Facultad de Ingeniería - UDeLaR

Proyecto de grado

Aplicaciones Lúdicas de Soporte a la Enseñanza de Lenguas

Autores:
Alejandro Tosi
Analía Percovich

Tutores:
Aiala Rosá
Luis Chiruzzo

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito para la obtención del título en Ingeniería en Computación

Grupo de Procesamiento de Lenguaje Natural
Instituto de Computación

24 de junio de 2019
Montevideo, URUGUAY

«*Education is the most powerful weapon which you can use to change the world.*»

Nelson Mandela

Agradecimientos

Dedicamos este trabajo a nuestras familias y amigos que siempre nos apoyaron a lo largo del proyecto, estuvieron presentes en esos momentos de frustración, agradecemos por comprender la falta de tiempo y motivarnos todos los días para seguir adelante.
Un grato reconocimiento a las empresas de las que formamos parte, a nuestros compañeros de trabajo por la flexibilidad y comprensión.
Agradecemos a todas las personas de ANEP que participaron, en especial a Aldo Rodríguez, Valentina Dubini, a los maestros Johanna Revetria y Gerardo Saracho, y a las escuelas rurales que nos recibieron con excelente disposición.
Un especial agradecimiento a nuestros tutores Aiala Rosá y Luis Chiruzzo, que nos han guiado durante todo el proceso siempre con dedicación y motivación.

Resumen

En nuestro país se ha establecido la enseñanza del inglés como primera lengua extranjera obligatoria a introducirse desde la Educación Primaria. A partir de esta resolución fue creado el programa "Ceibal en inglés" para que todos los escolares que no tienen la posibilidad de tener un docente de inglés en el aula, reciban clases de inglés. Este programa consiste en enseñar el idioma a través de videoconferencias con un docente (que puede encontrarse en el extranjero) y el apoyo del maestro del aula.

Dado que en las zonas rurales de nuestro país hay baja conectividad a Internet, la solución de las clases de inglés por videoconferencias no es viable para muchas escuelas rurales. Es por esto que el Programa de Políticas Lingüísticas (PPL) de la ANEP ideó una nueva metodología llamada *e-coaching* que consiste en la capacitación de forma continua del maestro del aula a través de clases, materiales, herramientas, etc., para que él mismo pueda enseñar el idioma a sus alumnos.

En el marco de una colaboración entre el Programa de Políticas Lingüísticas y el Grupo PLN de la FIng se propuso la construcción de aplicaciones lúdicas que no requieran de acceso a internet y den soporte a la enseñanza del inglés en todas las escuelas. La propuesta nace a partir de estudiar las diferentes herramientas y técnicas que brinda el área de PLN en la enseñanza, y cómo los juegos didácticos son considerados un excelente recurso a la hora de aprender otro idioma.

En este trabajo se implementó una aplicación web fácil e intuitiva que permite a los docentes y alumnos generar tres tipos de juegos sin conexión a internet y con información ya procesada. Los juegos implementados son: crucigramas, sopas de letras y batalla naval. Además, la aplicación cuenta con un módulo extra para los docentes, el cual les permite generar tableros de crucigramas a partir de un texto trabajado en clase y corregir los tableros generados en caso de que contenga errores.

Para llevar a cabo la aplicación, fue necesario implementar una serie de recursos apoyándonos en técnicas y herramientas de PLN, como por ejemplo: un extractor de definiciones, un segmentador de oraciones, entre otras. También se realizaron visitas a escuelas rurales para mostrar la aplicación con el fin de mejorar la experiencia de los usuarios y se participó de eventos organizados por ANEP para presentar este trabajo.

Palabras claves: procesamiento del lenguaje natural, PLN en la enseñanza, juegos didácticos, extracción de defniciones, word embedding, segmentador de oraciones

Índice general

Capítulo 1

Introducción

Hoy en día, nuestra forma de comunicarnos se ha visto cada vez más condicionada por el dominio del idioma inglés debido a la globalización y a los nuevos canales de comunicación que se han ido desarrollando alrededor del mundo. Esta lengua es una de las más usadas en la actualidad por el campo económico, la industria, el turismo, la tecnología, etc., y muchas veces se hace referencia a ella como el "idioma global" o "la lengua franca" de la era moderna (Quezada Narvaéz, 2011). En la actualidad aprender este idioma se ha vuelto un tema de necesidad para desenvolverse cómodamente tanto en el ámbito personal como en el laboral, y cada vez más personas lo eligen como lengua extranjera.

En particular en nuestro país, el Consejo Directivo Central (CODICEN) ha establecido la enseñanza del inglés como primera lengua extranjera obligatoria a introducirse desde la educación primaria. Es por esto que, como forma de responder a esta resolución, la Administración Nacional de Educación Pública (ANEP) ha puesto en práctica diversas modalidades para su enseñanza que van desde la forma tradicional, que consiste en un profesor de inglés que dicta el curso en forma presencial, hasta una modalidad más atípica llevada a cabo a través de un sistema de videoconferencias instalado en las escuelas aprovechando los recursos tecnológicos.

Esta última modalidad es un programa iniciado en 2012 conocido como "Ceibal en inglés" que tiene como objetivos generales democratizar el acceso al inglés como lengua extranjera a los niños en las escuelas y ofrecer la formación en ella a los maestros de la ANEP. Como objetivos específicos propone alcanzar la universalización de la enseñanza de este idioma en un principio para cuarto, quinto y sexto grado de educación primaria en sus dos modalidades (presencial o por videoconferencia) y para el año 2030 alcanzar un nivel de lengua A2[1] (o Inicial Avanzado, según el Marco Común Europeo de Referencia para las Lenguas (*CEFR*)) para todos los niños que egresan de Primaria (Brovetto, 2013).

Actualmente este programa se encuentra en todas las escuelas de Montevideo y en algunas escuelas del interior y propone una forma alternativa y complementaria de la enseñanza del inglés a través de la tecnología de videoconferencias, es decir, una comunicación en tiempo real de audio y video a través de internet realizada entre el maestro del aula y sus alumnos con un docente de inglés que se puede encontrar en nuestro país o en el extranjero.

[1] El nivel A2 se adquiere cuando el estudiante es capaz de comprender y comunicarse en términos sencillos sobre cuestiones que le son conocidas o habituales como información básica sobre sí mismo y su familia, lugares de interés, compras, ocupaciones, etc. Además sabe describir en términos sencillos aspectos de su pasado y su entorno así como cuestiones relacionadas con sus necesidades inmediatas.[14]

Muchos maestros sin conocimientos o con conocimientos mínimos del idioma optan por este método, por un lado, para que puedan capacitarse en la lengua aprovechando al docente remoto y por otro, para que que sus alumnos puedan aprender inglés otorgándoles igualdad de oportunidades para acortar la brecha entre los que pueden obtener una educación en esta lengua y los que no (Fregossi, 2014).

Dado que en las escuelas rurales de nuestro país hay baja conectividad a Internet, la solución de las clases de inglés por videoconferencias para maestros con poco conocimiento de inglés no es viable. Es por esto que, a partir de 2017, se ideó una nueva metodología llamada *e-coaching* o *entrenamiento no presencial* que consiste en la capacitación del maestro del aula a través de clases de apoyo continuas, llevadas a cabo por personas coordinadoras del Programa de Políticas Lingüísticas (PPL) de la ANEP. Esta capacitación se da por distintas vías de comunicación, donde se brinda soporte al docente, por ejemplo, para planificar el programa a dictar, para la introducción de nuevo material, ayuda en la corrección de las tareas de los alumnos, etc., y así el maestro del aula pueda dar él mismo las clases de inglés a sus alumnos además del programa curricular.

Esta nueva modalidad presenta un gran desafío para los involucrados, ya que es corriente que en las escuelas rurales la enseñanza sea *multigrado*, es decir, se da simultáneamente a alumnos que pertenecen a distintos cursos en un mismo salón.

Fue entonces que, para apoyar la metodología de *e-coaching* y a los docentes que la implantan, en el marco de una colaboración entre el PPL de ANEP y el Grupo de Procesamiento del Lenguaje Natural (PLN) de la Facultad de Ingeniería (FIng) se propuso la construcción de aplicaciones lúdicas que no requieran de acceso a internet y den soporte a la enseñanza de inglés a escolares, y de esa forma complementar las actividades y materiales del docente.

En un estudio cualitativo (Målgren y Ledin, 2012) de docentes de lenguas extranjeras y sus pensamientos sobre el juego como herramienta en el aula, se afirma que una de las maneras que tienen los docentes para facilitar el aprendizaje a sus estudiantes en adquirir una lengua nueva es el juego y esto puede consistir en actuar, jugar y cantar. Los estudiantes adquieren el conocimiento de la segunda lengua fijando su conocimiento en la mente muchas veces sin describir la estructura, ni la gramática. Este es un método que ayuda a todos los estudiantes porque es divertido y menos formal. Es trascendental que los docentes de inglés incluyan en sus clases desde preescolar diferentes juegos que de acuerdo con la edad de sus estudiantes y los tópicos estudiados en clase, les faciliten el aprendizaje de la misma.

La propuesta nace a partir de que el conocimiento y los recursos existentes de PLN, los cuales ya se aplican para todo tipo de actividades, como la traducción automática, transcriptores de voz a texto, análisis de sentimientos, asistentes digitales, entre muchas otras, son ideales para utilizar en el área de la enseñanza de lenguas.

1.1. Análisis del problema

El primer desafío para este proyecto es la elección de juegos a implementar, los cuales deben ser desarrollados aplicando técnicas de PLN sobre textos con un nivel de inglés apropiado y con reglas de juego simples. Una vez seleccionados los juegos,

debemos obtener material en inglés para llevarlos a cabo, esto implica recolectar una gran cantidad de textos en inglés con un nivel adecuado para escolares, por ejemplo con un nivel A1[2] del *CEFR* o, por lo menos, similar.

Luego de obtener el material debemos elegir las herramientas y técnicas que brinda el área de PLN para obtener los resultados deseados y poder impactarlo en los juegos a desarrollar. Es importante en este paso tener en cuenta que la aplicación debe apuntar a una independencia de la conexión a internet debido a que se puede estar en lugares de baja conectividad, por lo que se hace foco en que sea auto-contenida.

Dado el tipo de usuario de las aplicaciones, también se considera importante que su interfaz sea lo más simple posible haciendo énfasis en la usabilidad. Para esto, es relevante la visita a algunas escuelas para que los niños puedan ir probando y evaluando la aplicación lo cual permitirá que el producto final sea de buena calidad.

1.2. Objetivos

Los objetivos del proyecto son:

- Investigar sobre recursos disponibles necesarios para el desarrollo de aplicaciones lúdicas de enseñanza de inglés. Dichos recursos incluyen materiales como *corpus* de textos en inglés y herramientas de apoyo de PLN disponibles para el procesamiento de los mismos. La investigación también implica el estudio de trabajos de PLN aplicados a la enseñanza, así como el estado del arte de las herramientas y técnicas a utilizar.
- Diseñar y desarrollar juegos en base a los conocimientos adquiridos en la etapa de investigación. Se toma como punto de partida el desarrollo de un generador automático de crucigramas, para luego sumar otro conjunto de aplicaciones, las cuales no dependan de conexión a internet para su uso y se rijan por reglas fáciles para su resolución.
- Tener experiencias personales con involucrados en la enseñanza primaria, incluyendo coordinadores, maestros y alumnos, con el fin de mejorar la experiencia de usuario de las aplicaciones.
- Crear un producto instalable en las computadoras de los maestros y alumnos para que sea efectivamente utilizado como apoyo a la enseñanza del inglés.

[2] El nivel A1 se adquiere cuando el estudiante es capaz de comprender y utilizar expresiones cotidianas de uso muy frecuente así como frases sencillas destinadas a satisfacer necesidades de tipo inmediato.[14]

1.3. Cronología del Proyecto

En el cuadro 1.1 se presenta una línea de tiempo con la tareas realizadas e hitos del presente proyecto.

Cuadro 1.1 Cronología

Marzo-Abril de 2018	Comienza el proyecto con un estudio bibliográfico sobre PLN para enseñanza de lenguas, en particular, aplicaciones lúdicas educativas. Se estudian posibilidades de adaptación de generador de crucigramas realizado en proyecto de grado anterior. Se realizan reuniones con coordinadores del Programa de Políticas Lingüísticas de la ANEP para definir visitas a escuelas y asistencias a eventos del año lectivo.
Mayo de 2018	Se comienza a implementar el Extractor de definiciones con análisis de constituyentes y el módulo generador de crucigramas.
Junio de 2018	Se visita a una escuela de San José. Se presenta la primera versión de la aplicación.
Julio de 2018	Se evalúa y construye la versión final del generador de crucigramas. Se empieza a trabajar con análisis de dependencias y se desarrolla la funcionalidad *on-demand*. Se asiste al evento *WinterCamp* de ANEP organizado por el PPL, donde se presenta el avance del proyecto.
Agosto de 2018	Se visita a escuela de Paysandú para evaluar el uso de la herramienta. Se empiezan a utilizar la técnica de Word Embedding para la generación de recursos para las sopas de letras. Se experimenta con el Extractor de definiciones y se comienza con la evaluación de los resultados.
Setiembre de 2018	Comienza el desarrollo del módulo de sopas de letras. Se implementan mejoras del Extractor de definiciones en base al análisis de sus resultados.
Octubre de 2018	Se presenta la herramienta en el *11º Foro de Lenguas* de ANEP. La misma cuenta con los módulos de crucigramas y sopas de letras. Se comienza a trabajar con *Word Embedding* para mejorar la calidad de las definiciones obtenidas.
Noviembre de 2018	Se desarrolla el segmentador de oraciones para generar recursos para la batalla naval. Se comienza con la implementación del módulo para la generación de tableros de dicho juego. Se continúa con mejoras en el Extractor. Se experimenta con heurísticas para la obtención de definiciones más correctas
Diciembre de 2018	Se finaliza de la implementación de la herramienta.
Enero-Abril de 2019	Se comienza con la escritura del informe.
Mayo de 2019	Culmina la escritura del informe. Se prepara la presentación y se lleva a cabo la defensa del proyecto.

1.4. Organización del documento

El presente informe se organiza de la siguiente forma:

Capítulos

- Capítulo 1: Contiene la motivación del proyecto, un análisis del problema a resolver, se plantean los objetivos definidos y se presenta una cronología del proyecto.
- Capítulo 2: Se presenta el estado del arte en los campos de conocimiento relacionados a este trabajo.
- Capítulo 3: En este capítulo se describen los juegos a implementar y sus reglas, se presentan los recursos iniciales con los que cuenta el equipo para comenzar a trabajar y se plantea un esquema con la solución propuesta.
- Capítulo 4: Se exponen los recursos generados por el equipo para cumplir con los objetivos propuestos.
- Capítulo 5: Describe los juegos implementados, el funcionamiento de la aplicación y el diseño de la interfaz.
- Capítulo 6: Contiene las experiencias vividas por el equipo, a través de las visitas a escuelas rurales y asistencias a eventos a cargo de ANEP.
- Capítulo 7: Concluye el trabajo realizado tras la implementación, experimentación y evaluación de la herramienta. Además se plantea el posible trabajo a futuro.

Apéndices

- Apéndice A: Contiene el glosario con las abreviaciones y conceptos utilizados a lo largo del documento.
- Apéndice B: Contiene información extra sobre recursos iniciales o generados.
- Apéndice C: Se describen brevemente las tecnologías utilizadas para crear la aplicación y sus recursos.
- Apéndice D: Se describen los algoritmos que fueron descargados y modificados acorde a las necesidades de cada juego.

Capítulo 2

Estado del Arte

En este capítulo se desarrolla el marco teórico del proyecto. Se describen las técnicas utilizadas durante el proceso y se hace referencia a algunos trabajos relacionados.

2.1. Introducción

El Procesamiento del Lenguaje Natural (PLN) es una rama de la Ciencia de la Computación, Inteligencia artificial y la Lingüística a la que le concierne la interacción entre computadoras y el lenguaje natural humano (Reshamwala, Pawar y Mishra, 2013).

Muchas empresas y universidades, incluyendo algunas de gran renombre internacional, están impulsando la investigación en el área. El PLN es utilizado para una gran variedad de tareas, incluyendo entre muchas otras:

- **Extracción de Información (IE[1])**: Es la tarea de extraer información de texto no estructurado, semiestructurado o estructurado, pero legible por una máquina (ver sección 2.2.3). (Singh, 2018, Cui, Wei y Zhou, 2018)
- **Búsqueda de Respuestas (QA[2])**: Se centra en construir sistemas que respondan automáticamente las preguntas planteadas por los seres humanos en un lenguaje natural. (Kundu y Ng, 2018, Jangho Lee et al., 2017)
- **Traducción Automática (MT[3])**: Es la tarea de convertir automáticamente un lenguaje natural a otro, preservando el significado del texto de entrada y produciendo un texto fluido en el idioma de salida. (Lample et al., 2018, Shaw, Uszkoreit y Vaswani, 2018)

La ejecución de forma automática o semi-automática de tareas como las anteriores tiene su grado de dificultad, por lo que se suelen dividir en subtareas organizadas en una secuencia de pasos, llamada *pipeline*, que suele tener como entrada un texto plano[4] en lenguaje natural.

[1] Por sus siglas en inglés: Information Extraction
[2] Por sus siglas en inglés: Question Answering
[3] Por sus siglas en inglés: Machine Translation
[4] Texto que no está etiquetado computacionalmente, especialmente formateado o escrito en código. [3]

A modo de ejemplo [17], en la figura 2.1 se ilustra y detalla un posible *pipeline* de PLN.

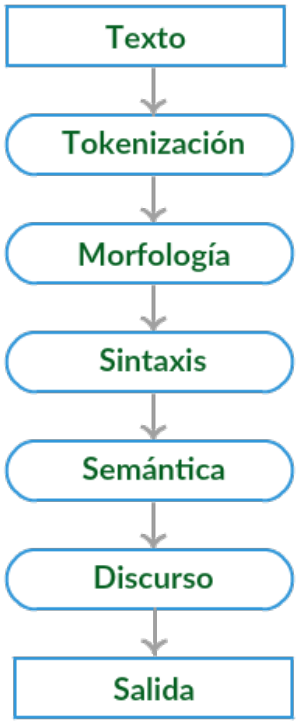


FIGURA 2.1: Ejemplo de pipeline de PLN

- La **tokenización** es el proceso de segmentar un texto en tokens (palabras, términos u otros).
- La **morfología** estudia la categorización de las palabras de acuerdo a su *POS-Tag* (ver sección 2.2.1), y los procesos que intervienen en la derivación y composición de las palabras[15].
- La **sintaxis** es el estudio de las relaciones formales entre palabras (Bose, 2004), y en PLN, el análisis sintáctico se utiliza para evaluar cómo se alinea el lenguaje natural con las reglas gramaticales.
- La **semántica** es el estudio del significado de las expresiones lingüísticas (Bose, 2004). Dentro del PLN, el análisis semántico involucra la aplicación de algoritmos para comprender el significado y la interpretación de las palabras y cómo se estructuran las oraciones.
- El análisis de **discurso** estudia grupos estructurados coherentes de oraciones. Los monólogos y diálogos son tipos de discurso[4].

2.2. Técnicas y recursos para la obtención y el procesamiento de textos

A continuación se introducen las técnicas y recursos relevantes para este trabajo.

2.2.1. POS tagging

El etiquetado gramatical (*POS tagging o part-of-speech tagging*) es el proceso que recibe un texto de entrada y devuelve la categoría gramatical para cada una de sus palabras, basado en su definición y contexto, donde las categorías gramaticales son sustantivo, verbo, adjetivo, etc.

FIGURA 2.2: Ejemplo de etiquetado gramatical

Además de la clasificación gramatical anterior, también devuelve otros rasgos de las palabras como por ejemplo número (singular, plural) y persona (primera, segunda o tercera).

Esta tarea requiere de un análisis morfológico previo y del contexto de una palabra para determinar la etiqueta correcta, desambiguando entre todas las posibilidades. Este proceso también recibe el nombre de análisis léxico.

2.2.2. Parsing

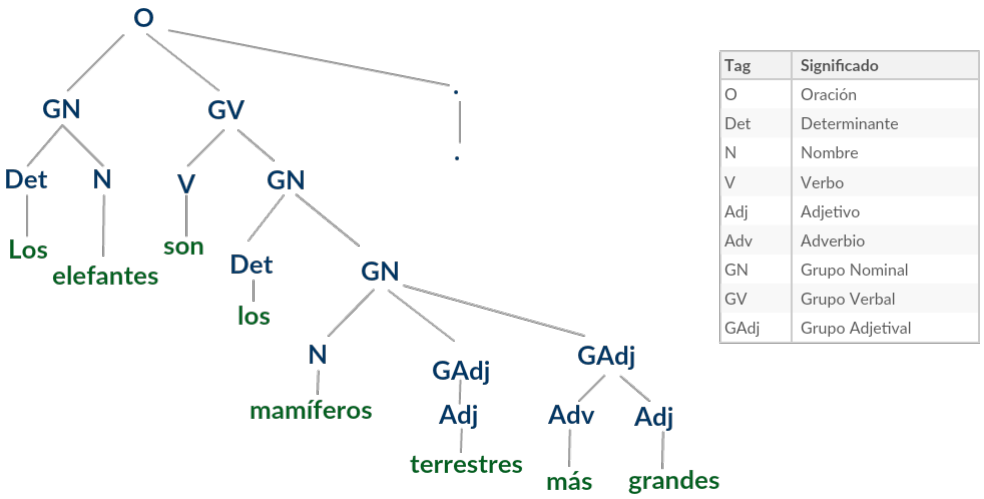
El análisis sintáctico o *parsing* hace referencia a las relaciones de concordancia y jerarquía que tienen las palabras cuando se relacionan entre sí. Existen distintos tipos de análisis sintácticos, a continuación detallaremos dos de ellos que fueron los utilizados en este proyecto:

Análisis de constituyentes

Un constituyente es una palabra o una secuencia de palabras que cumplen una función sintáctica dentro de la estructura jerárquica de una oración.

La idea de un constituyente es que una frase pueda ser dividida en subfrases de acuerdo a una gramática libre de contexto, si es posible. Esta gramática consiste en un conjunto de reglas de producción, cada cual indicando cómo se pueden agrupar los símbolos del lenguaje (Jurafsky y Martin, 2009).

El resultado de un análisis de constituyentes sobre una oración es una estructura denominada *árbol sintáctico de constituyentes*, como se muestra en la figura 2.3 para la frase *"Los elefantes son los mamíferos terrestres más grandes."*:



Tag	Significado
O	Oración
Det	Determinante
N	Nombre
V	Verbo
Adj	Adjetivo
Adv	Adverbio
GN	Grupo Nominal
GV	Grupo Verbal
GAdj	Grupo Adjetival

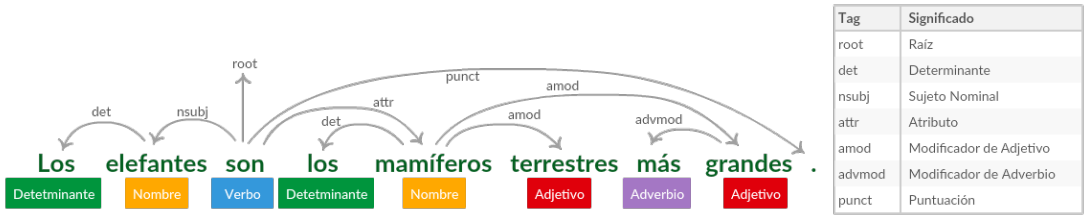
FIGURA 2.3: Ejemplo de árbol de constituyentes

Hay situaciones en que no es posible obtener un árbol de constituyentes o se obtiene más de uno debido a las ambigüedades que, en este tipo de análisis, no son fáciles de solucionar de forma automática.

Análisis de dependencias

El análisis de dependencias de una oración tiene como resultado un grafo o árbol que representa conexiones binarias entre las palabras llamadas *relaciones de dependencia*. A la palabra sintácticamente subordinada en una relación de dependencia se le denomina *dependiente* y a la palabra de la que depende, *padre*. El tipo de dependencia es la etiqueta que se le asocia a cada relación y que resume la información sintáctica que liga a la palabra subordinada con la subordinante (Jurafsky y Martin, 2009).

En la figura 2.4 se presenta el grafo de dependencias para la oración *"Los elefantes son los mamíferos terrestres más grandes"*.



Tag	Significado
root	Raíz
det	Determinante
nsubj	Sujeto Nominal
attr	Atributo
amod	Modificador de Adjetivo
advmod	Modificador de Adverbio
punct	Puntuación

FIGURA 2.4: Ejemplo de árbol de dependencias

Este tipo de análisis es muy útil para las tareas de simplificación de textos, análisis de sentimientos, etc. La principal diferencia con el análisis de constituyentes que se puede observar a partir de los dos ejemplos presentados, es que en el primero las palabras solo se encuentran en las hojas, mientras que en el segundo las palabras pueden estar en cualquier parte del árbol.

Esta característica aporta mayor complejidad al análisis de dependencias, aunque permite más facilidad de análisis en lenguajes que tienen más libertad en el orden de las palabras, por ejemplo: checo (Martínez, 2010). Esto explica el interés actual para la evaluación multilingüe de los programas de análisis de dependencias.

Otra ventaja del análisis de dependencias sobre el de constituyentes es que es más eficiente y rápido.

2.2.3. Extracción de información

La Extracción de Información (IE) es la tarea de obtener (de manera automática) información no estructurada y/o semi-estructurada que es legible por una computadora (Piskorski y Yangarber, 2012).

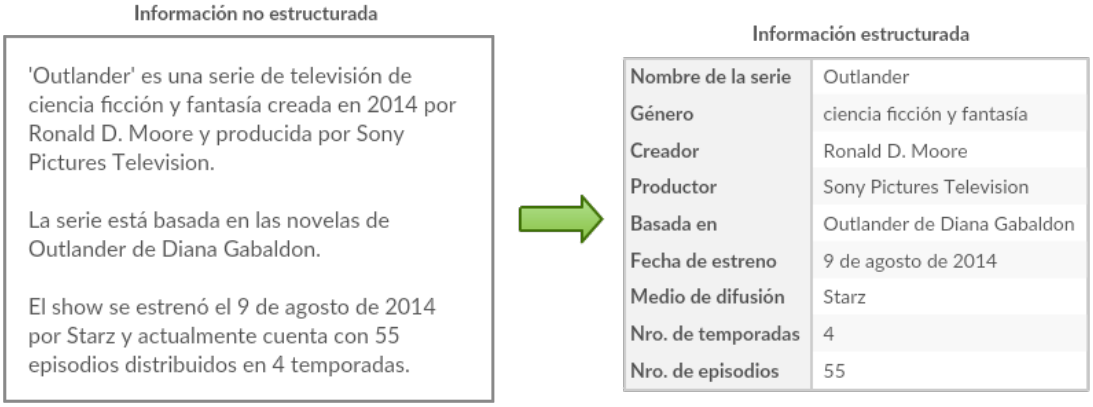


FIGURA 2.5: Ejemplo de extracción de información

El proceso de extracción de dicha información estructurada implica la identificación de ciertas estructuras a pequeña escala, como grupos nominales que denotan a una persona o un grupo de personas, referencias geográficas y expresiones numéricas, entre otras, y la búsqueda de relaciones semánticas entre ellas.

Las principales tareas de la extracción de información son:

- **Reconocimiento de entidades con nombre:** aborda el problema de la identificación y clasificación de tipos predefinidos de entidades nombradas, como organizaciones, personas, nombres de lugares, expresiones temporales, etc..
- **Resolución de correferencias:** requiere la identificación de múltiples menciones de la misma entidad en el texto.
- **Extracción de relaciones semánticas entre entidades:** es la tarea de detectar y clasificar relaciones predefinidas entre entidades identificadas en el texto. En la siguiente sub-sección se detalla la tarea de *Extracción de definiciones* que forma parte de esta categoría.
- **Extracción de eventos:** refiere a la tarea de identificar eventos y obtener información detallada y estructurada sobre ellos, idealmente identificando quién hizo qué, cuándo, dónde, a través de qué métodos y por qué.

A menudo, el proceso de **Extracción de Información** se confunde con el de **Recuperación de Información** (IR). La tarea de IR es seleccionar de una colección de documentos un subconjunto que sea relevante para una consulta en particular.

Los sistemas de IE son, en principio, más difíciles de construir e intensivos en conocimiento que los sistemas IR. Sin embargo, las técnicas de IE e IR pueden considerarse complementarias y potencialmente pueden combinarse de varias maneras. IR se usa a menudo en IE para filtrar previamente una colección de documentos muy grande a un subconjunto manejable, al que se pueden aplicar las técnicas de IE.

Alternativamente, el IE podría ser utilizado como un sub-componente de un sistema de IR para identificar estructuras para la indexación de documentos inteligentes.

Web scraping

El *web scraping* es una técnica de IR para extraer información de sitios web. El *scraping* de datos se enfoca en transformar el contenido no estructurado de un sitio web en datos estructurados los cuales pueden ser almacenados en una base datos. Al software programado para scrapear se le suele llamar *bot*, *spider* o *crawler*.

A lo largo de este proyecto se utilizó `Scrapy`[26] para implementar las *spiders* que extrajeron la información de los sitios web de interés.

Extracción de definiciones

La *extracción de definiciones* forma parte de la tarea más amplia de extracción de relaciones, que como ya fue mencionado, consiste en identificar relaciones semánticas entre entidades dentro de un texto. Las relaciones pueden ser entre palabras, frases o combinaciones de estas, y las extracciones pueden ser a nivel de oraciones (Hearst, 1992, Ortega Mendoza, 2007, Navigli y Velardi, 2010, Espinosa-Anke y Saggion, 2014, Esteche y Romero, 2015) o de múltiples oraciones (Gupta et al., 2019, He et al., 2018, Shwartz, Goldberg y Dagan, 2016).

Hearst, (1992) basa su trabajo de *extracción de hipónimos* (también parte de la extracción de relaciones) en la construcción léxico-sintáctica de patrones que pueden

detectar *hiponimia* en oraciones. Los patrones que utiliza son relativamente simples, en el sentido de que basta con identificar grupos nominales (abreviado en inglés como NP: *Noun Phrase*) y secuencias de palabras predefinidas siguiendo determinado orden. Un ejemplo de estos patrones es:

$$\boxed{NP_0 \textbf{ such as } \{NP_1,\ NP_2\ ...\ (\textbf{and}|\textbf{or})\}\ NP_n}$$

De esta forma, analizando si existen oraciones que cumplan con un patrón como el del ejemplo, se podrían extraer pares *«hipónimo,hiperónimo»*. Por ejemplo, la oración *"**Cool colors, such as blues and greens**, are popular choices for bedrooms and more relaxed home spaces because of their versatility."* cumple con el patrón mencionado (marcado en **negrita**). De esta oración se pueden extraer entonces los pares *«blues, Cool colors»* y *«greens, Cool colors»*.

La extracción de definiciones es una tarea muy similar a la de extracción de hipónimos o hiperónimos, y consiste en obtener a partir de textos una palabra a definir, el *definiendum*, y el enunciado que lo define, la *definición*. Para la tarea específica de obtención de pares «definiendum, definición», se puede decir que un paso previo sería la clasificación de oraciones en "definitorias" y "no definitorias" (Espinosa-Anke y Saggion, 2014). Una oración definitoria se le llama a aquella que define una palabra o un término. Para el español, ejemplos de estas oraciones pueden ser las que usan el verbo "ser" en algunas de sus conjugaciones, como la de la figura 2.6.

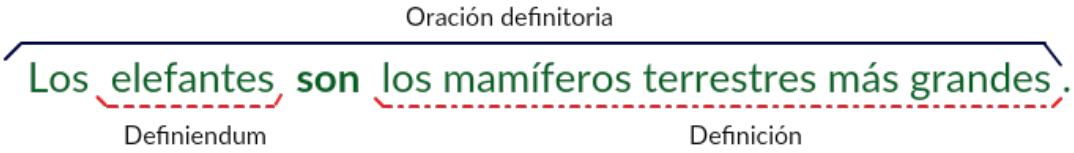


FIGURA 2.6: Ejemplo de oración definitoria

Lo que se encuentra en el ejemplo, es la presencia de un patrón que puede abstraerse como "determinante[5] <*definiendum*> **son** <*definición*>". Utilizando este patrón pueden encontrarse otras oraciones que potencialmente contengan un definiendum y su respectiva definición. La técnica de búsqueda por patrones puede realizarse:

1. Utilizando un conjunto inicial preestablecido de patrones para obtener **pares**.
2. En base a un conjunto preestablecido de pares, que puede incluir unos previamente obtenidos en *1.*, buscando luego nuevos **patrones** que asocian a dichos pares.

Hearst, (1992), Ortega Mendoza, (2007), y Esteche y Romero, (2015) trabajan aplicando los dos pasos, mientras que otros solo trabajan con un conjunto preestablecido de patrones (Przepiórkowski et al., 2007).

[5]Los determinantes definen una clase de palabra cuyos elementos determinan al sustantivo o al grupo nominal y se sitúan generalmente antes de un sustantivo. [24]

Algunos patrones son evaluados sobre representaciones resultantes de un análisis de constituyentes, otros se pueden definir como expresiones regulares (Przepiórkowski et al., 2007), o incluso se ha trabajado con combinaciones de ambos métodos (Esteche y Romero, 2015). También se ha trabajado en base a caminos o árboles de dependencias, siendo estas últimas representaciones de oraciones muy utilizadas en trabajos recientes que aplican métodos de aprendizaje automático (Shwartz, Goldberg y Dagan, (2016), He et al., (2018), Zhang, Qi y Manning, (2018), Gupta et al., (2019)).

La aplicación de métodos distribucionales para obtener información semántica como entrada a clasificadores ha dado buenos resultados en la extracción de relaciones. Por ejemplo Shwartz, Goldberg y Dagan, (2016), y He et al., (2018) utilizan *Word Embedding* (técnica que se detallará en la sección 2.2.5) y *Sentence Embedding* [19] respectivamente.

Navigli y Velardi, (2010) se basan en un enfoque distinto al de los patrones, trabajando con una variante de *Word Lattices*[6]. Utilizan este método tanto para extracción de definiciones como de hiperónimos.

2.2.4. Word Sense Desambiguation

Word Sense Desambiguation es la tarea de desambiguar el sentido de una palabra según el contexto en la que aparece. En definitiva, se puede suponer que palabras similares tienden a tener distribuciones contextuales similares (se detalla más sobre esto en la sección 2.2.5). El problema se puede abordar con varias metodologías, algunas de ellas se detallan a continuación.

Desambiguación supervisada: Está basada en un conjunto etiquetado de entrenamiento. Gale, Church y Yarowsky, (1992) entrenan con un corpus paralelo[7] en inglés y francés, tomando ventaja del hecho de que una palabra ambigua en un idioma no necesariamente lo es en otro. Construyen un *clasificador Bayesiano*[8] para la tarea de desambiguación y reportan una precisión de 90 %. Raganato, Jose Camacho-Collados y Navigli, (2017) realizan un estudio comparativo, concluyendo que el método de desambiguación supervisada da mejores resultados que métodos que usan bases de conocimiento. A continuación se detalla el conjunto de métodos que incluyen a este último.

Desambiguación basada en diccionarios: Se utilizan recursos léxicos como diccionarios, tesauros (por ejemplo: *WordNet*[34]) u otras bases de conocimiento. Lesk, (1986) se basa en la idea simple de que las definiciones de las palabras en un diccionario probablemente sean buenos indicadores de los significados que definen. Yarowsky, (1992) utiliza el tesauro *Roget's*[25] para resolver la tarea aplicando modelos estadísticos. Dagan y Itai, (1994) utilizan una metodología similar a la de Gale, Church y Yarowsky, (1992), en el sentido de que las palabras pueden ser desambiguadas buscando traducciones en otros idiomas. En este caso trabajan con textos en inglés, hebreo y alemán. Más recientemente, José Camacho-Collados, Pilehvar y Navigli, (2015) obtienen resultados de estado del arte utilizando *BabelNet* [2], una enciclopedia y red semántica multilingüe.

[6] Un grafo de palabras, o *word lattice* en inglés, es un grafo dirigido acíclico, o más formalmente, una variante de un autómata de estado finito (Dyer, Muresan y Resnik, 2008).

[7] Un corpus paralelo es aquel formado por textos en una lengua fuente con sus respectivas traducciones en una o más lenguas destino, es decir, cada texto posee su traducción correspondiente. [1]

[8] Clasificador probabilístico basado en el Teorema de Bayes.

Desambiguación no supervisada: Se utiliza un corpus de textos sin etiquetar. Schütze, (1998) utiliza un método estadístico Bayesiano similar al del trabajo de Gale, Church y Yarowsky, (1992), y el corpus fue construido con textos del periódico *New York Times*. Trabajos más recientes como el de Bartunov et al., (2015) utilizan un *dump* de *English Wikipedia* de 2010[31].

2.2.5. Modelado de Lenguaje

Se le llama *Modelado de lenguaje* a la tarea de predecir la siguiente palabra o caracter en un texto[11]. Un *modelo de lenguaje* es una representación del lenguaje natural construida con el fin de que sea "entendible" para una máquina [18], y son la base para poder realizar la tarea en cuestión.

Los trabajos de Harris, (1954) (donde el autor cuestiona si el lenguaje natural tiene una *estructura distribucional*), Firth, (1957) (conocido por su famosa cita *"You shall know a word by the company it keeps"*) y Deerwester et al., (1990) (que basan su trabajo en la teoría de *Latent Semantic Analysis*[9]) derivan en la llamada *Hipótesis distribucional*. Según lo afirman Levy y Goldberg, (2014), esta hipótesis fue la que inspiró la creación del conjunto de modelos de lenguaje y técnicas de aprendizaje de atributos conocida como *Word Embedding*.

Word Embedding

Para la construcción de los modelos de lenguaje asociados a *Word Embedding*, se utilizan palabras o frases de un vocabulario para crear **vectores** de números reales que permiten de alguna forma generar cierto grado de asociación entre una palabra en particular y el resto de las palabras del vocabulario. El modelo resulta en un espacio de dimensión baja respecto al tamaño del vocabulario (Berón y Jardim, 2017).

El conocimiento ha sido explotado en varias tareas, incluyendo la ya mencionada *Word Sense Desambiguation* (Brown et al., (1991)), el Análisis de Sentimiento (Howard y Rude, (2018), Liu et al., (2019), Cliche, (2017)) la Traducción Automática (Edunov et al., (2018), Wu et al., (2019)), entre otras.

Una propiedad que cumplen los vectores asociados a palabras que tienen características similares (según un contexto dado), es que se encuentran cercanas en el espacio, esto es, que la *similitud coseno* entre ellos es cercana a 1.

Fórmula de similitud coseno entre vectores

$$similitudCoseno(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{\|\vec{v}\|\|\vec{w}\|} = \frac{\sum_{i=1}^{n}\vec{v_i}\vec{w_i}}{\sqrt{\sum_{i=1}^{n}\vec{v_i}^2}\sqrt{\sum_{i=1}^{n}\vec{w_i}^2}}$$

[9] Análisis Semántico Latente [LSA por sus siglas en inglés] es una teoría y un método para extraer y representar el significado de uso contextual de las palabras mediante cálculos estadísticos aplicados a un gran corpus de texto [13]

Un problema que se puede dar es que el vector de una palabra **ambigua** podría no estar próximo a vectores que representan palabras con características similares, como pueden ser sinónimos de la palabra original, o hipónimos de un hiperónimo común.

Para ilustrar este problema, se presenta en la figura 2.7 un ejemplo. El vector asociado a *Bat*, que en inglés refiere a *murciélago* (hipónimo de animal) y al *Bat* deportivo, podría no estar muy próximo a los vectores de otros animales como *Lion* o *Whale* como sería deseado.

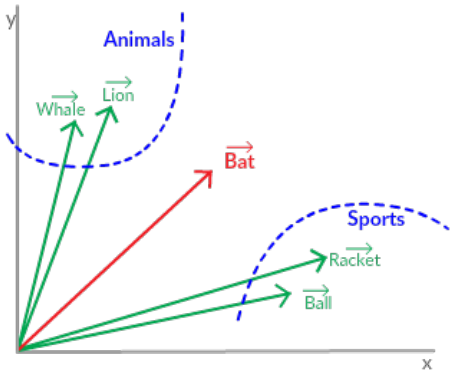


FIGURA 2.7: Ejemplo de ambigüedad de palabras en WordEmbedding

Otro problema que puede darse es que la frecuencia de las palabras en los corpus afecten a la construcción de sus representaciones vectoriales. Las palabras menos frecuentes tienen menos contextos que aquellas de mayor frecuencia. Trabajos como el de Qiu et al., (2014) y Gong et al., (2018) tratan este problema.

Existen varias técnicas para generar o "aprender" los vectores asociados a los *Word Embedding*, incluyendo *Word2Vec* (de Google), *GloVe* (de la universidad Stanford) y *FastText* (de Facebook)[33]. A continuación se detalla la técnica `Word2Vec` que fue la aplicada en este proyecto.

Word2vec

`Word2Vec` (Mikolov et al., 2013) está inspirada en una *red neuronal artificial* de dos capas que procesa texto. Su entrada es un corpus y su salida es un conjunto de vectores que representan las palabras de ese corpus. Si bien `Word2Vec` no es una red neuronal profunda[10], convierte el texto en una forma numérica que las redes profundas pueden entender.

Se utilizan dos métodos para analizar los textos y obtener los vectores:

- *Continuous bag of words (CBOW).*
- *Skip-Gram.*

El método basado en *CBOW* toma el contexto de cada palabra como entrada e intenta predecir la palabra correspondiente al contexto, mientras que el método *Skip-Gram* realiza el análisis de forma inversa, tomando una palabra y tratando de predecir el contexto en la que iría.

[10]Redes neuronales con varias capas intermedias y con varias neuronas en cada una

En la figura 2.8 se presenta un diagrama mostrando esta diferencia (Landthaler et al., 2017).

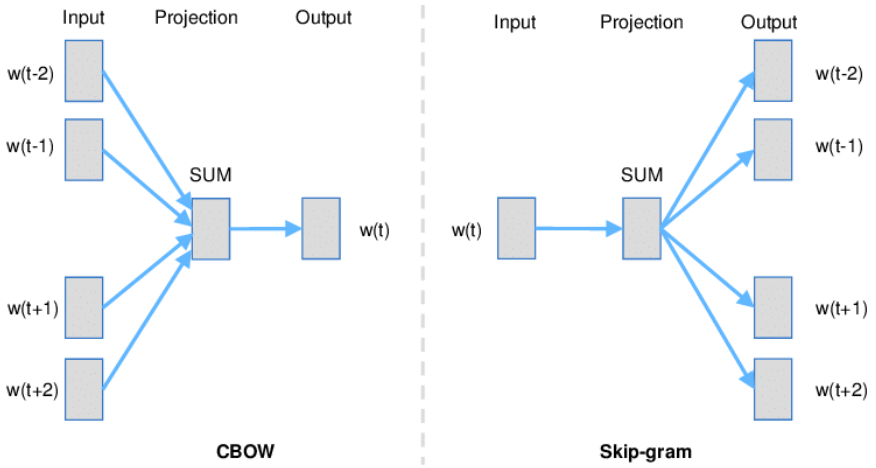


FIGURA 2.8: Comparación métodos CBOW y Skip-Gram

Comparando ambos métodos se puede decir que los dos tienen sus propias ventajas y desventajas. Según Mikolov, *Skip-Gram* funciona bien con una pequeña cantidad de datos, representa bien las palabras "raras" y devuelve mejores resultados especialmente en las relaciones semánticas. Por otro lado, *CBOW* es más rápido y tiene mejores representaciones para palabras más frecuentes.

2.3. PLN aplicado a la enseñanza

Aunque la educación es probablemente una de las áreas de aplicación más antiguas de la investigación de PLN (Litman, 2016), nuevos fenómenos como los MOOC[11], las redes sociales y *big data*[12] han provocado una explosión del interés actual en esta área, aumentando los vínculos entre los investigadores en PLN y otras áreas de Inteligencia Artificial.

Litman, (2016) plantea que la aplicación del procesamiento del lenguaje natural a la educación generalmente sigue un ciclo de vida iterativo (Figura 2.9).

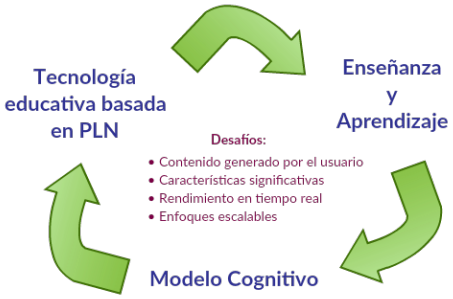


FIGURA 2.9: Ciclo PLN en la enseñanza - Adaptado del artículo de Litman, (2016)

Un problema de investigación en el área de PLN para aplicaciones educativas suele estar inspirado en las necesidades de un profesor o estudiante del mundo real (parte superior derecha de la figura). Por ejemplo, dada la enorme cantidad de alumnos e instructores en los MOOC, es difícil para un instructor leer todas las publicaciones en los foros de discusión e identificar las publicaciones que requieren la intervención de un instructor. A continuación, avanzando hacia el inferior de la figura en dirección de la flecha, las restricciones en las soluciones al problema se formulan teniendo en cuenta la teoría relevante o los hallazgos basados en datos de la literatura. Por ejemplo, incluso antes de los MOOC, había una literatura pedagógica sobre la intervención del instructor. Finalmente, avanzando hacia la parte superior izquierda del diagrama, se diseña, implementa y evalúa una tecnología basada en PLN. Todo esto basado en un análisis de error, el ciclo probablemente se repita.

Litman describe tres aspectos fundamentales del PLN los cuales ayudan a mejorar la tecnología educativa de varias maneras:

- Enseñar y aprender sobre temas relacionados con el lenguaje, por ejemplo, leer, escribir, hablar.
- Usar el lenguaje para enseñar cualquier tema, por ejemplo, enseñar en las disciplinas como historia, biología, etc..
- Procesar el lenguaje para satisfacer las necesidades de los estudiantes, profesores e investigadores, por ejemplo, material de estudios.

Como ejemplo del primer aspecto, el PLN se está utilizando para automatizar la calificación de los textos de los estudiantes con respecto a dimensiones lingüísticas como la corrección gramatical o la estructura organizativa. El análisis sintáctico se ha utilizado para detectar y corregir errores de escritura, como el uso incorrecto de las preposiciones para estudiantes de ESL[13] o estudiantes sordos (Michaud y Mccoy, 2006). El análisis

[11] Acrónimo de Massive Online Open Courses. Se trata de un curso a distancia, accesible por internet al que se puede apuntar cualquier persona y prácticamente no tiene límite de participantes.

[12] Término que hace referencia al concepto relativo a conjuntos de datos tan grandes y complejos como para que hagan falta aplicaciones informáticas no tradicionales de procesamiento de datos para tratarlos adecuadamente.

[13] Acrónimo de English as a Second Language

semántico se ha utilizado para evaluar el significado de las respuestas de los alumnos con respecto a las respuestas de referencia, tanto en la evaluación (Dzikovska et al., 2010) como en los niveles de análisis más generales, por ejemplo, calificar la coherencia de los ensayos de los estudiantes (Miltsakaki y Kukich, 2004).

Como ejemplo del segundo aspecto, las tecnologías de diálogo se utilizan para lograr los beneficios de la tutoría personal desarrollando herramientas para asistir a los estudiantes en sus tareas, especialmente en los dominios STEM[14], de una manera rentable y escalable.

La investigación realizada por Diane Litman se ha centrado en el diseño y la evaluación de un sistema de tutoría basado en el diálogo, ya que se ha demostrado que los estudiantes que trabajan individualmente con tutores humanos a menudo obtienen calificaciones más altas que los estudiantes que trabajan con un sistema de tutoría inteligente (ITS[15]). Para llevar esto a cabo, se aprendió de los corpus de diálogo de tutoría utilizando métodos basados exclusivamente en los datos (*data-driven methods*). Para respaldar su uso desarrollaron modelos probabilísticos de simulación de usuario y la representación de los diferentes escenarios, pudiendo así optimizar la elección de comportamientos del tutor pedagógico automatizado.

El tercer aspecto para el PLN en la educación es procesar de manera útil el texto y el habla de cualquier otra forma que pueda apoyar a los estudiantes y profesores, así como a los investigadores y desarrolladores de sistemas.

Principalmente con los maestros en mente, el PLN se está utilizando para intentar automatizar tareas que tradicionalmente han requerido un esfuerzo manual, por ejemplo, la creación de un plan de estudio, materiales de evaluación o el procesamiento de los intercambios que se dan en diferentes tipos de foros en donde estudiantes y docentes interactúan y se generan demasiados datos para procesarlos manualmente. La similitud semántica se muestra prometedora en la identificación de conceptos básicos de los recursos de educación en ciencias (Sultan, Bethard y Sumner, 2014), mientras que la simplificación del texto se está estudiando como un método para permitir la reutilización de materiales existentes en todos los niveles de competencia de los estudiantes (Candido Jr et al., 2009).

Para los estudiantes, el PLN se está utilizando para ayudarlos a navegar mejor en los materiales relacionados con textos y cursos basados en el habla, desarrollando herramientas que permiten a los estudiantes acceder y procesar mejor los materiales de conferencias en línea relacionados con el contenido de sus cursos (Glass et al., 2007).

Por otro lado, Alhawiti, (2014) plantea que la aplicación del PLN en la educación para el aprendizaje electrónico es un enfoque muy importante, ya que hoy en día hay varias fuentes electrónicas en línea disponibles que ayudan a los estudiantes y docentes a acceder a los materiales. Con respecto a esa disponibilidad, una preocupación importante está asociada con el aumento en el uso de blogs, Wikipedia y recursos no confiables. Esto requiere un procesamiento automático inteligente para evitar el uso de tales recursos no confiables y promover el uso de recursos auténticos.

[14] Acrónimo que se refiere a las áreas de conocimiento en las que suelen trabajar los científicos y los ingenieros: Science, Technology, Engineering and Mathematics.

[15] Por sus siglas en inglés, “Intelligent tutoring system“, es un sistema utilizado para la tutoría, que por lo general funciona sin la intervención de un profesor o tutor humano.

Como aplicaciones educativas innovadoras de PLN, cita a *Text Evaluator*[29], también conocida como Source Rater, que es una herramienta de mayor precisión para medir la complejidad de un texto, para utilizarlo en la creación de nuevos pasajes de comprensión de lectura y artículos. Esta herramienta une un conjunto grande de características cognitivas con sofisticados enfoques psicométricos para presentar categorizaciones de complejidad de texto que están altamente asociadas con la categorización proporcionada por educadores experimentados.

Language Muse[12] es otro ejemplo de aplicación educativa de PLN, que ayuda a los docentes a crear y asignar actividades de escritura y lectura que apoyan a los estudiantes de inglés y lectores con dificultades. Después de cargar texto, Language Muse lo procesa en un segundo y genera automáticamente actividades para seleccionar. También tiene una biblioteca de texto integrada y permite a los usuarios filtrarla por etiquetas. Los estudiantes pueden agruparse en clases, lo que les permite a los maestros gestionarlos de manera eficiente y hacer un seguimiento de su progreso.

Khaled Alhawiti también resalta que la aplicación del PLN en la educación también es efectiva para la minería de datos, la recuperación de información, la evaluación de la calidad y la evaluación de resultados.

En conclusión, el procesamiento del lenguaje natural y su aplicación en la educación brindan una posible solución a los diversos problemas y barreras del sistema educativo. Permite automatizar tareas que insumen una gran dedicación de tiempo a los docentes y además generar en forma automática herramientas didácticas y material educativo para los estudiantes, entre muchas otras cosas.

2.4. Los juegos didácticos para aprender inglés

Los juegos didácticos deberían ser parte de la vida de todo niño. No solo fomentan su capacidad cognitiva, sino que le ayudan a desarrollar distintas capacidades como la memoria, autoestima, concentración y el desarrollo social, entre otras (Castrillón Díaz, 2017).

El juego en el aprendizaje del inglés proporciona al estudiante un gran incentivo para aprenderlo y usarlo mejor, aumentan la motivación, y hasta suponen un cierto descanso de las formas tradicionales de enseñanza. Además, se considera un enfoque innovador y diferente de aprender el idioma.

Versi - School of English[32] plantea cinco razones por las que usan el juego en sus clases de inglés:

- **Promueven la motivación:** Los juegos ofrecen un incentivo para usar el inglés en la clase. Son divertidos y a los niños les gusta jugar. A través del juego los niños experimentan, descubren e interactúan con sus compañeros y su entorno.
- **Favorecen el aprendizaje significativo:** El juego forma parte de un proceso natural y significativo por el cual los niños construyen su propio aprendizaje, conocimientos y habilidades prácticas.
- **Desarrollan habilidades lingüísticas y sociales:** A través del juego los niños desarrollan diferentes destrezas tales como escuchar, hablar, leer y escribir, adquirir vocabulario y estructuras gramaticales, así como también habilidades sociales, donde aprenden a utilizar el idioma para comunicarse y socializarse con los demás.
- **Fomentan la creatividad:** A partir del juego los niños pueden explorar su imaginación y aplicar su creatividad en diferentes contextos.
- **Hacen que el aprendizaje sea más memorable:** Divertirse mientras aprenden también ayuda a los alumnos a retener mejor la información porque el proceso es agradable y memorable.

A través de los juegos, los niños pueden aprender inglés de la misma manera en que aprenden su lengua materna, sin ser conscientes de que la están estudiando y aprendiendo, además son ideales para crear la motivación que se desdibuja entre la obligatoriedad y la rutina de la clase.

Capítulo 3

Solución propuesta

En este capítulo se describen los juegos elegidos para desarrollar, los recursos iniciales con los que contaba el equipo para comenzar a trabajar y se plantea una arquitectura de la solución implementada para cumplir con los objetivos propuestos.

3.1. Los juegos

Los juegos a desarrollar serán tres: crucigramas, sopas de letras y la batalla naval. A continuación se hará una introducción a cada juego describiendo sus formatos y reglas, y se presentan ejemplos para cada uno.

3.1.1. Crucigramas

Un crucigrama es una cuadrícula (tablero) en donde cada casillero puede estar vacío (casillero blanco) o deshabilitado (casillero de color), y un conjunto de pistas.

Las pistas son pequeños textos que tratan de describir las palabras que deberán colocarse en la cuadrícula, utilizando casilleros vacíos contiguos horizontal o verticalmente e iniciando en un casillero específico según se indique, insertando un carácter alfabético en cada uno.

A continuación se presenta a modo de ejemplo el siguiente crucigrama[6], en el cual fue resuelta la pista 1 de las horizontales:

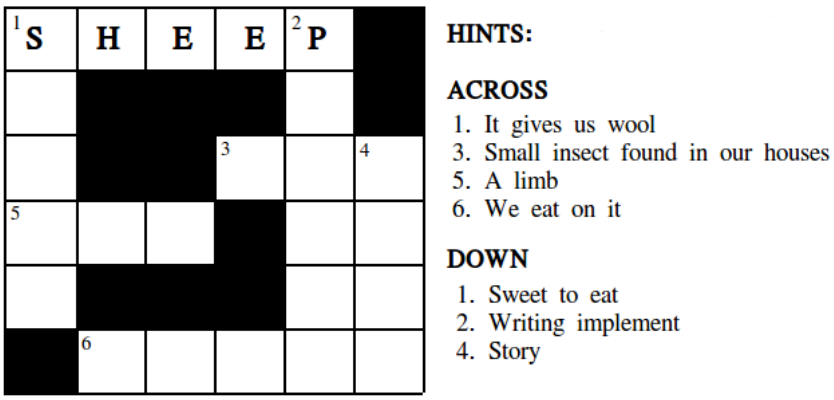


FIGURA 3.1: Ejemplo de crucigrama en inglés

El crucigrama está completo cuando todos los casilleros en blanco han sido completados, es decir, todas las pistas fueron resueltas.

Los crucigramas son un buen estimulante de aprendizaje individual y grupal. Ayudan a ampliar el vocabulario y entrenan la comprensión lectora. Descifrar correctamente un crucigrama también requiere una ortografía exacta y, a veces, de la evaluación de diferentes opciones que se pueden inferir de una pista impulsando el aprendizaje de sinónimos. A menudo los crucigramas cumplen el rol de diccionario y, en otros casos estimulan el uso del mismo, para términos de las pistas que el niño no comprende para su resolución (Jones, 2007).

3.1.2. Sopa de letras

Este es un juego simple que consiste en una cuadrícula (tablero) donde cada casillero contiene una letra y el objetivo es descubrir un número determinado de palabras que se forman enlazando las letras de forma horizontal, vertical o diagonal. Existen otras variantes que contienen las palabras desde abajo hacia arriba o de derecha a izquierda.

El siguiente tablero corresponde a una sopa de letras en la que se deben buscar las palabras indicadas en la lista, en este caso ya fue encontrada la palabra *soup*:

WORDS TO FIND:

egg
rice
burger
soup
cake

T	A	B	O	F	C
S	O	U	P	L	A
J	F	R	I	O	K
P	E	G	G	R	E
Y	O	E	P	M	N
E	L	R	I	C	E

FIGURA 3.2: Ejemplo de sopa de letras en inglés

El juego finaliza cuando todas las palabras que se deben buscar son encontradas.

La sopa de letras es una buena forma de aprender nuevo vocabulario y corregir faltas de ortografía. Estimulan la memoria visual, la atención del niño y, pueden jugarse de forma individual o grupal. También ayuda a reconocer patrones para aprender a escribir con fluidez, por ejemplo, reconocer que si aparece una letra Q, esta siempre está seguida de la letra U, o que a menudo la H se encuentra luego de la T, etc. (Van Gemert, 2016).

3.1.3. Batalla Naval

Este es un juego tradicional de adivinación que involucra a dos participantes. Los jugadores manejan dos tableros cada uno divididos en casillas. Cada tablero representa una zona diferente del mar abierto: la propia y la contraria.

En la correspondiente a su zona el jugador coloca sus barcos y registra los «tiros» que le realiza el oponente; en la otra zona, se registran los tiros propios al tiempo que se deduce la posición de los barcos del contrincante.

El juego a implementar en esta ocasión es una variante del original [30] porque el tablero se compondrá de columnas que tienen sujetos de oraciones y en las filas hay predicados que coinciden en número y persona, por lo que el jugador para identificar un casillero deberá construir una frase: *sujeto + predicado.*

A modo de ejemplo presentamos el siguiente tablero de 3x3:

	Most students	Betty and Brian	The adults
are tired.			
also like swings.			
do magic tricks.		X	

Entonces para identificar el casillero que contiene la *X*, el usuario debe comunicarle la frase *"Betty and Brian do magic tricks"* a su contrincante.

En el siguiente ejemplo [30] se presenta un tablero y las naves que se deben posicionar en él. En este caso se encuentra ya posicionada la nave *battleship*:

My Grid

	She	He	My father	Your mother	My friend	My dog	Your cat	Our teacher
likes reading books.								
doesn't like running.								
is a baseball fan.								
can play the piano.		**A**	**A**	**A**	**A**			
likes chocolate.								
has 10 cats.								
is a movie star.								
eats lunch at 1 o'clock.								

FIGURA 3.3: Ejemplo de batalla naval en inglés

El juego finaliza una vez que uno de los dos jugadores derribó todas las naves del participante contrario.

Este juego es una gran herramienta para practicar las habilidades de lectura, habla y escucha. Para hundir las naves del oponente, se debe leer inglés y armar oraciones. Para hacer un ataque, se necesita pronunciar la oración claramente. Finalmente, hay que entender lo que el oponente está diciendo para averiguar qué casilla del tablero está tratando de atacar.

3.2. Recursos iniciales

Los recursos iniciales con los que se cuenta para comenzar a implementar los juegos son:

- Proporcionados por el Programa de Políticas Lingüísticas de ANEP:
  - Lista de términos de los niveles *Pre A1 Starters*[23] y *Movers* [16] de *Cambridge Assessment English*. Cada uno de los términos se encuentra asociado a una clasificación temática dada por la unidad (*topic*) que se trabaje en el aula, por ejemplo: la palabra **giraffe** pertenece a la unidad **At the Zoo**.
    Esta lista cuenta con 834 términos, incluyendo adjetivos, verbos, nombres propios, etc.
    De ahora en adelante se hará referencia a esta lista de palabras como «*Lista inicial*».
  - Conjunto de textos de ejemplo (Ver apéndice B) con un promedio de 200 términos cada uno, para poder definir el nivel de inglés a considerar en los juegos.
- Proyecto de grado realizado anteriormente (Esteche y Romero, 2015), que tuvo como objetivo la generación automática de crucigramas en español a partir de textos de prensa.
- Corpus de *Simple English Wikipedia*[27]
  Es una traducción simple de 142.910 artículos de *English Wikipedia* (casi un 3 %). *Simple English Wikipedia* se creó para facilitar la lectura a niños, personas que estén aprendiendo el idioma, que se le dificulte leer o aprender, o quienes podrían encontrar difíciles de entender ciertos términos o artículos de la enciclopedia principal.
- Diccionario *Wordsmyth for children* [35]
  Wordsmyth es un diccionario completo con entradas que incluyen pronunciaciones, animaciones, sinónimos, fotografías y etimología de palabras. En este diccionario es posible buscar definiciones en distintos niveles de dificultad: principiante, medio y avanzado. En esta oportunidad nos interesa utilizar el nivel **principiante**, esto es: *Wordsmyth for children*.
- Proporcionado por `Plan Ceibal`[22]: Dos computadoras "Ceibalitas" modelos `Clamshell JP` y `XO 1.0`, y una tablet modelo `Tablet Haier 723` para poder realizar las pruebas a medida que se va implementado la aplicación.

3.3. Esquema de la solución

En las siguientes secciones se detalla el esquema de la solución al problema, comenzando por describir los recursos que fueron implementados y los requerimientos funcionales y no funcionales de la aplicación.

3.3.1. Recursos desarrollados

Como planteamos anteriormente los juegos implementados son tres: crucigramas, sopas de letras y batalla naval.

En particular para los dos primeros decidimos hacerlos temáticos ya que contamos con la *«Lista inicial»* de palabras que está clasificada por la unidad o tema que se trabaja en el aula. Así que, una vez definidas las categorías temáticas con las que se puede jugar, uno de los recursos generados fue una nueva lista a partir de la depuración y categorización de los términos de la *«Lista inicial»* para generar los tableros de los juegos según un tema elegido por el usuario.

A esta lista resultante la llamamos *«Lista categorizada»* de palabras la cual nos ayudó a tener un contexto del nivel de inglés a considerar para los siguientes recursos desarrollados.

Con respecto a los textos que nos brindó ANEP fue necesario complementar este material inicial construyendo un corpus de mayor tamaño extrayendo información de otros sitios.

Por un lado, fue importante asegurar que el vocabulario utilizado en el corpus a construir fuera de nivel básico, y en lo posible que se tratase de artículos educativos para extraer de ellos pistas que se utilizan en los crucigramas. Por otro lado, fue importante que incluyera frases "cortas" que, en lo posible, describieran actividades y/o situaciones para poder extraer de ellas los sujetos y predicados que se utilizan en los tableros de la batalla naval.

Para la generación de crucigramas, fue necesario implementar un procedimiento que, dado un conjunto de textos en inglés, extraiga pares del tipo *«palabra, pista»* donde el término *palabra* quede descrito (o definido) por la frase identificada como *pista*. Luego, a partir de esto, se pueden construir tableros y ofrecerle al usuario sus pistas correspondientes para poder completar el crucigrama. En este paso fue importante que los pares *«palabra, pista»* tuvieran asociada una categoría temática de las ya definidas para la *Lista Categorizada* de forma tal que sean fáciles de identificar cuando el usuario selecciona con qué categoría jugar.

Con respecto a las sopas de letras, nos interesa que se generen tableros de forma variada, por lo que fue primordial agregar más palabras a la *Lista Categorizada* que cumplan con una dificultad similar a las que ya le pertenecen. Para ayudarnos a clasificar como "sencillas" las palabras candidatas a ingresar a la lista nos apoyamos en el uso de la frecuencia con que aparece cada palabra en el corpus de *Simple English Wikipedia* ya que podemos suponer que una palabra es más sencilla cuánto más frecuente es su uso (Jiang y John Lee, 2017).

Para la batalla naval, fue necesario implementar un segmentador de oraciones para obtener el sujeto y predicado de cada una, además del número y persona en el que se encuentra, para luego construir los tableros de los participantes. Es importante saber el número y persona para que al momento de generar un tablero este contenga sujetos y predicados con los mismos rasgos, pudiendo de esta forma identificar un casillero con una frase que sea consistente gramaticalmente.

La figura 3.4 muestra la relación entre los recursos generados y los juegos construidos en grandes rasgos.

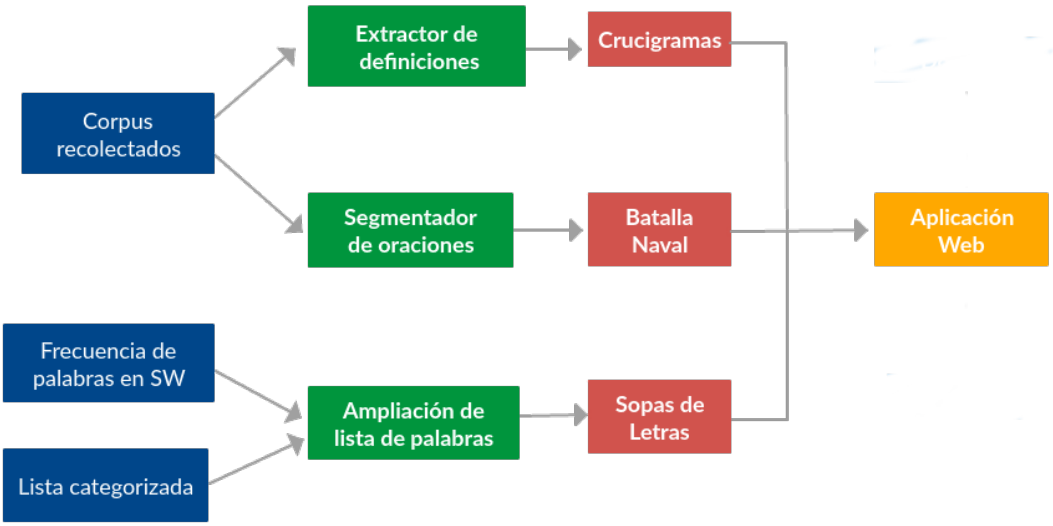


FIGURA 3.4: Relación entre recursos a desarrollar y juegos

3.3.2. Características de la aplicación

Requerimientos no funcionales

Debido a que la aplicación será utilizada por docentes pero en particular, por niños escolares, la misma debe tener una interfaz fácil e intuitiva y una manera de jugar atractiva de forma tal que los niños se sientan motivados para aprender y divertirse al mismo tiempo. Además es importante que no dependa de una conexión a Internet por las razones presentadas en el primer capítulo.

Otro requisito que se nos presentó desde un principio, es que la aplicación debe funcionar en las Ceibalitas de los escolares y maestros, por lo que fue necesario desarrollar una aplicación compatible con su sistema operativo y con un nivel de procesamiento de información que sea acorde a sus capacidades.

Requerimientos funcionales

En primera instancia la construcción de los juegos se realizó en base a información que ya fue procesada anteriormente y persistida en archivos incluidos en la aplicación, pero dado que el objetivo principal de esta aplicación es dar soporte a los docentes, una manera de hacerla más interesante es que el docente también pueda generar un juego de forma automática a partir de un texto trabajado en clase (*on-demand*).

A partir de este último requerimiento, nos planteamos realizar la aplicación con dos modalidades diferentes: una para estudiantes y otra para los docentes. En la modalidad de los estudiantes se pueden generar todos los juegos a partir de la información ya pre-procesada, mientras que, en la modalidad de los docentes, también pueden generarse de la misma forma pero además se le agregó la opción de generar crucigramas a partir de un texto con el que desean trabajar con sus alumnos.

Debido a que la extracción de los pares *«palabra, pista»*, tanto en la información que fue pre-procesada como en la opción *on-demand*, pueden contener errores, fue necesario que la modalidad para docentes tenga la funcionalidad de corregirlos y persistir las correcciones en los archivos correspondientes.

Con respecto a las sopas de letras, y aprovechando la información ya generada para los crucigramas, se decidió implementar en principio dos niveles de dificultad: el primer nivel es proporcionándole al usuario la lista de palabras a buscar, y el segundo nivel ofreciéndole solo las pistas de las palabras a buscar.

La última funcionalidad requerida para la aplicación es la exportación e importación de los tableros generados. Esta funcionalidad le proporciona al docente comodidad y agilidad a la hora de generar un tablero de su agrado y para poder exportarlo en un archivo para que sus estudiantes lo importen en su computadora y todos se encuentren bajo las mismas consignas de juego. Esta funcionalidad es fundamental para el juego de la batalla naval, ya que los tableros de ambos participantes deben ser iguales, y como la aplicación los genera aleatoriamente, uno de los participantes debe generarlo y luego exportarlo para que su contrincante lo importe en su computadora.

La figura 3.5 muestra la solución en líneas generales.

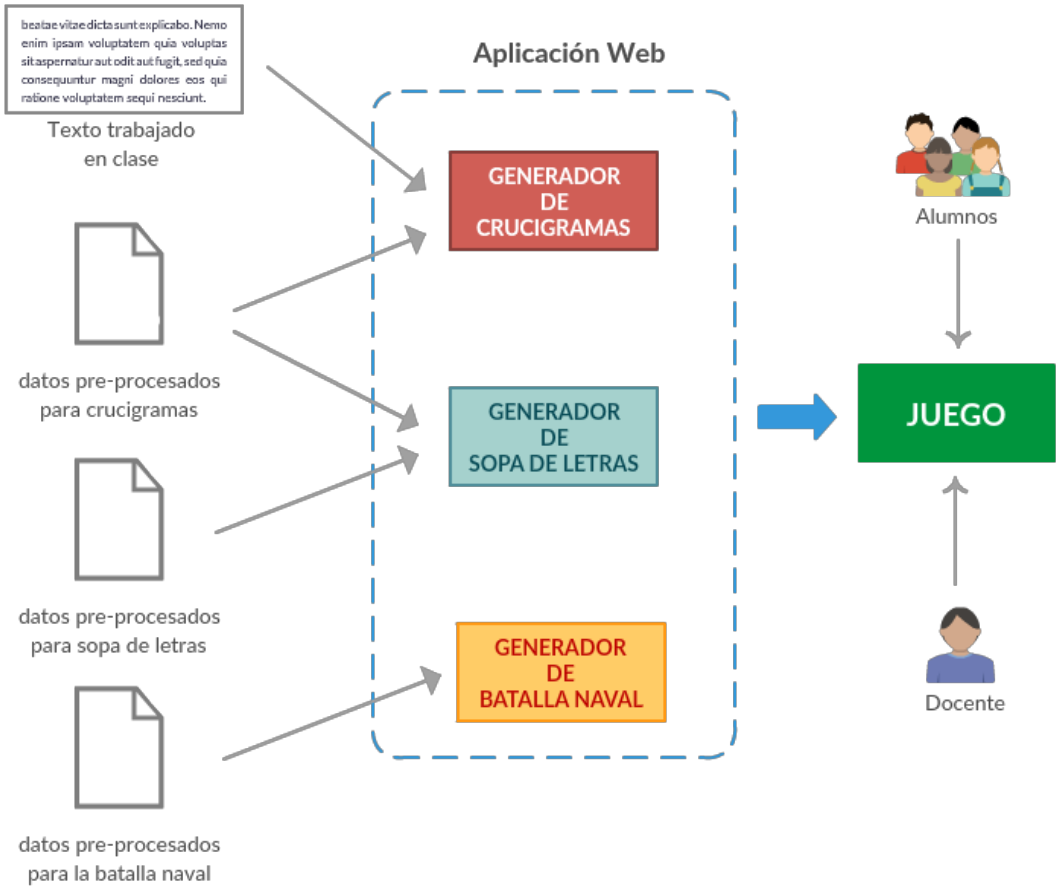


FIGURA 3.5: Solución propuesta en líneas generales

Capítulo 4

Desarrollo de recursos

En este capítulo se describen los recursos que fueron generados para implementar la solución propuesta. Además se detallan las técnicas y heurísticas aplicadas, los experimentos realizados y los resultados obtenidos.

4.1. Lista Categorizada de palabras

Se toma la «*Lista Inicial*» de palabras y se la depura quitando aquellos términos que son compuestos por más de una palabra (por ejemplo, la acción "pick up"), ya que no tiene sentido incluirlas en un crucigrama o sopa de letras.

También se eliminan las palabras que son preposiciones, verbos, conjunciones, determinantes, pronombres y algunos adverbios y adjetivos, ya que por más que estos formen parte de los textos que conforman oraciones gramaticalmente correctas, estos generalmente no se encuentran definidos como conceptos.

Entonces la lista resultante contiene, en su mayor parte, **nombres y adjetivos**, y tiene un total de 198 palabras.

Una vez obtenida la lista reducida de palabras, se les asigna una de las siguientes categorías:

- **Animals**: contiene 32 nombres de animales, por ejemplo: dog.
- **Body**: compuesta por 16 nombres de partes del cuerpo humano, por ejemplo: *arm*.
- **Colors**: compuesta por nombres de 10 colores.
- **Calendar**: inicialmente tiene los nombres de los 12 meses y los 7 días de la semana.
- **Family**: contiene 21 palabras que describen los miembros de una familia, por ejemplo: *mother, grandfather*, etc.
- **Food**: tiene 40 palabras relacionadas a comidas, frutas y verduras, por ejemplo: *bread, lemon*, etc.
- **House**: compuesta por 22 palabras relacionadas a muebles que se pueden encontrar en una casa y nombres de las distintas habitaciones, por ejemplo: *chair, kitchen*, etc.
- **Numbers**: contiene los dígitos del 0 al 9.

- **Sports**: constituída por 16 palabras relacionadas a deportes, por ejemplo: *ball, cycling, football*, etc.
- **Weather**: tiene 12 conceptos relacionados al clima, como por ejemplo *hot, windy*, etc.

De ahora en más se hará referencia a esta lista como *Lista Categorizada* de palabras.

4.2. Frecuencias de palabras

Ya que es de interés trabajar con un vocabulario en inglés sencillo, es beneficioso tener una medida de dificultad o complejidad de las palabras que se pretende utilizar en los juegos. Una buena medida es la frecuencia de palabras en un corpus; la teoría detrás de esto es que a mayor frecuencia, menor grado de dificultad (Jiang y John Lee, 2017).

Para esto se implementó un script que, utilizando la librería `NLTK`, cuenta los pares *«palabra,POS-tag»*[1] distintos para todas las palabras en el corpus de *Simple English Wikipedia*. Es decir, que en este trabajo nos referiremos a la cantidad de apariciones de una palabra como su **frecuencia**.

Cabe destacar que por más que dicho recurso utilice un vocabulario más simple que *English Wikipedia*, no deja de incluir términología compleja, por ejemplo, los sustantivos *grammatische*, *praetorium* y *hepacivirus* que se mencionan solo una vez en todo el corpus.

El hecho de tener en cuenta el *POS-tag* de la palabra es para desambiguar su significado, ya que en el inglés es muy común que una misma palabra refiera a, por ejemplo, un sustantivo y un verbo al mismo tiempo (por ejemplo: *water*, *run*, *study*, etc.).

Se obtuvieron 669.090 pares distintos, y fueron persistidos en una base de datos `MongoDB` por cuestiones de eficiencia en el acceso a los datos. El objetivo de esta base de datos es ser utilizada en la *Ampliación de la Lista Categorizada* (sección 4.4).

[1] Las etiquetas POS-tag utilizadas y sus significados se encuentran en el libro *Speech and Language Processing (2Nd Edition)* de Jurafsky y Martin, 2009

En el ejemplo de estructura `JSON` que sigue a continuación, se puede ver que el sustantivo *river* (NNP: Proper noun - singular) es más frecuente y según esta teoría (Jiang y John Lee, 2017) "más simple" que el adjetivo *indian* (JJ: Adjective):

```
{
    (...)
    {"frecuencia" : 17060,
     "palabra" : "river",
     "POS tag" : "NNP"},
    (...)
    {"frecuencia" : 3086,
     "palabra" : "indian",
     "POS tag" : "JJ"},
    (...)
}
```

4.3. Corpus

Para complementar el material inicial se realizó una búsqueda en la web de textos en inglés que tuvieran un nivel similar a los textos de ejemplo brindados por ANEP. Luego, para obtener la información disponible en los sitios seleccionados se recurrió a la técnica de `Web Scraping`, con las herramientas `Scrapy` para `Python` y `MongoDB` para persistir los datos.

4.3.1. Simple English Wikipedia

El corpus de *Simple English Wikipedia* utilizado fue un *dump* del 6 de abril de 2015. El mismo fue descargado de la página de *PIKES*[21] (Corcoglioniti, Rospocher y Palmero Aprosio, 2016), compuesto por un total de 109.708 archivos de texto donde cada uno de estos contiene un artículo en texto plano.

Dada la gran cantidad de información a manejar, los archivos fueron persistidos en una base de datos `MongoDB`, dentro de una *colección* donde cada *documento* incluye el contenido de cada artículo, con el fin de facilitar el acceso a los mismos.

El objetivo de este corpus es ser utilizado para extraer pares *«palabra, pista»* que luego serán la entrada de los crucigramas y sopas de letras como se verá más adelante.

4.3.2. Ducksters

Ducksters[5] es un portal educativo para niños que contiene artículos clasificados en diversas categorías como historia, ciencia, geografía, matemáticas, entre otras. Estos artículos están escritos para que sean fáciles de leer y entender para los niños. (Ver ejemplo en apéndice B). Además contiene actividades que incluyen juegos, deportes, películas, música y más.

Dentro del sitio se eligió la categoría *Science* para extraer sus artículos que están divididos en las siguientes subcategorías: *Biology, Earth Science, Chemistry, Physics y*

Animals. A su vez, cada una de estas subcategorías contiene clasificaciones de sus temas, por ejemplo, en la subcategoría *Animals* tenemos *Reptiles, Mammals, Birds,* etc. (Figura 4.1)

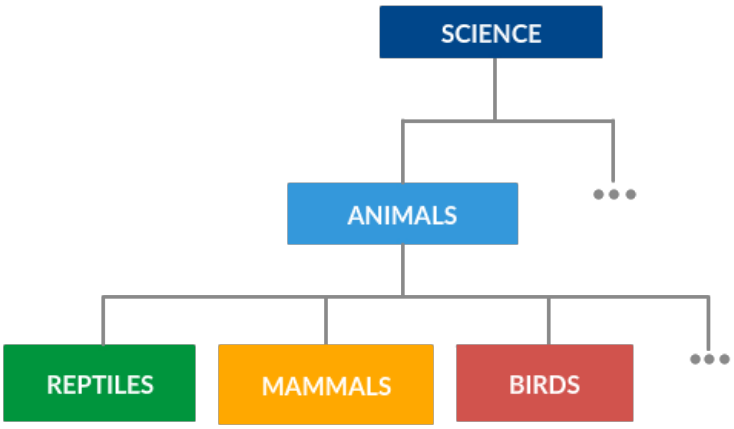


Figura 4.1: Categorización en Ducksters

La extracción para la categoría *Science* en números quedó según se muestra en el cuadro 4.1.

Subcategoría	Cantidad de clasificaciones	Total de artículos
Animals	11	81
Earth Science	7	62
Biology	5	35
Chemistry	4	68
Physics	7	77

Cuadro 4.1: Ducksters - Categoría Science en números

El objetivo de este corpus es ser utilizado para la extracción de pares *«palabra, pista»* que se utilizan en los crucigramas y sopa de letras, complementando el corpus de *Simple English Wikipedia*.

4.3.3. Wordsmyth for children

En este diccionario para niños se aplicó la técnica de `Web Scraping` para la extracción de definiciones de palabras pertenecientes a la *Lista Categorizada*. El objetivo de esta recopilación es complementar las definiciones obtenidas de los corpus descritos anteriormente con el *Extractor de definiciones*.

Como ya fue mencionado, en este diccionario es posible buscar definiciones en distintos niveles de dificultad: principiante, medio y avanzado. La modalidad **principiante** tiene la gran ventaja de que utiliza un vocabulario muy simple y concreto adecuado para

niños, y es la modalidad elegida para la extracción.

A modo de ejemplo se presenta la siguiente definición de la palabra *cheese* en *Wordsmyth* para el nivel de principiantes:

> **Cheese**
>
> a solid food made from milk.

En el nivel avanzado la definición de *cheese* es:

> **Cheese**
>
> a food made from milk curd that has been pressed into a mass and aged or seasoned in any of a wide variety of ways.

De este diccionario se extrajeron las definiciones para niños de todas las palabras pertenecientes a la *Lista Categorizada Ampliada* (ver sección 4.4), es decir, un total de 371 palabras.

4.3.4. ESLFast

ESLFast[9] es un sitio gratuito para estudiantes de ESL, *English as Second Language*, que incluye cuentos, consejos de conversación y lectura, práctica de estructura de oraciones y una sección de vocabulario.

Es un sitio muy simple de manejar que contiene su material dividido primero en dos grandes categorías: *For beginners* y *For intermediate learners*. En este caso, se realizó la extracción de la categoría *For beginners* para todas sus subcategorías (ver figura 4.2) las cuales contienen en promedio 100 textos cortos compuestos por, aproximadamente, 12 oraciones cortas.

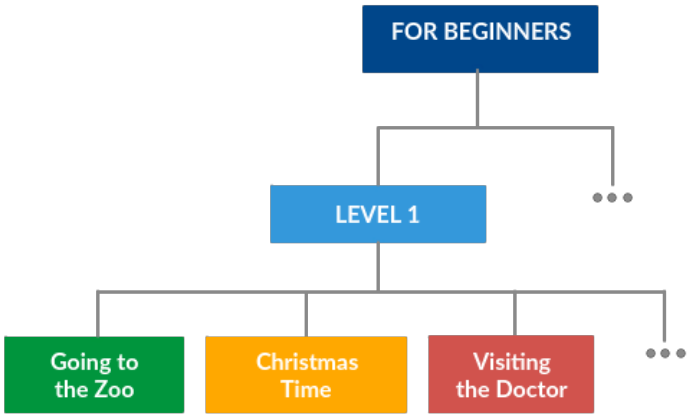


Figura 4.2: Categorización en ESLFast

A continuación se presenta, a modo de ejemplo, el texto llamado *"Going to the Zoo"*

Going to the Zoo

She goes to the zoo. She sees a lion. The lion roars.
She sees two elephants. The elephants have a long trunk.
She sees a turtle. The turtle is slow.

She sees a rabbit. The rabbit has soft fur. She sees a gorilla.
The gorilla is eating a banana.

En la extracción de información de este sitio se obtuvieron 931 textos cortos de los cuales se recogieron un total de 11.102 oraciones diferentes.

El objetivo de este corpus es ser utilizado como entrada para el *Segmentador de oraciones* obteniendo los sujetos y predicados de cada oración, que se aplicarán en los tableros de la batalla naval.

4.4. Ampliación de la Lista Categorizada

En esta sección se define la *Lista Categorizada Ampliada* y se detalla el proceso que la genera.

4.4.1. Implementación

La *Lista Categorizada* de palabras tiene como objetivo ser la entrada para las sopas de letras, esto es, dada una categoría, por ejemplo: *body*, construir un tablero con un subconjunto de palabras pertenecientes a ella.

El motivo de agregarle más palabras es disminuir la posibilidad de generar tableros similares a medida que el usuario está jugando y también incorporar nuevas palabras a su vocabulario. La tarea de ampliación de esta lista se realiza por categoría, es decir, se generan sublistas por categoría y se las amplía por separado.

Por ejemplo, para la categoría *body*, el proceso de aplicación sería como se muestra en la figura 4.3.

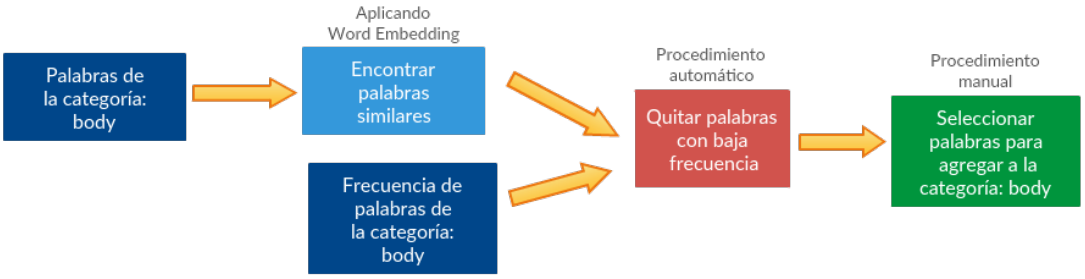


FIGURA 4.3: Proceso de ampliación de categoría *body*

Es importante que las palabras a agregar a cada sublista tengan un nivel de dificultad similar al de las palabras que conforman la *Lista Categorizada*. Para esto se utilizan las frecuencias *«palabra,POS-tag»* obtenidas de *Simple English Wikipedia* como referencia del grado de dificultad que presentan las potenciales palabras que formarán parte de las categorías.

Para asegurar el nivel similar, se compara la frecuencia asociada a la potencial palabra con la frecuencia promedio de las palabras incluidas en la categoría a expandir. Si dicha frecuencia promedio es menor a la de la potencial palabra, entonces es descartada por ser considerada más compleja de lo deseado.

Criterio de elección de potencial palabra

$$frecuencia(potencial\_palabra) >= \frac{\sum_{p \in categoria} frecuencia(p)}{\#palabras\_categoria}$$

En particular, se aplicó el método de `Word2Vec` para la generación de modelos. Para entrenar el modelo en `Word2Vec` en principio se utilizó el corpus de *Simple English Wikipedia* pero un gran porcentaje de las palabras de la *Lista Categorizada* no tenían un vector asociado, es por esto que los experimentos fueron realizados con un modelo ya entrenado del corpus de *Google News*[10] con, aproximadamente, 3 millones de vectores de palabras en inglés de 300 dimensiones.

4.4.2. Experimentos

A continuación se describen los dos experimentos de ampliación de sublistas de palabras y se comparan los resultados.

Experimento 1

El primer experimento realizado consistió en encontrar las palabras cercanas a la sublista de palabras que pertenecen a la categoría dada.

Para este proceso se desarrolló un procedimiento en `Python` que recibe como parámetros de entrada la sublista de palabras de la categoría a expandir, sus frecuencias y el modelo entrenado con `Word2Vec`.

Por ejemplo para la categoría *body* el proceso realiza los siguientes pasos:

1. Se calcula el vector $\vec{v}$ resultante de la suma de los vectores de cada palabra de la categoría, esto es:

$$\vec{v} = \vec{arm} + \vec{back} + \vec{leg} + ... + \vec{head}$$

2. Con `Word2Vec` se obtienen las 1000[2] palabras más cercanas al vector $\vec{v}$.

3. De esas 1000 palabras se quitan de forma automática las que tengan baja frecuencia (como se explicó anteriormente).

4. De la lista resultante de palabras se seleccionan manualmente las que ahora pasarán a formar parte de la categoría.

[2]Esto se obtiene con la función *most_similar* y configurando el parámetro *TOP_N=1000* en el modelo `Word2Vec`

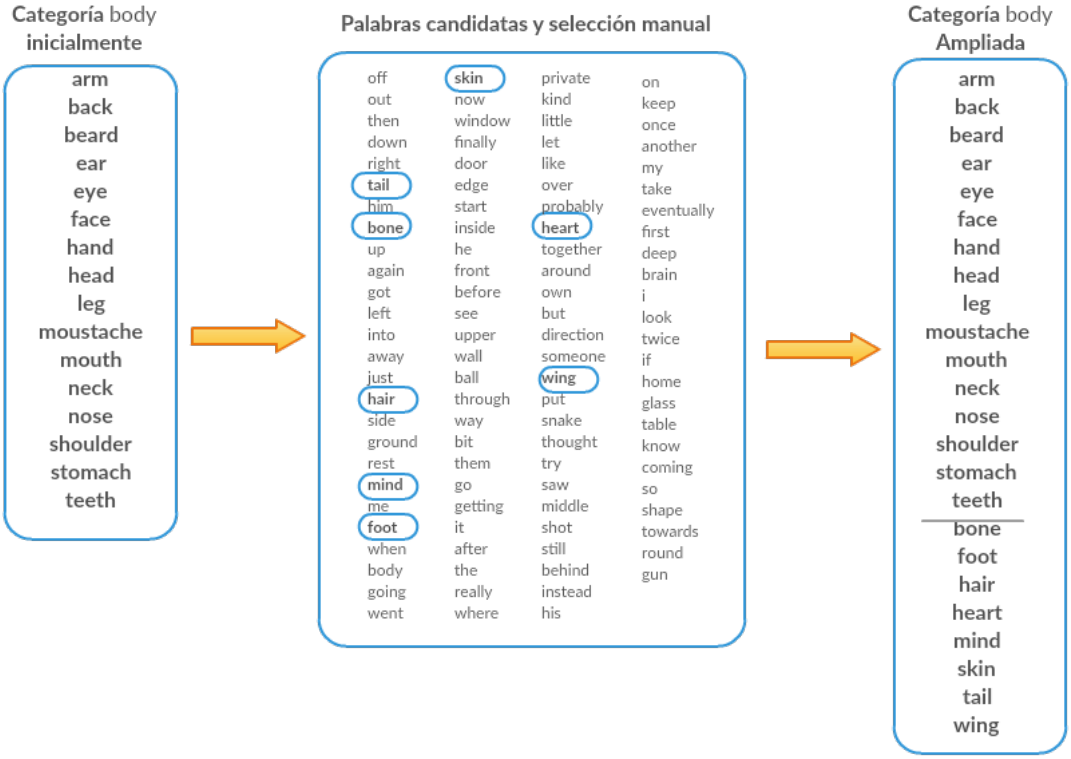


FIGURA 4.4: Ampliación Categoría body - Experimento 1

El resultado de este experimento (Figura 4.4) para la categoría *body* fue de 102 palabras candidatas luego del filtro automático por frecuencia, de las cuales solo 8 fueron tomadas como correctas para agregar a la sublista inicial, obteniendo una cantidad de 24 palabras en total.

Experimento 2

Este experimento consiste en "potenciar" el vector de entrada de `Word2Vec` con la categoría para encontrar sus palabras similares y así obtener una mejor calidad de la sublista.

Esta idea surge del interés de simplificar la tarea de revisión del resultado ya que se obtiene una gran cantidad de palabras las cuales hay que evaluar de forma manual para seleccionar las que formarán parte de la sublista definitiva.

También, se desea ayudar a la desambiguación de las palabras de entrada, ya que por ejemplo: para la palabra *back* de la categoría *body* se obtuvieron en este caso palabras como *left, out, right*, entre otras, que es correcto que se encuentren cercanas a *back* pero en el contexto de partes del cuerpo no lo es.

Para este experimento se utiliza un procedimiento que fue desarrollado en `Python` que recibe como parámetros de entrada la sublista de palabras de la categoría a expandir, sus frecuencias y el modelo entrenado con `Word2Vec`.

En este procedimiento, primero se calcula el vector promedio, llamado *centroide*, de el vector de cada palabra de la sublista y el vector de la categoría, por ejemplo para la

categoría *body* que tiene 16 palabras se tiene:

$$\vec{x_1} = (\vec{arm} + \vec{body})/2$$
$$\vec{x_2} = (\vec{back} + \vec{body})/2$$
$$...$$
$$\vec{x_{16}} = (\vec{eye} + \vec{body})/2$$

Luego se calcula el vector promedio de todos los obtenidos en el paso anterior:

$$\vec{v} = (\vec{x_1} + \vec{x_2} + ... + \vec{x_{16}})/16$$

Una vez obtenido $\vec{v}$ se realizan los pasos 2, 3 y 4 descritos para el experimento 1.

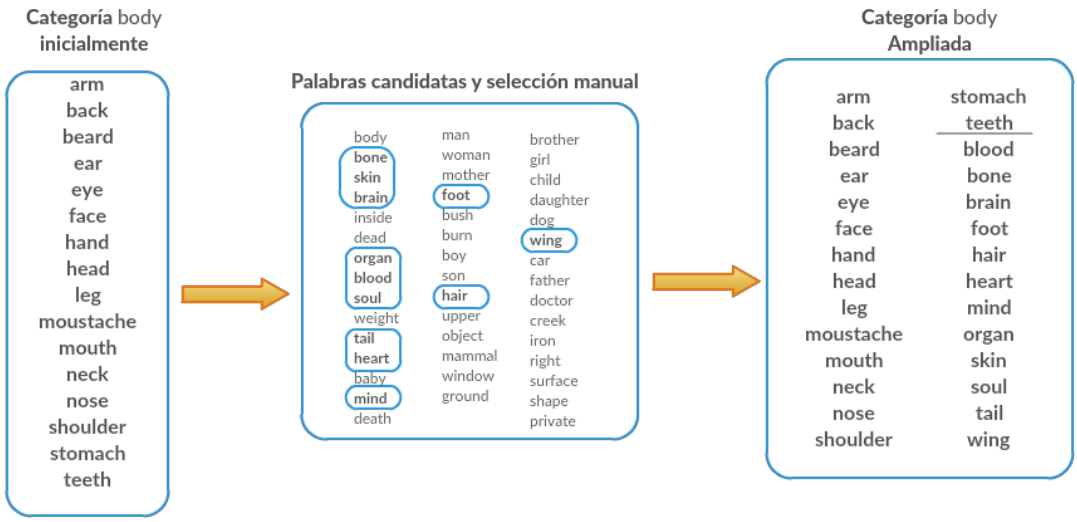


FIGURA 4.5: Ampliación Categoría body - Experimento 2

En la figura 4.5 se muestra el resultado del experimento 2. Este último arrojó un resultado mucho mejor que el experimento 1 ya que se obtuvieron solo 45 palabras nuevas pero de las cuales 12 fueron tomadas como correctas para agregar a la lista inicial, obteniendo una cantidad de 28 palabras en total para la categoría *body*.

4.4.3. Resultados

El hecho de obtener una menor cantidad de palabras nuevas (102 en el Experimento 1 contra 45 del Experimento 2) gracias a la desambiguación, ayudó a simplificar la tarea de seleccionar manualmente las palabras que se tomaron como correctas para formar parte de la sublista.

En general, el resultado de aplicar el método del experimento 2 fue mejor para siete de nueve[3] categorías con respecto al método del experimento 1[4], pero de igual manera en todos los casos se obtuvieron nuevas palabras, es por esto que tomamos el resultado de este último método como el apropiado para la ampliación de las sublistas.

[3]La categoría *numbers* no fue ampliada a partir de este método, simplemente se agregaron los números del 10 al 20

[4]Las categorías que dieron mejor resultado en el experimento 1 fueron: *sports y family*

Es válido aclarar que, para algunas categorías, las palabras añadidas no cumplen estrictamente la descripción dada en la sección 4.1. Por ejemplo, en la categoría *body* también se agregaron partes del cuerpo animal como la palabra *wing* o *tail*. En la categoría *calendar* se añadieron términos que describen períodos de tiempo, como por ejemplo *weekend*, y otras palabras como *yesterday*.

En el siguiente gráfico se pueden apreciar los resultados de la ampliación de la *Lista Categorizada* de palabras discriminado por categorías:

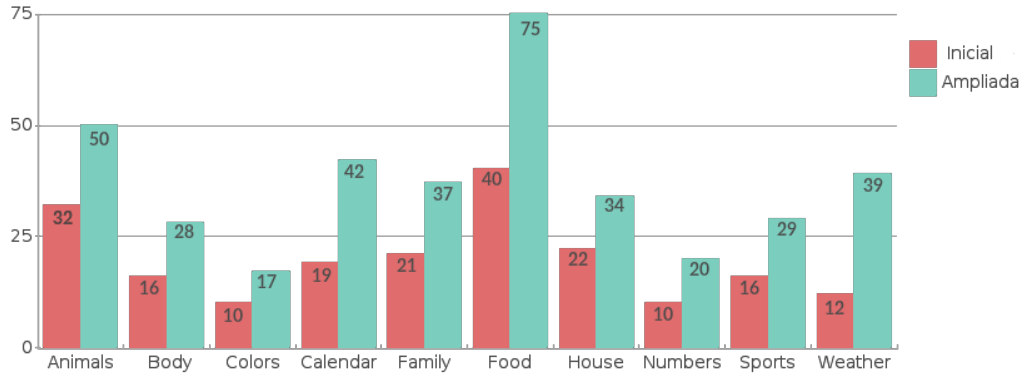


FIGURA 4.6: Cantidad de palabras por categoría - Inicial vs. Ampliada

La lista resultado que es la concantenación de todas las sublistas resultantes del experimento 2, se le llamará *«Lista Categorizada ampliada»* y contiene un total de 371 palabras.

4.4.4. Persistencia

Al finalizar la ampliación de la lista de palabras para cada categoría, se persiste la información en un archivo con formato `JSON`, el cuál contiene un documento con los siguientes datos:

```
{
    "animals": ["bat","bear","bird","cat",(...)]
    "body": ["arm","back","beard","ear",(...)]
    "colors": ["brown","black","blue","gray",(...)]
    (...)
}
```

Este archivo se denomina: *data_wordsearch.json*.

4.5. Extractor de definiciones

Este capítulo detalla cómo fue implementado el módulo Extractor de Definiciones. Se presentan algunos problemas presentados durante el proceso de implementación y cómo fueron atacados.

4.5.1. Implementación

La naturaleza de los corpus *Simple English Wikipedia* y *Ducksters* (en el sentido de que ambos provienen de enciclopedias) asegura que se encuentren una cantidad considerable de oraciones con formato de definición. Teniendo esto en cuenta y el nivel de inglés básico de sus textos, se decide utilizar la técnica de búsqueda por patrones, similares a los definidos en el trabajo de Hearst, (1992).

Para el caso del corpus de *Simple English Wikipedia*, generalmente los artículos comienzan con oraciones que definen el título del artículo.

Elephant

"Elephants are the largest living land mammals.[...]."[7]

Por otro lado, el sitio educativo para niños *Ducksters* brinda artículos interesantes donde se pueden obtener definiciones que describen más bien alguna propiedad del concepto a definir y en términos sencillos.

Penguins

"[...]Penguins are very funny animals.[...]."[20]

Los corpus no son la única entrada al extractor de definiciones, ya que la idea es que en el generador de crucigramas, la funcionalidad *on-demand* permita al maestro ingresar un texto propio. En este caso puede ocurrir que la entrada contenga pocas o ninguna oración que cumpla con los patrones de búsqueda de definiciones implementados.

A modo de ejemplo, se puede suponer que el maestro ingresa un texto que contiene la siguiente oración:

Texto ingresado

"[...]Many animals live in Africa, including lions,
giraffes, zebras, hyenas and monkeys.[...]"

A diferencia de los ejemplos de los corpus, esta oración no cumple con el siguiente patrón:

NP (**is**|**are**) *NP*

Es por esto que el Extractor no reconoce una oración definitoria en base al patrón en cuestión. Suponiendo que el texto es más extenso, se puede ignorar la oración y seguir buscando definiciones en las siguientes, pero esto no garantiza que se encuentre alguna. Sin embargo, el texto incluye sustantivos como *animals*, *Africa*, *lions*, entre otros, y

quizás se encuentre la definición de alguno de estos conceptos en el archivo de definiciones pre-procesadas. En estos casos el Extractor busca en el archivo la definición de sustantivos que aparecen en la oración y, por ejemplo, toma la definición de *Lion*:

```
{
    (...)
{"definicion" : "A large, strong mammal in the cat family that
 ↪  lives in Africa.",
"definiendum" : "Lion"}
    (...)
}
```

Usando la *lematización* se puede asociar la definición persistida con la palabra encontrada en el texto, y de esta forma incluirla en un crucigrama para enriquecerlo. En este caso el lema de *Lions* es *lion*, que es el definiendum tomado del archivo.

El extractor de definiciones se construyó en `Python` usando las librerías `NLTK` y el parser de `Stanford` para el análisis sintáctico. Para tareas de análisis semántico se utilizó el modelo `Word2Vec` al igual que en la sección anterior. Inicialmente se construye el Extractor utilizando la representación de árboles de constituyentes de oraciones para evaluar el cumplimiento de los patrones de búsqueda sobre las mismas. Dado que esto presentó dificultades, se trabaja posteriormente con árboles de dependencias.

Extracción utilizando análisis de constituyentes

El patrón de búsqueda que se usó en esta primera instancia fue:

{det} noun (**is**|**are**|**was**|**were**) NP

Esto es, una secuencia de:

- Un determinante opcional.
- Un sustantivo.
- Una de las conjugaciones del verbo "*to be*" en segunda o tercera persona del presente o pasado.
- Una frase nominal.

No es de interés incluir expresiones multi-palabra en un crucigrama o sopa de letras, y es por esto que se toman en cuenta definiendums a la izquierda del verbo principal con las características definidas en el patrón, descartando el determinante en caso de haberlo.

El algoritmo que realiza la búsqueda con este patrón es fácil de programar, ya que dichas oraciones conforman un árbol de constituyentes con un formato estándar de oración definitoria como se muestra en la figura 4.7.

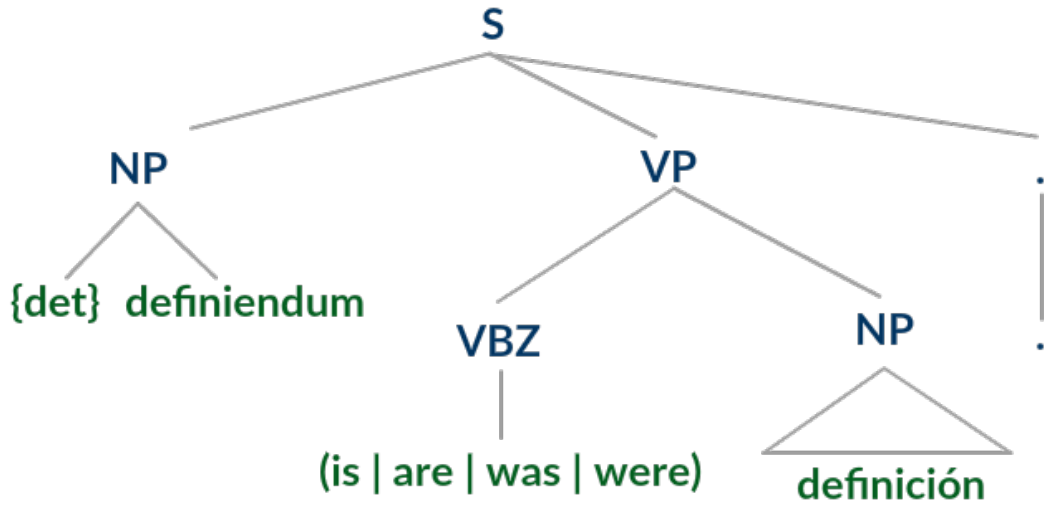


FIGURA 4.7: Árbol de constituyentes asociado al patrón "{det} noun (**is**|**are**|**was**|**were**) NP"

Se programó el algoritmo usando el parser de constituyentes sintácticos de `Stanford`. Para ejemplos como *"Warm is the opposite of cool"*, *"A wife is a married woman"*, o *"Lizards are reptiles"* se obtuvieron pares *«definiendum, definición»* correctos. Sin embargo, se detectó que para ciertos casos, debido a un mal funcionamiento del parser, los pares obtenidos no eran los esperados.

En el ejemplo de la figura 4.8 se pretende ilustrar la obtención de una definición incorrecta para el definiendum *game*.

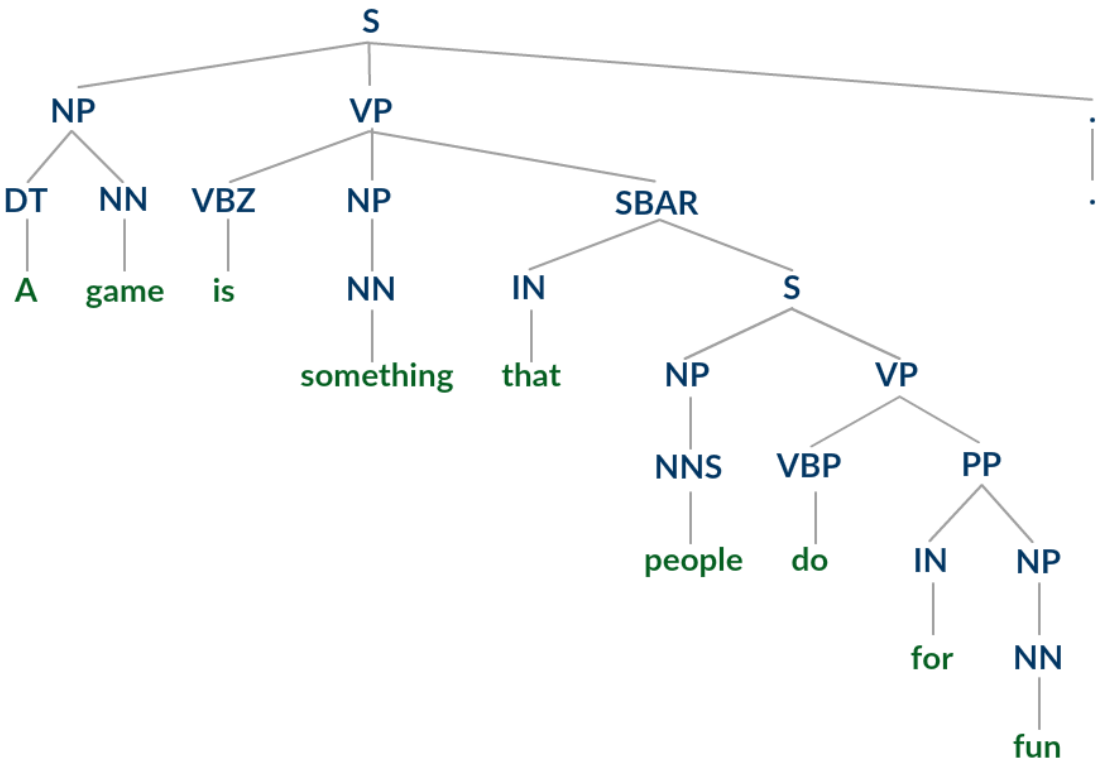


FIGURA 4.8: Árbol de constituyentes asociado a "A game is something that people do for fun."

Se puede observar que el árbol de constituyentes de la oración definitoria no se alinea con el esquema del árbol asociado al patrón de búsqueda. Esto se debe a que el grupo verbal (VP[5]) principal, que contiene al verbo *is* (VBZ[6]), tiene como tercer hijo un subárbol con raíz *SBAR*[7].

De esta forma el algoritmo extrae el par *«Game, Something»*. El subárbol asociado al mencionado *SBAR*, debería estar representado como segundo hijo del *NP* que tiene como primer hijo el sustantivo (*NN*[8]) *something*. De esta forma con el patrón de búsqueda se hubiera extraído la definición completa, es decir "*Something that people do for fun*".

Dado este problema del parser de constituyentes de `Stanford` se decide trabajar con su parser de dependencias ya que suele analizar con mayor precisión.

Extracción utilizando análisis de dependencias

El patrón de búsqueda previamente definido en el análisis de constituyentes usa conjugaciones del verbo *"to be"*, que es un verbo copulativo, esto es, un verbo que de por sí no tiene contenido semántico, sino que sirve para unir significados. Se puede decir entonces que las oraciones definitorias son un ejemplo de oraciones copulativas (aquellas que tienen como verbo principal un verbo copulativo). En español, los verbos *ser*, *parecer* y *resultar* son claros ejemplos.

[5]VP: Verbal Phrase
[6]VBZ: Verb, 3rd person singular present
[7]Subordinate clause: en español "Cláusula subordinada", es un grupo sintáctico que corresponde a la parte de una oración compuesta y funciona dentro de ella como un adjetivo, adverbio o sustantivo.
[8]NN: Noun, singular or mass

El parser de `Stanford` trata a los verbos copulativos como dependientes de su complemento[28]. O sea que para la oración *"A game is something that people do for fun"*, en lugar de ser *is* la raíz, es el sustantivo *something* que tiene como dependiente a dicho verbo con rótulo *cop*.

La figura 4.9 muestra el diagrama o árbol de dependencias asociado a la oración.

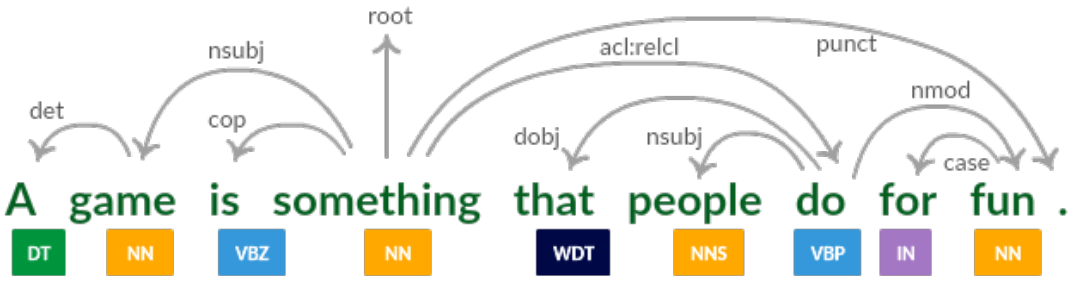


FIGURA 4.9: Diagrama de dependencias asociado a "A game is something that people do for fun."

Para el ejemplo, a partir de la raíz *something* (en este caso el núcleo de lo que sería el grupo nominal asociado a la definición), se puede navegar y obtener a todos sus dependientes en la oración. En el diagrama se puede observar que lo que sería la raíz de la claúsula subordinada del sustantivo *something*, el verbo *do*, es un dependiente de dicho sustantivo con rótulo *acl:relcl* (relcl de "relative clause"[8]). También se puede observar que la palabra *game* es un dependiente con rótulo *nsubj* (de "nominal subject"), indicando que es el sujeto de la oración y que es un sustantivo.

Si se abstrae del ejemplo anterior, se podría concluir que para obtener un par *«definiendum, definición»* a partir de un árbol de dependencias asociado de una oración definitoria, se puede:

- Obtener el definiendum como el dependiente de la raíz con etiqueta de sujeto.
- Obtener la definición recorriendo los dependientes de la raíz ignorando al verbo copulativo y al sujeto.

El esquema 4.10 ilustra la abstracción de un árbol de dependencias de una oración definitoria.

FIGURA 4.10: Diagrama de dependencias asociado a una oración definitoria

Siguiendo la idea anterior, se implementó un algoritmo basado en análisis de dependencias que extrajo correctamente el par *«Game, Something that people do for fun»*.

Con esta versión del extractor se experimentó sobre el corpus de *Simple English Wikipedia* que presentó algunos problemas, los cuales se detallan a continuación junto con las decisiones que se tomaron al respecto.

Correferencias: El problema se presentaba cuando el algoritmo analizaba oraciones donde el grupo nominal que contenía al definiendum era una correferencia a un grupo nominal de una oración anterior.

El siguiente ejemplo es un fragmento de texto del artículo *Komodo National Park* de *Simple English Wikipedia*:

> Komodo National Park
>
> "[...]The national park was founded in 1980 to protect the Komodo dragon. This animal is the world's largest lizard.[...]"

En este caso, *"This animal"* es una co-referencia a *"the Komodo dragon"*. El algoritmo, al no tener en cuenta estos casos, obtenía para *animal* la definición incorrecta *"The world's largest lizard"*. Este problema se atacó restringiendo la búsqueda de oraciones definitorias. Se tomaron en cuenta únicamente aquellas oraciones donde, en caso de tenerlo, el determinante del definiendum fuera un artículo indefinido. Para el caso del inglés serían *"A"* o *"An"* (descartando *the, this, that*, entre otros que en algunos casos indican una potencial correferencia).

Definiciones que incluyen definiendum: Para la oración *"Grapes are the fruit of a woody grape vine"*, se obtuvo la definición del concepto *grape* como *"the fruit of a woody* ***grape*** *vine."*

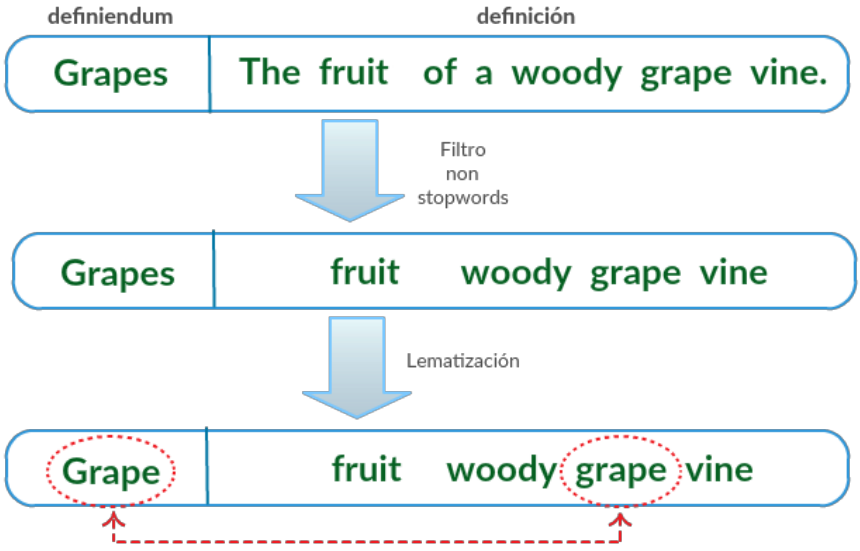


Figura 4.11: Diagrama de una definición que incluye su definiendum

Teniendo en cuenta que las definiciones tienen el objetivo de ser pistas para adivinar palabras en los juegos, se excluyeron definiciones donde el lema del definiendum estaba incluido en los lemas de las palabras de su correspondiente definición.

Voz pasiva: Se notó que de algunas oraciones definitorias no se extraían definiciones, como por ejemplo la que se muestra en la figura 4.12.

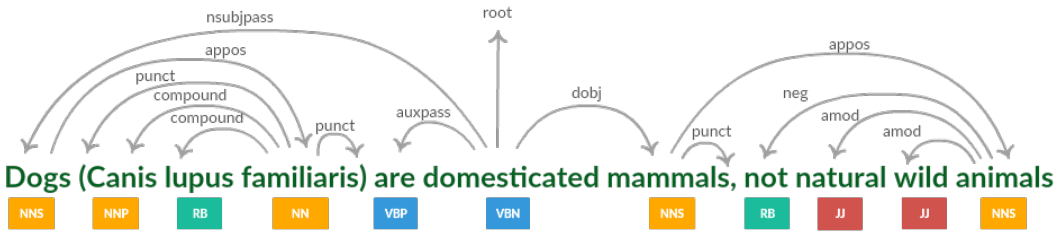


FIGURA 4.12: Diagrama de dependencias asociado a "Dogs (Canis lupus familiaris) are domesticated mammals, not natural wild animals"

Del diagrama de dependencias 4.12 se puede observar que:

- La raíz no es un sustantivo. El parser de `Stanford` toma al adjetivo *domesticated* como raíz, y como dependiente con rótulo *dobj* (objeto directo) al sustantivo *mammals*. Esto se da porque *domesticated* está mal interpretado como verbo (VBN [9])) por el parser.
- El verbo *are* es dependiente de *domesticated* con rótulo *auxpass* (de passive auxiliary), indicando incorrectamente que es verbo auxiliar de *domesticated*, e interpretando incorrectamente a la oración como un ejemplo de voz pasiva.
- El rótulo de la dependencia entre el sustantivo *Dogs* y la raíz *domesticated* no es *nsubj*, sino *nsubjpass* (de "passive nominal subject"), indicando incorrectamente que *Dogs* es sujeto pasivo de la oración.

La **voz pasiva** se usa usualmente cuando se le quiere dar más énfasis al paciente que al agente en la oración. Ejemplos de oraciones en voz pasiva son:

- ***Wine*** *is made by the fermentation of the sugar in grapes.*
- *A* ***cake*** *is made in a similar way to bread but sugar, fat and milk are added to the dough and often more ingredients.*
- ***Snow*** *is used for some winter sport activities like skiing and sledding.*

Teniendo esto en cuenta y que para implementar el generador automático de crucigramas lo que se buscan son palabras con sus respectivas pistas, se pueden tomar dichos pacientes y grupos verbales como pares *«palabra, pista»*. Los sustantivos *wine, cake* y *snow*, o **pacientes** en términos semánticos, son el foco de las oraciones ya que los complementos del verbo *is* son grupos verbales que **describen** características de dichos sustantivos. De hecho, los ejemplos de *wine* y *snow* provienen de los artículos de *Wine* y *Snow* respectivamente en *Simple English Wikipedia*.

[9] VBN: Verb, past participle

Se tomó la decisión de contemplar el funcionamiento incorrecto del parser y aceptar también como oraciones posibles para obtener definiciones o descripciones aquellas que tienen:

- Un verbo como raíz.
- Un dependiente primer hijo de la raíz con rótulo *nsubjpass*.
- Como dependiente de la raíz con rótulo *auxpass* y segundo hijo al verbo *is, are, was* o *were*.
- Con rótulo *dobj* un dependiente y tercer hijo de la raíz.

De esta forma se obtienen pares *«definiendum, definición»* de oraciones definitorias mal interpretadas como voz pasiva por el parser y pares *«palabra, descripción»* de oraciones realmente en voz pasiva. De ahora en más se hace referencia a *palabra* y *pista* o par *«palabra, pista»* para abarcar a ambas estructuras.

El esquema de la figura 4.13 ilustra la abstracción de un árbol de dependencias para las oraciones en voz pasiva analizadas por el parser de `Stanford`.

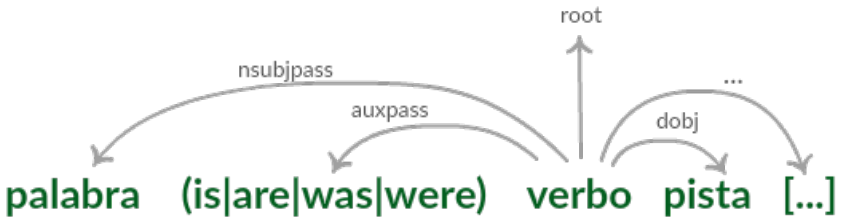


FIGURA 4.13: Diagrama de dependencias para las oraciones en voz pasiva

Versión final del Extractor

Volviendo al ejemplo de la figura 4.12, se puede observar también que el sustantivo *familiaris* es dependiente de *Dogs* con rótulo *appos* (de appositional modifier), indicando que *"Canis lupus familiaris"* modifica a *Dogs*. Generalizando, si al obtener la *palabra* se tiene en cuenta solamente al sujeto, y se ignora a sus modificadores (suponiendo que no especifican demasiada información extra), se pueden cubrir más *pistas* existentes en oraciones de un corpus.

Bajo las consideraciones anteriores, se programó el algoritmo de extracción de definiciones o pistas que recibe como entrada una oración en texto plano. El mismo fue aplicado sobre los corpus de *Simple English Wikipedia* y *Ducksters*, obteniendo así un *Conjunto Inicial de Pares «palabra, pista»*. También está incluido en el módulo Extractor que procesa los textos ingresados por el usuario de la aplicación en su modalidad *on-demand*.

En el cuadro 4.2 se detallan los resultados obtenidos en términos numéricos.

Corpus	Cantidad de pares «palabra,pista»	Cantidad de palabras distintas	Promedio de pistas por palabra
Simple English Wikipedia	51.607	33.060	1,5
Ducksters	688	352	1,9

Cuadro 4.2: Resultados numéricos del *Conjunto Inicial de Pares*

Analizando los 688 pares que fueron extraídas del corpus de *Ducksters*, se tiene que 275 (aproximadamente un 40 %) son extraídos de oraciones en voz pasiva según el parser de `Stanford`, de los cuales solo dos resultan de un análisis incorrecto:

- *«Giraffes, Not endangered, but many do live in protected areas».*
- *«Lava, Melted or liquid rock»*

En ambos casos se malinterpreta a un adjetivo como un verbo. Por ejemplo, en el segundo caso, el parser etiqueta incorrectamente a la palabra *Melted* con VBN (Verb, past participle) (ver figura 4.14).

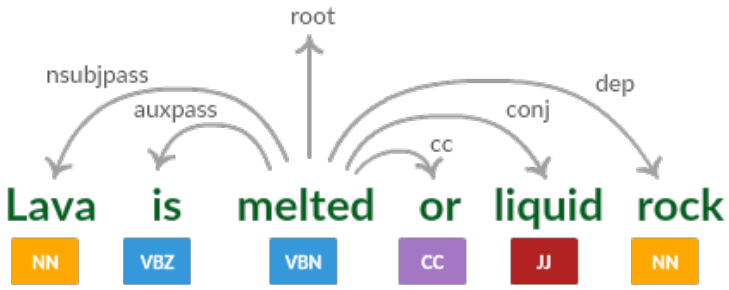


Figura 4.14: Diagrama de dependencias para “Lava is melted or liquid rock”

4.6. Conjunto de pares «palabra, pista»

En esta sección se detalla cómo se construye el conjunto de pares *«palabra,pista»* como datos pre-procesados para crucigramas, que se utiliza como entrada tanto para el generador automático de crucigramas como para el de sopa de letras.

4.6.1. Análisis del Conjunto Inicial de Pares

Estudiando el *Conjunto Inicial de Pares* se observa que las pistas son correctas en el sentido de que son grupos sintácticos bien formados.

Profundizando un poco más en el análisis, se observa que de los pares obtenidos de

Simple English Wikipedia son pocos los apropiados para utilizar en los juegos. Se debe tener en cuenta que estos pretenden enseñar inglés como segunda lengua a niños, por lo que es de interés solucionar los siguientes problemas:

1. **Las palabras o pistas son de uso poco frecuente**. Por ejemplo para *Liskeard* y su respectiva pista: *"A town near the A38 road in Cornwall, England"*, la cantidad de apariciones de *Liskeard* y *A38* en el corpus de *Simple English Wikipedia* es de 8 y 11 respectivamente. Para comparar, se tiene que las de *blue, green* y *red* en el mismo corpus son de 3.330, 3.539 y 5.918 respectivamente.

2. **Las palabras o pistas usan vocabulario inapropiado para niños**. Son pocos los pares que presentan este problema, pero es un punto importante a tener en cuenta.

3. **Las pistas dependen demasiado del contexto de los textos desde donde son extraídas**. Por ejemplo, para la palabra genérica *puppy* se obtiene la pista específica *"Usually born completely white and the spots develop a few weeks after they are born"*. Esto se da porque el par *«palabra,pista»* es extraído del artículo *"Dalmatian (dog)"* de *Simple English Wikipedia*. La pista contiene una correferencia: *the spots* referencia a la oración inmediatamente anterior en el artículo.

4. **Se encuentran múltiples pistas para una misma palabra**. Dentro del conjunto hay palabras ambiguas, donde interesa seleccionar aquella que tiene el significado de uso más frecuente. También sucede que alguna de las pistas puede estar asociada al significado de interés pero su calidad no es buena. Por ejemplo, para *black* se extraen 14 pistas, de las cuales:

   - Cinco hacen referencia al color. Un ejemplo es *"A dark color, the darkest color there is"*, que sería una buena pista para un crucigramas para niños. Otro ejemplo es: *"Not a desired color in the show ring but not a disqualification"*, que no es una buena pista ya que presenta el problema anterior (depende demasiado del contexto).
   - Una referencia a una compañía: *"A British book publishing company"*.
   - Ocho referencian a apellidos de personas (ejemplos: *"Also known by his nicknames, Jables or JB"* y *"A strong supporter of Franklin D. Roosevelt for President"*).

4.6.2. Conjunto Intermedio de Pares

Para minimizar los problemas *1.* y *2.*, lo primero que se realizó fue filtrar el conjunto inicial de pares quedándose solo con aquellos para los cuales la palabra está incluida en la *Lista Categorizada Ampliada* (ver sección 4.4.3). Esto se realizó en busca de que las palabras a incluir en los juegos sean las adecuadas para enseñar en el nivel de inglés de interés para este proyecto. De esta forma se reduce el conjunto inicial, obteniendo una versión que llamaremos *«Conjunto Intermedio de Pares»* el cual contiene 1.202 pares.

Del mismo se obtienen los siguientes resultados:

Corpus	**Cantidad de pares «palabra,pista»**	**Cantidad de palabras distintas**	**Promedio de pistas por palabra**
Simple English Wikipedia	1.135	260	4,3
Ducksters	67	36	1,8

Cuadro 4.3: Resultados numéricos del *Conjunto Intermedio de Pares*

Una observación interesante del cuadro es cómo aumenta el promedio de pistas por palabra respecto al *Conjunto Inicial de Pares*. En el caso del corpus de *Simple English Wikipedia* aumenta de **1,5** a **4,3**. Esto indica que palabras frecuentes o de baja complejidad, como lo son las de la *Lista Categorizada Ampliada*, tienden a aparecer más como sujeto en oraciones que usan ciertas conjugaciones del verbo *"to be"*, donde de alguna forma dichas oraciones describen o definen a las palabras. Esto en parte se debe a que las palabras de uso más frecuente tienden a ser más ambiguas (Baars y Gage, 2010). En la siguiente sección se verán ejemplos concretos de este fenómeno.

Lo segundo que se realizó, en particular para reducir los problemas *3.* y *4.*, fue definir ciertos criterios y experimentar con dos heurísticas que son detalladas a continuación con el fin de construir lo que llamaremos *«Conjunto Final de Pares»*.

4.6.3. Conjunto Final de Pares

Para la construcción de este conjunto se decide seleccionar para palabras del *Conjunto Intermedio de Pares* la pista más "correcta" con la ayuda de criterios y métricas detallados más adelante. Se destacan dos dimensiones que determinan la correctitud de una pista para un juego:

- Por un lado, un humano debería ser capaz de identificar la palabra a partir de la pista sin mayores dificultades. Esta dimensión de correctitud está asociada a la *completitud*.
- Por otro lado, la pista debe definir o describir a la palabra suficientemente bien como para que un humano no la confunda con otro significado que no es de interés. Esta dimensión está asociada a la *ambigüedad*.

Criterio de calidad

El criterio para decidir si una pista es de buena o mala **calidad**, se define en base a las dimensiones de *completitud* y *ambigüedad* ya detalladas. Para que una pista sea correcta en este sentido, debe cumplir con ambas condiciones, esto es, ser completa y que refiera significado de interés. El criterio es difícil de definir, ya que clasificar un par «palabra, pista» como de buena o mala **calidad**, es una opinión subjetiva.

Los ejemplos del cuadro 4.4 pretenden ilustrar el criterio definido, tomando como pistas de buena **calidad** las clasificadas como *positivas* y en el caso opuesto como *negativas*.

Categoría	**Palabra**	**Pista positiva**	**Pista negativa**
House	Bed	"A piece of furniture that people sleep on."	"The smallest lithostratigraphic unit."
Animals	Bat	"Mammals in the order Chiroptera."	"A successful group."
Colors	Green	"The colour of grass or leaves."	"A color."
Colors	Orange	"Between the red and yellow colors in a rainbow."	"A type of citrus fruit which people often eat."

Cuadro 4.4: Ejemplos de criterio de calidad

Del cuadro 4.4 se observa que:

- La pista positiva para *Bed* es de buena **calidad** ya que la define unívocamente como cama. La pista negativa define muy bien al concepto de *Bed* como unidad geológica, pero en este caso no es el significado de interés.
- Para el ejemplo de *Bat* la pista positiva la define unívocamente pero incluye vocabulario complejo. Se toma *"A successful group"* como negativa, porque por más que contenga vocabulario simple y esté asociada al significado de interés (es una afirmación respecto a los murciélagos que proviene del artículo *Bat*), como pista en sí es demasiado vaga.
- El ejemplo de *Green* muestra dos pistas con vocabulario sencillo, la diferencia está en que la pista positiva especifica suficientemente bien al color, y la negativa es demasiado genérica.
- El ejemplo de *Orange* es interesante, ya que las dos pistas refieren a dos significados de uso muy frecuente. Se marca como negativa a la que refiere a la fruta ya que la categoría asociada a la palabra es *Colors* y no *Foods*.

En este sentido se define la métrica **Precisión de calidad** como el porcentaje de pares clasificados como positivos respecto al total.

Criterio de Desambiguación

Para este criterio se tuvo en cuenta solamente la dimensión de ambigüedad y no la de completitud. El cuadro 4.5 ejemplifica pistas positivas y negativas según este criterio.

Categoría	Palabra	Pista positiva	Pista negativa
House	Bed	"A piece of furniture that people sleep on"	"The smallest lithostratigraphic unit"
Animals	Bat	"Mammals in the order Chiroptera"	"A kind of club, though the size and shape depend on the rules"
Animals	Dog	"Smuggled out of Hungary during the Second World War"	"Made of cheese"

Cuadro 4.5: Ejemplos de criterio de desambiguación

Del cuadro se puede observar que:

- Las pistas para *Bed* coinciden en su clasificación con las del criterio de **calidad** ya que, de nuevo, el significado de interés es el de la cama.
- El ejemplo de *Bat*, teniendo en cuenta que la categoría asociada es *Animals*, tiene una pista negativa ya que refiere al *Bat* usado en los deportes. La pista positiva utiliza vocabulario complejo, pero define al concepto de interés.
- Para *Dog*, se tiene al ejemplo positivo como tal ya que sí habla de perros, en particular de la raza *Hungarian Vizsla* (la pista es extraída de un artículo con este nombre). La pista *"Made of cheese"* proviene del artículo *Fact* bajo el título *"False statements"* y es por esto que se toma como negativa. Se puede decir que esta pista refiere a un perro hipotético y no a uno real como es de interés.

El criterio para clasificar las pistas como positivas o negativas en este caso es objetivo. Esto es así porque el contexto de las oraciones desde donde se extraen las pistas viene dado por los artículos que contienen la oración, ayudando así a determinar el significado de la palabra asociada. A partir de lo anterior se define la métrica **Precisión de desambiguación** análogamente a la **Precisión de calidad**.

Ejemplos de conjuntos de evaluación

Para evaluar la **Precisión de calidad** se puede utilizar cualquier **conjunto** de pares *«palabra, pista»* que estén categorizados, ya que la categoría ayuda a darle el contexto a la palabra. Sin embargo, para evaluar la **Precisión de desambiguación**, se requiere que sea un conjunto de pares categorizados que correspondan a palabras ambiguas que contengan más de una pista con diferentes significados.

Un ejemplo de conjunto para la evaluación de la **Precisión de calidad** sería el de la figura 4.15.

Categoría	Palabra	Pista
Animals	Bat	Also a symbol of ghosts, death and disease.
House	Bed	The smallest lithostratigraphic unit.
Body	Ear	Also used in other ways.
Foods	Orange	Also a very good source of dietary fibre.
Animals	Bat	A successful group.
Animals	Bat	A kind of club, though the size and shape depend on the rules.
House	Bed	A piece of furniture that people sleep on.
Foods	Orange	An important food source in many parts of the world for several reasons.
Weather	Warm	The opposite of cool.
Foods	Orange	A type of citrus fruit which people often eat.
Foods	Orange	The color of Autumn and harvest.
Animals	Bat	Mammals in the order Chiroptera.
Foods	Orange	A city in New South Wales, Australia.
Foods	Orange	A town of Juneau County in the state of Wisconsin in the United States.
Sports	Hockey	A game based on this sport.
Foods	Orange	Grown in many parts of California.
Foods	Orange	A color.
Foods	Orange	A village of Cuyahoga County, Ohio, United States.
Animals	Bat	Sometimes used as a weapon in fights or attacks.
Sports	Hockey	Played by both men and women at the Olympic games, and at world championships.

FIGURA 4.15: Ejemplo de conjunto para evaluar Precisión de calidad

Por otro lado, la figura 4.16 muestra un ejemplo de conjunto para evaluar la **Precisión de desambiguación**.

Categoría	Palabra	Pista
Animals	Bat	Also a symbol of ghosts, death and disease.
House	Bed	The smallest lithostratigraphic unit.
Foods	Orange	Also a very good source of dietary fibre.
Animals	Bat	A successful group.
Animals	Bat	A kind of club, though the size and shape depend on the rules.
House	Bed	A piece of furniture that people sleep on.
Foods	Orange	An important food source in many parts of the world for several reasons.
Foods	Orange	A type of citrus fruit which people often eat.
Foods	Orange	The color of Autumn and harvest.
Animals	Bat	Mammals in the order Chiroptera.
Foods	Orange	A city in New South Wales, Australia.
Foods	Orange	A town of Juneau County in the state of Wisconsin in the United States.
Foods	Orange	Grown in many parts of California.
Foods	Orange	A color.
Foods	Orange	A village of Cuyahoga County, Ohio, United States.
Animals	Bat	Sometimes used as a weapon in fights or attacks.

FIGURA 4.16: Ejemplo de conjunto para evaluar Precisión de desambiguación

Como se puede apreciar en la figura 4.15 las palabras *Ear* y *Warm* tienen una sola pista, y *Hockey* tiene dos pistas asociadas que refieren al deporte. Para estos ejemplos no tiene sentido evaluar la capacidad de desambiguación ya que no hay significados de pista disintos para desambiguar, y es por esto que no aparecen en el conjunto. En base a estos conjuntos de ejemplo se ilustrará la aplicación de las dos heurísticas mencionadas en la próxima subsección.

Definición de Heurísticas

Con el objetivo de obtener la pista más correcta por palabra, se definen las siguientes heurísticas.

Heurística 1 - Promedio de similitudes: Para un par *«palabra, pista»*, se calcula el **promedio de las similitudes** entre los vectores asociados a las palabras de la pista (descartando las *stop-words* ya que en pocos casos aportan significado semántico considerable) y el vector asociado a la palabra a definir o describir. Las representaciones vectoriales de las palabras son las mismas que se utilizaron para la *Ampliación de la Lista Categorizada* (ver capítulo 4.4).

Cálculo del Promedio de similitudes

$$promedioSimilitudes(palabra, pista) = \frac{\sum_{i=1}^{n}(similitudCoseno(\vec{palabra}, \vec{palabraPista_i}))}{n}$$

El $\boldsymbol{n}$ de la fórmula es la cantidad total de palabras de la pista sin contar las *stop-words*.

A continuación, a modo de ejemplo, se muestra cómo se calcula el **promedio de similitudes** para el par *«Bed, A piece of furniture that people sleep on.»*:

$$\begin{aligned}&(similitudCoseno(\vec{bed}, \vec{piece})\\ &+similitudCoseno(\vec{bed}, \vec{furniture})\\ &+similitudCoseno(\vec{bed}, \vec{people})\\ &+similitudCoseno(\vec{bed}, \vec{sleep}))/4\end{aligned}$$

La idea detrás de esta heurística es que este promedio ayude a seleccionar para una palabra dada, una de sus pistas que "más se asemeja" a la misma. En base a lo anterior, si se calcula la similitud promedio para todos los pares de un conjunto, se obtendría un subconjunto con la pista "más correcta" por palabra.

Heurística 2 - Similitud Basada en Centroide: En esta heurística se calcula para un par *«palabra, pista»* la similitud coseno entre:

1. El **centroide** de la palabra y su categoría.
2. El **centroide** de las palabras de la pista sin contar las *stop-words*.

El primer centroide se calcula de esta forma con el propósito "potenciar" a la palabra con su categoría como se realiza en el procedimiento de ampliación de la *Lista Categorizada* (ver sección 4.4), de alguna forma desambiguando la misma en caso de ser una palabra ambigua. Esto se hace ya que un problema que presentan las representaciones vectoriales utilizadas es que no tienen en cuenta la ambigüedad de las palabras, por lo que "sumarle contexto" daría un vector que representa mejor el significado de interés.

A partir del segundo centroide, se obtiene una representación vectorial de la pista. Luego, calculando la similitud coseno entre el centroide de la pista y el centroide de os vectores de la palabra y la categoría, se obtendría cuán "cercana" es la pista a la palabra.

Similitud Basada en Centroide

$$similitudBasadaEnCentroide(categoría, palabra, pista) =$$
$$similitudCoseno((\vec{categoría} + \vec{palabra})/2, \sum_{i=1}^{n} \vec{palabraPista\_i}/n)$$

A continuación se presenta muestra cómo se calcula la **Similitud Basada en Centroide** para el par *«Bed, A piece of furniture that people sleep on»* de la categoría *House*:

$$similitudCoseno((\vec{house} + \vec{bed})/2, (\vec{piece} + \vec{furniture} + \vec{people} + \vec{sleep})/4)$$

$$similitudCoseno((\vec{colors} + \vec{black})/2, (\vec{dark} + \vec{color} + \vec{darkest} + \vec{color})/4)$$

Aplicación de Heurísticas

El esquema de la figura 4.17 muestra el proceso de aplicación de una heurística sobre un conjunto de pares de entrada obteniéndose una una medida de precisión como salida.

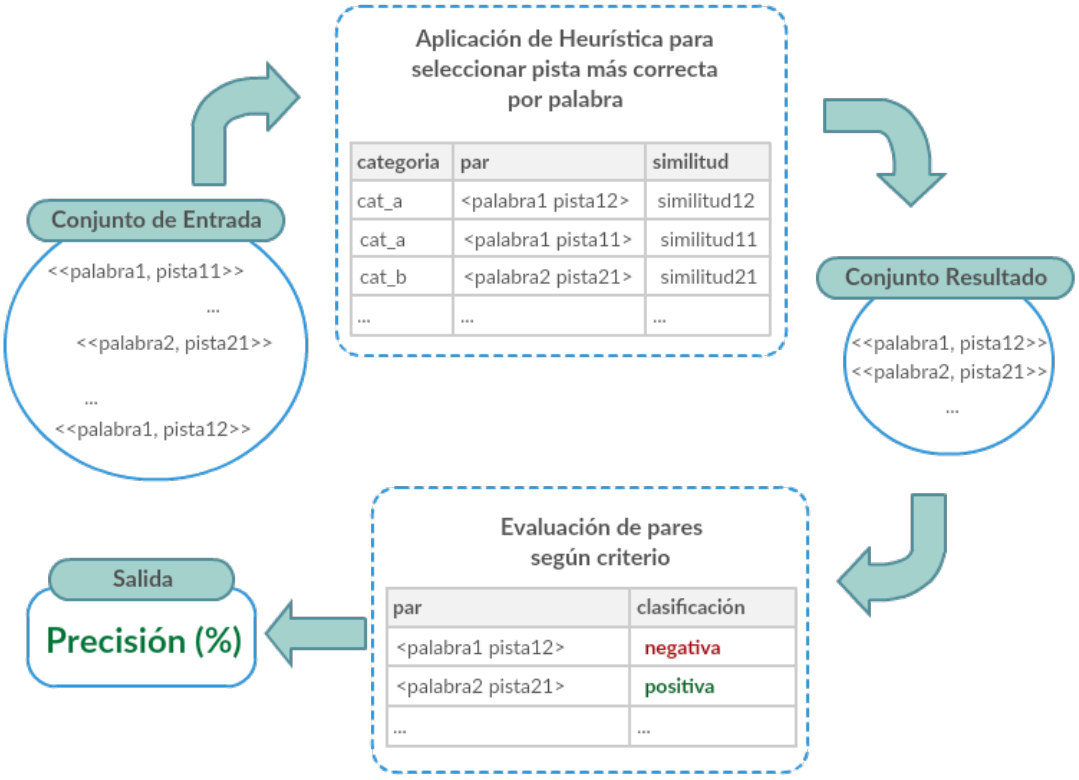


FIGURA 4.17: Proceso de aplicación de heurística para evaluar una precisión

Al aplicar la heurística **Promedio de Similitudes** sobre conjuntos de pares *«palabra, pista»*, se obtiene como resultado intermedio a los pares agrupados por palabra y ordenados por el valor **promedio de similitudes** como se muestra en la figura 4.18.

Categoría	Palabra	Pista	Promedio de Similitudes
House	Bed	A piece of furniture that people sleep on.	0.1591698509
House	Bed	The smallest lithostratigraphic unit.	0.0105992407
Animals	Bat	Mammals in the order Chiroptera.	0.1952548799
Animals	Bat	Also a symbol of ghosts, death and disease.	0.094790916
Animals	Bat	Sometimes used as a weapon in fights or attacks.	0.0592006759
Animals	Bat	A kind of club, though the size and shape depend on the rules.	0.0415365357
Animals	Bat	A successful group.	0.0285989963
Body	Ear	Also used in other ways.	0.0623353761
Foods	Orange	A color.	0.1754965877
Foods	Orange	Grown in many parts of California.	0.1206199702
Foods	Orange	The color of Autumn and harvest.	0.0939656318
Foods	Orange	Also a very good source of dietary fibre.	0.0897590403
Foods	Orange	A type of citrus fruit which people often eat.	0.0796434225
Foods	Orange	A town of Juneau County in the state of Wisconsin in the United States.	0.078809301
Foods	Orange	A city in New South Wales, Australia.	0.0786443649
Foods	Orange	A village of Cuyahoga County, Ohio, United States.	0.0674248451
Foods	Orange	An important food source in many parts of the world for several reasons.	0.0642791545
Weather	Warm	The opposite of cool.	0.1909018956
Sports	Hockey	A game based on this sport.	0.1712143306
Sports	Hockey	Played by both men and women at the Olympic games, and at world championships.	0.123214845

FIGURA 4.18: Ejemplo de Conjunto de Pares ordenado según el **Promedio de similitudes**

Por otro lado, al aplicar la **Similitud Basada en Centroide** sobre conjuntos de pares *«palabra, pista»* se los obtiene también agrupados por palabra pero en este caso ordenados por el valor **similitud casada en centroide** (ver figura 4.19).

Categoría	Palabra	Pista	Similitud Basada en Centroide
House	Bed	A piece of furniture that people sleep on.	0.4329408109
House	Bed	The smallest lithostratigraphic unit.	0.0784713775
Animals	Bat	Mammals in the order Chiroptera.	0.5271179676
Animals	Bat	Also a symbol of ghosts, death and disease.	0.3052933812
Animals	Bat	Sometimes used as a weapon in fights or attacks.	0.1831768602
Animals	Bat	A kind of club, though the size and shape depend on the rules.	0.1721144021
Animals	Bat	A successful group.	0.0389671102
Body	Ear	Also used in other ways.	0.1615584344
Foods	Orange	A type of citrus fruit which people often eat.	0.4683513939
Foods	Orange	Also a very good source of dietary fibre.	0.3346984684
Foods	Orange	An important food source in many parts of the world for several reasons.	0.3040527105
Foods	Orange	Grown in many parts of California.	0.2603227794
Foods	Orange	The color of Autumn and harvest.	0.2344199121
Foods	Orange	A color.	0.2153041214
Foods	Orange	A village of Cuyahoga County, Ohio, United States.	0.1766938567
Foods	Orange	A town of Juneau County in the state of Wisconsin in the United States.	0.17621167
Foods	Orange	A city in New South Wales, Australia.	0.1499511451
Weather	Warm	The opposite of cool.	0.2591627538
Sports	Hockey	A game based on this sport.	0.4676141739
Sports	Hockey	Played by both men and women at the Olympic games, and at world championships.	0.3427009583

FIGURA 4.19: Ejemplo de Conjunto de Pares ordenado según **Similitud Basada en Centroide**

Resultado y Evaluación de métricas

Se utilizaron dos conjuntos distintos de datos para evaluar por un lado la métrica **Precisión de calidad** y por otro la **Precisión de desambiguación**, ya que este último requiere de las condiciones anteriormente mencionadas.

El utilizado para medir la **Precisión de calidad** fue construido en base a los pares del *Conjunto intermedio de pares* obtenidos del corpus de *Simple English Wikipedia*, que al tener en común sus palabras con las de la *Lista categorizada ampliada*, se le agrega a cada par su categoría asociada. De ahora en más llamaremos a este *«Conjunto de*

Evaluación de Calidad» y está conformado por un total de 1.135 elementos.

Por otro lado, se define el conjunto para medir la **Precisión de desambiguación**, el cual se crea a partir de la selección de aquellas instancias del *Conjunto de Evaluación de Calidad* donde las palabras ambiguas tienen pistas que refieren a por lo menos dos de sus significados. De esta forma se obtiene un conjunto de pares reducido con 629 elementos, al cual llamaremos *«Conjunto de Evaluación de Desambiguación»*.

A modo de comparación, se presenta en el cuadro 4.6 los resultados de ambos conjuntos en términos numéricos.

Conjunto	**Cantidad de pares «palabra,pista»**	**Cantidad de palabras distintas**	**Promedio de pistas por palabra**	**Cantidad de categorías**
Evaluación de Calidad	1.135	260	4,3	10
Evalución de Desambiguación	629	87	7,2	9

CUADRO 4.6: Resultados numéricos de los conjuntos de evaluación de *Calidad* y de *Desambiguación*

Una primera observación interesante es cómo aumenta el promedio de pistas por palabra en el *Conjunto de Evaluación de Desambiguación* respecto al de *Calidad* (de **4,3** a **7,2**), debido en parte a que las palabras ambiguas naturalmente presentarían más pistas por tener múltiples significados. También se nota que la cantidad de categorías disminuye en 1 para el *Conjunto de Evaluación de Desambiguación*. Esto se dio porque la categoría *Numbers* no incluía más de una pista para ningún número con más de un significado.

Resultados de la heurística 1 - Promedio de similitudes

Para comenzar, en el *Conjunto de Evaluación de Calidad* se seleccionan 260 pares, y se los clasifica manualmente según la métrica de **Precisión de calidad**, obteniendo así un total de 166 pares positivos y 94 negativos. La figura 4.20 muestra ejemplos de pares seleccionados, indicando en verde si son positivos y en rojo si son negativos.

Categoría	Palabra	Pista
House	Bed	A piece of furniture that people sleep on.
Animals	Bat	Mammals in the order Chiroptera.
Body	Ear	Also used in other ways.
Foods	Orange	A color.
Weather	Warm	The opposite of cool.
Sports	Hockey	A game based on this sport.

FIGURA 4.20: Ejemplos de pares «palabra,pista» clasificados según la **Precisión de calidad** con mayor *promedio de similitudes*

En el *Conjunto de evaluación de desmbiguación* se seleccionan 87 pares, donde 23 pistas fueron incorrectamente desambiguadas, y 64 fueron correctamente desambiguadas (ver ejemplos en la imagen 4.21).

Categoría	Palabra	Pista
House	Bed	A piece of furniture that people sleep on.
Animals	Bat	Mammals in the order Chiroptera.
Foods	Orange	A color.

FIGURA 4.21: Ejemplos de pares *«palabra, pista»* clasificados según la **Precisión de desambiguación** con mayor *promedio de similitudes*

Los resultados de ambas precisiones utilizando la heurística de **Promedio de similitudes** son:

- *Precisión de calidad*: **63.8 %**.
- *Precisión de desambiguación*: **73.6 %**.

Resultados de la heurística 2 - Similitud Basada en Centroide
Nuevamente, en el *Conjunto de Evaluación de Calidad* se seleccionan 260 pares, y según la métrica de **Precisión de calidad** se obtienen 178 pares positivos y 82 negativos. La figura 4.22 muestra ejemplos de pares positivos y negativos seleccionados.

Categoría	Palabra	Pista
House	Bed	A piece of furniture that people sleep on.
Animals	Bat	Mammals in the order Chiroptera.
Body	Ear	Also used in other ways.
Foods	Orange	A type of citrus fruit which people often eat.
Weather	Warm	The opposite of cool.
Sports	Hockey	A game based on this sport.

FIGURA 4.22: Ejemplos de pares «palabra,pista» clasificados según la **Precisión de calidad** con mayor *Similitud Basada en Centroide*

En el *Conjunto de evaluación de desmbiguación* 9 pistas fueron incorrectamente desambiguadas, y 78 fueron correctamente desambiguadas. En la imagen 4.23 se muestran algunos ejemplos.

Categoría	Palabra	Pista
House	Bed	A piece of furniture that people sleep on.
Animals	Bat	Mammals in the order Chiroptera.
Foods	Orange	A type of citrus fruit which people often eat.

FIGURA 4.23: Ejemplos de pares *«palabra,pista»* clasificados según la **Precisión de desambiguación** con mayor *Similitud Basada en Centroide*

Los resultados de ambas precisiones utilizando la heurística de **Similitud Basada en Centroide** son:

- *Precisión de calidad*: **68.5 %**.
- *Precisión de desambiguación*: **89.7 %**.

Comparación de heurísticas y Conjunto Final de Pares

La heurística de **Similitud Basada en Centroide** es la que da mejores resultados, tanto para la *Precisión de Calidad* como para la *Precisión de Desambiguación*. Por esta

razón se incluyen los pares seleccionados de dicha heurística, donde un **68.5 %** de las palabras tienen una calidad suficientemente buena, y casi un **89.7 %** de las palabras tienen una pista que refiere al significado de interés.

A diferencia de *Simple English Wikipedia*, los pares obtenidos de *Ducksters* tienen mayor probabilidad de ser correctos, en parte porque es un sitio enfocado a niños. Esto también se asegura luego de evaluarle la *Precisión de Calidad*, que es de un **82,1 %**. Por esta razón no se aplica la heurística de **Similitud Basada en Centroide**, incluyéndose entonces todas las pistas obtenidas de esta fuente.

20Para finalizar se complementó el *Conjunto Final de Pares* con las definiciones de las palabras pertenecientes a la *Lista Categorizada Ampliada* que fueron extraídas del Diccionario `Wordsmyth`. La unión de estos tres conjuntos da el *Conjunto Final de Pares* y en la tabla 4.7 que sigue muestra los resultados numéricos del mismo.

Fuente	**Cantidad de pares «palabra,pista»**	**Cantidad de palabras distintas**	**Promedio de pistas por palabra**
Simple English Wikipedia	260	260	1,0
Ducksters	67	36	1,8
Wordsmyth	371	371	1,0
Conjunto Final de Pares	**698**	**371**	**1,9**

Cuadro 4.7: Resultados numéricos del Conjunto Final de Pares

4.6.4. Persistencia

La extracción de definiciones a partir de los corpus se persiste en un archivo con formato `JSON`, bajo el nombre *data_crossword.json* el cual contiene un documento con los siguientes datos:

```
{
    "animals": [
        {"definicion" : "the adult female of cattle",
         "definiendum" : "cow"},
        {"definicion" : "a short form of hippopotamus",
         "definiendum" : "hippo"},
        (...)],
    "food": [
        {"definicion" : "A fruit tree, and the fruit itself",
         "definiendum" : "lime"},
        {"definicion" : "the flesh of animals when used as food",
         "definiendum" : "meat"},
        (...)],
    (...)
}
```

Cuando la extracción se realiza on-demand el archivo generado se denomina *data_crossword_ondemand.json* y tiene el siguiente formato:

```
{
    "clues": [
        {"definicion":"A pale reddish color",
         "definiendum":"Pink"},
        {"definicion": "A married man.",
         "definiendum": "husband"},
        (...)]
}
```

4.7. Segmentador de oraciones

4.7.1. Implementación

El *Segmentador de oraciones* es un procedimiento desarrollado en `Python` con `NLTK` y el parser de `Stanford` que, dada una oración de entrada, devuelve su sujeto y predicado junto con el número y persona correspondiente (Figura 4.24), persistiendo los resultados en una base de datos implementada en `MongoDB`. El objetivo de la segmentación de oraciones es utilizar los sujetos y predicados para generar los tableros de la batalla naval (como se verá en el siguiente capítulo).

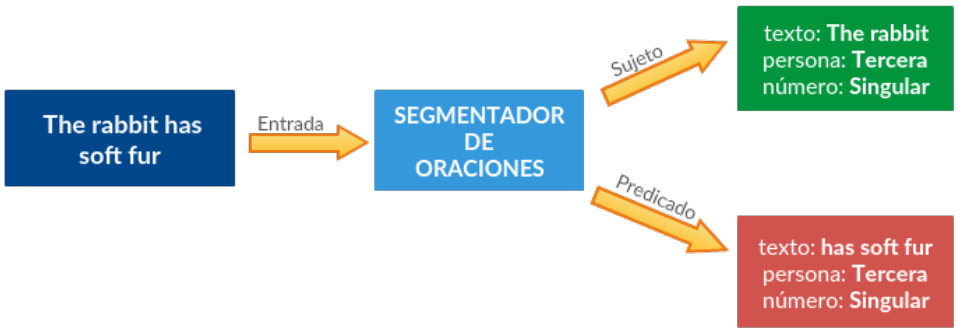


FIGURA 4.24: Segmentador de oraciones

Para obtener ambas partes de la oración, primero se genera su árbol de dependencias utilizando el parser de `Stanford`. Una vez obtenido el árbol, se procede a analizar la categoría gramatical de su raíz para reconstruir el sujeto y predicado de la oración. A su vez, el parser de `Stanford` devuelve para cada palabra su categoría gramatical.

Si la raíz del árbol pertenece a la categoría *verbo*, entonces una de sus ramas (izquierda o derecha) constituye el sujeto de la oración y la otra, incluyendo la raíz, constituye el predicado.

Para simplificar, en esta instancia solo contemplaremos los casos en que el sujeto se encuentra del lado izquierdo, esto es, antes del verbo. Para obtener el número y persona de la oración solo se analiza el sujeto ya que se asume que las oraciones están bien formadas y por lo tanto coinciden en esos dos rasgos.

4.7.2. Casos a analizar

A continuación se analizarán algunos casos, tomando como ejemplos oraciones que corresponden al texto de ejemplo de la subsección 4.3.4 perteneciente al corpus ESLFast.

Caso 1

En estos casos la raíz del árbol pertenece a la categoría gramatical *verbo*, es decir, coincide con cualquiera de las siguientes etiquetas de `Stanford` para verbos:

- VB - Verb, base form
- VBD - Verb, past tense
- VBG - Verb, gerund or present participle
- VBN - Verb, past participle

- VBP - Verb, non 3rd person singular present
- VBZ - Verb, 3rd person singular present

Para visualizar este caso presentamos las oraciones de la figura 4.25.

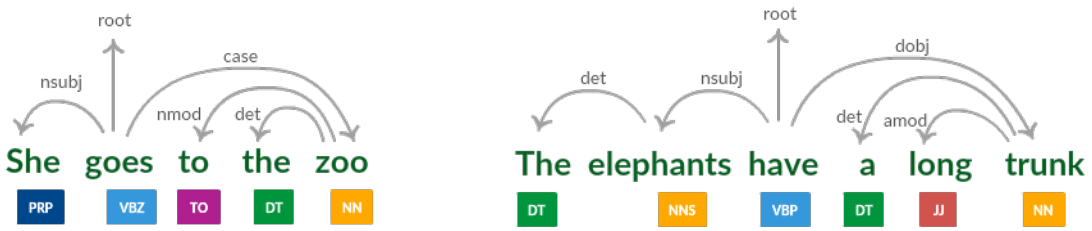


FIGURA 4.25: Árbol de dependencias - Caso 1

En ambas oraciones la raíz pertenece al grupo de etiquetas considerado antes.

Con respecto al número y persona de la oración se analiza la relación de la raíz con el dependiente a su izquierda: si la relación está etiquetada como "nsubj" o "nsubjpass" entonces se pasan a analizar los siguientes casos:

- Si el dependiente es la palabra *I*, entonces el número es *singular* y la persona es *primera*.
- Si el dependiente es la palabra *We*, entonces el número es *plural* y la persona es *primera*.
- Si el dependiente está etiquetado gramaticalmente como "NNS" o "NNPS", o la palabra es *They*, entonces el número es *plural* y la persona es *tercera*.
- En cualquier otro caso, el número es *singular* y la persona es *tercera*.

Es necesario aclarar que el corpus construido a partir de ESLFast no contaba con oraciones en segunda persona por lo que esta casuística no fue contemplada a la hora del análisis de número y persona.

En resumen, los resultados obtenidos para las oraciones de ejemplo se muestran en el cuadro 4.8.

Sujeto	**Predicado**	**Número**	**Persona**
She	goes to the zoo	*singular*	*tercera*
The elephants	have a long trunk	*plural*	*tercera*

CUADRO 4.8: Análisis de oraciones - Caso 1

Analizando la cantidad de oraciones donde el árbol de dependencia tiene como raíz un verbo obtuvimos un total de 8799, esto es un **79 %** del total de oraciones.

Caso 2

Este caso corresponde a oraciones incorrectamente etiquetadas por el parser de `Stanford`, donde la raíz es etiquetada como *sustantivo* pero según el contexto en el que se encuentra la palabra pertenece a la categoría *verbo*. Las etiquetas del parser para los sustantivos son las siguientes:

- NN - Noun, singular or mass
- NNS - Noun, plural
- NNP - Proper noun, singular
- NNPS - Proper noun, plural

El ejemplo para este caso se puede apreciar en la figura 4.26.

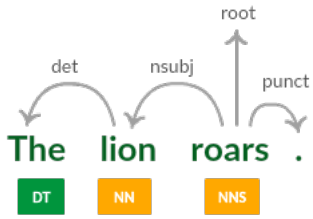


FIGURA 4.26: Árbol de dependencias - Caso 2

Se puede ver que el etiquetado es incorrecto puesto que, la palabra *roars* en este contexto es un verbo.

Ahora analizando la cantidad de oraciones donde el árbol de dependencia tiene como raíz un nombre obtuvimos un total de 1171, esto es un 10 % del total de oraciones, entre los cuales se tienen árboles correcta e incorrectamente etiquetados. Como el porcentaje es mucho menor al obtenido en el Caso 1, directamente se decidió descartar estas oraciones en el Segmentador.

Caso 3

El último caso a analizar es cuando el árbol está etiquetado de forma correcta pero la raíz no es un verbo, es decir, pertenece a las demas categorías: *adjetivos, adverbios, pronombres, etc.*

Un ejemplo para este caso es el que se presenta en la figura 4.27, donde la categoría de la raíz es el adjetivo *slow*. Esto ocurre debido al análisis de los verbos copulativos que hace el parser de `Stanford`, como se mencionó en la sección 4.5.1.

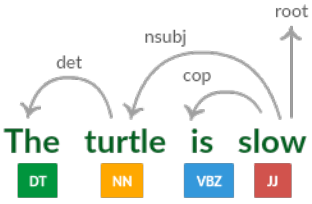


FIGURA 4.27: Árbol de dependencias - Caso 3

Consideramos que analizar las oraciones que corresponden a este caso de estudio debido a las diferentes casuísticas que se obtienen no resultaría una tarea útil, teniendo en cuenta que la cantidad de oraciones son solo un **11 %** del total. Es por esto que se decidió descartar estas oraciones en el *Segmentador*.

4.7.3. Resultados

Una vez finalizada la segmentación de todas las oraciones correspondientes al caso 1, es decir, un total de 8799, se eliminaron los sujetos y predicados repetidos y se obtuvieron los resultados de los cuadros 4.9 y 4.10.

Sujetos	*Singular*	*Plural*	*Total*
Primera Persona	1	1	2
Tercera Persona	810	316	1126
Total	811	317	**1128**

Cuadro 4.9: Resultados obtenidos para los sujetos

Predicados	*Singular*	*Plural*	*Total*
Primera Persona	57	22	79
Tercera Persona	6075	1021	7096
Total	6132	1043	**7175**

Cuadro 4.10: Resultados obtenidos para los predicados

Debido a que la obtención de estos estos sujetos y predicados tiene como objetivo ser utilizados para los tableros de la batalla naval, es necesario eliminar los casos que contienen muchos caracteres a los efectos de que se ajusten a la interfaz de la cuadrícula. Es por esto que se eliminaron los sujetos que contenían más de 16 caracteres y los predicados de más de 40 caracteres. Esto nos deja un total de 917 sujetos y 6134 predicados a persistir.

4.7.4. Persistencia

Al finalizar la segmentación de todas las oraciones, los resultados se persisten en dos archivos con formato `JSON`, uno para los sujetos y predicados que se encuentran en singular y otro para los que están en plural.

A modo de ejemplo se presenta a continuación un fragmento del archivo que contiene los datos en singular:

```
{
    {"sujetos":[
        {"persona":"T","sujeto":"The cat"},
        {"persona":"T","sujeto":"My father"},
        {"persona":"P","sujeto":"I"},
        (...)]
    "predicados":[
        {"persona":"T","predicado":"works at the bank."},
        {"persona":"T","predicado":"was obsessed with donuts."},
        {"persona":"P","predicado":"am practicing drawing.",
        (...)]}
}
```

Estos archivos se denominan *data_battleship_singular.json* y *data_battleship_plural.json*.

Capítulo 5

Aplicación *Fun with words*

5.1. Introducción

La aplicación web implementada lleva el nombre *"Fun with words"* y, como se expuso antes, contiene 3 tipos de juegos: crucigramas (*crosswords*), sopas de letras (*wordsearch*) y batalla naval (*battleship*). Los mismos fueron implementados específicamente para la "Ceibalita" modelo `Clamshell JP` [22] y el navegador `Mozilla Firefox`[1].

A continuación se presenta una captura de la pantalla incial de la aplicación:

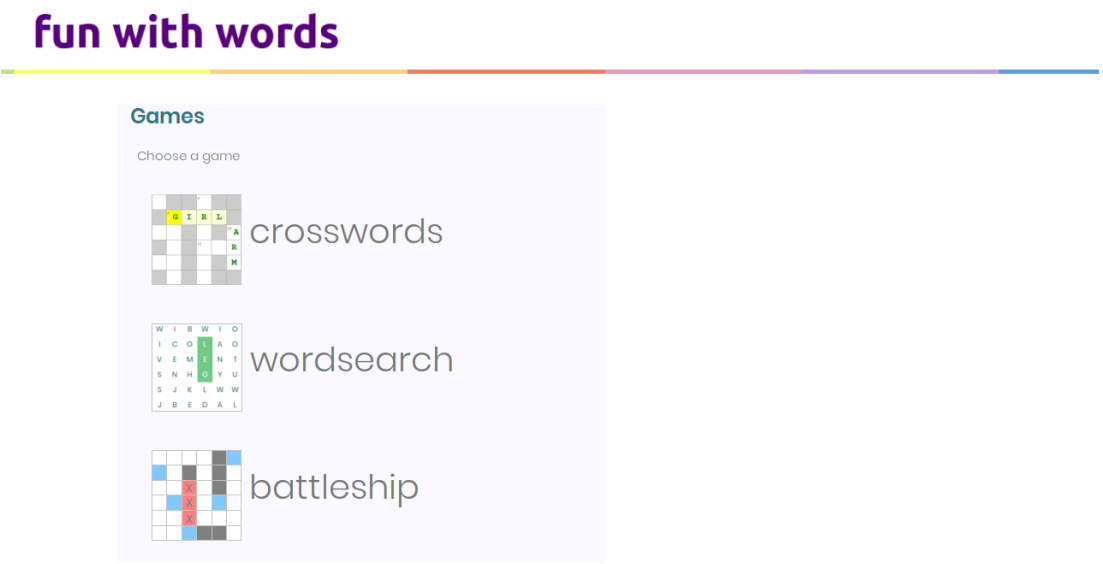


FIGURA 5.1: Captura de pantalla - Aplicación Fun with words

Una vez que el usuario selecciona el juego que desea jugar, se carga un panel a la derecha con los parámetros disponibles para ese juego y un botón más abajo con el que se confirman los datos ingresados para que la aplicación genere el tablero correspondiente.

En las secciones siguientes se describirá de forma más detallada cada uno de los juegos y su configuración.

[1]Mozilla Firefox es un navegador web libre y de código abierto desarrollado para Linux, Android, iOS, macOS y Microsoft Windows coordinado por la Corporación Mozilla y la Fundación Mozilla.

5.2. Crucigramas

5.2.1. Implementación

Para comenzar a jugar el usuario debe elegir entre dos modalidades:

1. **Generar** un tablero eligiendo los parámetros requeridos para su creación
   - Tamaño del tablero: las opciones disponibles en este caso son: 8x8, 10x10, 12x12 o 15x15.
   - Categoría de las palabras: se dispone de las categorías que fueron descritas en el capítulo anterior y además se agrega la categoría. *"all words"* que es la unión de todas.
2. **Importar** un tablero a través de un archivo `JSON` que ya fue generado por la aplicación anteriormente.

A continuación, se presenta una captura de pantalla de las opciones del juego una vez que el usuario eligió la opción *crucigramas*:

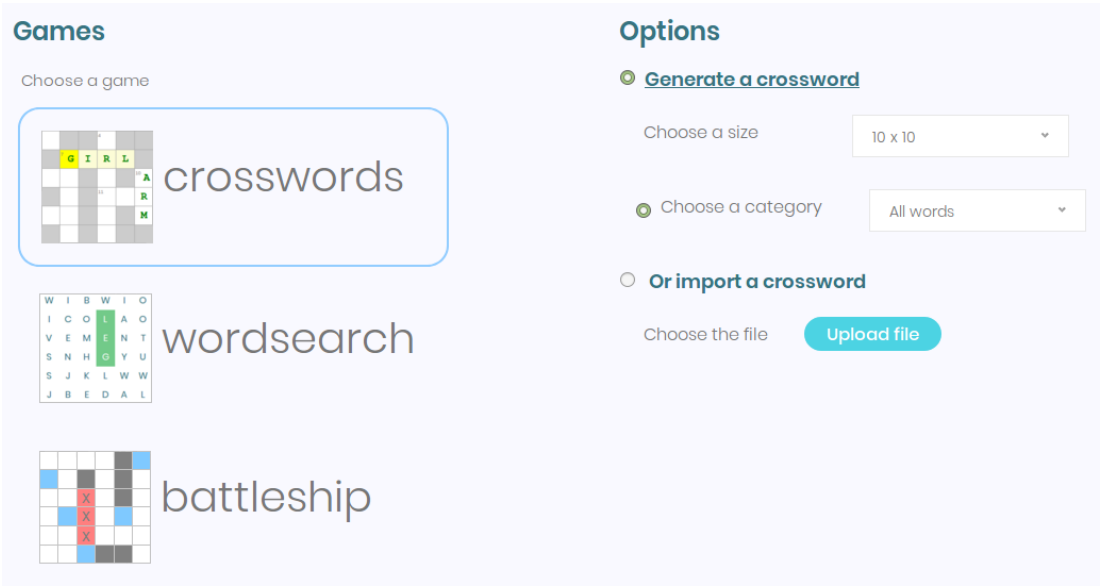


Figura 5.2: Captura de pantalla - Parámetros Crucigramas

El encargado de la creación del tablero es un algoritmo implementado en `Javascript` por Richard Rulach (Ver apéndice D) el cuál fue adaptado según las necesidades de este proyecto. Los datos que se utilizan para su generación son los pares *«palabra,pista»* que se encuentran en el archivo *data_crossword.json* antes descrito.

En la figura 5.3 se muestra en líneas generales el proceso de generación de crucigramas.

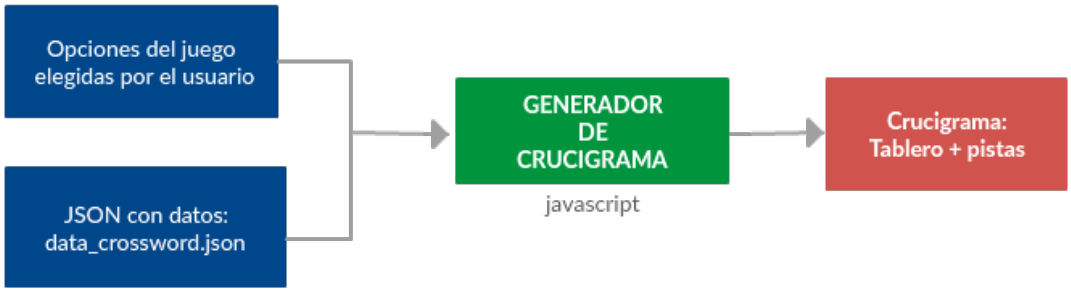


FIGURA 5.3: Solución para crucigramas en lineas generales

El algoritmo recibe todas las palabras de la categoría seleccionada y comienza a construir el tablero del tamaño requerido incorporando y quitando palabras al azar.

A modo de ejemplo, en la figura 5.4 se muestra cómo resulta un tablero generado cuando se selecciona tamaño 10x10 y categoría *Food*.

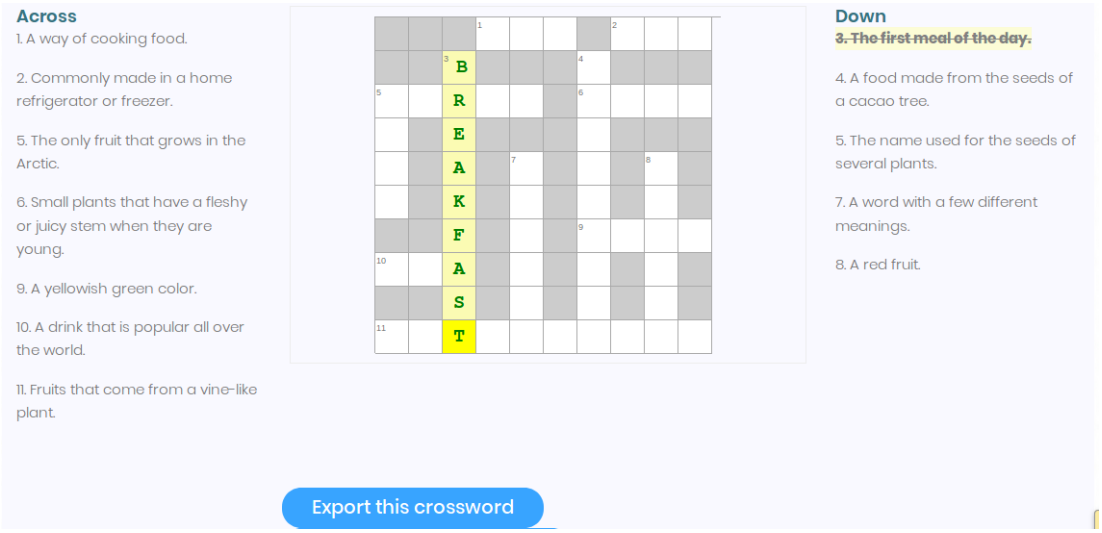


FIGURA 5.4: Captura de pantalla - Crucigramas - Tablero y pistas

Describiendo ahora la interfaz, se puede apreciar que el tablero a completar se encuentra en el medio y las pistas de las palabras horizontales y verticales del lado izquierdo y derecho, respectivamente.

Cuando el usuario selecciona una pista cualquiera, se iluminan con color amarillo los casilleros correspondientes a esa palabra para que el usuario la encuentre fácilmente. También funciona a la inversa: si el usuario selecciona un casillero del tablero, se ilumina con amarillo la pista que le corresponde.

Cuando el usuario completa las casillas con la palabra correcta, las letras se colorean de verde y la pista correspondiente se tacha. Si la palabra es incorrecta las letras se colorean de rojo.

Una vez completado todo el tablero se despliega un mensaje de éxito.

Debajo del tablero se encuentra el botón para exportarlo en caso de querer cargarlo en otra computadora. El archivo que se descarga está en formato `JSON` conteniendo toda la configuración del tablero generado.

Versión para docentes

Este juego tiene implementada una versión para docentes (Figura 5.5), que tiene como ventaja generar un crucigrama *"on-demand"* a partir de un texto trabajado en clase. También se le agregan las funcionalidades de modificar/corregir una pista del tablero generado si el docente la considera como incorrecta, o directamente puede eliminar la palabra.

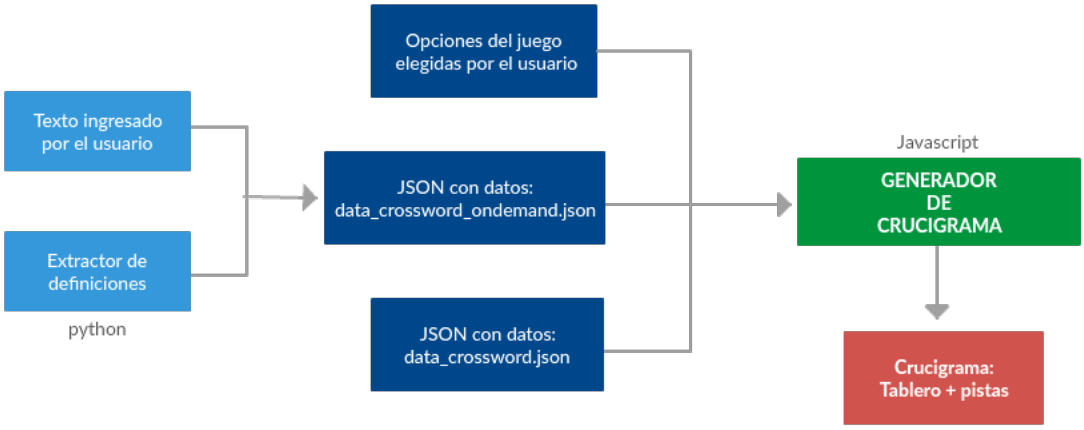


FIGURA 5.5: Solución para crucigramas - versión docentes

A continuación se muestra una captura de pantalla de las opciones del juego para esta versión. Se mantienen las mismas que la versión anterior, pero ahora el docente puede seleccionar si prefiere construirlo a partir de una categoría dada o de un texto que ingresará en el campo destinado para eso.

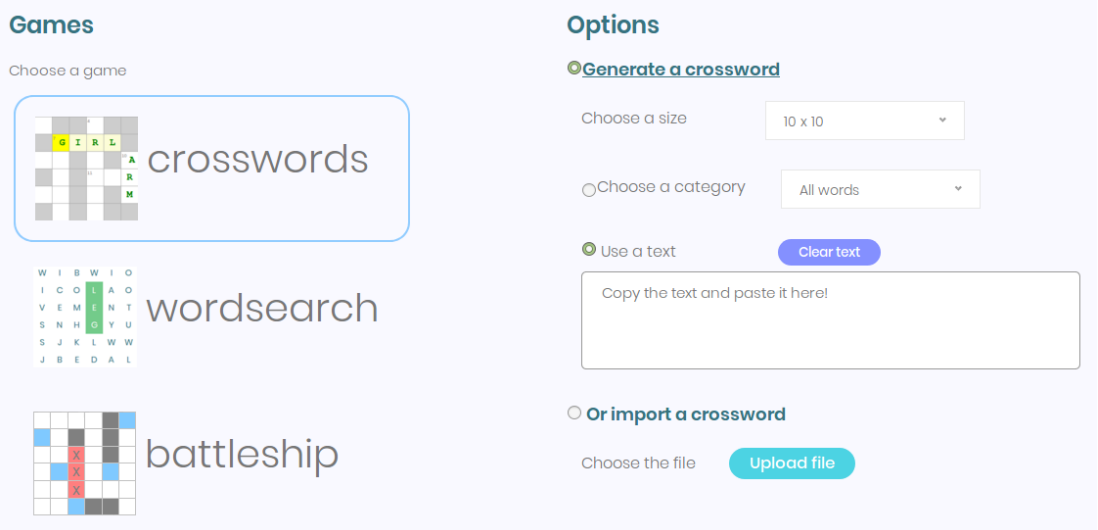


FIGURA 5.6: Captura de pantalla - Crucigramas - versión docentes

A modo de ejemplo, en la figura 5.7 se presenta un tablero de 8x8 generado a partir del siguiente texto:

Africa

Africa is a very big continent. Many animals live in Africa, including lions, giraffes, zebras, hyenas and monkeys.

Lions and hyenas are carnivores, meaning they eat meat. Hyenas are known to have their front legs longer than their back ones.

Giraffes are very tall animals, in part because of their long necks. Giraffes are tall enough to eat from trees.

Zebras look like horses and are known for having black and white stripes.

If you like wild animals, you should definitely visit Africa!

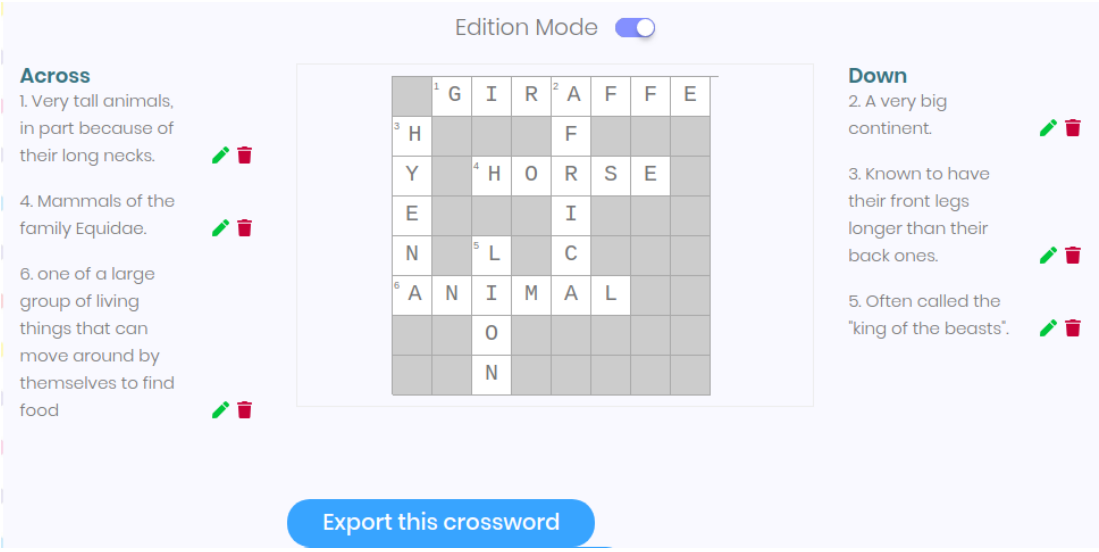


FIGURA 5.7: Captura de pantalla - Crucigrama a partir de un texto - Edition mode ON

En este ejemplo se puede ver que los pares *«palabra, pista»* correspondientes a los números 1,2y 3 fueron extraídos directamente del texto y los pares 4, 5 y 6 corresponden a los casos en que el algoritmo busca de forma alternativa sustantivos y trae sus definiciones de las ya procesadas que se encuentran en el archivo *data_crossword.json*.

Con respecto a la interfaz, encima del tablero se encuentra un botón *switch* junto al texto *"Edition Mode"* que, cuando se encuentra en modo habilitado, tiene la funcionalidad de mostrar los botones de edición y eliminación para cada pista y además completa los casilleros con las palabras correctas para hacer más fácil su corrección.

Al momento de corregir una pista, se envía la nueva información al archivo `JSON` para persistir el cambio. La opción de corrección de pistas siempre se encuentra disponible para esta versión, aún si el tablero fue generado a partir de un texto o con el `JSON` que contiene datos ya procesados.

5.3. Sopas de letras

5.3.1. Implementación

El esquema de la figura 5.8 corresponde al proceso en alto nivel para generar las sopas de letras.

FIGURA 5.8: Solución para sopas de letras en líneas generales

El encargado de la creación del tablero es un algoritmo implementado en `Javascript` por el usuario *BunKat* de GitHub (Ver apéndice D) el cuál fue adaptado según las necesidades de este trabajo. Una vez que el jugador elige las opciones, el algoritmo escoge al azar del archivo `JSON` que corresponde la cantidad de palabras de la categoría seleccionada y las acomoda en el tablero teniendo en cuenta las orientaciones seleccionadas, rellenando los casilleros que quedaron vacíos con letras de forma aleatoria. Luego, se le presenta al usuario el tablero y las pistas que correspondan según el nivel elegido.

A medida que el usuario va encontrando las palabras en el tablero y seleccionándolas con el mouse, el algoritmo realiza la validación y, si es correcta, la colorea de verde y la tacha de la lista de pistas.

En esta instancia también se tiene disponible un botón para exportar el tablero y pistas generado a un archivo `JSON` para poder importarlo dónde y cuándo se desee.

5.3.2. Modalidades y parámetros del juego

A continuación se presenta una captura de pantalla de la aplicación cuando se selecciona como juego las sopas de letras (*wordsearch*).

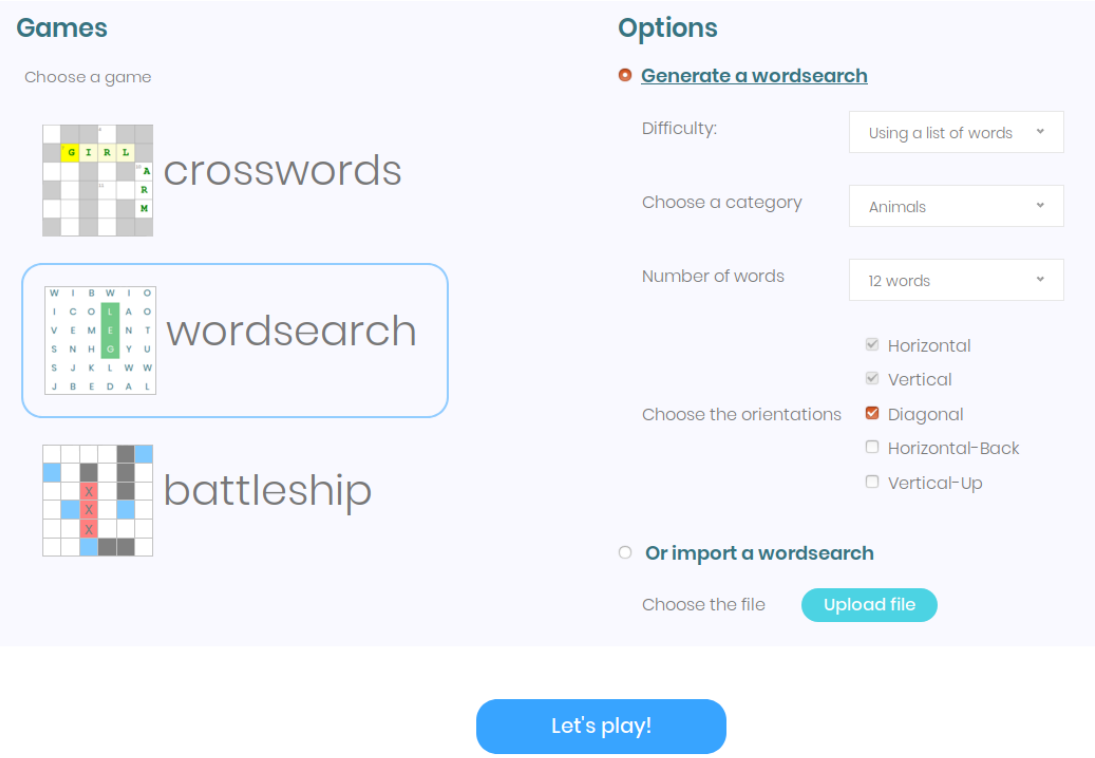


FIGURA 5.9: Captura de pantalla - Parámetros Sopas de letras

El usuario tiene disponible dos modalidades para jugar una sopa de letras (Figura 5.9):

1. **Puede generar** un tablero al momento de jugar, eligiendo distintos parámetros para su creación.
2. **Puede importar** un tablero a través de un archivo `JSON` que ya fue generado por la aplicación anteriormente.

Con respecto a los parámetros del juego, al seleccionar la modalidad 1, el usuario tiene que elegir entre las siguientes opciones:

- **Nivel de dificultad** - se debe seleccionar uno de los siguientes niveles:

  Nivel 1 - Tener disponible la lista de palabras a buscar.

  Nivel 2 - Tener disponible las pistas que corresponden a las palabras a buscar, por lo que debe adivinar la palabra objetivo a través de la pista dada.

  Nivel 3 - Tener disponible solo la categoría y la cantidad de palabras a buscar.

- **Categoría de palabras** - el usuario debe seleccionar la categoría de las palabras a buscar. Estas categorías son las descritas en el capítulo anterior y además se agrega la categoría *"all words"* que es la unión de todas.
- **Cantidad de palabras** - debe elegir la cantidad de palabras a buscar. Los valores disponibles son: 8, 10, 12 y 15 palabras.
- **Orientación de las palabras** - en este caso ya se tienen seleccionadas por defecto las opciones *horizontal y vertical* y además se pueden agregar *diagonal, horizontal-back y vertical-up*.

Una vez elegida la modalidad y/o las opciones el usuario debe hacer click en el botón *"Let's play!"* de abajo y aparecerá el tablero generado con sus respectivas pistas.

5.3.3. Niveles de dificultad

Nivel 1

En este nivel el usuario tiene disponible la lista de palabras que debe encontrar. En la figura 5.10 se muestra un diagrama de su implementación en líneas generales.

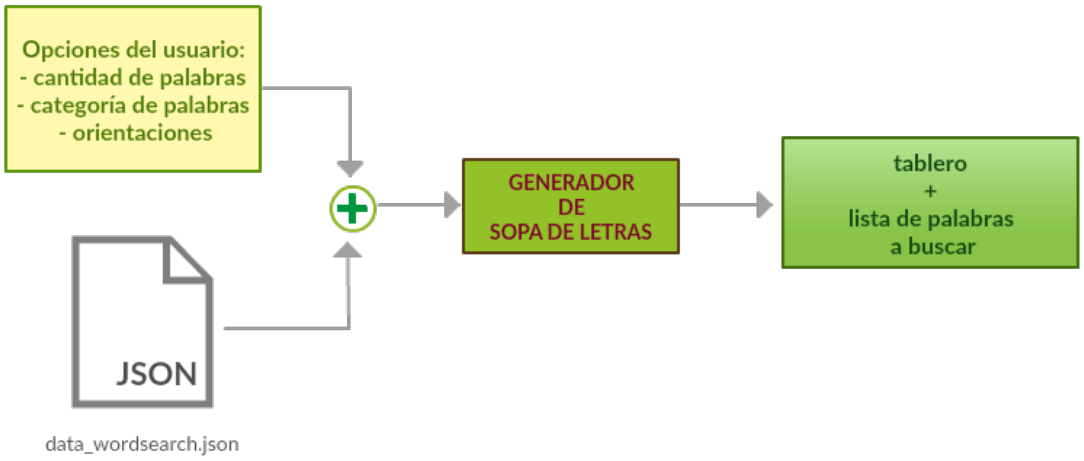


FIGURA 5.10: Implementación sopa de letras - nivel 1

En este caso, la información para crear el tablero se obtiene del archivo *data_wordsearch.json*. Un ejemplo de sopa de letras para este nivel se puede ver en la figura 5.11.

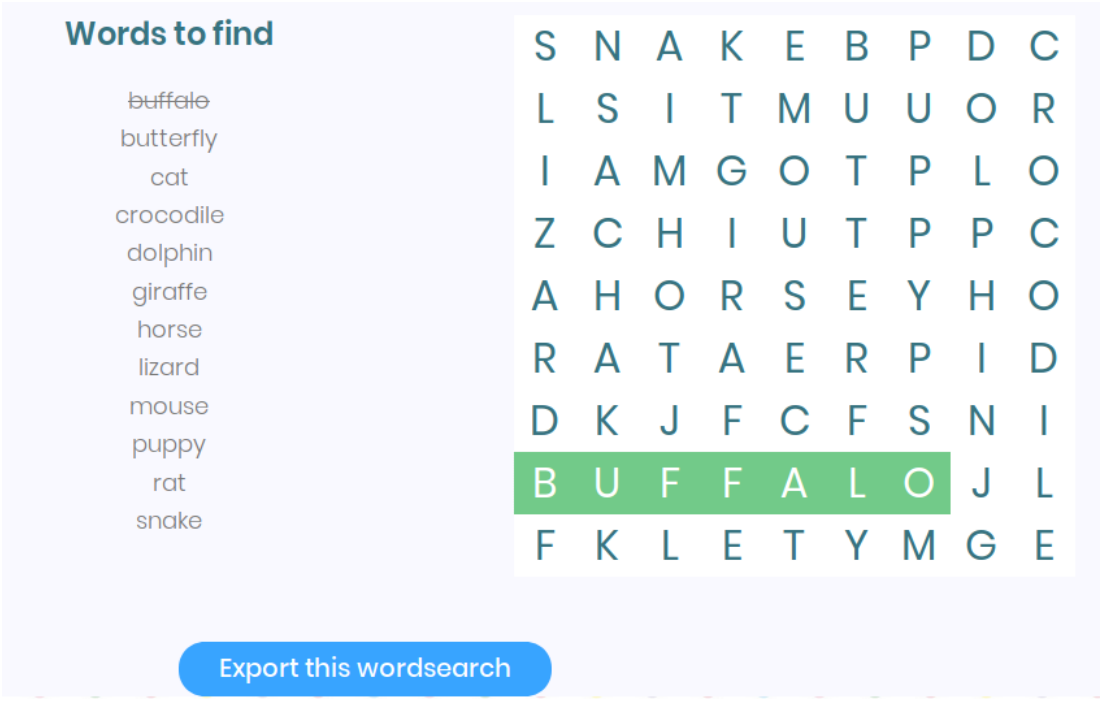


FIGURA 5.11: Captura de pantalla - Sopa de letras - Ejemplo de nivel 1

Nivel 2

En este nivel se tienen disponibles las pistas que corresponden a las palabras a buscar entonces el usuario debe adivinar la palabra objetivo a través de la pista dada.

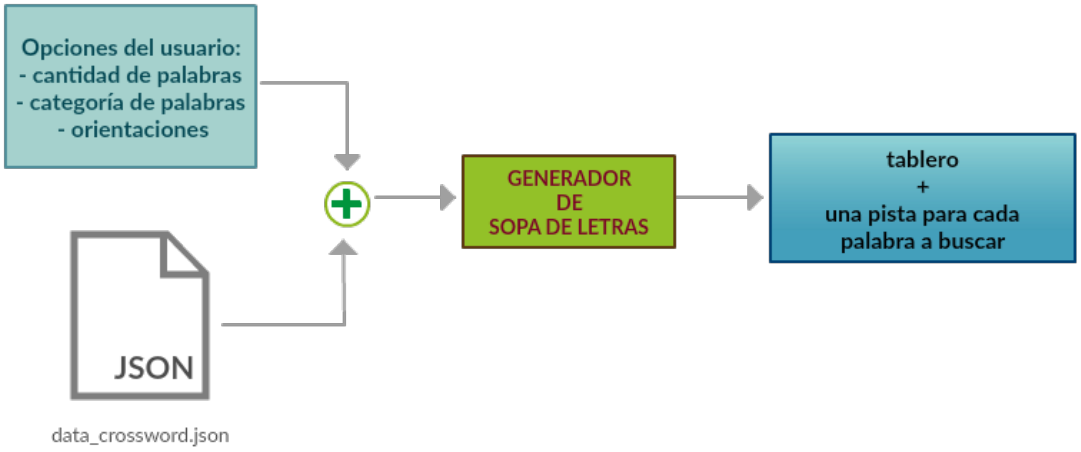


FIGURA 5.12: Implementación sopa de letras - nivel 2

En este caso, la información para crear el tablero se obtiene del archivo *data_crossword.json* ya que este es el que contiene los pares *«palabra,pista»* que se utilizan para los crucigramas (Figura 5.12).

Un ejemplo de sopa de letras para el nivel 2 se puede ver en la figura 5.13.

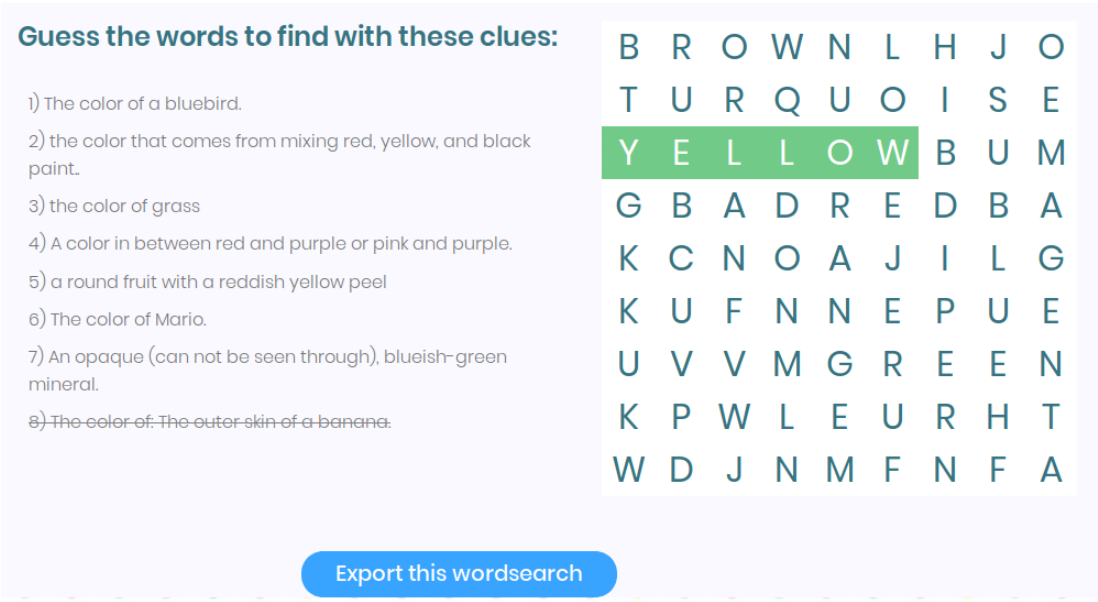


FIGURA 5.13: Captura de pantalla - Sopa de letras - Ejemplo de nivel 2

Nivel 3

Este nivel es considerado el más difícil ya que solo se tiene el nombre de la categoría y la cantidad de palabras a buscar (Figura 5.14).

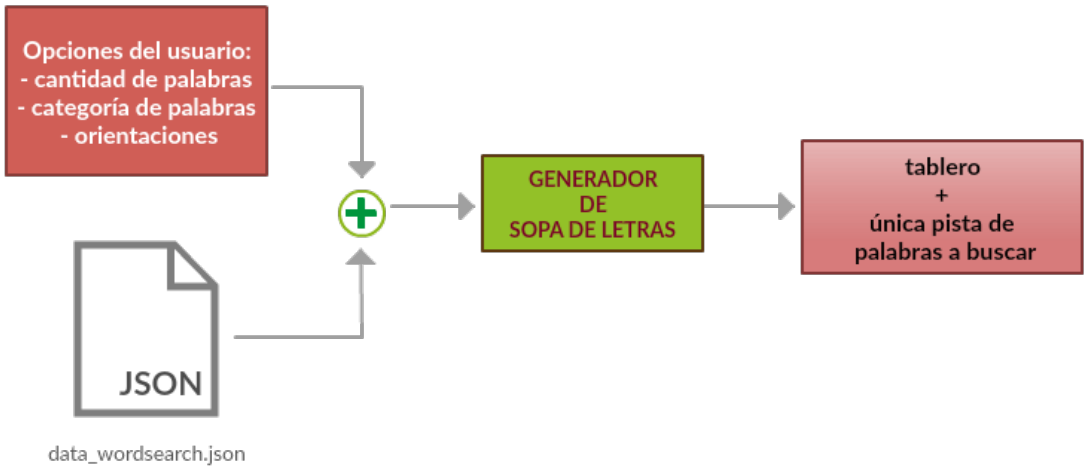


FIGURA 5.14: Implementación sopa de letras - nivel 3

En este caso al igual que en el nivel 1, la información para crear el tablero se obtiene del archivo *data_wordsearch.json*.

Dadas las siguientes opciones elegidas:

- Cantidad de palabras: **10**.
- Categoría: **House**.
- Orientaciones: **horizontal y vertical**.

La sopa de letras generada se presenta en la figura 5.15.

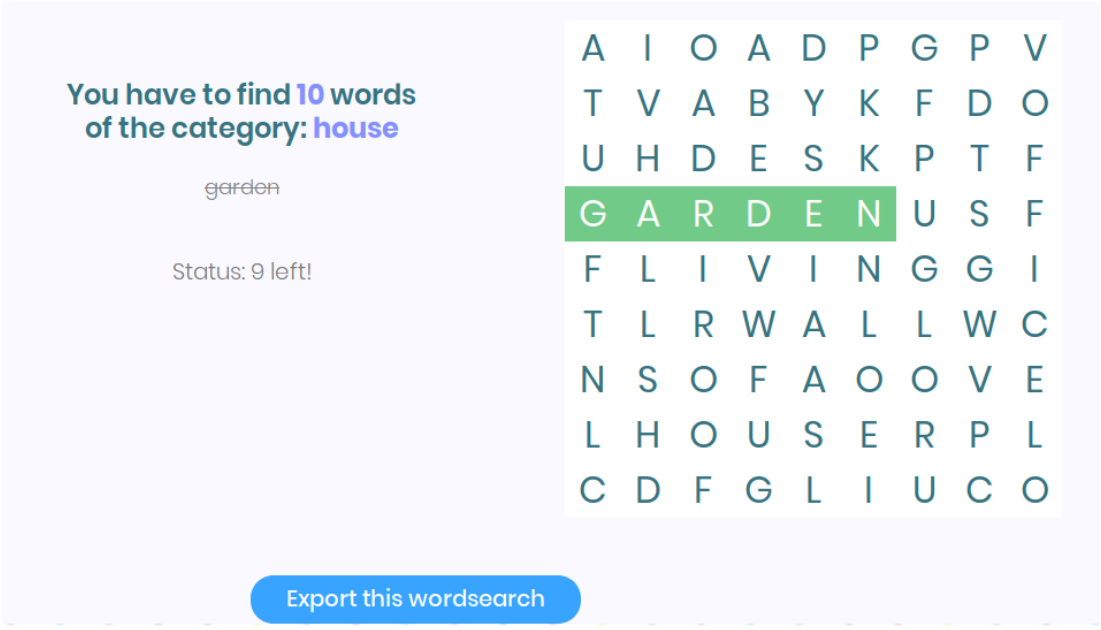


FIGURA 5.15: Captura de pantalla - Sopa de letras - Ejemplo de nivel 3

En este nivel, cuando el usuario encuentra una palabra correcta, se agrega en una lista a la izquierda del tablero y se le va informando cuántas palabras le quedan por encontrar.

5.4. Batalla naval

5.4.1. Implementación

Inicialmente la pantalla que se le presenta al usuario cuando selecciona el juego *battleship* es la siguiente:

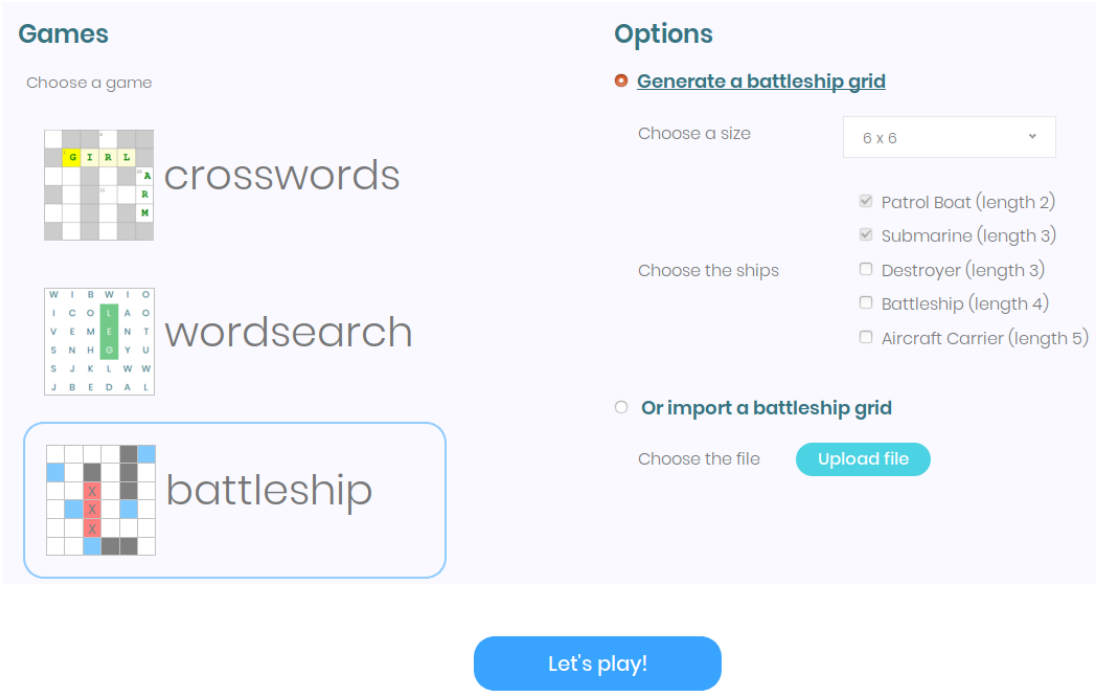


FIGURA 5.16: Captura de pantalla - Parámetros Batalla Naval

Este juego también tiene las dos modalidades que los anteriores:

1. **Puede generar** un tablero para jugar con un compañero.
2. **Puede importar** un tablero a través de un archivo `JSON` que ya fue generado por la aplicación anteriormente.

Si desea generar un tablero, el usuario debe seleccionar el tamaño que desea entre las opciones 6x6, 7x7 y 8x8. También debe elegir las naves con las que desea jugar y sus opciones son:

- *Patrol boat* con una longitud de 2 casilleros.
- *Submarine* con una longitud de 3 casilleros.
- *Destroyer* con una longitud de 3 casilleros.
- *Battleship* con una longitud de 4 casilleros.
- *Aircraft Carrier* con una longitud de 5 casilleros.

Por defecto el juego ya cuenta con las primeras dos naves seleccionadas. Es probable que una vez que el usuario genere el tablero inmediatamente deba exportarlo para que su contrincante lo pueda importar en su computadora así tendrían la misma configuración de sujetos y predicados. Es por eso que la opción de exportar también está disponible en este juego.

Para la implementación de la batalla naval (Figura 5.17) se utilizaron los archivos `JSON` generados por el *Segmentador de oraciones*, los parámetros elegidos por el usuario y un algoritmo implementado en `Javascript` por Bill Mei (Ver apéndice D), el cuál fue modificado según las necesidades de este proyecto.

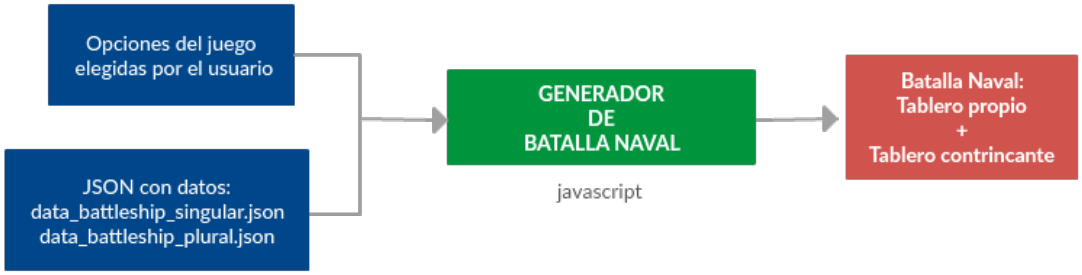


FIGURA 5.17: Solución para batalla naval en líneas generales

Lo primero que realiza el algoritmo es escoger al azar entre las opciones {*singular, plural*} y entre {*primera, tercera*}. Luego a partir del tamaño seleccionado por el jugador y los valores de persona y número anteriores, toma al azar del archivo `JSON` correspondiente 6, 7 u 8 sujetos y predicados para luego construir el tablero.

En la figura 5.18 se muestra el resultado de haber elegido el tamaño 6x6 y todas las naves.

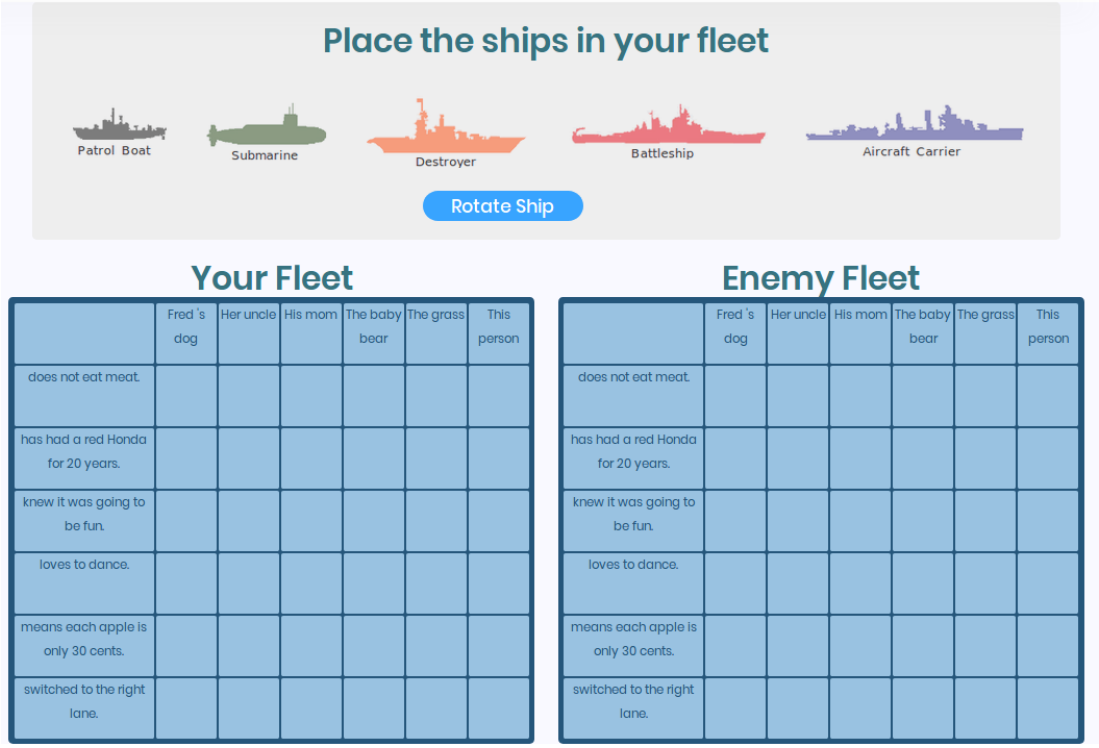


FIGURA 5.18: Captura de pantalla - Batalla Naval - tablero 6x6

Las naves que tiene disponible el usuario se encuentran en un recuadro encima de los tableros. Para colocarlas en el propio debe elegir una y ubicarla encima de modo que aparecen los casillero en gris indicando la posición deseada.
También hay disponible un botón *"Rotate ship"* para girar la nave a vertical u horizontal.

La situación luego de posicionar la nave *Submarine* en vertical sobre el tablero propio se muestra en la figura 5.19.

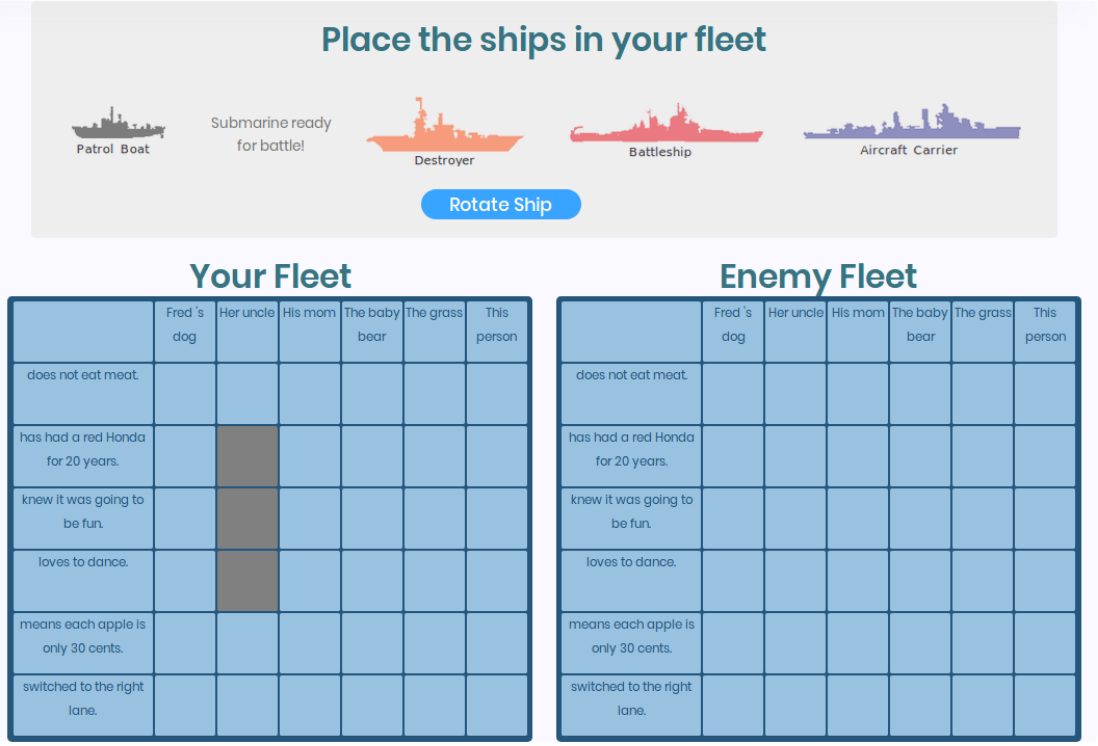


FIGURA 5.19: Captura de pantalla - Batalla Naval - Submarino posicionado

Las oraciones a construir por su contrincante para derribar el *Submarine* deben ser:

- *"Her uncle has had a red Honda for 20 years."*
- *"Her uncle knew it was going to be fun."*
- *"Her uncle loves to dance."*

Una vez que el usuario posiciona todas las naves en el tablero aparece el botón *"Start Game"* para que se habilite la interacción con los tableros como muestra la figura 5.20.

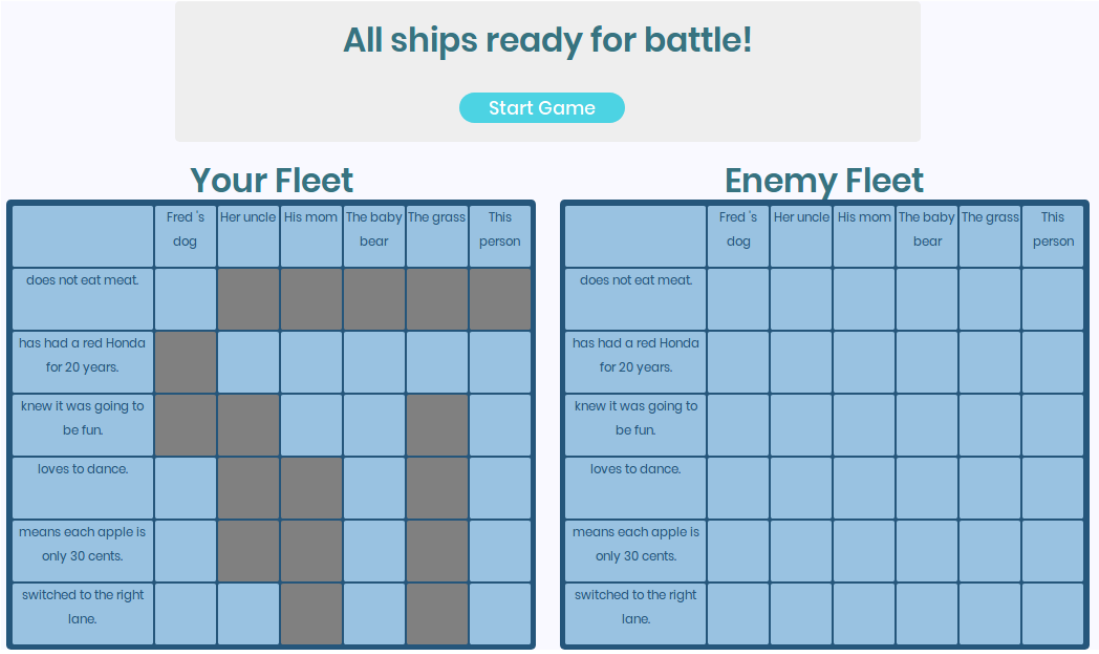


FIGURA 5.20: Captura de pantalla - Batalla Naval - Comenzar juego

La figura 5.21 muestra la interacción con el tablero propio. Una vez que el contrincante ejecuta un tiro existen dos opciones:

- **Miss**: el casillero corresponde al agua y se marca con color blanco.
- **Hit**: el casillero corresponde a una de las naves. En este caso el casillero se marca con rojo.

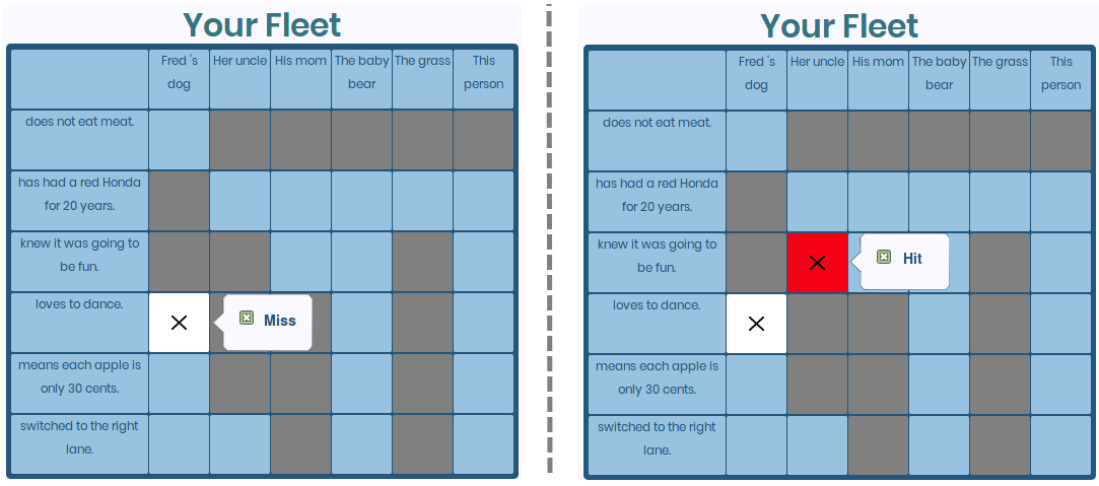


FIGURA 5.21: Captura de pantalla - Batalla Naval - Tablero propio

Con respecto al tablero del contrincante, cuando el usuario elige un casillero para el destino del tiro, el contrincante le comunica si fue acertado (hay una nave) o si fue al agua. Es por esto que al hacer clic en los casilleros del tablero enemigo se despliegan las dos opciones: *Water* o *Ship*. En el primer caso el casillero se colorea de blanco y en caso de que haya una nave se colorea de rojo (Figura 5.22).

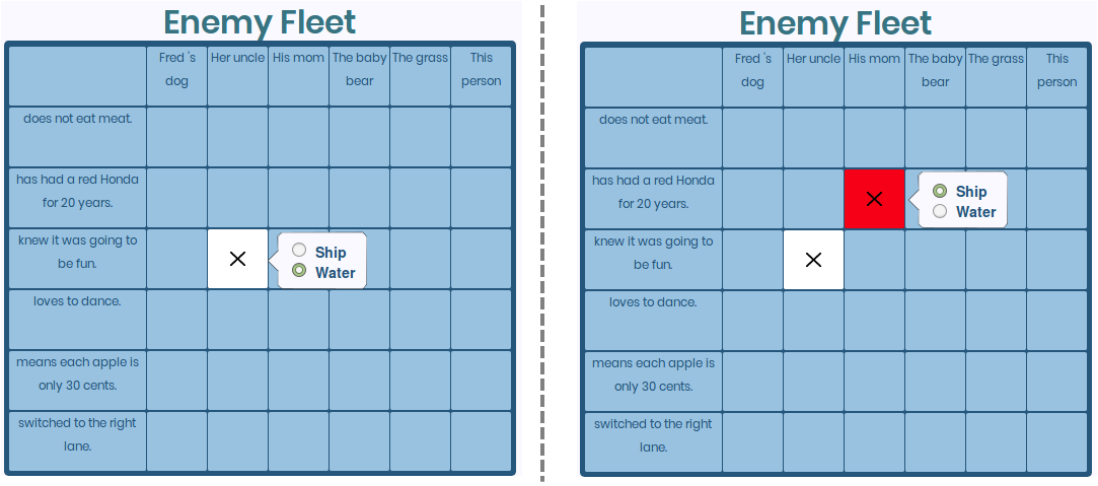


FIGURA 5.22: Captura de pantalla - Batalla Naval - Tablero contrincante

El juego finaliza una vez que uno de los dos jugadores derribó todas las naves del contrario. Este control no está contemplado en la aplicación, el mismo queda a cargo de los jugadores.

Capítulo 6

Experiencias

6.1. Visitas y eventos

En esta sección se presentan las experiencias del equipo cuando se visitaron las escuelas rurales asignadas y la asistencia a los eventos de ANEP.

Junio de 2018 - Visita Escuela Nro. 44, San José

Se realiza una visita a la maestra Johanna Revetria a cargo de la enseñanza de inglés multigrado[1] a los alumnos de 4to, 5to y 6to año (aproximadamente 14 niños en total), los cuales cuentan con Ceibalitas pero escasa conexión a Internet.

En esta instancia, se realizó una presentación de los crucigramas solo a la maestra y se le proporcionó un instalador para que pudiera instalarlo en las computadoras de los alumnos y comenzar a probarlo. Al cabo de unas semanas tuvimos una devolución buena sobre la herramienta ya que lo pudieron instalar y usar sin dificultades.

Figura 6.1: Escuela Nro. 44, San José

[1] Simultánea a varios cursos

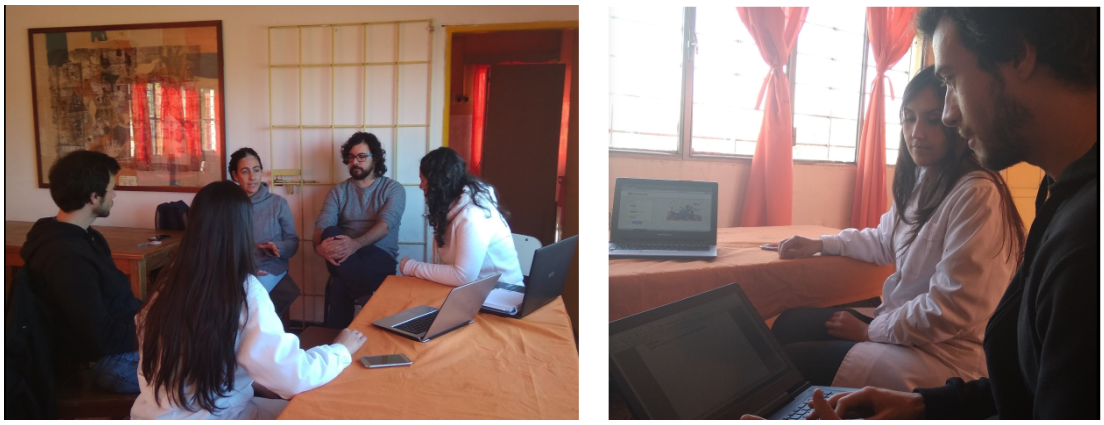


FIGURA 6.2: Presentación a la maestra Johanna, San José

Julio de 2018 - Evento Wintercamp

Este evento se realizó en la ciudad de Trinidad y consistió de un encuentro de maestros de enseñanza primaria pública, con el objetivo de realizar presentaciones académicas, actividades didácticas y exploración de técnicas pedagógicas en el marco de la enseñanza del inglés.

Para comenzar uno de los tutores de este proyecto realizó una introducción sobre PLN a los asistentes del evento. Luego se hizo una presentación sobre la implementación de la herramienta, que en su momento incluía una primera versión del generador automático de crucigramas. Se comentaron las técnicas utilizadas las dificultades encontradas, y los planes de avance en el desarrollo de nuevos juegos.

La presentación culminó con una demostración de la herramienta, e inmediatamente después de eso hubo un intercambio de ideas donde intervinieron coordinadores y maestros, con el fin aclararles dudas, y de su parte recibir sugerencias de mejora de la aplicación e ideas para posibles juegos nuevos.

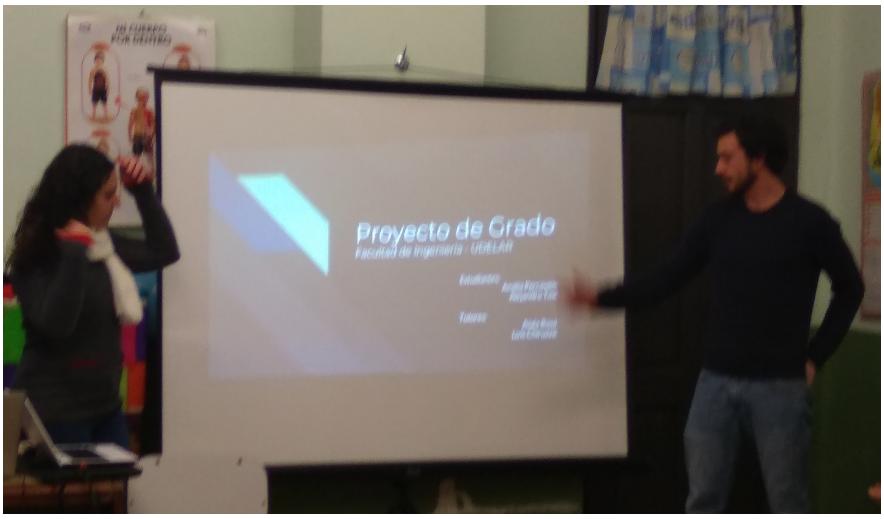


FIGURA 6.3: Presentación en el Wintercamp

Agosto de 2018 - Visita Escuela Nro. 76, Paysandú

En esta oportunidad se realizó una visita al maestro Gerardo Saracho a cargo de la enseñanza de inglés multigrado de 4to, 5to y 6to de esta escuela que es uno de los pocos internados del país. Los chicos (desde inicial a 6to) y sus docentes conviven de lunes a viernes y tienen clases de inglés de dos a tres veces por semana.

En esta ocasión se instaló la herramienta en las Ceibalitas del maestro y de los alumnos con los dos primeros juegos implementados, crucigramas y sopas de letras. La respuesta de los niños fue muy buena, había mucha motivación y quedaron muy contentos ya que se dieron cuenta que sabían más inglés de lo que creían.

FIGURA 6.4: Escuela Nro. 76, Paysandú

FIGURA 6.5: Intercambio con maestro Gerardo y alumnos - Paysandú

Octubre de 2018 - Evento 11 Foro de Lenguas de ANEP

El Foro de Lenguas es un evento anual de encuentro, reflexión y discusión sobre el aprendizaje y la enseñanza de lenguas organizado por la línea transversal de Políticas Lingüísticas de ANEP. Al mismo asisten quienes a través de la práctica educativa, la investigación, o ambas, contribuyen a la generación de conocimiento y para el trabajo en el aula.

En esta instancia del Foro, se realizó una presentación introductoria al PLN a cargo de los tutores del proyecto y luego una breve explicación sobre cómo fue implementada la herramienta, culminando con una demostración de los juegos crucigramas y sopas de letras. La devolución de los docentes fue muy buena, incluso se recopilaron nuevas ideas para llevar a cabo como trabajos a futuro.

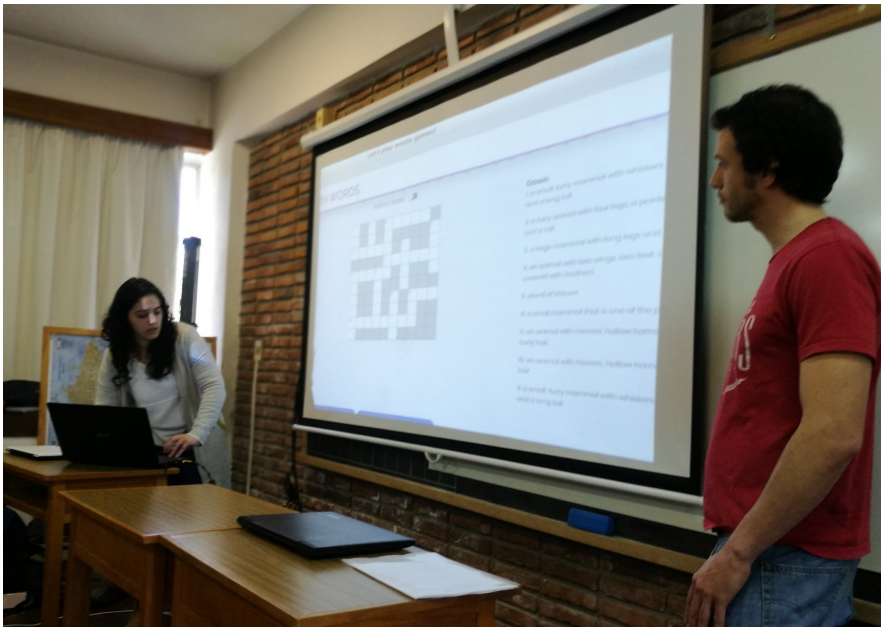


Figura 6.6: Presentación en el Foro de Lenguas

Capítulo 7

Conclusiones y trabajo a futuro

En este proyecto se trabajó en la construcción de una herramienta que genera juegos de forma automática para dar soporte a la enseñanza de inglés en las escuelas.

Comenzamos recolectando todo el material que nos brindaba la ANEP para tener una referencia del nivel de inglés que se trabaja en los cursos de Primaria, este material hubo que complementarlo con textos obtenidos de otras fuentes generando así un corpus de mayor tamaño que tiene como objetivo ser la entrada principal de los recursos implementados aplicando técnicas de PLN. Encontrar el material complementario con el nivel de inglés adecuado, en este caso nivel A1, no fue una tarea sencilla ya que calificar la dificultad un texto no es algo inmediato ni preciso.

Por otro lado, muchas herramientas dedicadas a la educación para niños están fuertemente vinculadas a contenidos audiovisuales (imágenes, videos, canciones, etc.) lo cual hace que no sean aptos para este proyecto. También hubo que descartar materiales de textos en inglés que no estaban en un formato apropiado para su procesamiento, por ejemplo, el formato *pdf*.

Con respecto a la selección de los juegos, el principal problema fue encontrar o definir qué juegos implementar a partir del procesamiento de textos, ya que como se mencionaba anteriormente en este contexto es usual apoyarse en contenido multimedia para el entretenimiento y la enseñanza de un idioma.

Uno de los juegos fue basado en un proyecto de grado anterior que consistía en la generación automática de crucigramas a partir de textos de prensa en español. Por otro lado, se implementaron sopas de letras con distintos niveles de dificultad para hacerlas más interesantes. Por último, se desarrolló el juego de la batalla naval con tableros conformados por sujetos y predicados de oraciones.

A partir de la selección de los juegos se pudieron desarrollar los diferentes recursos de procesamiento de texto necesarios para plasmarlos en una aplicación web y cumplir con los requerimientos propuestos.

Con respecto al algoritmo extractor de definiciones, fue un trabajo arduo identificar las distintas casuísticas de lo que define una pista válida para una palabra de un crucigrama que debe manejar el nivel de inglés apropiado. Se logró controlar esta tarea estudiando el comportamiento del parser de Stanford y analizando los resultados del mismo frente a las oraciones de los corpus generados. Esto culminó con una evaluación de la calidad y capacidad de desambiguación de un subconjunto de pares «palabra, pista» con dos heurísticas basadas en *Word Embeddings*. Los resultados de la evaluación fueron considerablemente distintos y se usó el mejor para construir el conjunto de pares

final, el cual fue reducido a un tamaño significativamente menor al conjunto inicial.

La calidad de algunas pistas no es la más adecuada, por lo que se incluye en la aplicación la funcionalidad de modificación de las mismas por parte de los docentes.

En la tarea de ampliación de la *«Lista de palabras categorizada»* el uso de los *Word Embeddings* fue clave, ya que permitió encontrar de forma fácil palabras con el mismo nivel de dificultad que las palabras con las que ya contábamos. Tanto para esta tarea como para la de seleccionar los mejores pares «palabra,pista», el hecho de usar el nombre de la categoría para contextualizar las palabras dio mejores resultados que las otras técnicas.

El segmentador de oraciones no presentó mayores dificultades en su implementación ya que al utilizar el análisis de dependencias en las oraciones se obtiene fácilmente el sujeto y predicado de las mismas. El corpus utilizado en esta instancia facilitó fuertemente la tarea de evaluar los distintos casos a contemplar por el segmentador ya que se compone de oraciones simples y con relaciones de dependencias similares.

Sobre la aplicación web generada, podemos decir que cumple con el principal requerimiento: no depende de una conexión a internet y no requiere instalar demasiados paquetes adicionales a los ya incorporados en las Ceibalitas.

Gracias a las visitas a las escuelas y a los eventos en los que se iban presentando los avances de la aplicación, tuvimos una retroalimentación muy positiva en lo que respecta a su diseño y facilidad de uso. Tuvimos la oportunidad de estar presentes cuando alumnos de Paysandú instalaron sin problemas la aplicación, generaron crucigramas y sopas de letras para jugar tanto individualmente como en equipo y pudieron resolverlos exitosamente apenas con un poco de ayuda del docente a cargo. Esto quiere decir que el nivel de inglés empleado en los juegos podría considerarse apropiado para los usuarios de la aplicación.

Finalizando el proyecto, se destacaron dos cosas importantes: para empezar, los corpus generados a partir de la extracción de sitios web pueden considerarse un material muy valioso debido a lo difícil que es encontrar recursos de estas características. Como segundo punto queremos destacar que los métodos y heurísticas empleados en la generación de los recursos para llevar a cabo el proyecto pueden considerarse útiles para futuros desafíos y están abiertos a mejoras.

Para concluir podemos decir que se cumplieron los objetivos planteados al comienzo del proyecto, logrando como resultado final una aplicación web que creemos que será de gran utilidad para dar soporte a la enseñanza de inglés en las escuelas.

Como trabajo a futuro se pueden plantear los siguientes puntos:

- En la extracción de definiciones se pueden agregar más patrones de búsqueda, también se podrían utilizar hiperónimos para encontrar definiciones más correctas y tener en cuenta recursos como *synsets de WordNet, Topical Word Embedding* y métodos de aprendizaje automático para evitar obtener definiciones sintáctica y/o semánticamente incorrectas.
- Mejorar la calidad de las pistas, por ejemplo, escribirlas en formato de pregunta (ej: *«Elephant,Which is the largest living land mammal?»*), o utilizar pronombres

(a partir de *"A mother is a female parent (a male parent is called a father)"* obtener y construir el par *«Mother,"Someone who is a female parent (a male parent is called a father"»*)

- Desarrollar la opción *on-demand* para todos los juegos, por ejemplo, buscar los sustantivos de un texto trabajado en clase para construir una sopa de letras.
- En los tableros generados para la batalla naval se podrían utilizar *Modelos de Lenguaje* para poder controlar los casos en que las oraciones a construir tengan una probabilidad de ocurrencia predefinida para descartar casos como: *the cat is reading a book*.
- Utilizar las frecuencias de las palabras (en algún contexto dado) de una definición, exceptuando las *stop-words*, para determinar su dificultad y así descartar definiciones complejas.
- Programar el descarte automático de definiciones obtenidas de oraciones con correferencias detectadas.
- Generar una base de datos de textos ya procesados por la aplicación para ofrecerle al maestro, para que pueda trabajarlo en clase según sus necesidades
- Adaptar la dificultad de los juegos automáticamente a medida que el usuario está jugando
- También sería ideal agregar más juegos, por ejemplo, algunos de los que nos comentaron en las visitas a las escuelas fueron: ordenar párrafos de cuentos, el *ta-te-ti* donde las fichas son los verbos en distintas conjugaciones de tiempo según la columna del tablero a elegir, completar letras de canciones con fonemas que se pronuncien parecido, etc.

Apéndice A

Glosario

Algoritmo En el ámbito matemático se aprecian definiciones como: conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.

Análisis semántico La semántica es el estudio del significado asociado a las estructuras formales del lenguaje (sintaxis). Así como el análisis sintáctico hace referencia a cómo las palabras se disponen en una oración, el análisis semántico se centra en determinar su significado.

ANEP **A**dministración **N**acional de **E**ducación **P**ública

Categoría gramatical Clasificación de una palabra según su función dentro de una oración (verbo, sustantivo, adjetivo, etc.).

Corpus Conjunto de anotaciones que se utiliza para representar las características de un texto.

Definición Término que describe a un definiendum dado.

Definiendum Término que se está definiendo.

Dump Una gran cantidad de datos que se mueven de un sistema, archivo o dispositivo informático a otro.

FIng **F**acultad de **I**ngeniería

Grupo sintáctico Está formado por una palabra o un grupo de palabras cohesionadas alrededor de un núcleo que les da nombre. Según su núcleo el grupo sintáctico se puede llamar: Grupo Nominal, Grupo Adjetival, Grupo Adverbial, Grupo Verbal, o Grupo Preposicional.

Hipónimo Palabra de carácter más específico que un correspondiente hiperónimo, palabra de carácter más general. A la relación semántica entre estas palabras se le llama hiponimia (o hiperonimia, en el sentido inverso). Como posibles ejemplos de esta relación tenemos que “elefante” es hipónimo de “vertebrado”, y que “vertebrado” es hipónimo de “animal”.

Inteligencia Artificial Área multidisciplinaria que intenta imitar la inteligencia humana para crear y diseñar entidades capaces de resolver cuestiones por sí mismas.

JSON JavaScript Object Notation (JSON) es un formato basado en texto estándar para representar datos estructurados en la sintaxis de objetos de JavaScript.

Lema Forma básica de una palabra que busca eliminar toda inflexión.

Lematización Es el proceso lingüístico de hallar el lema correspondiente a una forma flexionada.

On-demand A demanda.

Parser Es una de las partes de un compilador que transforma su entrada en un árbol de derivación.

PLN **P**rocesamiento del **L**enguaje **N**atural

POS tagging **P**art **O**f **S**pech tagging.

Predicado Dada una oración, el predicado describe la acción que realiza el sujeto o lo que se dice del sujeto. En el predicado siempre hay un verbo, que concuerda en persona y número con el sujeto.

Redes Neuronales Artificiales Es un método de aprendizaje automático vagamente inspirado en sistemas neuronales biológicos. Una red neuronal artificial se compone de un conjunto de elementos interconectados (las "neuronas" como tipo abstracto de datos) que trabajan conjuntamente para resolver problemas específicos (por ejemplo: reconocimiento de patrones o clasificación de información) a través de un aprendizaje a partir un conjunto de datos de ejemplo. Se nota una analogía con cómo funciona el cerebro, en el sentido de que la experiencia o los ejemplos en el mundo real son una forma de aprendizaje.

Similitud semántica Es la medida de la interrelación existente entre dos palabras cualesquiera en un texto.

Simplificación de textos Proceso de transformar un texto en un equivalente que es mas fácil de entender por una audiencia determinada.

Sintagma Palabra o grupo de palabras que constituyen una unidad sintáctica y que cumplen una función determinada con respecto a otras palabras de la oración.

Sujeto Dada una oración, el sujeto indica quién realiza la acción o de quién se dice algo.

Stop-words palabras que son filtradas (descartadas) antes o después del procesamiento de un texto, porque no agregan ninguna información sustancial (por ejemplo determinantes, y preposiciones). Varían según la tarea que se esté realizando, y generalmente son palabras muy frecuentes.

Token Símbolo utilizado como unidad mínima de trabajo en el análisis de cierto texto. En este trabajo un token equivale a una palabra o un signo de puntuación.

Tokenización Proceso de separar un texto en tokens.

Voz pasiva Es una construcción verbal de las oraciones que se utiliza cuando se le quiere dar más énfasis al afectado (sujeto paciente) por la acción asociada al verbo principal en la oración. Su contraparte, la voz activa, centra la atención en el realizador de la acción (sujeto agente). Como ejemplo, tenemos que para la oración en voz activa "El león come carne" se tiene como sujeto agente a "El león" y como objeto directo del verbo "come" a "carne". La oración semánticamente equivalente en voz pasiva "La carne es comida por el león" tiene como sujeto paciente a "carne" y como complemento agente de "es comida" a "el león".

Apéndice B

Recursos

B.1. Extracción de un texto brindado por ANEP

"Dolphins are part of the whale family. They are smaller than most whales and they have small teeth. Dolphins are very clever animals. They learn things very quickly and a dolphin can make noises to talk to another dolphin. Dolphins live with their families. They like to play in the water and to jump out of the water and back in again. A lot of people who sail boats say that dolphins like to be near people. They come very near to boats and sometimes they swim with the boats for days.

There are 350 kinds of parrot in the world. They are clever animals. A lot of parrots are green, but you can find parrots which are red, yellow and blue. They live in trees and rocks in hot places. They have big heads and short necks. They are very good at climbing trees. Most parrots do not eat meat. They eat fruit and plants. Parrots fly to many places every day to look for food. When they are eating, they hold their food in one foot. These birds make a lot of noise when they are with their families. (...)"

B.2. Extracción de Simple English Wikipedia

"**Elephant**: Elephants are the largest living land mammals. The largest elephant recorded was one shot in Angola, 1974. It weighed 27,060 pounds (13.5 tons) and stood 13 feet 8 inches tall. At birth, an elephant calf may weigh 100 kg (225 pounds). The baby elephant develops for 20 to 22 months inside its mother. No other land animal takes this long to develop before being born. In the wild, elephants have strong family groups. Their ways of acting toward other elephants are hard for people to understand. They "talk"to each other with very low sounds. Most elephants sounds are so low, people cannot hear them. But elephants can hear these sounds far away. (...)"

B.3. Extracción de artículo de Ducksters

"**Penguins**: Penguins are one of the most beloved animals in the world. Penguins are found in many areas in the southern hemisphere. Most people think of penguins as living in very cold climates like the icy continent of Antarctica, but they also live in more temperate areas like the Galapagos Islands, Australia, and South Africa.

Penguins are very funny animals. They are birds that cannot fly, but love to swim! A typical penguin can spend at least half of its time swimming in the water. (...)"

Apéndice C

Tecnologías

Python lenguaje de programación interpretado y de código abierto. Tiene la gran ventaja de que su sintaxis apunta a un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y funcional.

NLTK es un módulo de Python que contiene muchas funciones diseñadas para su uso en el análisis lingüístico de documentos y en el procesamiento de lenguaje natural.

Scrapy framework de código abierto desarrollado en Python que permite administrar peticiones, preservar sesiones de usuario y seguir redirecciones. Una de las mayores ventajas de `Scrapy` es que es muy eficiente, es capaz de scrapear más cantidad, más rápido y a menos coste de CPU que otras alternativas.

HTML la sigla corresponde a HyperText Markup Language. HTML es un lenguaje de marcado que se utiliza para el desarrollo de páginas de Internet.

CSS la sigla corresponde a Cascading Style Sheets (Hojas de Estilo en Cascada). Es el lenguaje utilizado para describir la presentación de documentos HTML o XML. CSS describe cómo debe ser renderizado el elemento estructurado en pantalla, en papel, hablado o en otros medios.

Javascript (JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador. Es un lenguaje script multiparadigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa.

JQuery jQuery es una librería de JavaScript que se enfoca en simplificar la manipulación del DOM, llamadas AJAX y manejo de Event.

ElectronJs es un framework para JavaScript que permite el desarrollo de aplicaciones de escritorio mediante el uso de tecnologías web. Esta desarrollado por GitHub (lo que garantiza revisiones constantes), es de código abierto y multiplataforma.

MongoDB bases de datos NoSQL orientada a documentos, es decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON. Una de las diferencias más importantes con respecto a las bases de datos relacionales, es que no es necesario seguir un esquema. Los documentos de una misma colección (concepto similar a una tabla de una base de datos relacional), pueden tener esquemas diferentes.

Apéndice D

Algoritmos

A continuación se hace referencia a los códigos fuente utilizados para la construcción de los juegos. En todos los casos los códigos fueron modificados acorde a las necesidades de este proyecto.

Crucigramas El código javascript utilizado para la construcción de los tableros de crucigramas (*backend*) pertenece a Richard Rulach bajo una licencia Apache y fue tomado de `https://github.com/richardrulach/js-xwords`.

El código *javascript, css y html* para la interacción de los crucigramas (*frontend*) se encuentra bajo una licencia GNU y su autor es Matt Wiseley. El mismo puede encontrarse en: `https://github.com/wiseley/javascript-crossword`.

Sopa de letras Para la construcción e interacción de los tableros de las sopa de letras se utilizó el código Javascript perteneciente al usuario BunKat bajo una licencia MIT. El sitio de donde fue descargado es: `http://github.com/bunkat/wordfind`.

Batalla Naval Los tableros para la batalla naval fueron basados en una idea de Bill Mei publicada en `https://github.com/billmei/battleboat`. Este código también se encuentra bajo una licencia MIT.

Bibliografía

Alhawiti, Khaled M. (2014). «Natural Language Processing and its Use in Education». En: *International Journal of Advanced Computer Science and Applications(ijacsa)*.

Baars, Bernard J. y Nicole M. Gage (2010). *Cognition, Brain, and Consciousness: Introduction to Cognitive Neuroscience (2nd edition)*. Academic Press, pág. 373.

Bartunov, Sergey et al. (2015). «Breaking Sticks and Ambiguities with Adaptive Skip-gram». En: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Berón, Rodrigo y Ezequiel Jardim (2017). «SCARR - Sistema clasificador automático de respuestas según relevancia». En: *Proyectos de Grado - Grupo PLN*.

Bose, Ranjit (2004). «Natural Language Processing: Current state and future directions». En: *International Journal of the Computer, the Internet and Management Vol. 12, number 1 (January – April, 2004) pp 1 - 11*.

Brovetto, Claudia (2013). *Aprendizaje abierto y aprendizaje flexible - Capítulo 9: Ceibal en inglés*. URL: https://www.anep.edu.uy/sites/default/files/images/Archivos/publicaciones/plan-ceibal/aprendizaje_abierto_anep_ceibal_2013.pdf.

Brown, Peter F. et al. (1991). «Word-Sense Disambiguation Using Statistical Methods». En: *ACL*.

Camacho-Collados, José, Mohammad Taher Pilehvar y Roberto Navigli (2015). «A Unified Multilingual Semantic Representation of Concepts». En: *ACL*.

Candido Jr, Arnaldo et al. (2009). «Supporting the adaptation of texts for poor literacy readers: A text simplification editor for Brazilian portuguese.» En: *Proc. 4th Workshop Innovative Use of NLP for Building Educational Applications*.

Castrillón Díaz, Lía Trinidad (2017). «Los Juegos y su Rol en el Aprendizaje de una Lengua.» En: *Universidad de la Sabanas*.

Cliche, Mathieu (2017). «BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs». En: *Proceedings of the 11th International Workshop on Semantic Evaluation*.

Corcoglioniti, Francesco, Marco Rospocher y Alessio Palmero Aprosio (2016). «Frame-based Ontology Population with PIKES». En: *IEEE Transactions on Knowledge and Data Engineering, 2016, vol. 28, no. 12, pp. 3261-3275*.

Cui, Lei, Furu Wei y Ming Zhou (2018). «Neural Open Information Extraction». En: *ACL*.

Dagan, Ido y Alon Itai (1994). «Word Sense Disambiguation Using a Second Language Monolingual Corpus». En: *Computational Linguistics* 20, págs. 563-596.

Deerwester, Scott et al. (1990). «Indexing By Latent Semantic Analysis». En: *Journal of the American Society for Information Science* 41, págs. 391-407.

Dyer, C, S Muresan y P Resnik (2008). «Generalizing word lattice translation». En: *Proceedings of ACL/HLT 2008*, págs. 1012-1020.

Dzikovska, Myroslava et al. (2010). «BEETLE II: a system for tutoring and computational linguistics experimentation». En: *Proceedings of the ACL 2010 System Demonstrations*.

Edunov, Sergey et al. (2018). «Understanding BackTranslation at Scale». En: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Espinosa-Anke, Luis y Horacio Saggion (2014). «Descripción y Evaluación de un Sistema de Extracción de Definiciones para el Catalán». En: *SEPLN*.

Esteche, Jennifer y Romina Romero (2015). «Extracción de definiciones y generación automática de crucigramas a partir de textos de prensa». En: *Proyectos de Grado - Grupo PLN*.

Firth, John Rupert (1957). *Studies in Linguistic Analysis*. Oxford, Blackwel.

Fregossi, Federico (2014). «La Ruta de Ceibal en inglés: Entre oportunidades y controversias». En: *Tesis de Magisterio, Montevideo, Uruguay*.

Gale, William A., Kenneth W. Church y David Yarowsky (1992). «Using Bilingual Materials to Develop Word Sense Disambiguation Methods». En:

Glass, James et al. (2007). «Recent progress in the MIT spoken lecture processing project». En: *Proceedings of Interspeech*.

Gong, Chengyue et al. (2018). «FRAGE: Frequency-Agnostic Word Representation». En: *NIPS 2018*.

Gupta, Pankaj et al. (2019). «Neural Relation Extraction Within and Across Sentence Boundaries». En: *Proceedings of AAAI 2019*.

Harris, Zellig S. (1954). «Distributional Structure». En: *Word* 10.2-3, págs. 146-162.

He, Zhengqiu et al. (2018). «SEE-Syntax-aware Entity Embedding for Neural Relation Extraction». En: *In Proceedings of the AAAI 2018*.

Hearst, Marti A (1992). «Automatic acquisition of hyponyms from large text corpora». En: *Proceedings of the 14th conference on Computational linguistics-Volume 2*.

Howard, Jeremy y Sebastian Rude (2018). «Universal Language Model Fine-tuning for Text Classification». En: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Jiang, Shu y John Lee (2017). «Distractor Generation for Chinese Fill-in-the-blank Items». En: *The Twelfth Workshop on Innovative Use of NLP for. Building Educational Applications. Proceedings of the Workshop*.

Jones, Kerry (2007). «Teaching with Crossword Puzzles». En: URL: https://www.vocabulary.co.il/2007/09/teaching-with-crossword-puzzles/.

Jurafsky, Daniel y James H. Martin (2009). *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc. ISBN: 0131873210.

Kundu, Souvik y Hwee Tou Ng (2018). «A Question-Focused Multi-Factor Attention Network for Question Answering». En: *AAAI*.

Lample, Guillaume et al. (2018). «Phrase-Based and Neural Unsupervised Machine Translation». En: *EMNLP*.

Landthaler, Joerg et al. (2017). «Extending Thesauri Using Word Embeddings and the Intersection Method». En: *Second Workshop on Automated Semantic Analysis of Information in Legal Text*.

Lee, Jangho et al. (2017). «Training IBM Watson using Automatically Generated Question-Answer Pairs». En: *HICSS*.

Lesk, Michael E. (1986). «Automatic Sense Disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone». En: *SIGDOC*.

Levy, Omer y Yoav Goldberg (2014). «Dependency-Based Word Embeddings». En: *ACL*.

Litman, Diane (2016). «Natural Language Processing for Enhancing Teaching and Learning». En: *Thirtieth AAAI Conference on Artificial Intelligence*. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12310.

Liu, Xiaodong et al. (2019). «Multi-Task Deep Neural Networks for Natural Language Understanding». En: *CoRR*.

Målgren, Ann-Sofie y Camilla Ledin (2012). «La importancia del juego para adquirir una lengua extranjera: Un estudio cualitativo de profesores de lenguas extranjeras y sus pensamientos sobre el juego como herramienta en el aula». En: *Independent thesis Advanced level*. URL: http://www.diva-portal.org/smash/get/diva2:558137/FULLTEXT01.pdf.

Martínez, Miguel Ballesteros (2010). «Mejora de la Precisión para el Análisis de Dependencias usando Maltparser para el Castellano». En: *Proyecto de Fin de Máster de Sistemas Inteligentes*.

Michaud, Lisa y Kathleen Mccoy (2006). «Capturing the Evolution of Grammatical Knowledge in a CALL System for Deaf Learners of English.» En: *I. J. Artificial Intelligence in Education* 16, págs. 65-97.

Mikolov, Tomas et al. (2013). «Distributed Representations of Words and Phrases and their Compositionality». En: *Advances in neural information processing systems*.

Miltsakaki, E. y K. Kukich (2004). «Evaluation of text coherence for electronic essay scoring systems.» En: *Natural Language Engineering*.

Navigli, Roberto y Paola Velardi (2010). «Learning Word-Class Lattices for Definition and Hypernym Extraction». En: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Ortega Mendoza, Rosa María (2007). «Descubrimiento Automático de Hipónimos a partir de Texto no Estructurado». En: *Tesis de lic. México: Instituto Nacional de Astrofísica, Óptica y Electrónica*.

Piskorski, Jakub y Roman Yangarber (2012). «Information Extraction: Past, Present and Future». En: *Book - Theory and Applications of Natural Language Processing*.

Przepiórkowski, Adam et al. (2007). «Towards the automatic extraction of definitions in Slavic». En: *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing*.

Qiu, Lin et al. (2014). «Learning Word Representation Considering Proximity and Ambiguity». En: *AAAI*.

Quezada Narvaéz, Carolina (2011). «La popularidad del inglés en el siglo XXI». En: *Tlatemoani - Universidad de Málaga*.

Raganato, Alessandro, Jose Camacho-Collados y Roberto Navigli (2017). «Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison». En: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, págs. 99-110.

Reshamwala, Alpa, Prajakta Pawar y Dhirendra Mishra (2013). «Review on Natural Language Processing». En: *IRACST – Engineering Science and Technology: An International Journal (ESTIJ), ISSN: 2250-3498, Vol.3, No.1, February 2013*.

Schütze, Hinrich (1998). «Automatic Word Sense Discrimination». En: *Computational Linguistics* 24, págs. 97-123.

Shaw, Peter, Jakob Uszkoreit y Ashish Vaswani (2018). «Self-Attention with Relative Position Representations». En: *NAACL-HLT*.

Shwartz, Vered, Yoav Goldberg y Ido Dagan (2016). «Improving Hypernymy Detection with an Integrated Path-based and Distributional Method». En: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Singh, Sonit (2018). «Natural Language Processing for Information Extraction». En: *CoRR* abs/1807.02383.

Sultan, Arafat, Steven Bethard y Tamara Sumner (2014). «Towards automatic identification of core concepts in educational resources». En: *14th ACM/IEEE-CS Joint Conference on Digital Libraries*.

Van Gemert, Lisa (2016). «Gifted Guru». En: URL: http://www.giftedguru.com/the-benefits-of-wordsearches/.

Wu, Felix et al. (2019). «Pay Less Attention with Lightweight and Dynamic Convolutions». En: *International Conference on Learning Representations*.

Yarowsky, David (1992). «Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora». En: *COLING*.

Zhang, Yuhao, Peng Qi y Christoper Manning (2018). «Graph Convolution over Pruned Dependency Trees Improves Relation Extraction». En: *Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018*.

Sitios web de referencia

[1] *AXOLOTL* (2015). http://www.corpus.unam.mx/axolotl. Último acceso: 06/05/2019.

[2] *BabelNet*. https://babelnet.org/. Último acceso: 07/05/2019.

[3] *Computer Science Chapter 1* (2018). https://quizlet.com/310225797/computer-science-chapter-1-flash-cards/. Último acceso: 06/03/2019.

[4] *Discourse Analysis* (2016). https://hpi.de/fileadmin/user_upload/fachgebiete/plattner/teaching/NaturalLanguageProcessing/NLP2016/NLP07_DiscourseAnalysis.pdf. Último acceso: 03/05/2019.

[5] *Ducksters - Education site*. https://www.ducksters.com/. Último acceso: 25/02/2019.

[6] *Ejemplo tomado de*. http://www.kidsfront.com/crossword/images/color/sm/crossword1.jpg. Último acceso: 07/04/2019.

[7] *Elephant - Definition*. https://simple.wikipedia.org/wiki/Elephant. Último acceso: 15/03/2019.

[8] *Enhanced dependencies - Relative clauses* (2014). http://universaldependencies.org/u/overview/enhanced-syntax.html. Último acceso: 27/03/2019.

[9] *ESLFast - A huge free online English learning resource* (2006). https://www.eslfast.com/. Último acceso: 26/02/2019.

[10] *GoogleNews-vectors* (2016). https://github.com/mmihaltz/Word2Vec-GoogleNews-vectors. Último acceso: 15/03/2019.

[11] *Language modeling*. https://nlpprogress.com/english/language_modeling.html. Último acceso: 06/05/2019.

[12] *Language Muse*. https://languagemuse.org/. Último acceso: 17/04/2019.

[13] *Latent Semantic Analysis (LSA)* (1998). http://lsa.colorado.edu/whatis.html. Último acceso: 07/05/2019.

[14] *Marco Común Europeo de Referencia para las Lenguas Modernas*. https://academico.unizar.es/estudios-de-grado/marco-comun-europeo-de-referencia-para-las-lenguas-modernas. Último acceso: 02/04/2019.

[15] *Morfología en Lingüística* (2016). https://www.significados.com/morfologia/. Último acceso: 03/05/2019.

[16] *Movers - Word list picture book* (2018). https://www.cambridgeenglish.org/images/149680-yle-movers-word-list.pdf. Último acceso: 15/04/2019.

[17] *Natural Language Processing (NLP) - Yuriy Guts* (2016). https://image.slidesharecdn.com/nlp-160709201345/95/natural-language-processing-nlp-38-638.jpg?cb=1468095414. Último acceso: 03/05/2019.

[18] *NLP: Explaining Neural Language Modeling* (2017). https://mchromiak.github.io/articles/2017/Nov/30/Explaining-Neural-Language-Modeling/#.XNNlZo5KhPY. Último acceso: 08/05/2019.

[19] *Paper Summary: Evaluation of sentence embeddings in downstream and linguistic probing tasks* (2018). https://towardsdatascience.com/paper-summary-evaluation-of-sentence-embeddings-in-downstream-and-linguistic-probing-tasks-5e6a8c63aab1. Último acceso: 06/05/2019.

[20] *Penguins - Definition*. https://www.ducksters.com/animals/penguins.php. Último acceso: 15/03/2019.
[21] *PIKES - Evaluation using Simple English Wikipedia* (2018). http://pikes.fbk.eu/eval-sew.html. Último acceso: 14/03/2019.
[22] *Plan Ceibal - Dispositivos*. https://www.ceibal.edu.uy/es/dispositivos/. Último acceso: 20/02/2019.
[23] *Pre A1 Starters - Word list picture book* (2018). https://www.cambridgeenglish.org/images/starters-word-list-picture-book.pdf. Último acceso: 15/04/2019.
[24] *Real Academia Española - determinante*. https://dle.rae.es/?id=DaJfrXM. Último acceso: 06/05/2019.
[25] *Roget's Thesaurus* (1999). http://www.roget.org/. Último acceso: 07/05/2019.
[26] *Scrapy*. https://scrapy.org/. Último acceso: 13/04/2019.
[27] *Simple English Wikipedia*. https://simple.wikipedia.org/. Último acceso: 20/02/2019.
[28] *Stanford typed dependencies manual* (2008). https://nlp.stanford.edu/software/dependencies_manual.pdf. Último acceso: 26/03/2019.
[29] *Text Evaluator* (2014). https://www.ets.org/c/23491/. Último acceso: 17/04/2019.
[30] *The 7 Best Games to Learn English In Groups and Alone* (2017). https://www.prepscholar.com/toefl/blog/games-learn-english/. Último acceso: 08/05/2019.
[31] *The Wesbury Lab Wikipedia corpus* (2010). http://www.psych.ualberta.ca/~westburylab/downloads/westburylab.wikicorp.download.html. Último acceso: 08/05/2019.
[32] *Versi - School of English* (2004). https://versi.es. Último acceso: 12/04/2019.
[33] *What the heck is Word Embedding* (2011). https://towardsdatascience.com/what-the-heck-is-word-embedding-b30f67f01c81. Último acceso: 08/05/2019.
[34] *WordNet - Una Base de Datos Léxica para el Inglés*. https://wordnet.princeton.edu/. Último acceso: 07/05/2019.
[35] *Wordsmyth for children - The premier Educational Dictionary-Thesaurus*. https://kids.wordsmyth.net. Último acceso: 20/02/2019.