como las concordancias de las palabras buscadas y los delimitadores de campos de año, autor, título, país, tema y publicación. Las *regex*, por tanto, permiten identificar de manera precisa estos elementos dentro del CORDE, facilitando que la información se transforme en un formato estructurado listo para su exportación.

Una vez se hayan recuperado los testimonios, el navegador se cerrará y el usuario podrá comprobar en la terminal que, en la última línea escrita, se informa de lo siguiente: 'Resultados guardados de forma exitosa'. De todos modos, cabe recordar que el *script* se ha diseñado para descargar únicamente las concordancias, por lo que no funcionará en caso de que, en la opción de obtención de ejemplos, se seleccione *Documentos*. Por eso, si el CORDE devuelve muchos testimonios y solo se visualizan por documentos, se recomienda acotar las búsquedas a intervalos temporales menores para poder obtener directamente las concordancias.

Por último, una vez finalizada la ejecución del código, es posible acceder a los resultados en una carpeta que se crea sin intervención manual con el nombre de *recursos*, situada en el mismo directorio en el que se encuentran los archivos y la terminal. En ella, se irán almacenando distintas subcarpetas con la fecha del día de la consulta y, dentro, el documento en formato XLSX, que presenta una estructura como la siguiente:

Nº	CONCORDANCIA	AÑO	AUTOR	TÍTULO	PAÍS	TEMA	PUBLICACIÓN
1	za de animales dañinos, lobos, zorros, garduñas, gatos monteses, linces, tejone	1902	Anónimo	Ley [Leyes,	ESPAÑA	10.Ordenar	Est. Tip. de los Hij
2	sa y dijo a los del bando contrario: - ¡Vaya unos gatos más buenos que compra e	1909	Baroja, Pío	Zalacaín el	ESPAÑA	12.Relato e	Ricardo Senabre,
3	nada. Ellos riñen, en el interior, como perros y gatos, pero le dejan a uno en paz	1909	Baroja, Pío	Zalacaín el	ESPAÑA	12.Relato e	Ricardo Senabre,
4	e nos hizo dueños de los tejados, pues ni aun los gatos se atrevían a andar por el	1902	Pérez Galdo	Las tormen	ESPAÑA	12.Relato e	Biblioteca Virtual
5	la villa, y alejando de mi aposento la caterva de gatos y perros que en la casa ter	1902	Pérez Galdo	Las tormen	ESPAÑA	12.Relato e	Biblioteca Virtual
6	casa de fraile donde reinaba una vieja rodeada de gatos: ¡Tac-tac! ¡Tac-tac! Era l	1905	Valle-Inclár	Sonata de i	ESPAÑA	12.Relato e	Leda Schiavo, Esp
7	ombrece, y de puro nervioso echo chispas como los gatos. ¡Miseria, nulidad de	1905	Pardo Bazá	La Quimera	ESPAÑA	12.Relato e	Marina Mayoral, (
8	que descubre el estado de sus nervios. Somos dos gatos pelo arriba, dos sistema	1905	Pardo Bazá	La Quimera	ESPAÑA	12.Relato e	Marina Mayoral, (
9	o de la tierra de promisión. En Madrid, hasta los gatos hacen la naveta, * van y v	1905	Pardo Bazá	La Quimera	ESPAÑA	12.Relato e	Marina Mayoral, (
10	. Se echaban todos los cerrojos, se recogían los gatos, los perros, los asnos, y m	1908	Larreta, En	La gloria de	ARGENTINA	12.Relato e	Victoriano Suárez
11	aba cada vez que se le acercaba uno de los muchos gatos que la rondaban requi	1903	Bobadilla, I	A fuego len	CUBA	12.Relato e	Biblioteca Virtual
12	a la luna. Al cerrar las maderas vio un rimero de gatos rodar por las tejas, arañár	1903	Bobadilla, I	A fuego len	CUBA	12.Relato e	Biblioteca Virtual
13	aullaba por las calles solitarias y dormidas. Los gatos se paseaban por las aceras	1903	Bobadilla, I	A fuego len	CUBA	12.Relato e	Biblioteca Virtual
14	ta-, hay que echar los hijos al arroyo como a los gatos. ¡Qué ideas tan originales	1903	Bobadilla, I	A fuego len	CUBA	12.Relato e	Biblioteca Virtual

Imagen 5. Ejemplificación de las concordancias descargadas para 'gatos'

4.5. Reflexiones finales

En conclusión, el presente artículo tiene como objetivo contribuir a la optimización del trabajo de recuperación y almacenamiento de datos lingüísticos. Para ello, se ha diseñado una propuesta que se sirve de las herramientas que ofrece la lingüística computacional para integrar Selenium y Python en el análisis de una base de datos de referencia como el CORDE. La automatización de la extracción de datos lingüísticos en este corpus reduce significativamente el tiempo y minimiza los errores que podrían cometerse a la hora de recoger datos, de modo que tratamos de contribuir metodológicamente en la mejora de los estudios a gran escala, ya que, hasta la fecha, las concordancias se recuperan mediante métodos manuales.

Gracias a la capacidad de esta herramienta para procesar información de manera estructurada, los lingüistas y filólogos pueden obtener y conservar los resultados más rápido. La utilización

del código para la extracción de concordancias del CORDE puede constituir una herramienta útil que, puesta a disposición del investigador, ayude a reducir el tiempo invertido en el almacenamiento de los testimonios y a agilizar la descarga de forma automática. Al tratarse de una página dinámica, en la que no es posible acceder a toda la información almacenada en el *backend*, el *web scraping* se ha llevado a cabo sobre la parte visual de la aplicación (*frontend*) por medio de diferentes funciones y expresiones regulares (*regex*), cuyo objetivo principal consiste en encontrar diferentes patrones para poder fragmentar en columnas la información. No obstante, se trata de un código en proceso constante de modificación, de ahí que esté almacenado en el repositorio de GitHub, donde otros expertos en lingüística computacional pueden modificarlo y corregir posibles errores para perfeccionar su funcionamiento. Por tanto, su uso debe ir acompañado de una revisión manual y minuciosa para comprobar si se han almacenado correctamente todos los testimonios, y si están distribuidos en los campos dispuestos en el Excel (N.º, Concordancia, Autor, Título, País, Tema y Publicación).

Sea como fuere, a pesar de los retos que implica su implementación en una página dinámica como el CORDE, el uso de herramientas de *web scraping* representa un paso adelante en la consolidación de metodologías digitales aplicadas a la investigación de la lengua española.

Queda, como reto futuro, el diseño de una herramienta más accesible y amigable que permita a cualquier usuario, incluso sin conocimientos técnicos, beneficiarse del empleo de este *script*.

4.6. Referencias bibliográficas

Acevedo-Castiblanco, Jorge-Alexander; Suárez-Barón, Marco-Javier; Gonzalez-Sanabriaz, Juan-Sebastián (2023): «Categorización e integración de columnas de opinión contenido en páginas web aplicando técnicas de Procesamiento de Lenguaje Natural». *Ingeniería y Competitividad*, *25*(3), 1-14. https://doi.org/10.25100/iyc.v25i3.13220

Blas Arroyo, José Luis; Puertas Ribés, Elia (2024): «INCOM: Un corpus de inmediatez comunicativa para el estudio sociolingüístico del español en su historia». *Cultura, Lenguaje y Representación*, 33, 7-29. https://doi.org/10.6035/clr.7200

Calderón Campos, Miguel (2019): «Los corpus del español clásico y moderno: Entre la filología y la lingüística computacional». *Revista de lingüística teórica y aplicada*, *57*(2), 41-64. https://dx.doi.org/10.4067/S0718-48832019000200041

Calderón Campos, Miguel; Vaamonde Dos Santos, Gael (2020): «Oralia diacrónica del español: un nuevo corpus de la edad moderna». *Scriptum digital*, 9, 167-189. http://hdl.handle.net/10481/64696

Celi Párraga, Ricardo Javier; Boné Andrade, Miguel Fabricio, Mora Olivero, Aldo Patricio (2023): *Programación Web del Frontend al Backend*. Editorial Grupo AEA.

CHARTA = Red CHARTA (2015-2024): *Corpus hispánico y americano en la red: textos antiguos [CHARTA]*. https://corpora.uah.es/charta/ [consulta: 12/08/2025].

CORDE = Real Academia Española: *Corpus diacrónico del español* [en línea]. http://www.rae.es [consulta: 08/09/2025].

CORHEN = Torrens Álvarez, María Jesús (dir. y ed.) (2016-): *Corpus Histórico del Español Norteño [CORHEN]*. http://corhen.es/ [consulta: 03/09/2025]. ISSN 2530-0938

CorLexIn = Morala Rodríguez, José R. (dir): *Corpus Léxico de Inventarios (CorLexIn)*. http://web.frl.es/CORLEXIN.html [consulta: 08/09/2025].

CORPES XXI = Real Academia Española: *Corpus del Español del Siglo XXI* [en línea]. http://www.rae.es [consulta: 08/09/2025].

Díaz Bravo, Rocío (2018): «Las Humanidades Digitales y los corpus diacrónicos en línea del español: problemas y sugerencias». En: Bocanegro, Lidia y Romero-Frías, Esteban (eds.), *Ciencias sociales y Humanidades Digitales aplicadas. Casos de estudio y perspectivas críticas*, Granada y Nueva York, Universidad de Granada y Downhill Publishing. http://hdl.handle.net/10481/63214

Enrique Arias, Andrés (2012): «Dos problemas en el uso de corpus diacrónicos del español: perspectiva y comparabilidad». *Scriptum digital: revista de corpus diacrònics i edició digital en llengües iberoromàniques*, 1, 85-106.

INCOM = Grupo de investigación Sociolingüística (UJI): Corpus INCOM (Corpus de textos de inmediatez comunicativa). https://auth.sketchengine.eu/ [consulta: 09/10/2025].

Jurafsky, Dan; Martin, James H. (2021): *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. https://web.stanford.edu/~jurafsky/slp3/

Martínez, Roxana; Rodríguez, Rocío; Vera, Pablo; Parkinson, Christian (2019): «Análisis de técnicas de raspado de datos en la web - Aplicado al portal del estado nacional argentino». *XXV Congreso Argentino de Ciencias de la Computación*, 457-466. https://sedici.unlp.edu.ar/bitstream/handle/10915/91026/Documento_completo.pdf?sequence=1

ODE = Calderón Campos, Miguel; García-Godoy, María Teresa (2019-): *Oralia Diacrónica del Español (ODE)*. [Consulta: 10/10/2025]. http://corpora.ugr.es/ode

OSTA = Gago Jover, Francisco; Pueyo Mena, F. Javier (2025): *Old Spanish Textual Archive* [v 3.0]. Hispanic Seminary of Medieval Studies. On line at https://osta.oldspanishtextualarchive.org [consulta: 15/09/2025]

Hernández Tadeo, Alexis; Gómez Vásquez, Edy; Berdejo Rincón, César A.; Montero García, Jorge; Calderón Maldonado, Adrián; Ibarra Orozco, Rodolfo (2015): «Metodologías para análisis político utilizando web scraping». *Resarch in Computing Science*, 95, 113-121. https://rcs.cic.ipn.mx/2015_95/Metodologias%20para%20analisis%20politico%20utilizando%20Web%20Scraping.pdf

Post Scriptum = CLUL, ed. (2014): *P.S. Post Scriptum. Arquivo Digital de Escrita Quotidiana em Portugal e Espanha naÉpoca Moderna*. http://ps.clul.ul.pt [consulta: 15/09/2025]

Rojo, Guillermo (2010). «Sobre la codificación y explotación de corpus textuales: otra comparación del Corpus del español con el CORDE y el CREA, *Lingüística*, 24, 11-50.

Rojo, Guillermo (2016): «Corpus textuales del español». En Gutiérrez Rexach, Javier (coord.), *Enciclopedia de Lingüística Hispánica*, 2, 285-296.

Selenium. (2025, abril 6): *Documentation*. Recuperado dehttps://www.selenium.dev/documentation/

Salve Cuenca, Alejandro (2025): «El papel de las herramientas de web scraping en la enseñanza del análisis del discurso digital». En Hidalgo Downing, Raquel (coord.), *Tecnologías*

digitales aplicadas a la enseñanza de lenguas y a la lingüística, 303-325.

Torruella Boix, J.; Llisterri Casañas, J. (1999): «Diseño de corpus textuales y orales». *Filología e informática: nuevas tecnologías en los estudios filológicos*, 45-81. https://fegalaz.usc.es/~gamallo/aulas/lingcomputacional/biblio/LinguisticaDeCorpus.pdf

Vaamonde, Gael (2015): «Limitaciones en el uso de corpus diacrónicos del español: Nuevas aportaciones desde el proyecto de investigación "Post Scriptum" », *E-Aesla*, 1, 1-10.