Universidad Internacional de La Rioja (UNIR)

Escuela Superior de Ingeniería y Tecnología (ESIT)

Máster Universitario en Inteligencia Artificial

Procesamiento del Lenguaje Natural para determinar el grado de acierto o validez de una respuesta corta basado en palabras clave

Trabajo Fin de Máster

elaborado por: Sanz Fayos, Javier

Dirigido por: De La Fuente Valentín, Luis

Codirigido por: Elena Verdú Perez

Ciudad:Valencia

Fecha: 20 de Septiembre de 2021

Índice de Contenidos

Índice de Ilustraciones

Índice de Tablas

Resumen

En ocasiones, el contenido y calidad de un examen no se adecúa a los objetivos marcados para la prueba. Por eso, se pretende desarrollar una herramienta capaz de reconocer las respuestas de contenido más pobre, facilitando *feedback* al estudiante y/o al profesor. En este trabajo, se estudia la influencia del reconocimiento de palabras clave como método de evaluación semiautomático en relación a la calificación final de una respuesta. Para ello, se elaboran técnicas de segmentación, lematización y desambiguación del sentido de las palabras mediante un corpus seleccionado. Además de analizar palabras clave en la respuesta pre-procesada del alumno, se estudian sus sinónimos y antónimos recogidos en dos tesauros creados automáticamente. Los resultados obtenidos avalan el funcionamiento del método desarrollado pero indican una falta de relación entre la técnica de evaluación desarrollada y la real, evidenciando la falta de impacto de las palabras clave sobre la nota.

Palabras Clave: Evaluación semiautomática, palabras clave, Procesamiento del Lenguaje Natural, retroalimentación, tesauro.

Abstract

Sometimes, the content and quality of an exam does not match the objectives set for the test. For this reason, it is intended to develop a tool capable of recognizing the answers to the poorest content, providing feedback to the student and/or the teacher. In this work, the influence of keyword recognition is studied as a semi-automatic evaluation method in relation to the final rating of an answer. For achieving this, techniques of segmentation, lemmatization and word sense disambiguation are developed through a selected corpus. Not only keywords are analyzed in the student's pre-processed response, their synonyms and antonyms collected in two automatically created dictionaries are studied too. The obtained results support the functioning of the developed method but also indicate a lack of relationship between the developed evaluation technique and the real one, evidencing the lack of impact that keywords have on the mark.

Keywords: Dictionary, feedback, keywords, Natural Language Processing, semi-automatic evaluation.

Capítulo 1

Introducción

1.1. Motivación

Desde tiempos inmemoriales, el hombre ha buscado la manera en la que poder realizar una actividad en el menor tiempo posible, optimizando su esfuerzo y coste para explotar con mayor eficiencia los recursos disponibles en su entorno. Con el paso del tiempo, instrumentos y máquinas han sido desarrollados proporcionalmente a la tecnología del momento, como medio de resolución de ciertas necesidades. En la actualidad, se busca mejorar las capacidades de las máquinas a la hora de realizar tareas típicamente humanas, facilitando la colaboración y comunicación entre ambos para aprovechar la potencia que ofrecen éstas en múltiples campos, tales como medicina, videojuegos, asistencia, etc. Una vía de investigación en auge de esta materia, en el momento presente, es el Procesamiento del Lenguaje Natural (PLN), que se refiere a todo el conjunto de técnicas utilizadas para adquirir, procesar y comprender el lenguaje humano, tanto oral como escrito (Figura 1.1). Esta área de la Inteligencia Artificial (IA), se encuentra cada vez más incorporada en dispositivos o aplicaciones de uso diario como asistentes de voz (Siri o Alexa), chatbots, correcciones gramaticales automáticas, sugerencias en búsquedas de información, traducción automática o análisis de sentimiento, entre otros.

La corrección de exámenes y trabajos es una de las tareas más tediosas y menos agradecidas que deben realizar los maestros, pero a su vez, se trata de una de las más importantes. La corrección sirve tanto para evaluar los conocimientos adquiridos por los alumnos, como para reforzar el aprendizaje de los mismos exponiendo y argumentando los errores cometidos. Se trata de una tarea que es exigente, produce un gran desgaste y no siempre es

precisa, pues el profesor puede cometer errores en el juicio de alguna pregunta o en el cálculo final de la nota. La precisión en la evaluación, además, se ve más comprometida en preguntas de respuesta abierta porque, aunque permiten valorar de forma más personalizada las respuestas, la diferencia de criterios de corrección del profesorado puede provocar, en ocasiones, que la calificación que recibe el alumno esté ligada a cierta subjetividad (McNamara et al., 2015) (Rodrigues y Oliveira, 2014). Esta variabilidad en la nota, aunque se trate de mitigar con el uso de rúbricas, podría penalizar o beneficiar a ciertos estudiantes en determinadas pruebas (Brame, 2019).

Por este motivo, se pretende investigar y desarrollar una aplicación software, basada en PLN, que contribuya al desarrollo y mejoría de las aproximaciones existentes en la actualidad para la evaluación automática/semiautomática de contenido. Se buscará, por tanto, contribuir al avance de este tipo de sistemas para perfeccionar su precisión y objetividad, de forma que se consigan suprimir, así, los inconvenientes de la corrección tradicional en un futuro.

1.2. Planteamiento del trabajo

El lenguaje es una fuente rica de información en todos sus componentes. Sin embargo, el tratamiento de ésta encierra un conjunto de desafíos, tales como la ambigüedad o dimensionalidad del lenguaje, que dificultan su procesamiento. El primero de éstos, se encuentra relacionado con la capacidad de describir una situación de maneras distintas (sinonimia y antonimia) y con fenómenos que se producen en el lenguaje como la homonimia (palabra que tiene varios sentidos que no están relacionados entre sí), homografía (palabra que tiene la misma grafía que otra) u homofonía (palabra que suena igual que otra). El segundo desafío, está relacionado con la cantidad de palabras, significados y acepciones de las mismas que dificultan o enlentecen el cómputo de algunas operaciones en un programa software. De esta manera, para que la aplicación sea eficiente, se debe buscar un equilibrio entre información almacenada y rapidez de ejecución. Así, el tiempo de respuesta de la misma será adecuado para el entorno de ejecución en el que haya sido implementado.

En el presente documento, ambas dificultades serán contempladas, incidiendo principalmente en la primera de ellas, fundamental para el reconocimiento de palabras y significados en una oración determinada. Se pretenderá, de esta forma, identificar la aparición

Figura 1.1: Comunicación máquina humano

Fuente: (Ambróz, 2019)

de determinados conceptos a partir del análisis de todas las posibilidades de expresión de ese significado. Esta investigación, se llevará a cabo aplicada a un contexto estudiantil, y consistirá en evaluar las respuestas de los alumnos a determinadas preguntas de una materia. Es decir, se pretenderá valorar de forma automática y relativamente objetiva, a partir de palabras clave, la similitud entre la contestación de un alumno y la respuesta que pretende su profesor, proporcionando *feedback* sobre las diferencias y similitudes de contenido existentes. Para ello, se pretenderá dar respuesta a las siguientes preguntas de investigación:

- ¿En qué medida es válida la evaluación de una respuesta basándose en un conjunto de palabras clave para determinar la calidad de una respuesta?
- ¿Existe correlación entre el criterio utilizado por los examinadores y el que emplea este método?
- ¿Se puede generar *feedback* competente con este método para justificarle al alumno sus aciertos y errores de contenido en su respuesta?

Es importante destacar, por otra parte, que este trabajo desarrollará una sección del plano semántico de un proyecto investigativo de mayor calibre que pretende, a partir de la combinación de las salidas de múltiples métodos que evalúan diferentes aspectos de la respuesta del estudiante, diseñar un formato de evaluación basado en la rúbrica

ponderada de los diferentes planos, de forma que se consiga generar una calificación general y argumentada de dicha respuesta.

1.3. Estructura del trabajo

El presente documento se organiza de la siguiente manera: la segunda sección (Capítulo 2), es una contextualización de los antecedentes y diversas técnicas existentes en la actualidad en relación al proyecto; la tercera sección (Capítulo 3), presenta los objetivos del trabajo y la metodología utilizada para el mismo; la cuarta (Capítulo 4), explica detalladamente las decisiones tomadas para desarrollar el trabajo; la quinta (Capítulo 5), expone los resultados obtenidos tras aplicar las diversas técnicas; y la sexta (Capítulo 6), analiza los resultados y extrae las pertinentes conclusiones, relegando aquellas cuestiones no resueltas o no muy óptimas para trabajos futuros.

Capítulo 2

Contexto y Estado del Arte

2.1. Dificultades del PLN en la actualidad

El primer paso para el procesamiento informático del conocimiento lingüístico es la representación formal de dicho conocimiento. Para ello, es necesario disponer de un buen corpus, pues la adecuación del mismo influirá directamente en la calidad de la información que se pueda extraer de él. El corpus, además, debe estar formado de textos relacionados con el dominio o ámbito que se desea trabajar para conseguir lograr información más precisa. Este requerimiento puede llegar a ser, a veces, una labor difícil de completar correctamente. En ciertos dominios, como el médico, satisfacer esos requisitos puede llegar a ser complicado, fruto de la especificidad del vocabulario y la falta de preparación de algunos registros clínicos para poder aplicar técnicas de PLN (Torrijos, 2020).

El idioma es también una característica muy importante a tener en cuenta a la hora de seleccionar el corpus. En la actualidad, las librerías más importantes de PLN están desarrolladas, sobre todo, para lengua inglesa. De esta manera, y a pesar de que algunas incluyen soporte para otras lenguas, como el español, algunos comandos o técnicas que logran buenos resultados con el inglés pueden no ser tan efectivas con los demás idiomas, dificultando la tarea de procesamiento y obligando a programar a más bajo nivel para conseguir un resultado similar.

Por otra parte, existen múltiples recursos creados para representar la información lingüística, entre ellos, los glosarios especializados, taxonomías, tesauros y ontologías (Vilches-Blázquez, 2019). Estos recursos, están formados según el criterio de expertos en determinadas áreas. Un problema que surge, sobre todo para la desambiguación del sentido de las

palabras, es la falta de consenso que tienen entre ellos, pues no siempre están de acuerdo en qué palabra pertenece en qué sentido. Por ejemplo, en *Senseval-2*, donde se utilizaron distinciones de sentido detalladas, los anotadores humanos estuvieron de acuerdo únicamente en el 85 % de las ocurrencias de palabras (Fellbaum, 1997). Esto se debe a que el significado de las palabras es, en principio, infinitamente variable y sensible al contexto, por lo que es susceptible (en algunos casos) a interpretación (Snyder y Palmer, 2004). Además, fruto de esta división de opiniones, es posible encontrar diferentes sentidos de palabras recogidas en los recursos mencionados, pudiendo obtener diferentes representaciones en función del material utilizado.

Por último, existen dificultades en el procesamiento debido a la cuestión lingüística conocida como pragmática, en la que se plantea la necesidad de conocimiento del sentido común para identificar el sentido de las palabras, recurriendo de esta manera a la información imprimida en el contexto en el que se formula la sentencia para conocer su significado (Gil y Rodríguez, 1996). Este conocimiento, es importante también para realizar un correcto análisis de sentimiento de una determinada sentencia, independientemente del uso de ironía o retórica en ella, o para conseguir una identificación acertada de anáforas, catáforas o referencias en el texto. Es decir, los computadores analizan las oraciones de forma literal, mientras que los humanos podemos realizar abstracciones y entender metáforas. El problema de transferir este tipo de conocimiento a nivel de máquina es, precisamente, la dificultad de definir qué es ironía y qué no cuando, en muchas ocasiones, una persona humana también experimenta dificultades a la hora de interpretarlo o entenderlo (Ambróz, 2019).

2.2. PLN en educación

En el campo de la educación, el PLN busca extraer, modular y transformar a nivel computacional el conocimiento de un experto en un determinado campo, con el objetivo de construir y perfeccionar los modelos de conocimiento actuales. Se pretende que sirvan, dichos modelos, como herramientas de monitorización o asesoramiento en determinadas cuestiones, tales como la corrección o evaluación de distintos aspectos relacionados con el aprendizaje (ortografía, contenido, cohesión gramatical, etc.). La capacidad de emular el conocimiento que aporta el experto es posible dado el dominio específico que tiene el modelo del contenido a evaluar. Las reglas y conocimientos, de ese dominio en particular, son

obtenidas a partir de diversas técnicas basadas en herramientas de aprendizaje de ontologías, entre las que se erige el aprendizaje automático. A partir de ellas, es posible extraer conocimiento a partir de textos rápidamente, a gran escala, y corrigiendo posibles sesgos de naturaleza humana (sin la dependencia total del experto en la materia en cuestión) (Zhou, 2007).

Este escenario, propicia la posibilidad de mejorar la educación en línea actual (Maldonado et al., 2017). La base de una gran parte de los sistemas de evaluación automática actuales, como se ha avanzado anteriormente, está construida a partir del conocimiento de expertos y la asignación de conceptos clave de un dominio en concreto, pero también a partir de ontologías generadas automáticamente, con el fin de determinar el nivel de conocimiento plasmado por un alumno en una respuesta dada. *Assisted Study*, por ejemplo, es una herramienta que ayuda a la corrección sintáctica y semántica de exámenes de respuesta abierta, proporcionando *feedback* con el que argumentar la calificación calculada, de forma que el alumno sabe qué conceptos planteó correctamente y qué partes quedaron incompletas (Rodrigues y Oliveira, 2014). También, se favorece la posibilidad de construcción e integración automática de múltiples ontologías, de forma similar al sistema desarrollado por Deline et al. (2010), que integra una ontología de dominio por niveles: curso, programa, y estudiante; o al desarrollado por Frost y McCray (2012), cuyo sistema de búsquedas permite la actualización de la ontología en el campo de la medicina.

Existen también sistemas que proporcionan una serie de índices analizables que sirven para obtener métricas con las que evaluar de mejor manera al estudiante en cuestión. Westera et al. (2018) consigue generar hasta 200 índices para evaluar el aprobado o suspenso del alumno, en un entorno de enseñanza online relativamente recreativo. Este procedimiento de corrección se efectúa a partir de un vasto *dataset* de textos previamente calificados, que sirven como *input* de diversos métodos de aprendizaje automático, tales como regresión lineal, perceptrón multicapa o regresión de vectores de soporte. Otra solución integral e interesante de evaluación de textos, desde distintas perspectivas y considerando múltiples métricas, es *ReaderBench* (Dascalu, 2014). Se trata de un entorno software de código abierto que permite evaluar la complejidad de textos, resúmenes y explicaciones, y medir la colaboración social dentro de un grupo. Para ello, utiliza técnicas de minería de texto, procesamiento de lenguaje natural y herramientas de análisis de redes sociales. *ReaderBench*, además, utiliza una representación del discurso basada en la cohesión, y en su

diseño se han integrado los índices provistos por los sistemas *E-rater* (Ramineni, 2013), *iSTART* (McNamara et al., 2004) y *CohMetrix* (Graesser et al., 2011).

Sistemas, como el propuesto por Panaite et al. (2018), se nutren del conocimiento que proporcionan este tipo de índices para efectuar la calificación de respuestas cortas. Los índices, en este caso en particular, son obtenidos a partir de *ReaderBench*, y se utilizan para evaluar las respuestas con más sesgo que en las aproximaciones anteriormente mencionadas. La principal diferencia, entre ellos, es que éste consigue agrupar la respuesta con un porcentaje de acierto casi idéntico al de un experto humano en una de las cuatro siguientes categorías: pobre, suficiente, buena y excelente. Otro sistema similar, pero más centrado al entrenamiento del alumno en la escritura de ensayos, es *Writing Pal*, que utiliza PLN para valorar y proporcionar *feedback* de la calidad del escrito del alumno, incluyendo índices de evaluación para identificar la complejidad, cohesión, retórica y lingüística del texto.

La evaluación de este tipo de respuestas pueden ser, también, conseguidas de forma parecida a partir del uso de motores de puntuación automatizados, como *C-rater*. Esta aproximación, utiliza una estructura de argumento de predicado, referencia pronominal, análisis morfológico y sinónimos para asignar crédito total o parcial a una pregunta de respuesta corta. Este crédito, fruto de la comparación del modelo obtenido a partir del conocimiento experto y el obtenido de las respuestas de los alumnos, mide, por tanto, la desviación entre la contestación y aquello que espera el profesor. La medición la realiza, además, con una similitud cercana al 84 % (con respecto al criterio experto humano) (Leacock y Chodorow, 2003). Con este tipo de métodos, el profesor conseguiría reducir la carga de trabajo de evaluación, pues únicamente entraría a supervisar aquellas pruebas cuyo resultado predicho tuviera un bajo nivel de confianza.

En general, se busca dar soporte al estudiante, no sólo a partir de corrección o evaluación de determinadas cuestiones. *AutoTutor*, por ejemplo, es un sistema de tutoría de lenguaje natural capaz de dialogar y orientar al alumno usando un procedimiento similar al que emplearía un tutor humano. Para adaptarse mejor a las características particulares de cada alumno y buscar las características que mejor puedan ayudar o motivar su aprendizaje, es capaz de construir un modelo de éste. Así, se va perfeccionando a partir de la interacción y las respuestas del estudiante con el sistema, de forma que cada vez lo "comprende" mejor (Nye et al., 2014). Otros ejemplos de tutores o ITSs (Intelligent Tutoring

Systems, en inglés) implementados con cierto éxito son: *Why2*, *CIRCSIM-Tutor*, *GuruTutor*, *DeepTutor* o *MetaTutor*. Éstos, están también basados en mecanismos de aprendizaje adaptativo para el estudiante, y ofrecen garantías de efectividad en la tutoría similares a las que proporcionaría una persona humana (Rus et al., 2013).

Por otra parte, existen otro tipo de técnicas, basadas en PLN y aprendizaje automático, que se aplican para orientar al alumno en la realización de determinadas tareas. Por ejemplo, Chen et al. (Chen et al., 2018) incorpora en su sistema una componente de salida visual, en forma de sociograma, para representar la participación del alumno en foros de debate. De esta forma, se muestran las interacciones establecidas del alumno con todos los demás estudiantes, permitiendo el seguimiento de la línea de acción que éste ha tomado. Asimismo, se le facilita la labor de identificación de los conceptos que le faltan por defender, para que se pueda centrar en ellos.

Otros sistemas aplicados para analizar redes sociales, foros, o actividades de debate, como el desarrollado por Xie et al (Xie et al., 2018), miden el liderazgo ejercido por los estudiantes a partir de las contribuciones que realizan al debate, aunque en este caso en particular, sin arrojar retroalimentación del motivo de la calificación. Por otra parte, Hernández-Lara et al. (Hernández-Lara et al., 2019) permite analizar el contenido de los foros para predecir el rendimiento del aprendizaje del alumno y la evaluación de preguntas de respuesta corta incluyendo, a diferencia de la técnica anterior, cierto *feedback*.

Como se puede observar, los resultados existentes en la actualidad son prometedores, sin embargo, conseguir modelar conocimiento experto es laborioso y costoso, tanto en tiempo como en dinero. Se ha demostrado, además, que en ocasiones la abstracción de conceptos clave de una única explicación predefinida es más precisa que la realizada a partir de un corpus de explicaciones de varios expertos. Esto se debe, principalmente, a las distintas formas que tienen cada uno de ellos de entender el problema. Por tanto, no siempre supone una clara ventaja la realización de un entrenamiento de forma muy supervisada (Zouaq et al., 2011).

Por último, a pesar de que en la actualidad se están promoviendo métodos de abstracción automática del conocimiento experto, falta todavía investigar más en esta área para mejorar algoritmos, sobre todo de PLN, que perfeccionen las técnicas de identificación y

extracción de los componentes semánticos actuales, como hace *OntoCmap*, para construir mejores modelos de conocimiento y de evaluación de los estudiantes (Zouaq et al., 2011).

2.3. Desambiguación del sentido de las palabras

La ambigüedad en los sentidos de las palabras es un fenómeno que se produce en todas las lenguas cuando una misma palabra tiene múltiples sentidos. Por eso, en tareas de PLN, se busca conocer el significado real de esa palabra en distintas oraciones y contextos. Esta labor de desambiguación (Word Sense Disambiguation; WSD, en inglés), es particularmente compleja en los casos en los que la cantidad de sentidos de la palabra es elevada (Kalita y Barman, 2015). Por este motivo, es crucial seleccionar esta cuestión correctamente o, de lo contrario, se puede alterar completamente el significado de la oración, generando ineficiencias en ciertas aplicaciones fundamentadas en el conocimiento de la semántica como, por ejemplo, la traducción automática.

Para abordar la tarea de WSD, en la actualidad, existen dos enfoques principales: el profundo y el superficial. Los enfoques profundos trabajan con la presunción de acceso completo al conocimiento del mundo. Son, por tanto, aproximaciones difícilmente implementables, pues semejante cuerpo de conocimiento no existe, de momento, en un formato compatible para ser procesado por computador, salvo en dominios muy limitados (Elkan y Greiner, 1993). Los enfoques superficiales, por otra parte, dependen de reglas introducidas por humanos o generadas de forma automática por el ordenador a partir de un corpus correctamente etiquetado. Estos enfoques, aunque teóricamente son menos poderosos que los profundos, en la práctica tienen mejores resultados (Amores et al., 2016). Los enfoques superficiales se clasifican, a su vez, en tres bloques: los basados en diccionarios y conocimientos, los basados en métodos supervisados y los basados en no supervisados (Fellbaum, 1997).

De los métodos basados en diccionarios, destaca el algoritmo de *Lesk*, que selecciona el sentido de la palabra ambigua cuya definición en el diccionario (firma) comparte más cantidad de palabras con sus vecinos en la oración o con las firmas de éstos. Es decir, se basa en la hipótesis de que las palabras, en un determinado rango de vecindad, tenderán a compartir significado o estarán relacionadas entre sí (Lesk, 1986). Otra aproximación similar consiste en buscar la ruta más corta entre dos palabras, buscando la segunda

palabra entre las firmas de la primera. En caso de no encontrarlo, se analizan las firmas de las palabras que componen la definición anterior, y así sucesivamente. De esta manera, la primera palabra se desambigua seleccionando la firma cuya distancia es menor con la segunda palabra (Diamantini et al., 2015).

Los métodos de aprendizaje supervisado, por otro lado, se basan en el contexto para la extracción de significado, pues se considera que a partir de él se puede extraer suficiente información como para eliminar la ambigüedad. Para ello, se construyen modelos de clasificación de los sentidos de las palabras utilizando técnicas de aprendizaje supervisado, como máquinas vectoriales de soporte, redes bayesianas, árboles de decisión o redes neuronales. Estos modelos son entrenados a partir de un conjunto de datos y ejemplos en los que aparece la palabra objetivo. Un gran limitante de este tipo de métodos es que requieren muchos datos para ser efectivos y la anotación debe realizarse de forma manual, lo que supone un gran gasto de tiempo para una tarea que es tediosa. Además, son altamente dependientes del lenguaje y el dominio en el que han sido entrenados, por lo que únicamente sirven para desambiguar palabras concretas en contextos específicos (Amores et al., 2016).

Los no supervisados, tienen la ventaja respecto a los anteriores de que no necesitan partir de textos etiquetados (Amores et al., 2016). Se fundamentan en la premisa de que los sentidos pueden ser inducidos a partir del contexto. Para ello, se basan en grafos y en ciertas métricas de similitud para generar grupos de ocurrencia de palabras, con la palabra objetivo como vector de la agrupación, asignando las nuevas ocurrencias a cada grupo en función de su similitud (Schütze, 1998). Se pueden utilizar, además, bases de datos léxicas como herramienta de soporte para el mapeado de los sentidos, tales como *WordNet*, *ConceptNet* o *BabelNet*. Este enfoque es más robusto que el anterior, pues es independiente de la lengua y del dominio; sin embargo, requiere muchas iteraciones para calcularse y, en la actualidad, los valores conseguidos de efectividad en la desambiguación son bajos (Amores et al., 2016).

Existen, además, otras técnicas como el análisis morfosintáctico que pueden servir como métodos indirectos de desambiguación de palabras. En el análisis morfosintáctico, este fenómeno se produce cuando una palabra ambigua, que pertenece a más de una categoría gramatical, se etiqueta correctamente según el contexto de la frase analizada, descartando así los sentidos pertenecientes a las restantes categorías.

Como se puede observar, la tarea de WSD no está resuelta, y se encuentra en fase de investigación. Con el objetivo de desarrollarla y a fin de proporcionar y compartir datos y procedimientos, se realizan campañas públicas de evaluación específicas. Por ejemplo, *Senseval* es un concurso internacional de WSD que se celebra cada tres años desde 1998. Su objetivo es, entre otros, preparar uno o varios corpus de prueba y comparar y publicar el resultado de las diferentes técnicas aplicadas por los participantes, para que sirva como referencia en futuras investigaciones.

2.4. Análisis morfosintáctico

El proceso de etiquetado o análisis morfosintáctico, también conocido como *POS tagging* en inglés, consiste en asignar una etiqueta a cada una de las unidades que componen una oración. Estas etiquetas representan la categoría gramatical que tiene cada palabra en la frase, y se suelen mostrar en formatos específicos recomendados para la anotación, como el basado en *EAGLES*.

Existen tres grandes grupos para este propósito: los etiquetadores basados en reglas, los estadísticos y los híbridos (Rodríguez, 2007). Los primeros utilizan conocimiento lingüístico expresado en forma de reglas/sentencias/restricciones creadas manualmente para determinar la combinación aceptable de posibles etiquetas en cada caso. Un ejemplo de esta familia es *EngCG* (Karlsson et al., 1995), un sistema de alta precisión que integra un millar de reglas y que se implementa para la lengua inglesa.

Los segundos, obtienen generalizaciones y relaciones matemáticas mediante el entrenamiento que efectúan a partir de un corpus anotado de palabras (también denominadas *tokens*) con categorías gramaticales. Estos modelos aprenden a asignar las categorías gramaticales de las palabras de una secuencia a partir de la evidencia empírica abstraída. En este grupo, es creciente el uso de técnicas de aprendizaje automático, tanto supervisado como no supervisado (o al menos con supervisión muy limitada), que exprimen al máximo los corpus voluminosos que se disponen en la actualidad. Por ejemplo, las Redes Neuronales Recurrentes (RNN) se utilizan para analizar, por separado y orden de aparición, cada uno de los *tokens* de la secuencia recibida como *input*. Estas técnicas se utilizan para construir modelos sencillos del lenguaje, como los bigramas o los trigramas, pero también modelos complejos como el descrito por Brill (1995).

Para este tipo de análisis, una de las técnicas erigidas con mayor popularidad es los modelos ocultos de *Márkov* (Hidden Markov Model, HMM). Un modelo de *Márkov* es un modelo estocástico (probabilístico) que se utiliza para representar un sistema en el que los estados futuros dependen únicamente del estado actual. En este modelo, que puede concebirse de forma similar a un modelo de autómatas finito, los nodos representan a los estados y cada una de las líneas de flujo existentes representa una posible transición entre éstos (Jurafsky y Martin, 2008).

Para adquirir la capacidad de etiquetar de forma morfosintáctica, los HMM deben ser entrenados a partir de un corpus anotado que incluya la categoría gramatical de cada palabra. A partir de estos datos, el modelo establece una serie de relaciones de carácter probabilístico y fija las estimaciones de máxima verosimilitud para cada uno de los estados. Con estas operaciones, es posible calcular las probabilidades de transición y de emisión de ese HMM. Las probabilidades de transición se corresponden a la probabilidad de que se dé una etiqueta dada la etiqueta anterior; y las probabilidades de emisión representan la probabilidad de que una palabra concreta se asocie con una determinada etiqueta.

Con la información recogida de las probabilidades de transición y emisión, es posible elegir la secuencia de etiquetas más probable de la sentencia analizada usando algoritmos de decodificación del HMM, tales como el algoritmo de *Viterbi* o el algoritmo *forward* (Jurafsky y Martin, 2008).

Por último, los sistemas híbridos combinan técnicas de las dos anteriores familias para evitar las limitaciones que podrían llegar a tener por separado. De esta manera, se consigue construir un único desambiguador relativamente más completo o una combinación de éstos usando sistemas de votación, *bagging*, *boosting* o *bootstrapping*, entre otros (Rodríguez, 2007).

Capítulo 3

Descripción general de la contribución

3.1. Objetivo general

Investigar la posibilidad de hacer una evaluación semiautomatizada de la respuesta de un alumno a una pregunta concreta de un examen, basándose en el conjunto de palabras clave que el profesor espera que conteste.

3.2. Objetivos específicos

Es importante aclarar que con este estudio se busca facilitar la tarea de corrección del profesor y, de ninguna forma, se pretende sustituir su labor. Aquello que se desea, por tanto, es desarrollar un mecanismo de evaluación para filtrar las respuestas más pobres en cuanto a contenido de los alumnos, asegurando un mínimo de nivel en los documentos que recibiría el profesor. Además, dependiendo del criterio del profesor en la examinación, se podría dar la posibilidad o no de que el alumno pudiera recibir e interpretar cierto *feedback* con el que reformular su contestación.

Para alcanzar el objetivo general antes planteado, es necesario realizar un desglose de las acciones u objetivos intermedios que servirán como guía en el desarrollo del presente trabajo (Figura 3.1). Estos objetivos son los siguientes:

- Elaborar un diccionario que contenga las palabras clave (o *keywords*, en inglés) indicadas por el profesor, y ampliarlo automáticamente con los sinónimos y antónimos

de cada palabra.

- Recoger las respuestas de los alumnos y pre-procesarlas, implementando técnicas para filtrar los caracteres no significativos.
- Establecer una estrategia adecuada para la correcta comparación de palabras, teniendo en cuenta sus múltiples sentidos.
- Implementar un sistema de evaluación de la respuesta con *feedback* comprensible, basado en PLN, de las acciones que el programa ha tomado para justificar la calificación calculada.
- Comparar los resultados de la evaluación con las calificaciones reales otorgadas por profesores y comprobar si existe relación entre ambos métodos.

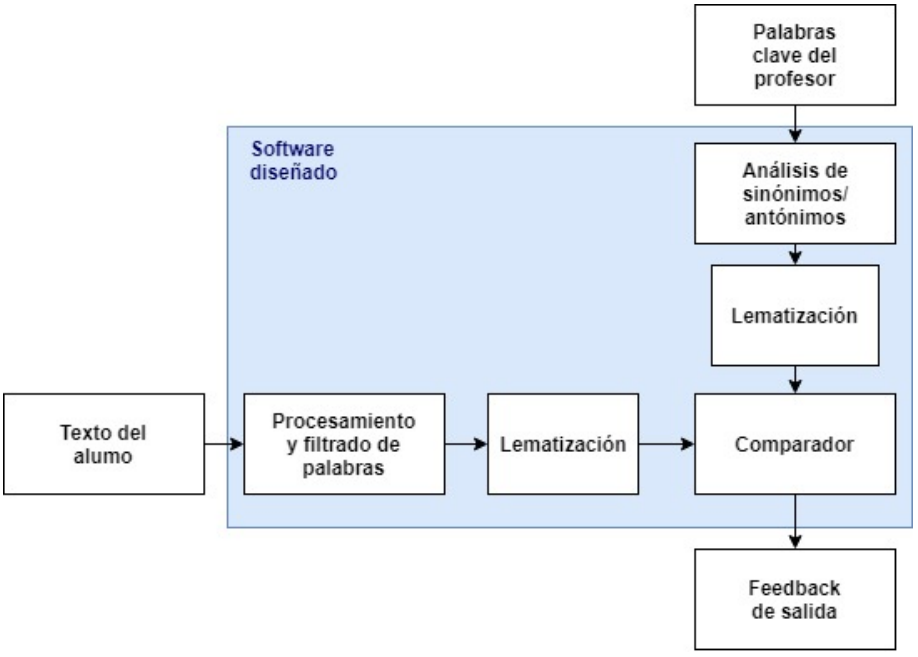


Figura 3.1: Diagrama de bloques general del proceso a desarrollar

3.3. Metodología

En el presente trabajo se empleará una adaptación de la metodología en espiral para el desarrollo del software. El funcionamiento de este tipo de metodología consiste en describir el ciclo de vida del lanzamiento del software por medio de espirales, que se repiten hasta que el producto se encuentra terminado (Figura 3.2). Estas espirales, son un abanico

de iteraciones en las que se va perfeccionando el producto, de forma que el software se desarrolla a partir de una serie de versiones incrementales donde el sistema diseñado es cada vez más completo (Galo, 2011) .

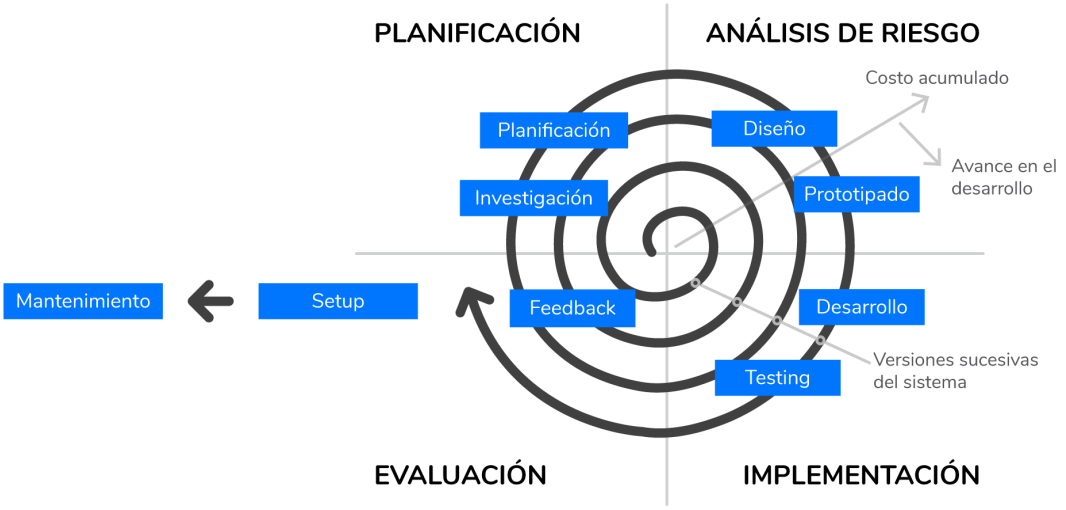


Figura 3.2: Fases de la metodología en espiral.

Fuente: (ASPgems, 2019)

Se implementó esta metodología en detrimento de otras debido a que las fases de las que consta y el modo de operación de la metodología en espiral es similar al trabajo investigativo que requiere este proyecto. De esta manera, la información proveniente del vasto conjunto de cambios en cada uno de los prototipos puede servir para reforzar la aproximación que se está realizando o para vislumbrar la necesidad de un cambio en ésta. Además, los resultados obtenidos de las pruebas en cada iteración sirven para actualizar el *baseline* del proyecto y pueden ser potencialmente interesantes para la comunidad científica, pues se trabaja con datos reales (debidamente anonimizados).

Las fases de las que se compone la metodología en espiral, relacionadas con este proyecto, son las siguientes:

- Planificación, donde se realizará un estudio del problema a resolver y una revisión del estado del arte actual, en busca de identificar las ventajas e inconvenientes de las aproximaciones existentes.
- Análisis de riesgo, donde se contemplarán y tratarán de prever posibles problemas o contratiempos que pudieran surgir durante el desarrollo de la investigación.

- Implementación, donde se realizarán labores de ingeniería del software para el desarrollo de un prototipo.
- Evaluación, donde se compararán y analizarán los resultados obtenidos con el *baseline* disponible de otros prototipos.

3.4. Descripción general de las partes o componentes de la propuesta

Esta investigación tiene como principal objetivo estudiar el impacto e idoneidad que tiene el reconocimiento de palabras clave en una respuesta como medida de evaluación total o parcial de un sistema de corrección de exámenes. Para ello, se formará primero un corpus representativo, aleatorio y equilibrado, fundamental para el desarrollo de técnicas de extracción de conocimiento útil para el software. Dada la dificultad y la cantidad de tiempo necesaria para construir un corpus de semejantes características, se optará por la selección de opciones ya trabajadas y fácilmente accesibles en Internet, como *WordNet* o *Wikicorpus*, que garanticen un mínimo de calidad en la información que se extraiga.

Se buscará, además, un corpus que no sea demasiado específico; o sea, que no esté compuesto principalmente por palabras altamente relacionadas con un determinado dominio. Se pretende, así, extraer información de la lengua en su totalidad; es decir, conocimiento perteneciente a diferentes variedades lingüísticas. Este tipo de corpus, es útil como fuente de datos para la elaboración de diccionarios generales y, por tanto, apropiado para la desambiguación de los sentidos de las palabras. De esta manera, la información aprovechable (de carácter semántico y/o sintáctico de los textos) se utilizará principalmente para WSD. Así, el significado de las palabras será siempre representativo y acorde a los requisitos que exige el profesor (relación directa con el significado de las palabras clave proporcionadas).

Para fundamentar el criterio de selección del corpus, se discutirán y analizarán las diferentes opciones con el tutor y cotutora de este proyecto, tras finalizar, de forma previa, una revisión de la literatura actual de este tema. El corpus final, estará compuesto de oraciones de textos en español, pues ese será el idioma en el que se evaluarán las respuestas de los alumnos. Es importante tratar de adecuar lo máximo posible el idioma del corpus a la lengua de escritura del examen, pues la estructura gramatical de otras vertientes del español y, sobre todo, de otros idiomas como el inglés, puede llegar a ser muy diferente.

Para trabajar el problema, se dispone de un *dataset* de 144 respuestas en formato *JSON* de una prueba real. Cada una de las entradas de este documento contiene la pregunta del examen, las palabras clave que deben incluirse en la respuesta, la respuesta del alumno, la nota que obtuvo el alumno con esa respuesta y un identificador anonimizado, tal y como se puede observar en el ejemplo adjuntado en la Sección I de los Anexos. A partir de las palabras clave de las preguntas, se formará un tesauro con éstas y con sus sinónimos y antónimos. Éstos, serán obtenidos mediante una búsqueda online reactiva en un diccionario de sinónimos y antónimos existente en Internet, cuya selección seguirá un proceso similar al tomado con el corpus. Este tesauro, se guardará en memoria interna del programa a desarrollar y será recorrido para identificar si la respuesta del alumno incluye palabras clave o palabras cercanas a éstas en cuanto a significado. Se utilizará el entorno de *Jupyter Notebook* que ofrece *Google Colab* para desarrollar el código, por su facilidad para programar en *Python* sin necesidad de instalación de paquetes y por su facilidad de integrar funciones de PLN.

Para facilitar el análisis del texto y el uso de determinadas técnicas de extracción de información (como el análisis morfosintáctico), se separará la respuesta en frases más simples. Con esta aproximación, además, se podrá comprobar la distribución de aparición de las palabras clave a lo largo de la respuesta, evitando así calificar positivamente una contestación que únicamente enumere las palabras clave (sin conexión o sentido). Se analizará, también, si existe alguna relación entre dicha distribución y la calificación calculada. A cada frase resultante obtenida, se le aplicará un proceso de *tokenización*, que separará cada cadena de caracteres en un *array* de palabras distinto. A continuación, se eliminarán los caracteres que no aportan valor al análisis, seleccionando y definiendo un conjunto de *stopwords* de la lengua española. Luego, se lematizarán tanto las palabras del *bag of words* restante de las contestaciones como las que componen el tesauro, para reducir la dimensionalidad y facilitar la comparación de las palabras.

En función del número total de palabras clave definidas por el tutor y el número de coincidencias en la respuesta del alumno, se ajustará el porcentaje final de acierto de la respuesta y se comparará con la calificación que tuvo en la realidad. En esta aproximación, no se contemplará introducir, inicialmente, ninguna ponderación en el conjunto de palabras clave. De esta manera, se considerará que todas tienen el mismo peso en la calificación, independientemente de la forma en la que aparezcan (de forma literal o mediante uno de

sus sinónimos/antónimos).

El sistema de *feedback* que se desarrollará, por tanto, estará basado en sentencias que servirán para informar del número de palabras clave incluidas y de su adecuación en la oración, a fin de prevenir su uso incorrecto en la misma (antónimos sin negar o sinónimos y palabras susceptibles de expresar un sentido diferente al deseado). Además, servirá como herramienta para justificar la calificación que se le asigne a la respuesta. Cabe destacar que esa nota estará basada únicamente en el número de *keywords* independientes identificados, y no en su frecuencia de aparición durante la oración. Por eso, se ajustarán y filtrarán las listas de palabras clave acertadas y de *feedback* proporcionado, de forma que no se repita información.

Por último, a partir de las notas reales de los exámenes incluidas en el *dataset*, se podrá realizar un análisis cualitativo para comprobar si existe o no relación entre este método y el criterio de evaluación del profesor o profesores que llevaron a cabo la corrección. Además, se analizará la influencia de cada palabra clave en función de la nota y de la aparición de otras palabras clave, para comprobar si alguno de los *keywords* tiene más importancia que los demás.

Capítulo 4

Desarrollo específico de la contribución

El sistema a desarrollar en el presente trabajo, *a priori*, puede preconcebirse como una técnica de comparación sencilla, palabra a palabra, en la que se recorre una cadena de caracteres (la respuesta del alumno) para buscar coincidencias. Sin embargo, esta labor aparentemente simple, esconde diversos entresijos que complican su procesamiento. El primero de ellos, está relacionado con las diferentes formas gramaticales en las que se puede representar una misma palabra. Por ejemplo, el sustantivo *chico* podría representarse, dependiendo del contexto de la oración, con género o número distinto; es decir, como *chica*, *chicos* o *chicas*. Cada una de esas variantes expresa un mismo significado, el de una persona joven, pues comparten la misma raíz. La diferencia, en este caso, está en el sufijo que se utiliza para completar el significado del lexema. Lo mismo sucede con otras categorías gramaticales como artículos, adjetivos o verbos, y con otro tipo de morfemas que complementan a la raíz como, por ejemplo, los existentes en las diferentes conjugaciones de los tiempos verbales.

El segundo de ellos, está relacionado con el problema ya mencionado de la desambiguación del sentido de las palabras. Por ejemplo, la palabra homónima *casa* tiene varios significados: hogar (1), estirpe (2), sociedad (3), contraer nupcias (4), acoplar (5) o combinar (6), entre otros. Si el reconocimiento de los *tokens* de las oraciones se ejecutara de forma literal, se podría tomar como válida una acepción que expresara un sentido distinto al de la palabra clave en cuestión.

La tercera gran dificultad que presenta esta cuestión, está relacionada con la riqueza del lenguaje español. Existen múltiples formas de hacer referencia a un mismo concepto de forma concreta, como los sinónimos o los antónimos. Sin embargo, también se puede expresar o definir un concepto indirectamente en un texto, mediante su explicación o a partir de una expresión. En este proyecto, se abordarán ciertas técnicas para facilitar el procesamiento de los textos, a partir de las que se pretende solventar, en la medida de lo posible, las tres dificultades mencionadas.

Tal y como se referenció en los capítulos previos, se antoja indispensable atesorar un buen corpus para poder extraer información con la que desarrollar PLN. En la actualidad, no es preciso descargar textos anotados para efectuar determinadas operaciones de PLN, pues existen librerías como *NLTK*, *Spacy*, *Freeling*, *Stanza*, *Scikit-learn* o *Gensim*, entre otras, que han sido trabajadas y optimizadas a partir de un gran abanico de textos y que ofrecen multitud de funciones y operaciones. No obstante, en esta investigación será preferible partir a más bajo nivel, a fin de poder personalizar, en mayor medida, la información que se extraerá en aras de la propia construcción de funciones.

Se apostó, consecuentemente, por la búsqueda de un determinado número de textos ya etiquetados con los que formar el corpus de trabajo. Se busca extraer conocimiento de carácter general y no de un dominio en concreto; es decir, no es necesario que el corpus incluya palabras demasiado específicas de un área en particular. De acuerdo con esta idea, se escogió *Wikicorpus* (Reese et al., 2010), que es un corpus trilingüe (catalán, español, inglés) gratuito, al menos en el momento de la realización de este estudio, enriquecido con información lingüística extraída de la *Wikipedia*. La anotación de sus textos, además, ha sido realizada utilizando *FreeLing* y el algoritmo de desambiguación de sentido de palabras *UKB*, que es quien asigna sentidos a *WordNet*, por lo que se trata de un corpus adecuado para hacer labores de WSD (Reese et al., 2010).

Se descargaron, inicialmente, 57 ficheros con frases anotadas de su página web. Cada fichero contiene, en promedio, unas 2 000 000 de palabras etiquetadas (el rango oscila entre 1 300 000 y 4 200 000 palabras, aproximadamente) y se identifica con el tag XML (ID) del documento que corresponde con el archivo de *Wikipedia* de donde se extrajo la información. Se realizó una primera fase de tratamiento de los datos con el fin de observar el formato, contenido y distribución de los mismos. La información relativa a cada palabra está representada en una nueva línea del fichero. En cada una de éstas, se indica el *token*

que representa la palabra, su lema, su etiqueta gramatical (*POS tag*) basada en el formato *EAGLES* y un código que expresa su sentido, tal y como se muestra a continuación:

convierte convertir VMIP3S0 00077276

Durante la realización de dicho estudio, se advirtió que en algunos ficheros existían ciertas líneas cuyo formato era distinto al esperado (Figura 4.1). Como se disponía de información suficiente, se descartaron aquellos ficheros que contuvieran alguna de esas líneas, reduciendo el conjunto de textos a 32 y el cómputo total de palabras a 60 880 634.

```
SE_10k_15k.txt
['no', 'se', 'no', 'se', 'NCCS000', '0']
['no', 'se', 'no', 'se', 'AQ0CS0', '0']
['no', 'conoce', 'no', 'conoce', 'VMIP3S0', '0']
['no', 'se', 'no', 'se', 'VMSI3S0', '0']
['de', '12', 'de', '12', 'Z', '0']
['15', 'años', '15', 'años', 'Z', '0']
['de', '9', 'de', '9', 'Z', '0']
['12', 'años', '12', 'años', 'Z', '0']
['cuatro', 'mil', 'cuatro', 'mil', 'AQ0CS0', '0']
['No', 'pido', 'no', 'pido', 'NP00000', '0']
SE_455k_460k.txt
['del', 'ego', 'ante', 'del', 'ego', 'ante', 'AQ0CS0', '0']
SE_430k_435k.txt
SE_15k_20k.txt
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
['Fz', '0']
```

Figura 4.1: Líneas de ficheros con distinto formato

4.1. Pre-procesamiento de información

En esta sección, se desarrollará el procedimiento y el criterio seguido para la selección de las técnicas de pre-procesamiento de las oraciones, profundizando, fundamentalmente, en las técnicas de lematización y de WSD. Se pretende, de esta manera, identificar las técnicas que más se ajusten a las características de este proyecto para facilitar la comparación e identificación posterior de palabras clave. Asimismo, se relacionarán las técnicas escogidas con las funciones desarrolladas en el programa *software* para desempeñar su mismo cometido.

4.1.1. Técnicas de lematización

Un texto se compone de oraciones formadas por palabras ricas en cuanto a volumen de información. Para elaborar ciertos algoritmos de PLN, no siempre es necesario utilizar todo su contenido. En estos casos, es recomendable pre-procesar el texto para simplificarlo, a partir de dos operaciones:

- *Tokenización*: Consiste en separar una frase en palabras, también llamadas *tokens*.
- Lematización: Consiste en reducir el significado de las palabras *tokenizadas* de la oración, transformando cada uno de los *tokens* a su forma canónica o lema.

Se aplicarán ambas técnicas al proyecto ya que la reducción a la forma canónica de las palabras supone eliminar sus diferentes formas gramaticales, tiempos verbales, etc. De esta manera, la tarea de identificación de las palabras clave se ve simplificada, pues la comparación entre los diversos lemas se puede efectuar de forma directa. Es importante resaltar que existen palabras ambiguas cuya correcta lematización depende de la oración, pues en función de su categoría gramatical se deberán reducir a una forma canónica u otra. El tratamiento de la ambigüedad de las palabras, se tratará con detalle en la Subsección 4.1.2.

A partir del procesamiento de los 32 archivos, es posible generar un diccionario que contenga las diferentes palabras del corpus y su lema asociado. Este diccionario, que en el programa se denominará como *LemmaDictionary*, se compondrá por varios niveles. El primer nivel lo formarán las claves de acceso al diccionario; es decir, el conjunto de diferentes palabras del corpus transformadas a minúsculas. El segundo nivel, estará formado por un diccionario cuyas claves serán las diferentes categorías gramaticales asociadas a la palabra del primer nivel y cuyo contenido será el lema correspondiente en cada caso.

Este diccionario permite lematizar palabras con bastante potencia y está estructurado para facilitar la tarea de WSD. Sin embargo, este recurso podría no ser suficiente si apareciera una palabra que no formulara entre las claves del mismo. Para evitar errores en el software, las funciones de lematización diseñadas tendrán la capacidad de recurrir, como segunda opción, al lematizador que proporciona el *pipeline* en español de *Stanza*.

4.1.2. Técnicas de WSD

Métodos valorados

Existen múltiples métodos de desambiguación de palabras y numerosas librerías y herramientas que incorporan soluciones a dicha cuestión. Por este motivo, se debe analizar el contexto y situación en el que se va a trabajar para tratar de escoger aquella opción que mejor se adapte a las necesidades del proyecto.

Se parte de la hipótesis basada en la inherente especificidad que poseen las palabras clave a analizar. Es decir, el profesor desea evaluar al alumno de un tema en concreto, por lo que el sentido de las palabras que espera que éste conteste estarán relacionadas con ese dominio. Por ejemplo, las palabras clave de un examen sobre el conflicto de la Segunda Guerra Mundial podrían ser: potencias del eje, nazis, aliados, Polonia, Berlín, bombas, campos de concentración y/o holocausto. En sí, consultando la *Web EuroWordNet Interface* (Adimen, s.f.), únicamente aparece "bomba" como palabra polisémica, significando artefacto explosivo y artefacto para mover líquidos. Esta regla se cumple en igual medida seleccionando otros posibles conceptos de dominios arbitrarios. Además, entre las distintas palabras clave de cada tema suelen abundar los nombres propios. Se considera, por tanto, que la mayoría de los conceptos más susceptibles de ser considerados clave, generalmente, no serán polisémicos y tendrán un sentido concreto.

Sin embargo, esta hipótesis no resuelve la cuestión, pues al igual que sucedía con la palabra "bomba", existen otras palabras con diversos sentidos cuya aparición se podría acentuar si el dominio del examen no fuera demasiado específico. Esta conjetura sirve para descartar aquellas opciones existentes que supongan un gran coste computacional y/o coste en cuanto a tiempo de respuesta del *software*, pues *a priori* no se necesitará mucha potencia operativa debido al número relativamente pequeño de palabras a desambiguar.

Una opción que se barajó, teniendo en cuenta todo lo mencionado hasta este punto, fue la de aplicar el algoritmo de *Lesk* que proporciona el kit de herramientas *NLTK*. Dicha función, además de requerir como parámetros la palabra ambigua y el contexto en el que se ha utilizado, precisa también de la asignación de una serie de *synsets*. Estos *synsets* son agrupaciones de palabras que expresan el mismo concepto, de forma que cada instancia sirve como clave para buscar el significado en *WordNet*. El problema de ésta y otras opciones similares es que utilizan el inglés recogido en *WordNet* o *EuroWordNet*

como lengua pívot para obtener conocimientos de la lengua que se precise (en este caso, el español). Se ha podido comprobar, con este tipo de soluciones, que la ambigüedad de la lengua natural original se puede transmitir a la siguiente, proporcionando un resultado poco acertado, pues se eleva al cuadrado la ambigüedad de la lengua de la rama siguiente. Para ejemplificar esta cuestión, se utilizarán dos frases en las que aparecerá la palabra ambigua casco, con sentidos distintos:

- El casco antiguo de Barcelona es muy bonito.
- El casco nuevo que te has comprado para la motocicleta no me gusta.

Según la Real Academia Española (Real Academia Española, s.f.), la palabra casco tiene los siguientes 21 sentidos:

1. *Cráneo (caja del encéfalo).*
2. *Fragmento que queda de un vaso o vasija al romperse, o de una bomba después de estallar.*
3. *Cáscara dura de algunos frutos.*
4. *Gajo (cada una de las divisiones interiores de algunas frutas).*
5. *Cada una de las capas gruesas de la cebolla.*
6. *Copa del sombrero.*
7. *Cobertura de metal o de otra materia, que se usa para proteger la cabeza de heridas, contusiones, etc.*
8. *Casco urbano.*
9. *Pieza de la armadura antigua que cubría y protegía la cabeza.*
10. *Armazón de la silla de montar.*
11. *Recipiente, como un tonel o una botella, cuando está vacío.*
12. *Cuerpo de la nave o avión con abstracción del aparejo y las máquinas.*
13. *Embarcación filipina de fondo plano y costados verticales, con batangas y velas de estera, que carga unas 50 t.*

14. *En las bestias caballares, uña del pie o de la mano, que se corta y alisa para sentar la herradura.*
15. *Casquete (empegado de pez para los tiñosos).*
16. *Cabeza (parte superior del cuerpo).*
17. *Cabeza (juicio, talento y capacidad).*
18. *Pieza que imita el casco de la armadura y sirve para timbrar el escudo, poniéndolo inmediatamente encima de la línea superior del jefe.*
19. *Suelo de una propiedad rústica, aparte de los edificios y plantaciones.*
20. *Cabeza de carnero o de vaca, quitados los sesos y la lengua.*
21. *Aparato compuesto por dos auriculares unidos por una tira curvada, que se ajusta a la cabeza y permite o mejora la recepción del sonido.*

Para utilizar el algoritmo de *Lesk*, se calculan los diferentes *synsets* de la palabra casco, a partir de los recursos instalables que tiene *NLTK* de *WordNet*. Como se ha explicado, la técnica utiliza el inglés como lengua pívot, por eso, los sentidos de los *synsets* aparecerán en inglés. Para contextualizar y entender mejor la cuestión, se muestran, entre paréntesis, estos sentidos también traducidos al español.

```
1 Synset('hoof.n.01') The foot of an ungulate mammal (el pie de un
    mamífero ungulado).
2 Synset('empty.n.01') A container that has been emptied (un recipiente
    que se ha vaciado).
3 Synset('hard_hat.n.02') A lightweight protective helmet (plastic or
    metal) worn by construction workers (un casco protector liviano (
    plástico o metal) que usan los trabajadores de la construcción).
4 Synset('helmet.n.02') A protective headgear made of hard material to
    resist blows (un casco protector de material duro para resistir los
     golpes).
5 Synset('helmet.n.01') Armor plate that protects the head (placa de
    armadura que protege la cabeza).
6 Synset('hull.n.06') The frame or body of ship (el armazón o cuerpo del
    barco).
```

```
Synset('potsherd.n.01') A shard of pottery (un fragmento de cerámica).
Synset('skid_lid.n.01') A crash helmet (un casco de seguridad).
Synset('toe.n.03') Forepart of a hoof (parte delantera de una pezuña).
Synset('hull.n.01') Dry outer covering of a fruit or seed or nut (
    recubrimiento externo seco de una fruta, semilla o nuez).
```

Como se puede observar, a simple vista, el número de *synsets* obtenido es claramente inferior al número de sentidos que proporciona la RAE. Los sentidos esperados, *a priori*, son los correspondientes a los números 7 y 8 de la RAE. Sin embargo, la acepción del significado "casco urbano" no existe entre los *synsets* adquiridos, por lo que se puede afirmar que la primera oración no se desambiguará correctamente. El resultado para cada oración, tras aplicar el algoritmo de *Lesk*, es el siguiente:

- Synset('toe.n.03') forepart of a hoof
- Synset('empty.n.01') a container that has been emptied

Como se esperaba, la primera oración no se desambigua adecuadamente. Sin embargo, la segunda tampoco se ha podido obtener correctamente, pues el significado que la palabra casco expresa, según el algoritmo, es "un recipiente que se ha vaciado". Se puede comprobar que, a pesar de existir dos *synsets* relacionados con la palabra inglesa "helmet" y uno con "skid_lid", que encajan más con el significado real de la palabra casco en la segunda oración, el algoritmo no consigue relacionarlos adecuadamente por utilizar otra lengua como pívot.

Otra opción contemplada fue utilizar el código del *synset* que incorporaba el corpus seleccionado como fuente para la creación de algoritmos de aprendizaje automático. Sin embargo, se terminó desestimando esta alternativa por dos motivos principales. En primer lugar, no se puede garantizar que existan, en el corpus, suficientes instancias de los sentidos de una misma palabra para elaborar con cierta garantía un modelo de este tipo. Es decir, en el corpus podría aparecer la palabra casco con 4-5 sentidos diferentes (lo que supone la pérdida de otros sentidos), pero su frecuencia de aparición podría no estar balanceada y podría no haber un conjunto significativo de muestras de cada una de ellas. Además, debido a que las palabras clave variarán dependiendo del examen, y la aplicación se desea hacer de carácter general, es muy difícil de determinar si el número de muestras para un *keyword* distinto será o no significativo en cada uso del software.

En segundo lugar, el código del *synset* incorporado en el corpus para cada palabra, de por sí, no aporta mucha información. Dichos códigos deben ser "traducidos" para identificar el sentido de una forma menos abstracta. La problemática de esta cuestión es que no se consiguió encontrar ninguna base de datos o diccionario público sobre el que poder realizar dichas transformaciones. Utilizando la *Web EuroWordNet Interface* (Adimen, s.f.) es posible traducir los códigos de forma manual (Figura 4.2), pero no se logró extraer información de esta web indirectamente a partir de técnicas de *web scraping*.

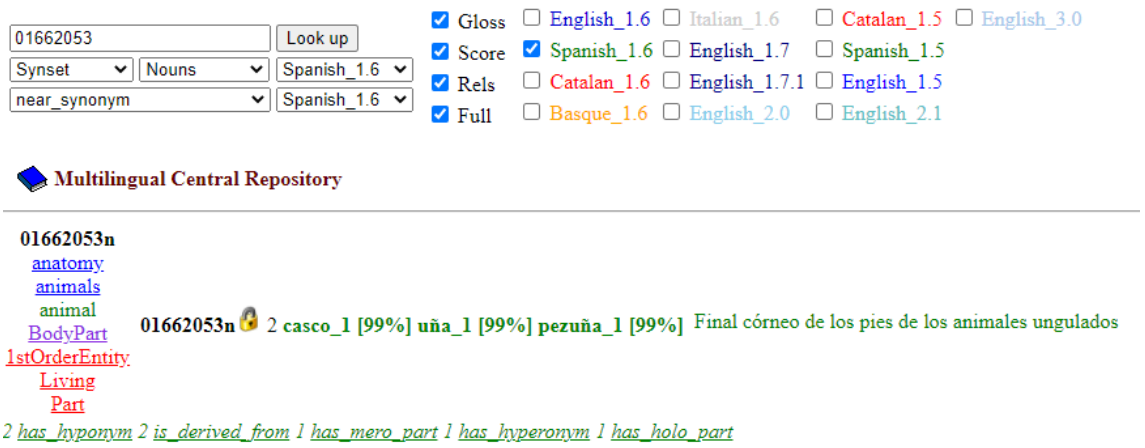


Figura 4.2: Traducción del synset utilizando la *Web EuroWordNet Interface*

Por otra parte, se consideró, también, utilizar la librería *Spacy*, que cuenta con una serie de núcleos específicos descargables para el español, permitiendo así la desambiguación en dicho idioma. Con *Spacy* se consiguieron buenos resultados; por ejemplo, partiendo de la frase "Yo quiero retirar 5.000 euros", los synsets que se calculan para cada palabra son los siguientes:

```
[]
[Synset('want.v.02'), Synset('accept.v.02')]
[Synset('withdraw.v.09')]
[]
[]
```

El *synset* de "*want*" (querer) o "*accept*" (aceptar) está profundamente relacionado con el sentido de "querer" en la frase. Lo mismo sucede con el *synset* de "*withdraw*" (retirar) y el sentido de la palabra "retirar" en la oración. Si se accediera a los lemas encontrados

por cada *synset* y se mostraran por pantalla, en contexto con la oración, el resultado sería el siguiente:

Yo (requerir|aceptar|necesitar|precisar|querer) (retirar|quitar|sacar) 5.000 euros

Como se puede observar, las palabras sugeridas expresan el mismo sentido en la oración, pues se podrían utilizar indistintamente y el significado de la oración no variaría. Los resultados con *Spacy* son buenos, el problema que presenta esta librería es que requiere especificar un dominio para proporcionar los *synsets*. Este dominio está asociado a un código, que es el que permite extraer la información de *WordNet* a partir de *Spacy*. Por ejemplo, algunos de los códigos del campo de la pedagogía que hay disponibles son los siguientes:

```
00584282-v  military pedagogy
00584395-v  military school university
00584526-v  animals pedagogy
00584634-v  pedagogy
00584743-v  school university
00585097-v  school university
00585271-v  pedagogy
00585495-v  pedagogy
00585683-v  psychological_features
```

Existe una página web, *WordNet Domains*, donde está disponible el contenido de cada dominio y su código asociado. El problema es que la información no es de acceso público, se requiere solicitar permisos para descargar los archivos y utilizarlos, necesitando un perfil específico para que te los otorguen. Como se pretende construir una aplicación software que sea accesible a cualquier individuo, en aras de facilitar la posible replicación de los resultados por la comunidad científica, se decidió desestimar, también, esta aproximación.

En el proceso de búsqueda de opciones para desarrollar la tarea de WSD en el programa, se advirtió que, muchas veces, es el orden del discurso el que determina el significado de las unidades y las relaciones entre ellas (Cárdenas, 2010). Es decir, dependiendo de la construcción de la oración, el sentido de una unidad puede ser uno u otro, pues existe cierta relación entre sintaxis y semántica. De esta forma, para algunas opciones de WSD

es necesario determinar la categoría gramatical de la palabra, que es, en definitiva, información proporcionada de la propia unidad y del contexto. Por ejemplo, para usar la *Web EuroWordNet Interface* (Adimen, s.f.) es necesario introducir no sólo el código del *synset*, sino también la categoría gramatical que representa, bien sea nombre, adjetivo, verbo o adverbio.

Se optó, de esta manera, por sustituir las técnicas normalmente concebidas para WSD por técnicas basadas en el análisis morfosintáctico de cada oración, ideadas para filtrar las acepciones de las palabras en cuestión. Esta idea es factible en este proyecto si se considera como válida la hipótesis postulada al principio de esta sección (basada en la inherente especificidad de las palabras clave), pues la técnica ayudaría a cribar ciertos sentidos de las palabras. Siguiendo dicha hipótesis, se considerarán las demás acepciones que no se consigan filtrar como válidas si aparecieran en la respuesta, pues se presupone que el alumno habrá escogido aquel sentido que más se adapte al dominio que se está evaluando. De esta manera, continuando con el ejemplo de la palabra "casco", se conseguirían descartar aquellos sentidos relacionados con la palabra "cascar" correspondientes a su primera persona del presente del indicativo ("yo casco"), indicados a continuación (Real Academia Española, s.f.):

1. *Quebrantar o hender algo quebradizo.*
2. *Dar a alguien golpes con la mano u otra cosa.*
3. *Estropear, dañar algo.*
4. *Quebrantar la salud de alguien.*
5. *Morir.*
6. *Charlar.*

Este recurso, además de desambiguar algunos significados, se antoja indispensable en otras partes de la rúbrica del proyecto general al que pertenece este trabajo (Sección 1.2), tales como la identificación de la cohesión léxica y gramatical en la respuesta del alumno. De esta manera, se preferirá desarrollar funciones que podrán ser reutilizables en otros aspectos de la mencionada rúbrica frente a otras soluciones de WSD cuyo resultado tampoco garantiza la total resolución del problema en este proyecto. Por otra parte, esta solución

adoptada no es completa ya que no resuelve, como tal, todo el problema semántico de WSD. Será, por tanto, una cuestión a profundizar en trabajos futuros, tal y como se refleja en el Capítulo 6.

Integración de la solución de WSD escogida

Recapitulando, la técnica que se desarrollará, concretamente, será un modelo basado en HMM que se decodificará utilizando el algoritmo de *Viterbi*. Este modelo se nutrirá de la información de los múltiples archivos que componen el corpus formado. A pesar de contar con etiquetas ya anotadas, se decidió transformar el sistema de anotación a uno más simple, considerando únicamente la primera letra del código *EAGLES*, que es la referente a la categoría gramatical de la palabra. De esta forma, se filtra la información restante del *token*, que no aporta conocimiento relevante para este proyecto. Al disminuir el número de etiquetas a trabajar, se reducen las matrices resultantes de transición y emisión del HMM. Se observó que dicho proceso de reducción de dimensionalidad, tenía un resultado positivo en la eficiencia final del etiquetador y, además, suponía una reducción en el tiempo de cómputo de las matrices. Para facilitar la identificación de las nuevas etiquetas en las tablas de probabilidad, se creó un diccionario interno en la clase *HMMBigram*, de forma que éstas se transformaran al nombre de la categoría gramatical en cuestión.

```
1 {"A":"Adjective", "D":"Determiner", "N":"Noun", "V":"Verb", "P":"
    Pronoun", "R":"Adverb", "C":"Conjunction", "S":"Adposition", "W":"
    Date", "Z":"Number", "I":"Interjection", "F":"Punctuation"}
```

La clase *HMMBigram*, cuenta con una serie de métodos para el cálculo y obtención de las probabilidades (Figura 4.3), entre los que destacan:

- ***CorpusProcessing***: Éste método se encarga de extraer y limpiar los recursos del corpus. Tras un primer análisis, se advirtió que los *tokens* y los lemas de algunas entradas estaban formadas por expresiones o abreviaturas seguidas de puntos o guiones, tal y como se muestra a continuación:

```
1 en_pie_de_guerra en_pie_de_guerra RG 0
2 D.M. d.m. NP00000 0
```

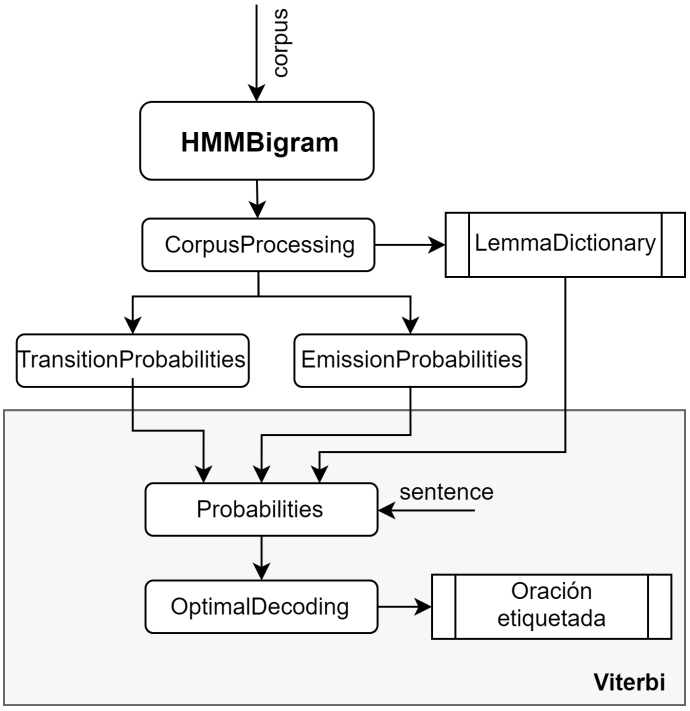


Figura 4.3: Diagrama de flujo del funcionamiento de las clases HMMBigram y Viterbi

Este tipo de entradas no son interesantes para el proyecto. El etiquetador que se pretende construir debe catalogar unidades, no términos o palabras compuestas. Por lo tanto, se aprovechará la lectura de cada línea del corpus, durante su importación al programa, para analizar si el *token* en cuestión contiene alguno de estos símbolos y, por tanto, si debe ser ignorado o procesado.

A partir de este procedimiento, además, se almacenarán los diferentes estados y *tokens* existentes junto al número de apariciones de cada uno. Esta información es vital para poder calcular, *a posteriori*, las tablas de probabilidades de transición y emisión. Para reducir el tamaño de dichas tablas, se transformarán las letras de cada *token* a sus respectivas minúsculas, de forma que "Hola" y "hola" se considerarán de forma análoga. Sin embargo, cabe destacar, también, que el análisis morfosintáctico se efectuará sobre la frase sin lematizar, pues se busca que sirva como herramienta para ayudar a lematizar una misma palabra a su forma canónica correcta. Esto provoca que el número de claves del diccionario de lemas y de *tokens* de la matriz de emisión sea muy alto.

- ***LemmaDictionary***: Este método devuelve un diccionario compuesto por el con-

junto de *tokens* diferentes extraídos del corpus y el lema asociado a la categoría gramatical de éstos. Se integrará la creación de este instrumento durante el tratamiento de los datos en el método *CorpusProcessing*, aunque su utilidad estará mayormente enfocada a tareas de lematización (Subsección 4.1.1).

- ***TransitionProbabilities***: Devuelve y/o calcula las probabilidades de transición entre las diferentes etiquetas.
- ***EmissionProbabilities***: Devuelve y/o calcula las probabilidades de emisión de los diferentes *tokens* con respecto a cada etiqueta.

Los dos últimos métodos de la clase manejan una gran cantidad de valores en memoria para efectuar sus operaciones. Como consecuencia del vasto conjunto de datos que componen el corpus, el consumo de memoria se dispara durante su ejecución, provocando la interrupción del *kernel* del archivo *Jupyter Notebook* tras exceder el límite de memoria RAM permitida en *Google Colab*. Con el fin de solventar dicha cuestión y reducir, también, el tiempo de cálculo de dichas tablas probabilísticas, se llevaron a cabo dos aproximaciones:

1. La primera de ellas fue eliminar información no muy significativa del corpus. Se filtraron, primero, todas las palabras pertenecientes a las categorías gramaticales "W" y "Z"; es decir, las correspondientes a *tokens* de fechas y números. Se trata de información no muy relevante porque es de carácter numérico-simbólico y jamás podrá servir para la desambiguación de los sentidos. Su función en la oración, además, suele ser complementaria. De esta forma, etiquetar una de estas palabras erróneamente no debería influir mucho en la sintaxis de los demás *tokens* de la oración, más allá del sintagma al que pertenezca.

   Se sabe, además, que si se tradujera la información numérica en categórica; es decir, si se escribiera la palabra "cinco" en vez de "5", el *token* sería catalogado como sustantivo o adjetivo (Real Academia Española, s.f.), no como número o fecha. Por tanto, y sabiendo que en esta aplicación no es necesario efectuar un análisis sintáctico con excesivo detalle, se consideró ésta como una buena opción para reducir la cantidad de información.

   Por otra parte, se eliminaron, también, los nombres propios. Esta medida tomada redujo con creces la dimensionalidad de la tabla de probabilidad de emisión, dada la

variedad de nombres propios diferentes existentes en el corpus. Como esta técnica, *a priori*, sí conlleva la pérdida de información significativa, se creó un nuevo *token* con el nombre de *tokenseliminados* para recoger y almacenar la mencionada información descartada, como si de un *buffer* se tratara. Se consigue, de esta manera, reducir el consumo de datos en memoria sin la consecuente pérdida de detalles significativos del corpus ni de sus relaciones.

2. La segunda medida tomada consistió en segmentar la información manejada durante el procesamiento. En lugar de emplear todo el corpus, se procedió a trabajar con cada fichero por separado. Para poder operar con esta técnica sin perder información, se creó una nueva clase: *HMM_Probabilities*, que se inicializa con las probabilidades calculadas de cada fichero. El listado de instancias de esta clase almacenará las probabilidades de transición y emisión calculadas en cada caso, manejando un menor número de datos para realizar dichas operaciones.

Con este procedimiento, se consiguió liberar suficiente espacio en memoria RAM como para evitar la desconexión de la plataforma *Google Colab*. Además, una vez terminados los cálculos probabilísticos (que son los que mayor consumo de RAM provocan), la información se almacena en memoria de "disco", dónde se dispone de mayor capacidad de almacenamiento. Terminada la operación, la RAM se vacía, permitiendo realizar nuevos cálculos sin problema alguno.

Por otra parte, cada tabla probabilística calculada se obtiene en un formato similar al mostrado en la Figura 4.4 y en la Figura 4.5. Las tablas obtenidas (que se adjuntarán en el repositorio de *GitHub* indicado en la Sección III de los Anexos), se guardarán como *CSV* con una particularidad: la tabla de probabilidades de emisión se almacenará con las columnas transpuestas, de lo contrario, se excede el límite de columnas que se pueden crear y se genera un error. Como el número máximo de filas es superior al de columnas, se consigue evitar la aparición de dicho error con la operación transpuesta.

Tras finalizar el procedimiento descrito, se dispone de 64 tablas de probabilidad (dos por cada fichero). Para gestionar e integrar las probabilidades calculadas en una sola tabla, se efectuó una media entre los valores calculados para cada archivo, asumiendo igual representatividad y calidad en la información de cada uno de ellos. Lógicamente, la tabla de probabilidades de emisión de cada archivo de *Wikicorpus* es distinta, pues también

	q0	Determiner	Noun	Verb	Adposition	Adjective	Punctuation	Conjunction	Pronoun	Adverb	Number	Date	Interjection	qF
q0	0.000000	0.249160	0.318593	0.109062	0.171052	0.004547	0.010995	0.028374	0.035704	0.055023	0.015718	0.001559	0.000213	0.000000
Determiner	0.000000	0.011758	0.808015	0.013182	0.009249	0.082384	0.004906	0.001100	0.025491	0.006354	0.017866	0.019659	0.000036	0.000000
Noun	0.000000	0.010561	0.048491	0.088396	0.245157	0.111023	0.363798	0.067022	0.033353	0.018431	0.012697	0.000488	0.000582	0.000000
Verb	0.000000	0.227699	0.086812	0.140377	0.295037	0.030314	0.064043	0.052566	0.040454	0.049960	0.012497	0.000121	0.000121	0.000000
Adposition	0.000000	0.512150	0.313212	0.050803	0.002159	0.019188	0.005896	0.006043	0.020845	0.007539	0.056865	0.005251	0.000048	0.000000
Adjective	0.000000	0.011215	0.257580	0.065085	0.221025	0.021900	0.294795	0.084907	0.029432	0.010011	0.003562	0.000459	0.000029	0.000000
Punctuation	0.000000	0.057034	0.251712	0.069913	0.071242	0.012565	0.102883	0.054243	0.031118	0.019076	0.055198	0.003539	0.000751	0.270725
Conjunction	0.000000	0.216529	0.291573	0.151233	0.105575	0.052964	0.025285	0.011827	0.062673	0.057161	0.023540	0.001536	0.000105	0.000000
Pronoun	0.000000	0.036758	0.025993	0.627845	0.090202	0.009408	0.047187	0.013894	0.098533	0.048465	0.001615	0.000045	0.000056	0.000000
Adverb	0.000000	0.073572	0.053709	0.277721	0.149489	0.144527	0.137833	0.032538	0.064719	0.054338	0.011197	0.000051	0.000306	0.000000
Number	0.000000	0.010498	0.255521	0.038235	0.093187	0.010946	0.498352	0.045974	0.013675	0.008336	0.024620	0.000611	0.000045	0.000000
Date	0.000000	0.033480	0.047508	0.078088	0.147760	0.003647	0.557748	0.058449	0.040120	0.004676	0.022258	0.006266	0.000000	0.000000
Interjection	0.000000	0.007380	0.474785	0.035670	0.019680	0.004920	0.382534	0.012300	0.025830	0.017220	0.011070	0.001230	0.007380	0.000000
qF	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Figura 4.4: Probabilidad de transición de uno de los ficheros

	el	ser	uno	institución	de	educación	superior	inspiración	cristiano	,	...
Determiner	0.314399	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Noun	0.001330	0.000073	0.000000	0.000247	0.000099	0.000426	0.000006	0.000065	0.000030	0.000000	...
Verb	0.000000	0.010618	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Adposition	0.000000	0.000000	0.000000	0.000000	0.426120	0.000000	0.000000	0.000000	0.000000	0.000000	...
Adjective	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.002114	0.000000	0.000344	0.000000	...
Punctuation	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.343961	...
Conjunction	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Pronoun	0.000000	0.000000	0.020610	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Adverb	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Number	0.000000	0.000000	0.000268	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Date	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
Interjection	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...

Figura 4.5: Probabilidad de emisión de uno de los ficheros

lo es su contenido y las palabras que incorpora. La media que se calculará para obtener la tabla final de emisión será, por tanto, relativa al número de apariciones contadas de cada *token*. Para ello, se generó un diccionario auxiliar, denominado "*divider*", que servirá como almacén o *buffer* del conteo de la aparición de cada palabra. Paralelamente, se sumarán los valores de las 32 tablas obtenidas de emisión. Si un valor ya existe en la tabla, la frecuencia de aparición de "*divider*" se incrementa, sino, se iguala a la unidad y se añade la columna a la tabla. Una vez terminado el sumatorio de los ficheros, la herramienta "*divider*" permitirá ajustar, correctamente, la media para cada palabra.

Para decodificar la información final extraída de ambas tablas de probabilidad, como se ha adelantado, se utilizó el algoritmo de *Viterbi*, introducido en la clase denominada por el

mismo nombre. En esta clase, se integran una serie de métodos privados a partir de los que se calculan las diferentes operaciones requeridas por el algoritmo (Figura 4.3). Además, la clase incorpora dos métodos públicos: *ViterbiProbability*, que devuelve la matriz de probabilidad de *Viterbi* obtenida para una determinada oración, y *SyntacticAnalysis*, que devuelve la secuencia de *tokens* de la oración con su correspondiente categoría gramatical.

Si en una oración aparece alguna palabra nueva; es decir, una no incluida en las claves del diccionario de lemas generado, el programa lo considerará como "*tokenseliminados*", por lo que no se producirá ningún error de ejecución. La columna de la tabla de probabilidad de emisión de esta clase, presenta valores de probabilidad de diversas categorías gramaticales, que reflejan el conjunto de etiquetas descartadas en el proceso de reducción de dimensionalidad previamente explicado. Como se dispone de un diccionario completo de palabras de uso cotidiano, se toma como asunción que la aparición de una palabra desconocida será o bien un nombre propio (la opción más probable) o bien una fecha o un número. Por este motivo, se añadió una penalización a aquellas categorías gramaticales de este tipo de palabras que no fueran ni sustantivo ni adjetivo, ajustando el camino que se obtendría tras decodificar el HMM, que sería incorrecto de lo contrario.

Esta asunción podría provocar que se efectuara un incorrecto análisis morfosintáctico si la palabra nueva fuera, por ejemplo, un verbo. Una opción valorada fue eliminar esta penalización y trabajar con los valores de probabilidad de "*tokenseliminados*". Sin embargo, la situación contemplada se presupone como alto improbable dada la extensión del diccionario, y las pruebas realizadas fueron más positivas utilizando la asunción como criterio de diseño del algoritmo, que sin utilizarla.

Por otra parte, se añadirá un índice a cada *token* etiquetado en caso de aparición de dos o más palabras iguales, a fin de considerar su información de forma independiente. De lo contrario, como la clave ya existe en el diccionario en el que se almacenan las palabras etiquetadas de la oración, únicamente se mostraría una de sus ocurrencias pues se estaría sobrescribiendo el contenido. Asimismo, el índice permite también identificar, visualmente y con mayor claridad, el orden de aparición de los mismos en la oración.

4.2. Diccionario de palabras clave

En esta sección, se detalla el procedimiento realizado para expandir el conjunto de palabras clave proporcionadas por el profesor en aras de conseguir crear y ampliar las herramientas de control terminológico con las que efectuar la comparación de palabras de la oración.

4.2.1. Creación del instrumento de control terminológico

Evaluar a un alumno requiere la definición de los objetivos que éste debe de alcanzar para establecer criterios de corrección. En este trabajo, se analiza el impacto y adecuación de este método como criterio de la rúbrica del proyecto mencionado que engloba éste y otros trabajos. En particular, se evaluará que el estudiante incluya todo el conjunto de palabras marcadas por el profesor, indicadas en la sección *Keywords* del archivo que éste proporciona (Anexos, Sección I). De esta forma, se valorará si el trabajo puede ser suficiente como herramienta de evaluación independiente y semiautomática del estudiante o si se puede extraer información interesante que combinar con las otras técnicas desarrolladas para los demás puntos de la rúbrica.

A partir de este momento, se particularizará el concepto de evaluación a este método, haciendo referencia a la salida de esta técnica, no a la corrección de texto de forma automática que se daría integrando las otras partes. Así pues, como base de evaluación para la identificación de palabras clave, se deberá tener en cuenta la variabilidad de formas en las que normalmente se expresa un mismo concepto. Para almacenar todos esos datos, será necesaria la transformación de un diccionario convencional en un tesauro, de forma que la información quede perfectamente indexada y organizada en niveles. Su estructura jerárquica, servirá de instrumento para poder establecer, de forma más clara, las relaciones de las distintas palabras en el programa. Es importante mencionar que no se integrará toda la información en un único instrumento de control terminológico, sino que se utilizarán dos tesauros, diferenciando así la información que proporcionan los sinónimos de los antónimos. Se pretende, así, facilitar su manejo en programa para poder efectuar ciertas comprobaciones con mayor comodidad.

Como instrumento para extraer, automáticamente, la información necesaria para complementar a cada palabra clave, se utilizó *WordReference* (WordReference, s.f.). *WordRe-*

ference es un conocido diccionario en línea que permite, además de consultar la definición de un concepto, proporcionar los sinónimos y antónimos de una misma palabra, teniendo en cuenta sus múltiples acepciones y categorías gramaticales. Se consideró interesante esta opción porque se trata de un diccionario web de uso gratuito y de relativo prestigio, hecho que garantiza un mínimo de calidad en la información que se extrae. Además, se puede acceder a su información indirectamente utilizando técnicas de *web scraping*, lo que permite que esta tarea se desempeñe de manera automática y no supervisada.

Tras efectuar las operaciones de *web scraping* para cada uno de los *keywords* (previamente lematizados) del listado del profesor, los términos obtenidos fueron lematizados utilizando el procedimiento anteriormente explicado (Subsección 4.1.1). Por cada uno de ellos, se consiguió una estructura en formato XML similar a la mostrada a continuación, correspondiente a la palabra clave "arbitrar".

```
\text{[<div class="trans clickable" onclick='redirectWR(event,"
    ESsinonimos")'><h3>arbitrar</h3><ul><li>interceder, mediar,
    intermediar, intervenir, interponer, resolver, fallar,
    sentenciar, dictaminar, decidir, establecer, juzgar, actuar</
    li><ul><li><span class="r">Antónimos: inhibirse, abstenerse</span
    ></li></ul><li>agenciarse, servirse, ingeniarse, valerse,
    lograr, conseguir</li><br/></ul></div>]}
```

Como se puede observar, la información deberá ser extraída y filtrada de los distintos nodos del documento, factor que se tendrá en consideración para el diseño de la función en el programa. Por otra parte, si la palabra clave es compuesta, como es el caso de "balanceo de carga", no se pueden extraer, en *WordReference*, sinónimos/antónimos del sentido general que expresa. Sin embargo, la propia página web sugiere la posibilidad de buscar el significado de cada unidad que compone la palabra compuesta por separado (Figura 4.6), que, precisamente, es el procedimiento que se siguió en esos casos. Para comprobar si la palabra clave es o no compuesta, se efectúa primero la *tokenización* del *keyword* para, posteriormente, calcular el tamaño de la lista de palabras resultante a esa operación. Si dicho tamaño supera la unidad, se trata de una palabra compuesta y, por tanto, se realiza *web scraping* para cada uno de los *tokens* obtenidos.

Existe un grupo de palabras de uso frecuente y particular a cada idioma cuyo significado es poco representativo. Este grupo recibe el nombre de palabras vacías (*stopwords*,

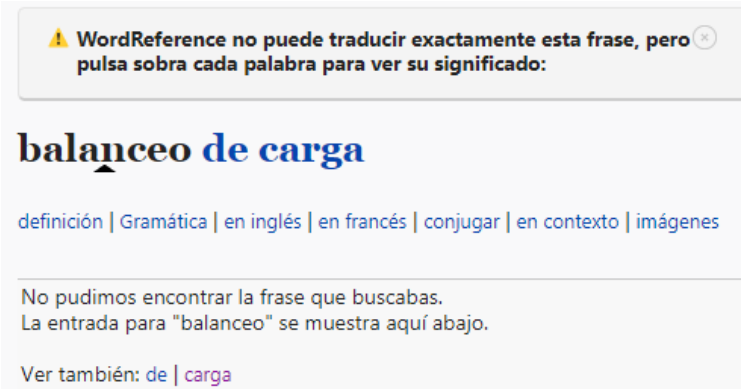


Figura 4.6: Resultado de la búsqueda de balanceo de carga en *WordReference*

en inglés) y se compone fundamentalmente de pronombres, artículos, preposiciones, conjunciones y demás palabras que cualquier motor de búsqueda suele ignorar. Como no se dispone de criterio suficiente en lingüística para poder discriminar siempre qué vocablo es *stopword* y qué no, se empleó *NLTK* para obtener un listado de palabras vacías del español. De esta manera, dicho conjunto es mucho más fiable, pues *NLTK* es una plataforma muy utilizada por desarrolladores y por la comunidad científica para el uso de PLN. Si alguno de los *tokens* extraídos de la palabra compuesta se encuentra en la lista de *stopwords*, se omitirá la búsqueda de sinónimos y antónimos de dicha unidad. Por ejemplo, se ignorará la palabra "*de*" de la clave "balanceo de carga".

Si se puede dividir una palabra compuesta en unidades, también se pueden buscar sinónimos de esas palabras resultantes por separado. Por ejemplo, "fluctuación" es un sinónimo de "balanceo" y "cargamento" es sinónimo de "carga", por lo que se podría obtener la palabra compuesta sinónima "fluctuación de cargamento". Es posible que, modificando todas las unidades de la palabra compuesta por sus sinónimos, el significado se pueda ver ligeramente o completamente alterado. Sin embargo, se busca conseguir mayor flexibilidad a la hora de identificar las palabras clave. De esta manera, la palabra compuesta "fluctuación de carga", que *a priori* tiene un significado muy similar, se podrá reconocer y considerar como correcta.

El problema que conlleva almacenar los sinónimos de las expresiones es que la unidad de una palabra compuesta, por sí misma, no es una palabra clave. De esta forma, no sería correcto identificar la palabra "carga" si no estuviera precedida por la preposición "de" y la unidad "balanceo" o alguno de sus sinónimos. Para solventar esta cuestión y

optimizar el funcionamiento del software, se decidió añadir un nivel más en los dos tesauros mencionados. Este nivel, proporciona información complementaria de la palabra clave en cuestión sobre los siguientes temas:

- ***Compound***: Representa con lógica booleana si el término es una palabra clave (0) o si, por contra, es parte de una palabra compuesta (1).
- ***Dictionary***: Permite acceder al nivel donde se almacena el tesauro obtenido de *WordReference*. La estructura de este tesauro se compone, primero, por los lemas de las acepciones de la palabra clave, segundo, por las categoría gramaticales de cada una de las acepciones y, tercero, por el listado de sinónimos/antónimos de cada acepción.
- ***GCategory***: Almacena la categoría gramatical correspondiente a la palabra clave proporcionada por el profesor. Esta información es importante como referencia para la función de WSD desarrollada.

Para ejemplificar esta cuestión, se muestra la información obtenida de cada tesauro a partir de la palabra clave "respaldo":

Tesauro de Sinónimos:

```
'respaldo': {
  'Compound': 0,
  'Dictionary': {'respaldar': {'VERB': ['ayudar',
                                        'amparar',
                                        'apoyar',
                                        'proteger',
                                        'avalar',
                                        'favorecer',
                                        'alentar',
                                        'defender',
                                        'patrocinar',
                                        'respaldo']},
                 'respaldo':   {'NOUN': ['espaldar',
                                        'respaldar',
                                        'apoyo',
                                        'protección',
```

```
                                        'ayuda',
                                        'amparo',
                                        'sostén',
                                        'auxilio',
                                        'patrocinio']}
                },
  'GCategory': 'NOUN'},
```

Tesauro de Antónimos:

```
'respaldo': {
  'Compound': 0,
  'Dictionary': {'respaldar': {'VERB': ['desamparar', 'desaprobar']},
                 'respaldo': {'NOUN': ['desamparo', 'desprotección']}}}
                    ,
```

Como se puede observar, las estructuras de ambos tesauros son diferentes porque el de antónimos no incorpora la categoría *GCategory*. Esto se debe a que únicamente se realizará la tarea de WSD para el tesauro de sinónimos. Esta decisión se fundamenta, por una parte, en la hipótesis basada en la tendencia existente a utilizar sinónimos y no antónimos para referenciar una palabra con otro término; y por la otra, en la falta de antónimos encontrados en una gran proporción del conjunto de palabras clave. Se considerará, por tanto, la aparición de un antónimo negado como evidencia suficiente para demostrar que el alumno pretendía hacer referencia a la palabra clave en cuestión, independientemente de su categoría gramatical.

4.2.2. Integración en el programa

En este punto, se detalla el procedimiento realizado para integrar en el programa la estrategia definida en la anterior subsección, de forma que queden relacionados los métodos del programa con la tarea o función que desempeñan.

El procesamiento del fichero proporcionado por el profesor (Anexos, Sección I) y la creación de los tesauros se realizará a partir de la clase *NLP_Questions* del programa desarrollado. Esta clase, se inicializará con la ruta a dicho fichero y con dos diccionarios. Los diccionarios pueden estar vacíos, de forma que el contenido que se obtenga sea particular

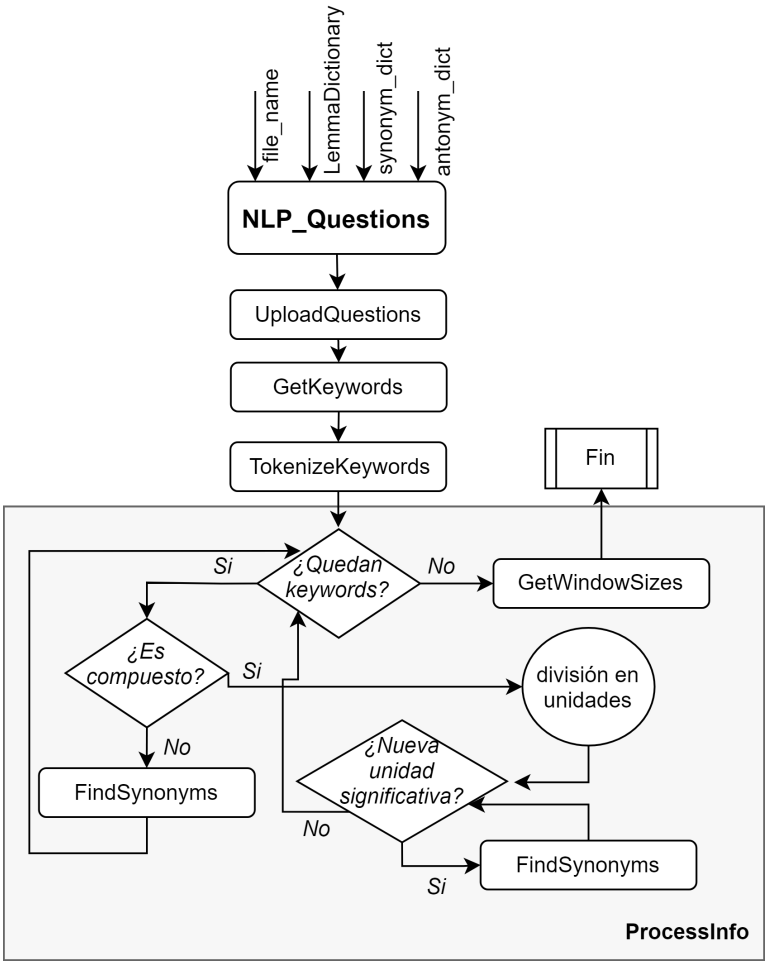


Figura 4.7: Diagrama de flujo del funcionamiento de la clase *NLP_Questions*

a un examen en concreto, o pueden contener información, ampliando así el bagaje de conocimiento previo almacenado. Para esta aplicación en concreto, no será recomendable la segunda opción, pues cuanto menos filtrada se encuentre la información en los tesauros, mayor será el coste computacional y, consecuentemente, mayor será el tiempo de ejecución del software. El procesamiento de los datos y las pertinentes operaciones se obtendrán al instanciar la clase, a partir de una serie de métodos privados que integran desde la lectura del archivo con formato *JSON* hasta la efectuación de operaciones de *web scraping* y construcción de los tesauros (Figura 4.7). La información calculada se podrá obtener a partir de los siguientes métodos públicos de la clase:

- ***Keywords***: Devuelve un diccionario cuyas claves corresponden a las palabras originales del listado del profesor y donde el contenido de cada clave es la propia palabra lematizada. Este método es puramente informativo; es decir, no se empleará para el procesamiento de las respuestas de los alumnos (para eso se utilizará el méto-

do *LemmatizedKeywords*). Para lematizar las palabras, se creará el método privado *Lemmatization* teniendo en cuenta que, en este caso, las palabras se encuentran fuera de contexto y, por tanto, no tendría sentido aplicar técnicas de WSD como el análisis morfosintáctico. Por este motivo, únicamente se utilizará el diccionario de lemas si la palabra clave se encuentra recogida en el mismo con una única acepción. De lo contrario, se recurrirá al *pipeline* en español de *Stanza*.

- ***LemmatizedKeywords***: Devuelve la lista de palabras clave lematizadas.
- ***Synonyms***: Devuelve el tesauro de sinónimos en el formato especificado en la Subsección 4.2.1.
- ***Antonyms***: Devuelve el tesauro de antónimos en el formato especificado en la Subsección 4.2.1.
- ***Windows***: Como se ha explicado, no todas las palabras clave son unidades, por lo que es necesario establecer un tamaño de ventana adecuado para poder identificar también, en cada caso, las palabras compuestas. Para obtener este tamaño, se calculan las diferentes longitudes de palabras existentes en el *set* de palabras clave, en el *set* de sinónimos y en el *set* de antónimos. De esta manera, el método *Windows* devuelve un *array* de vectores donde el contenido de la posición del *array* corresponde al conjunto de tamaños de ventana que se debe analizar en el determinado *set*. Por ejemplo, el vector obtenido (palabras clave - sinónimos - antónimos) en este proyecto es:

$$[\ [1,3],[1,3],[1]\ ]$$

- ***SaveDictionaries***: Permite guardar la información de los métodos previamente explicados, a fin de evitar su cómputo en cada uso del software.

4.3. Estrategia de identificación de palabras

En esta sección, se explicará el procedimiento diseñado para la identificación de palabras en la oración, relacionando cada parte de este proceso con la función creada en el programa para conseguir dicho propósito. Para procesar y comparar el contenido de la respuesta del alumno con el que espera el profesor, se creó la clase *NLP_Answers*. Dicha clase, requiere los siguientes objetos para poderse inicializar:

(a) Objetos extraídos de la clase *HMMBigram*: Como la clase *NLP_Answers* incorpora la clase *Viterbi* para el análisis morfosintáctico de las palabras, será necesario que reciba el diccionario de lemas y las tablas de probabilidad de observación y transición, que se representarán en la Figura 4.8 con un color gris claro.

(b) Objetos extraídos/compartidos con la clase *NLP_Questions*: La ruta al fichero con las respuestas de los alumnos, el tesauro de sinónimos y antónimos calculado, los *keywords* lematizados, y el vector de tamaños de ventana, que se representarán en la Figura 4.8 con un color gris oscuro.

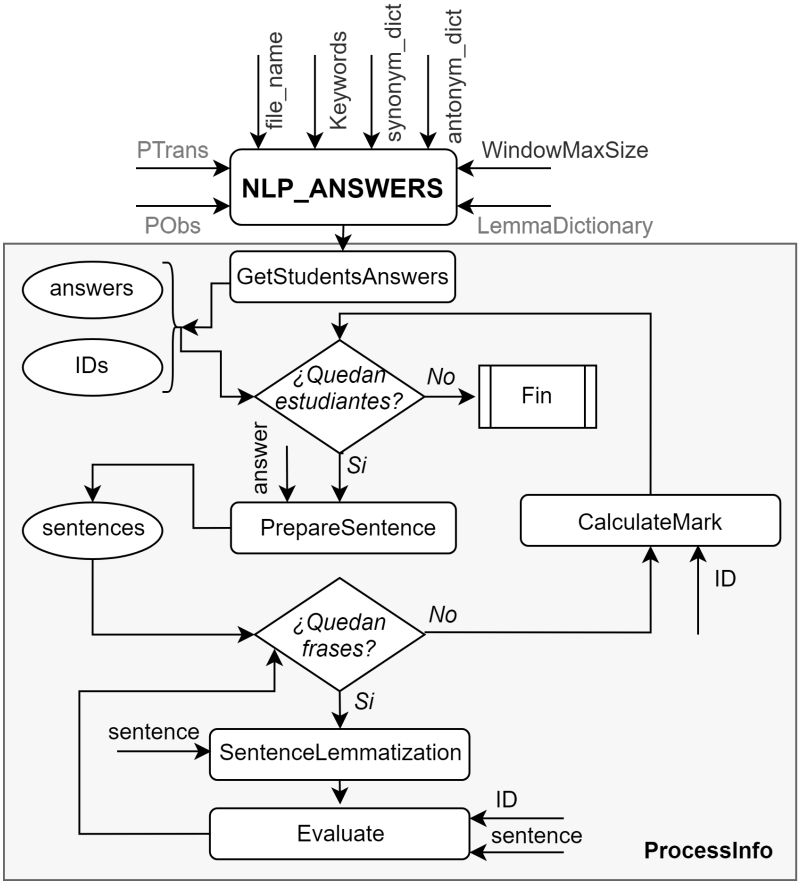


Figura 4.8: Diagrama de flujo del funcionamiento de la clase *NLP_Answers*

El procesamiento de todo el conjunto de información extraíble de esos objetos, se efectuará a partir de una serie de métodos privados, cuyas tareas comprenden desde el pre-procesamiento de los textos hasta la identificación de palabras clave o la evaluación de la respuesta en combinación con un motor de generación de *feedback* (Figura 4.8). De todo el conjunto de dichos métodos, destacan:

- ***GetStudentsAnswers***: Este método extrae la información del fichero especificado en la ruta y devuelve el par formado por la respuesta e ID anonimizado de cada estudiante.

- ***PrepareSentence***: Este método servirá como primera función para pre-procesar la información de las respuestas. Se encarga de segmentar la oración en frases más simples para favorecer el correcto análisis morfosintáctico. Para ello, se realiza primero una segmentación por puntos, ya que el punto indica el final de una sentencia. Para cada frase resultante, se analiza su tamaño; es decir, el número de palabras que la componen. Si el tamaño supera la cifra de 10 unidades, se considerará como una frase larga que se debe procesar. Este límite máximo de palabras está diseñado, nuevamente, en aras de favorecer el análisis morfosintáctico, pero se podría ajustar según el criterio del programador. Las frases largas que se obtengan, debido al uso de subordinadas, adversativas, enumeraciones u otro tipo de recursos léxicos, se dividirán en frases más simples a partir de una segmentación por coma.

  Por otra parte, se eliminarán ciertos caracteres que dificultan el procesamiento de algunas palabras. El símbolo \n (se utiliza como salto de línea), el punto, la coma, las comillas y el punto y coma, serán suprimidos ya que aparecen concatenados a una palabra, por lo que quedarán adheridos a ella en el momento de la *tokenización*. Aunque un humano, a simple vista, sí que pueda identificar dicha palabra, una máquina detecta que existen determinados caracteres en ese *string* que no posee la palabra clave a buscar y, por tanto, no logra identificarla. Es decir, la palabra no coincide con la clave y es considerada como "*tokenseliminados*". Por este motivo, será interesante filtrar dichas situaciones para que el análisis sea lo más correcto posible.

- ***SentenceLemmatization***: Este método se encargará de calcular la categoría gramatical y lematizar cada palabra de una frase. Se utilizará como función principal de pre-procesamiento, para reducir el contenido de información de la oración, por lo que se realizará una única vez por frase distinta en la respuesta del alumno. El método, dispone de un parámetro denominado *complete_analysis*, que permite ajustar la precisión del análisis. De esta forma, si el parámetro está a 0 (opción por defecto y la utilizada para el proyecto), se utilizará el etiquetador morfosintáctico en aras de desambiguar, únicamente, las palabras que cumplan los siguientes requisitos:

(a) La unidad debe figurar en la lista de palabras clave por sí misma.

(b) La unidad debe tener más de una posible categoría gramatical.

En caso contrario, se utilizará el *pipeline* en español de *Stanza* como método alternativo para lematizar y obtener la categoría gramatical de las palabras, tal y como se avanzó en la Subsección 4.1.2. De esta forma, se limitará el uso del análisis morfosintáctico exclusivamente a dichas situaciones, pues se trata de una función con relativo coste computacional que podría enlentecer el funcionamiento del software si se utilizara arbitrariamente. Contrariamente, la solución de *Stanza* es más rápida que la del etiquetador pero no es robusta frente a palabras polisémicas; por este motivo, ambas técnicas se complementan a la perfección. Cabe destacar que la nomenclatura de la categoría gramatical que proporciona *Stanza* difiere con la utilizada en el etiquetador. En este proyecto, esto no influirá en demasía, pues únicamente interesan las categorías gramaticales de las palabras que sean claves. Sin embargo, en caso de requerir un análisis más exhaustivo, será necesaria la integración de un diccionario similar al expuesto en la Sección 4.1.2, que permita traducir dicha información al mismo formato que el etiquetador.

Es posible que, con esta aproximación, algunas palabras que no son clave y sí son ambiguas puedan ser etiquetadas erróneamente, lo que podría suponer un problema si condicionara el sentido de las siguientes palabras. Sin embargo, el etiquetador morfosintáctico realizado es bastante robusto a este tipo de situaciones ya que incorpora información suficiente de la lengua y sus relaciones sintácticas como para reducir el problema a la unidad o, como mucho, a un bigrama o trigrama. Compensará, por tanto, perder un poco de precisión en el análisis a cambio de reducir el coste de procesamiento de cada frase. Además, si se deseara desambiguar todas las palabras polisémicas de la oración, independientemente de si son palabras clave o no, se podría conseguir ajustando el parámetro *complete_analysis* a 1.

Para comprobar si una palabra con varios sentidos es o no clave, se iterará por el lema de cada uno de ellos (almacenados en *LemmaDictionary*) en busca de una posible coincidencia, pues el listado de *keywords* se encuentra ya lematizado. Una vez calculada la categoría gramatical mediante el etiquetador, se accederá, de nuevo,

a *LemmaDictionary* para extraer el lema de la clave de dicha categoría gramatical. Esta información (lema-categoría gramatical), será almacenada en un *array* junto con los valores de los demás *tokens* de la oración. De esta manera, se dispondrá de la frase *tokenizada*, lematizada y desambiguada para poder realizar correctamente el análisis posterior.

- ***calculateGrammarCategory***: Este método se utilizará para extraer del *array* creado a partir del método *SentenceLemmatization* la categoría gramatical adecuada para cada palabra. Se contempló el escenario producido por la aparición del mismo *keyword* dos o más veces en la misma oración, con sentidos idénticos y/o distintos. Para solventar esta cuestión, se configuró la salida de este método como un *array*, almacenando en cada posición la categoría gramatical correspondiente a cada una de las distintas ocurrencias de la palabra según su aparición en la frase. En el programa, cada una de estas categorías será obtenida iterativamente a partir de un bucle.

  Se aprovechará, además, este proceso para almacenar la palabra en el listado de *keywords* encontrados y para generar y almacenar sentencias de *feedback* que expliquen el porqué de la selección (no es lo mismo encontrar una palabra con la categoría gramatical buscada que sin ésta). Con este procedimiento, es posible que algunas sentencias se dupliquen en el *buffer* de *feedback* creado si comparten categoría gramatical y proceso de identificación. Sin embargo, esto no será muy importante ya que en los métodos públicos creados para mostrar los resultados (Sección 4.4), se filtrarán y eliminarán aquellas sentencias repetidas, por lo que será preferible identificar correctamente el *token* a condicionar de alguna manera el proceso.

- ***ProcessInfo***: Este método se encarga de recorrer, iterativamente, las diferentes respuestas de los alumnos (transformadas a minúsculas) y llamar a los métodos pertinentes en cada caso (para pre-procesar la información, evaluar su contenido, etc), coordinando así su procesamiento. Es decir, su funcionamiento se asemejará a la labor que desempeña el *main* en programación (Figura 4.8).

- ***isWordInWindow***: Este método se utiliza, recursivamente, como función para identificar los distintos *keywords* de las frases. Recibe, como parámetros, la frase a analizar, el término del listado de palabras clave a buscar y el tamaño de ventana correspondiente. De esta forma, la función recorrerá la frase para comprobar si el

n-grama, cuyo tamaño estará definido por la ventana, coincide con la palabra clave en cuestión. Este proceso se repetirá por cada *keyword* disponible y por cada tamaño de ventana, tal y como se explica en el método *Evaluate* (a continuación de éste).

Es importante mencionar que, inicialmente, se tomó otra aproximación como estrategia de comparación. En lugar de recorrer la frase desplazando un filtro de tamaño correspondiente al de la ventana, como finalmente se implementó, se buscó analizar la frase palabra por palabra. De esta forma, se analizaba iterativamente si un determinado *token* de la frase estaba incluido en los tesauros como clave, como parte de una palabra compuesta, o como sinónimo/antónimo. Para ello, si existía coincidencia, se analizaban las palabras vecinas en un determinado rango o ventana proporcional al *keyword* que se estaba buscando. Es decir, si se detectara la palabra "balanceo", por ejemplo, se observaría que la palabra coincide con una unidad del *keyword* "balanceo de carga". Como esta palabra clave, concretamente, tiene una longitud de 3 unidades, se desplazaría un filtro de dicho tamaño desde la posición x-2 (siendo x la posición de la palabra detectada) hasta la posición x+2. Si durante ese proceso se encontraban coincidencias, se anotaba la palabra clave y se continuaba con el análisis de la oración, sino, se consideraba la palabra detectada como "falso positivo", y se descartaba su identificación.

Se comprobó que la aproximación anterior, valorada en un principio como más óptima que la implementada al final, tenía un rendimiento inferior a ésta. Tras realizar ciertas pruebas, se determinó que, con el aumento del número de tamaños de ventanas, lógicamente, el rendimiento de la aproximación final se reducía, mientras que el de la técnica anterior se mantenía relativamente constante, equiparándose en cuanto a prestaciones. Por tanto, la razón del cambio de técnica no fue precisamente por su rendimiento, sino por la dificultad de proporcionar *feedback* específico a cada situación. La técnica anterior, además, concentraba el análisis de múltiples sentencias de forma simultánea, tarea que se veía mayormente complicada con la aparición de palabras compuestas. No solo era necesario un método similar a *isWordInWindow* para asegurar la aparición del *keyword*, sino que también era necesario otro con el que detener el análisis de los demás *keywords* y saltar el número de n-gramas correspondientes para no repetir una misma evaluación de otra unidad del *keyword* compuesto identificado.

Cabe destacar que, en el momento del cambio de método (tras efectuar la comparación de rendimiento inicial entre ambos), los tesauros no estaban adecuados con índices como el de "*Compound*" para facilitar el procesamiento, lo cual dificultaba la identificación de las palabras compuestas al encontrar primero las unidades que lo componían. Es posible que, con los tesauros construidos de esa forma, el rendimiento del método anterior hubiera sido ligeramente mayor. Sin embargo, por comodidad de estructuración y capacidad de particularizar mejor las comparaciones para afinar mejor la justificación del *feedback*, se descartó dicha técnica y se apostó, finalmente, por la implementada en este trabajo.

Con todo, el rendimiento de la utilizada es bastante óptimo, pues tarda una media de 0.63 segundos por respuesta (1 minuto y 31 segundos en procesar las 144 respuestas) si se lematizara la oración sin considerar la categoría gramatical de las palabras ambiguas, lo que permitiría obtener el *feedback* de forma casi inmediata (admitiendo cierto error). En caso de calcular, primero, la categoría gramatical para minimizar el error de la lematización, el tiempo de procesamiento aumenta considerablemente, pues tarda una media de 8.33 segundos por respuesta. A pesar del incremento, se considera que dicho tiempo rebaja con creces la duración que tardaría el profesor a corregir la respuesta, por lo que se considerará adecuado.

- ***Evaluate***: Este método se compondrá de tres bloques: el primero, donde se analizará la existencia de *keywords* en la oración; el segundo, donde se analizará la aparición de sinónimos; y el tercero, donde se analizará la existencia de antónimos (Figura 4.9). Los tres bloques estarán diseñados de manera idéntica; es decir, para recorrer la frase por cada uno de los tamaños de ventana del filtro de cada *set* y por cada uno de los *keywords* de la lista. Sin embargo, el método presentará algunas diferencias en referencia a la forma de procesamiento de las unidades de cada bloque de código.

  En el primer bloque, tras ratificar la existencia de un *keyword* en la oración, se procederá a efectuar su análisis morfosintáctico, independientemente de la longitud del término. Si la palabra es simple, se compara la categoría gramatical calculada con la referencia almacenada en el índice "GCategory" del tesauro de sinónimos. Dependiendo de si hay o no coincidencia, se configurará el *feedback* de una forma u otra (se expandirá este punto en la Sección 4.4), de manera que siempre se tendrá anotada dicha evidencia cuando se consulte la razón de aparición de ese determinado

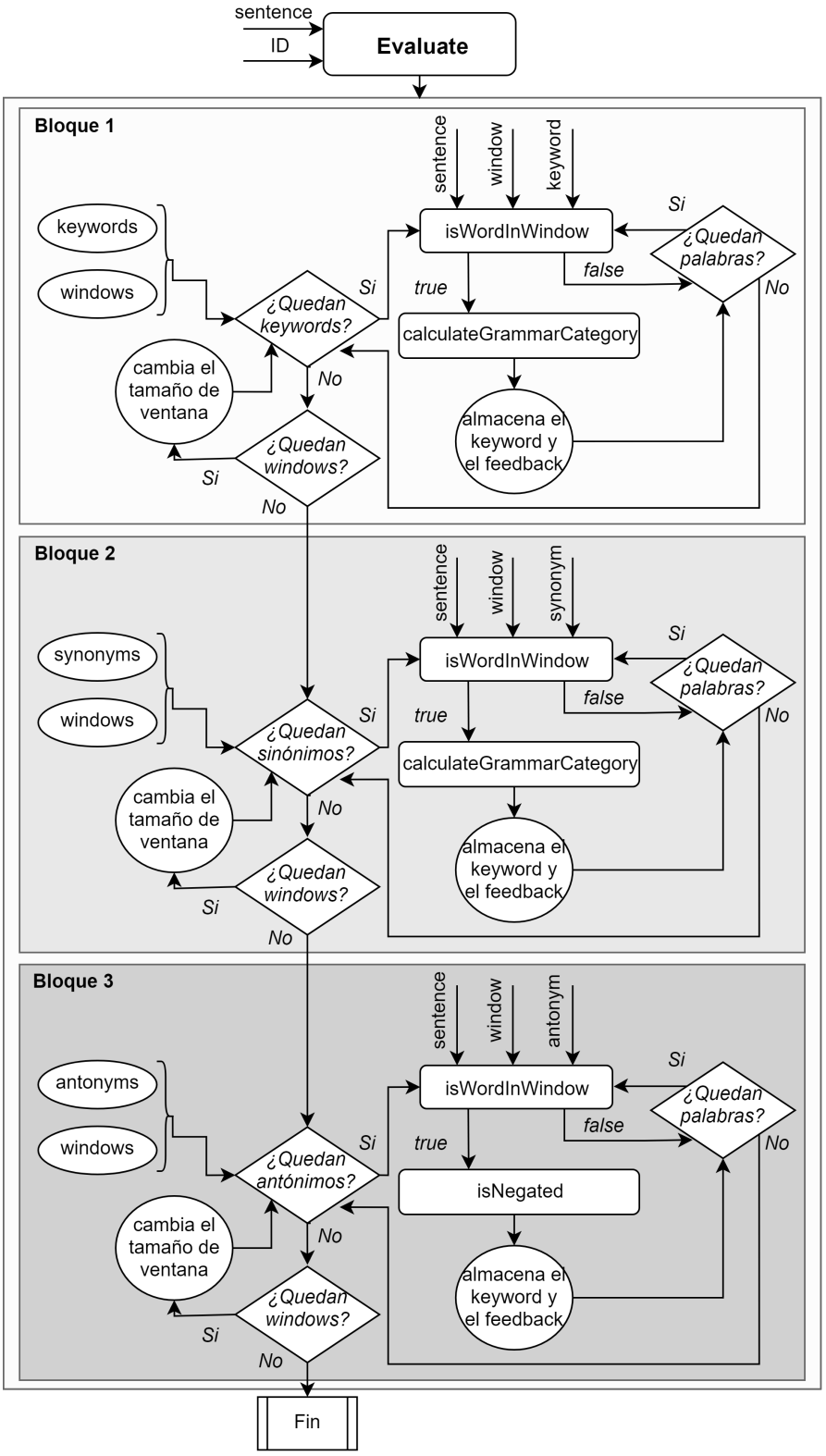


Figura 4.9: Diagrama de flujo del método *Evaluate* de la clase *NLP_Answers*

keyword. Por otra parte, si la palabra es compuesta, se efectuará el mismo procedimiento descrito para una palabra simple aplicado a cada unidad significativa del *keyword* en cuestión, descartando, así, las unidades consideradas como *stopword*. Si todas estas unidades tienen la categoría gramatical buscada, se configura una sentencia de *feedback*, sino, se configura otra (como el caso anterior y como ocurre con los demás bloques).

En el segundo bloque, el funcionamiento para los sinónimos simples será análogo al procedimiento del primer bloque, con la diferencia que se empleará únicamente el *pipeline* de *Stanza* como método para obtener la categoría gramatical. Sin embargo, las palabras compuestas se analizarán a partir de sus unidades más significativas, pero omitiendo la búsqueda de la categoría gramatical. De esta forma, si todas las unidades coinciden con la propia unidad clave o con un sinónimo de ésta, se considerará su aparición, sino, se ignorará dicho n-grama.

En el último bloque, se analizará la aparición del par formado por el antónimo de algún *keyword* y una palabra que exprese negación que figure en el siguiente grupo:

```
1 ["no", "nunca", "jamás", "nada", "ni", "siquiera", "ningún", "
    tampoco", "ninguno", "nadie", "ninguna"]
```

Para identificar un antónimo clave, se considerará suficiente evidencia la aparición del propio antónimo negado (método *isNegated* del programa), por lo que no será necesario realizar ningún análisis de carácter morfosintáctico, ni con el etiquetador ni con *Stanza*. Además, se omitirá el procesamiento de los antónimos que provienen de una palabra clave compuesta, pues se consideró su potencial aparición como alto improbable, dado el cúmulo de condiciones que se deberían cumplir para ello (aparición de los antónimos de las unidades más significativas de la palabra clave y aparición de un número impar de términos que expresen negación). Además, si propiamente es el antónimo la unidad que más se aleja en cuanto a significado de la verdadera palabra clave, si se combinaran dos o más de éstas, el resultado, aunque estuviera negado, podría no expresar con bastante probabilidad el significado buscado.

- ***CalculateMark***: Este método se encarga de efectuar las determinadas operaciones

necesarias para el cómputo de la nota del estudiante, criterios que se desarrollarán con mayor profundidad en el siguiente punto (Sección 4.4).

4.4. Evaluación y *feedback*

En esta sección se expondrán los criterios de evaluación escogidos para calificar la respuesta de cada estudiante y el método diseñado para generar el *feedback* más oportuno posible de cada contestación. Además, se relacionará este contenido con las funciones del programa encargadas de calcular y obtener dichas operaciones.

Se plantea, como hipótesis, la existencia de una relación entre el número de palabras clave acertadas y la nota que otorgó el profesorado. Esta hipótesis se fundamenta en que el conjunto de palabras clave de la pregunta es exactamente aquello que el profesor espera que el estudiante responda. Es posible que el profesor considere más importantes algunos de los conceptos que ha indicado como claves, aún así, debido al desconocimiento del tema a evaluar, se asumirá que todos los *keywords* tendrán una importancia similar. Por tanto, a mayor número de palabras clave distintas acertadas, mayor nota se le asignará al alumno. De esta forma, la nota del estudiante se calculará con la Ecuación 4.1:

Número de *keywords* incluídos en la respuesta / Número de *keywords* totales (4.1)

A partir de la información sustraída del proceso explicado en el apartado anterior, se obtendrán cuatro diccionarios que serán accesibles a partir de los métodos públicos de la clase *NLP_Answers*: *showFeedbackDistribution*, *showKeywordsDistribution*, *showFeedback* y *showMarks*. Los dos primeros almacenarán, por cada frase en la respuesta del alumno, la información relativa al número de palabras clave encontradas y al *feedback* relacionado con las mismas. Se podrá, así, examinar si existe alguna relación en la distribución de aparición de los *keywords* de cada estudiante. Los dos segundos diccionarios almacenarán el mismo contenido pero filtrado, de forma que se contarán y mostrarán únicamente el número de *keywords* y sentencias de *feedback* diferentes existentes en la respuesta. El diccionario *showMarks*, además, contendrá el cálculo de la nota otorgada a partir de este método (Ecuación 4.1).

La retroalimentación que proporcionará el software, se centrará únicamente en argumentar por qué se ha considerado como clave una determinada palabra. Así pues, por cada

keyword identificado y agregado, se dispondrá de la justificación de su selección. Como se ha explicado en el apartado anterior, existen casos en los que la identificación podría ser dudosa, y la palabra podría no expresar el sentido adecuado. En esta aproximación, se considerarán igualmente válidas este tipo de palabras, pues el *feedback* proporcionado, que se ajusta a cada posibilidad contemplada en la comparación, incluye detalladamente el proceso deliberativo seguido, clarificando la necesidad o no de que el profesor supervise dicha cuestión. Sin embargo, se podría contemplar la opción de no incorporarlos y esperar la verificación del profesor, que intervendría nuevamente dependiendo del *feedback* obtenido.

Se elaboraron, así, las siguientes 11 sentencias, donde la palabra en negrita representa en cada caso el valor de la variable con dicha nomenclatura en el programa. Cabe destacar que cada una de estas frases representa una situación de evaluación diferente de cada bloque explicado en el método *Evaluate* del anterior punto. De esta manera, las primeras 5 sentencias corresponden al primer bloque, las 4 siguientes al segundo bloque, y las 2 últimas, al tercer bloque.

- “La palabra clave ” + **keyword** + “ aparece correctamente utilizada en el texto”
- “La palabra clave ” + **keyword** + “ aparece en el texto, pero no con la categoría gramatical buscada, por lo que su sentido podría ser inadecuado”
- “La expresión clave ” + **keyword** + “ aparece correctamente empleada en el texto”
- “La expresión clave ” + keyword + “ aparece correctamente empleada en el texto”
- “La palabra ” + **component** + “ aparece en el texto y es una forma de expresar la palabra clave ” + **keyword** + “, pero su significado puede no ser el adecuado”
- “En el texto aparece la expresión ” + **element** + “, que es sinónima del lema de la palabra clave ” + **keyword**
- “En el texto aparece la palabra ” + **element** + “, que es sinónima del lema de la palabra clave ” + **keyword**
- “El sinonimo ” + **element** + “ de la palabra clave ” + **keyword** + “ aparece, pero no con la categoría gramatical buscada, por lo que su sentido podría ser inadecuado”
- “En el texto aparece la expresión ” + **expresion** + “, que es sinónima de la sentencia clave ” + **keyword**

- "En el texto aparece la palabra " + **element** + " negada, que es antónima de " + **keyword**
- "Aparece la palabra " + **element** + ", que es antónima de " + **keyword** + ", pero no se ha encontrado ninguna forma de negación, por lo que podría no compartir el significado buscado"

Para ejemplificar el formato que tendrían estos archivos, se mostrará en la Sección 5.2 el contenido obtenido para varios alumnos, teniendo en cuenta que cada línea de *showFeedbackDistribution* y de *showKeywordsDistribution* representa una frase distinta de la respuesta del estudiante.

Capítulo 5

Resultados

5.1. Ejemplos de funcionamiento del etiquetador morfosintáctico

En esta sección, se mostrará el funcionamiento del etiquetador morfosintáctico elaborado a partir de dos frases, incluyendo su correspondiente etiquetado. La primera de ellas, contiene las palabras "Pepe", "XXI" y "21" que, *a priori*, no deberían estar recogidas en el diccionario, pues se trata de palabras potencialmente categorizadas como "*tokenseliminados*". Sin embargo, durante la realización de la prueba, se advirtió que únicamente "21" no está incluida en éste, "Pepe" y "XXI" sí que lo están, al igual que "Carlos", "Juan" y otros nombres propios. Esto se debe a que, en el corpus, aparecen dichas palabras etiquetadas también con una categoría gramatical distinta a la de "NP..." (por ejemplo, AQ0CS0, RG o NCMS000). Se observará en el análisis de las oraciones, de esta forma, la respuesta del etiquetador tanto para palabras eliminadas como para palabras con determinadas categorías gramaticales suprimidas. Se pretende observar, también, si el efecto de la penalización introducida durante el funcionamiento del algoritmo de *Viterbi* provoca que el resultado sea correcto o no.

La segunda frase, a su vez, es un ejemplo de desambiguación de la palabra vino. La primera acepción de la frase está referida al verbo venir, pero la siguiente está referida al sustantivo; es decir, al producto que se obtiene de la vid. También incluye la palabra "1996", que se trata de una palabra no recogida tampoco en el diccionario. Además, en ambas oraciones aparecen diversos *tokens* más de una vez, por lo que se podrá comprobar también la fórmula creada para diferenciar su aparición en la frase.

- **El invitado vino con el vino tinto de la cosecha de 1996**

```
el / Determiner
invitado / Noun
vino / Verb
con / Adposition
el(2) / Determiner
vino(2) / Noun
tinto / Adjective
de / Adposition
la / Determiner
cosecha / Noun
de(2) / Adposition
1996 / Noun
```

- **Pepe me dijo que XXI significa 21 en números romanos en el siglo 21.**

```
pepe / Noun
me / Pronoun
dijo / Verb
que / Conjunction
xxi / Noun
significa / Verb
21 / Noun
en / Adposition
números / Noun
romanos / Adjective
en(2) / Adposition
el / Determiner
siglo / Noun
21(2) / Noun
```

Como *baseline* para poder comparar los resultados, se adjuntan dos tablas que incorporan el análisis morfosintáctico real de ambas frases (Tabla 5.1 y Tabla 5.2), complementadas con dos figuras en las que se muestra el análisis sintáctico de las mismas (Figura 5.1 y Figura 5.2).

Palabra	Categoría gramatical	Descripción
El	da0000	Artículo (definido)
invitado	nc0s000	Nombre común (singular)
vino	vmis000	Verbo (principal, indicativo, pretérito)
con	sp000	Preposición
el	da0000	Artículo (definido)
vino	nc0s000	Nombre común (singular)
tinto	aq0000	Adjetivo (descriptivo)
de	sp000	Preposición
la	da0000	Artículo (definido)
cosecha	nc0s000	Nombre común (singular)
de	sp000	Preposición
1996	w	Fecha

Tabla 5.1: Análisis morfosintáctico real de la frase 1.

Fuente: https://data.cervantesvirtual.com/analizador-sintactico-automatico

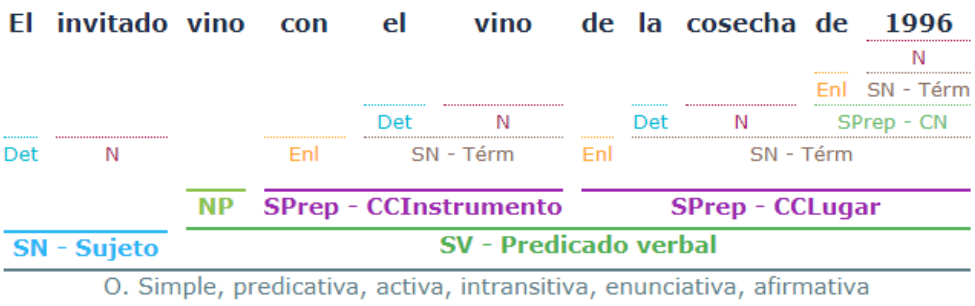


Figura 5.1: Análisis sintáctico de la primera frase.

Fuente: https://sintaxis.org/analizador/

Se observa que las palabras consideradas como "*tokenseliminados*", en ambas frases, han sido etiquetadas como sustantivo. En los análisis morfosintácticos complementarios (Tabla 5.1 y Tabla 5.2), dichas palabras se muestran como número o fecha, por lo que no pueden servir como referencia para saber si se han etiquetado correctamente. Sin embargo, juzgando el análisis sintáctico de ambas (Figura 5.1 y Figura 5.2), se comprueba que las dos constituyen el núcleo de un sintagma nominal, validando así la decisión del etiquetador. Se puede afirmar, consecuentemente, que la penalización introducida no ha influido negativamente en el propio etiquetado.

Palabra	Categoría gramatical	Descripción
Pepe	np00000	Nombre propio
me	pp000000	Pronombre personal
dijo	vmis000	Verbo (principal, indicativo, pretérito)
que	cs	Conjunción (subordinada)
XXI	np00000	Nombre propio
significa	vmip000	Verbo
21	z0	Número
en	sp000	Preposición
números	nc0p000	Nombre común (plural)
romanos	aq0000	Adjetivo (descriptivo)
en	sp000	Preposición
el	da0000	Artículo (definido)
siglo	nc0s000	Nombre común (singular)
21	z0	Número

Tabla 5.2: Análisis morfosintáctico real de la frase 2.

Fuente: https://data.cervantesvirtual.com/analizador-sintactico-automatico

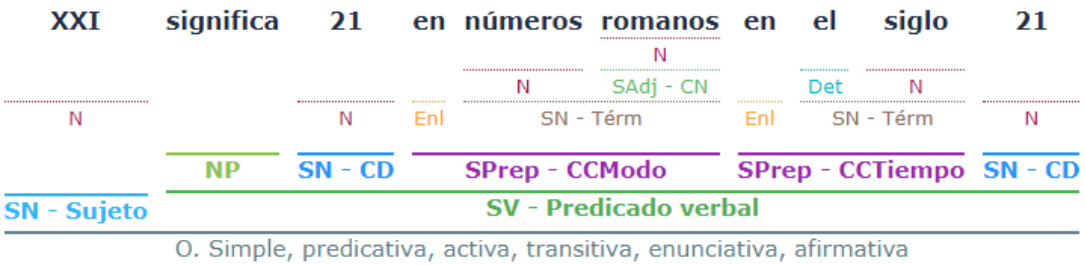


Figura 5.2: Análisis sintáctico de una parte de la segunda frase.

Fuente: https://sintaxis.org/analizador/

Se comprueba, por tanto, que el etiquetador es robusto a palabras nuevas, ya que ha conseguido efectuar el análisis completo y correcto de ambas oraciones. Además, ha sido capaz de diferenciar las distintas ocurrencias en ambas frases, asignando la etiqueta correspondiente en cada caso. No obstante, se observa que las palabras catalogadas como "*tokenseliminados*", debido a su proceso de formación, están bastante condicionadas a ser sustantivo, debido a que la probabilidad de emisión de sustantivo es superior a la de

adjetivo. De esta forma, en aquellas situaciones en las que la probabilidad de transición de un estado a otro sea relativamente equiprobable, el etiquetador asignará la palabra como sustantivo. Como se argumentó en la Sección 4.3, esto no es muy importante ya que no será necesario que el análisis sea siempre perfecto, y menos para este tipo de palabras complementarias que, además, suelen ser mayoritariamente sustantivos o nombres propios.

Por otra parte, se observa que palabras como "pepe" o "xxi", al no ser consideradas como "*tokenseliminados*" y estar incluidas en el diccionario, pierden adaptabilidad según la función que desempeñen en la oración. Por este motivo, todas las ocurrencias de "xxi" en el análisis de la segunda frase se catalogarán como sustantivo, porque es la única acepción almacenada. Se podría tratar de filtrar este tipo de palabras del diccionario utilizando las claves de las palabras que fueron eliminadas. Sin embargo, se advirtió que, en dicho caso, la consecuente pérdida de información (en otras palabras sí interesantes) hace que no compense la aproximación.

Por último, se comprueba que la palabra vino ha sido etiquetada correctamente dependiendo de su acepción en la oración, evidenciando que sí se ha conseguido desarrollar un método relativamente fiable para hacer WSD.

5.2. Ejemplos de funcionamiento del método desarrollado

En esta sección, se mostrará el *feedback* generado y los *keywords* obtenidos en relación a la respuesta dada por diversos estudiantes. De esta manera, se podrá comprobar de forma objetiva si el contenido calculado en cada ocasión es oportuno o no. Las tres primeras subsecciones (Subsección 5.2.1 - Subsección 5.2.3) servirán para mostrar los casos y poder comparar el contenido y su distribución. El análisis detallado de éstos, se profundizará en la Subsección 5.2.4.

5.2.1. Estudiante con ID eb9121746ee55b359a9eefa8f6378365

Respuesta proporcionada segmentada en frases:

```
['La arquitectura réplica-set se basa en la redundancia de datos y el
    incremento de su disponibilidad.',
 ' ',
 'Sigue un modelo maestro-esclavo que fue mejorado.',
 ' ',
```

```
'La instancia maestra recibe las operaciones de escritura y estas se aplican en las instancias dependiente para mantener la consistencia de los datos.',
' ',
'La mejora de este modelo permite la recuperación a fallos de forma automática.',
' ',
'Esto ocurre porque habrá un nodo principal',
' nodos secundarios y un nodo árbitro.',
' ',
'La asignación del nodo principal se realiza al azar y viene determinada por el nodo árbitro.',
' ',
'Los nodos entre ellos se envían pings cada dos segundos para saber la accesibilidad en cada instante',
' lo que permite al nodo árbitro elegir un nuevo nodo principal en caso de fallo.',
' ',
'El resto de nodos replicarán las operaciones realizadas sobre el primario',
' aumentando así la consistencia de los datos.',
' ',
'La configuración mínima de un réplica-set se consigue con tres nodos .',
'',
'La principal diferencia es que mongodump realiza copias de seguridad conectándose a una instancia mongod.',
' ',
'Si hay algún fallo y necesitamos recuperar los datos',
' estos puede que no estén actualizados a la última versión.',
' ',
'Por el contrario',
' la arquitectura réplica-set mantiene los datos actualizados en todo momento']
```

showFeedbackDistribution:

```
[['La palabra clave disponibilidad aparece correctamente utilizada en
   el texto'],
[],
[],
[],
[],
[],
[],
[],
['El sinonimo haber de la palabra clave disponibilidad aparece, pero no
    con la categoría gramatical buscada, por lo que su sentido podría
   ser inadecuado'],
[],
[],
[],
[],
['En el texto aparece la palabra enviar, que es sinónima del lema de la
    palabra clave exportar'],
[],
[],
[],
[],
[],
['En el texto aparece la palabra conseguir, que es sinónima del lema de
    la palabra clave arbitrar'],
[],
[],
[],
['El sinonimo haber de la palabra clave disponibilidad aparece, pero no
    con la categoría gramatical buscada, por lo que su sentido podría
   ser inadecuado'],
['La palabra clave actualizar aparece en el texto, pero no con la
   categoría gramatical buscada, por lo que su sentido podría ser
   inadecuado'],
[],
[],
['La palabra clave actualizar aparece en el texto, pero no con la
```

```
   categoría gramatical buscada, por lo que su sentido podría ser
   inadecuado']]
```

showKeywordsDistribution:

```
[['disponibilidad'],
 [],
 [],
 [],
 [],
 [],
 [],
 ['disponibilidad'],
 [],
 [],
 [],
 [],
 ['exportar'],
 [],
 [],
 [],
 [],
 [],
 ['arbitrar'],
 [],
 [],
 [],
 ['disponibilidad'],
 ['actualizar'],
 [],
 [],
 ['actualizar']]
```

showFeedback:

```
['La palabra clave actualizar aparece en el texto, pero no con la
   categoría gramatical buscada, por lo que su sentido podría ser
```

```
    inadecuado', 'En el texto aparece la palabra conseguir, que es
    sinónima del lema de la palabra clave arbitrar', 'La palabra clave
    disponibilidad aparece correctamente utilizada en el texto', 'En el
     texto aparece la palabra enviar, que es sinónima del lema de la
    palabra clave exportar', 'El sinonimo haber de la palabra clave
    disponibilidad aparece, pero no con la categoría gramatical buscada
    , por lo que su sentido podría ser inadecuado']
```

showMarks:

```
[['arbitrar', 'disponibilidad', 'actualizar', 'exportar'], 'Nota:
    0.44']
```

5.2.2. Estudiante con ID f4d0bf14886ef288809047c1c7bce5ce

Respuesta proporcionada segmentada en frases:

```
['Disponer de una configuración en mongoDB en Replica-set nos permite
    mejorar la disponibilidad de nuestro clúster al disponer de varios
    nodos configurados de tal forma que en caso de que uno falle otro
    asuma el ""mando.',
' ',
'Los nodos envían un heartbeat cada dos segundos para comprobar la
    disponibilidad del resto',
' de esta manera se mantiene siempre actualizado y en caso de fallo
    hay un margen pequeño de pérdidas.',
'',
'Una configuración replica-set para ser considerada como tal se deben
    configurar como mínimo tres nodos.',
' ',
'Los cuales son un nodo primario',
' un nodo secundario y un nodo arbitro.',
' ',
'Este último nodo se encarga de que cuando un nodo primario se cae y
    se procede a la votación de quien será el nuevo nodo primario',
' en caso de empate el nodo arbitro se encargará de desempatar y
    adjudicar el nodo primario al nodo secundario mejor ""posicionado.'
    ,
```

```
13  '',
14  'Los nodos secundarios pueden configurarse de diferentes maneras tales
        como: hidden (no puede ser nodo primario)',
15  ' retrasado (espera un tiempo hasta hacer los cambios)',
16  ' etc.',
17  '',
18  'La diferencia con mongodump reside en que este realiza una ""foto de
       los datos en el momento de realizarse la llamada a la función.',
19  ' ',
20  'Mientras que el replica-set se encarga de hacer backups constantes']
```

showFeedbackDistribution:

```
1  [['La palabra clave disponibilidad aparece correctamente utilizada en
      el texto'],
2  [],
3  ['En el texto aparece la palabra enviar, que es sinónima del lema de la
       palabra clave exportar', 'La palabra clave disponibilidad aparece
      correctamente utilizada en el texto'],
4  ['El sinonimo haber de la palabra clave disponibilidad aparece, pero no
       con la categoría gramatical buscada, por lo que su sentido podría
      ser inadecuado', 'La palabra clave actualizar aparece en el texto,
      pero no con la categoría gramatical buscada, por lo que su sentido
      podría ser inadecuado'],
5  [],
6  [],
7  [],
8  [],
9  ['La palabra clave arbitrar aparece en el texto, pero no con la
      categoría gramatical buscada, por lo que su sentido podría ser
      inadecuado'],
10 [],
11 ['La palabra clave votación aparece correctamente utilizada en el texto
      '],
12 ['La palabra clave arbitrar aparece en el texto, pero no con la
      categoría gramatical buscada, por lo que su sentido podría ser
      inadecuado'],
13 [],
```

```
[],
['En el texto aparece la palabra cambio, que es sinónima del lema de la
    palabra clave sustitución'],
[],
[],
[],
[],
[]]
```

showKeywordsDistribution:

```
[['disponibilidad'],
[],
['disponibilidad', 'exportar'],
['disponibilidad', 'actualizar'],
[],
[],
[],
[],
['arbitrar'],
[],
['votación'],
['arbitrar'],
[],
[],
['sustitución'],
[],
[],
[],
[],
[]]
```

showFeedback:

```
['La palabra clave actualizar aparece en el texto, pero no con la
   categoría gramatical buscada, por lo que su sentido podría ser
   inadecuado', 'La palabra clave votación aparece correctamente
   utilizada en el texto', 'La palabra clave arbitrar aparece en el
```

```
   texto, pero no con la categoría gramatical buscada, por lo que su
   sentido podría ser inadecuado', 'La palabra clave disponibilidad
   aparece correctamente utilizada en el texto', 'En el texto aparece
   la palabra enviar, que es sinónima del lema de la palabra clave
   exportar', 'El sinonimo haber de la palabra clave disponibilidad
   aparece, pero no con la categoría gramatical buscada, por lo que su
    sentido podría ser inadecuado', 'En el texto aparece la palabra
   cambio, que es sinónima del lema de la palabra clave sustitución']
```

showMarks:

```
['votación', 'exportar', 'sustitución', 'arbitrar', 'disponibilidad', '
   actualizar'], 'Nota: 0.67'],
```

5.2.3. Estudiante con ID cc74af2802f44d4dd32ff43c7035e086

Respuesta proporcionada segmentada en frases:

```
['La función de un replica-set es dar respaldo a los datos.',
 ' ',
 'Un replica-set es una estructura que se crea para que en caso de
    error en el nodo primario no perdamos toda la información.',
 ' ',
 'Para crear un replica set son necesarios 3 nodos',
 ' el nodo primario',
 ' el nodo secundario y el arbitro.',
 ' ',
 'La función que tienen estos nodos es: - nodo primario: guardar la
    información.',
 ' ',
 '- nodo secundario:  hacer una copia de esta información y mantenerla
    hasta que sea necesario',
 ' que por caída del nodo primario se ponga en funcionamiento.',
 ' ',
 '- arbitro: encargado de nombrar al nodo secundario que comienza a
    actuar en caso de que el primario deje de tener disponibilidad.',
 ' ',
 'Mongodump es un buen método de respaldo de la información',
```

```
 ' pero realiza la copia de seguridad en un momento dado',
 ' a diferencia de replica- set que se mantiene actualizada.']
```

showFeedbackDistribution:

```
['El sinonimo respaldo de la palabra clave respaldo aparece, pero no
    con la categoría gramatical buscada, por lo que su sentido podría
    ser inadecuado', 'La palabra clave respaldo aparece en el texto,
    pero no con la categoría gramatical buscada, por lo que su sentido
    podría ser inadecuado'],
  [],
  [],
  [],
  [],
  [],
  ['La palabra clave arbitrar aparece en el texto, pero no con la
     categoría gramatical buscada, por lo que su sentido podría ser
     inadecuado'],
  [],
  [],
  [],
  [],
  [],
  [],
  ['La palabra clave arbitrar aparece en el texto, pero no con la
     categoría gramatical buscada, por lo que su sentido podría ser
     inadecuado', 'La palabra clave disponibilidad aparece
     correctamente utilizada en el texto', 'En el texto aparece la
     palabra actuar, que es sinónima del lema de la palabra clave
     arbitrar'],
  [],
  ['El sinonimo respaldo de la palabra clave respaldo aparece, pero no
     con la categoría gramatical buscada, por lo que su sentido podría
      ser inadecuado', 'La palabra clave respaldo aparece en el texto,
      pero no con la categoría gramatical buscada, por lo que su
     sentido podría ser inadecuado'],
  [],
  ['La palabra clave actualizar aparece en el texto, pero no con la
```

```
    categoría gramatical buscada, por lo que su sentido podría ser
    inadecuado']],
```

showKeywordsDistribution:

```
[['respaldo'],
[],
[],
[],
[],
[],
['arbitrar'],
[],
[],
[],
[],
[],
[],
['arbitrar', 'disponibilidad'],
[],
['respaldo'],
[],
['actualizar']]
```

showFeedback:

```
['La palabra clave actualizar aparece en el texto, pero no con la
   categoría gramatical buscada, por lo que su sentido podría ser
   inadecuado', 'La palabra clave arbitrar aparece en el texto, pero
   no con la categoría gramatical buscada, por lo que su sentido
   podría ser inadecuado', 'La palabra clave respaldo aparece en el
   texto, pero no con la categoría gramatical buscada, por lo que su
   sentido podría ser inadecuado', 'En el texto aparece la palabra
   actuar, que es sinónima del lema de la palabra clave arbitrar', 'El
    sinonimo respaldo de la palabra clave respaldo aparece, pero no
   con la categoría gramatical buscada, por lo que su sentido podría
   ser inadecuado', 'La palabra clave disponibilidad aparece
   correctamente utilizada en el texto'],
```

showMarks:

```
[['disponibilidad', 'arbitrar', 'actualizar', 'respaldo'], 'Nota: 0.44'
    ],
```

5.2.4. Análisis de resultados

En general se demuestra, con estos tres ejemplos, que la identificación de una palabra clave en cualquier frase se adecúa al contenido que expresa el alumno en la misma. Los *keywords*, además, no aparecen mayormente concentrados en ninguna frase en concreto, sino que se distribuyen de forma natural por la respuesta del alumno. Es cierto que en determinadas frases aparecen identificados dos o más *keywords*, pero dicha aparición se observa que se debe a la relación que hay entre los mismos, no a la expresión forzada del estudiante para incluirlos. A su vez, el *feedback* generado, en cada caso, es coherente, conciso y perfectamente adecuado a cada frase evaluada.

Se observa, por otra parte, que en la primera frase de la respuesta del estudiante con ID "cc74af2802f44d4dd32ff43c7035e086" (Subsección 5.2.3), la palabra clave "respaldo" aparece identificada por duplicado y con sentencias de *feedback* distintas. Este hecho, aunque *a priori* parezca un error, se debe a que la palabra clave tiene dos sentidos con distinta categoría gramatical: respaldo y respaldar. El algoritmo, primero, identifica el *keyword* "respaldo" perfectamente, sin embargo, después lo vuelve a considerar porque hay un sinónimo de respaldar denominado también "respaldo". Se confirma, de nuevo, que el *feedback* obtenido es congruente.

Por último, se advierte que algunas líneas de la respuesta del estudiante están vacías. Se trata de un residuo que se forma, en ocasiones, tras usar la función *split()* con la que se separan las frases. Estas líneas, no suponen apenas coste computacional, pues su contenido es nulo. Sin embargo, de cara a trabajos futuros, se podrían filtrar durante el pre-procesamiento.

5.3. Relación entre la nota calculada y la real

En esta sección se mostrará la comparativa entre la nota real que obtuvo el estudiante en su examen y la calificación calculada en este trabajo a partir de las palabras clave (Figura 5.3). Se representará, en el eje de las ordenadas (Y), la calificación de 0 a 1 conseguida por ambos métodos y, por otra parte, cada unidad del eje de las abscisas (X) representará a un estudiante distinto. A partir de este momento, para facilitar la comparación de valores, se denominará "nota A" a la calificación real del estudiante y "nota B" a la calificación calculada a partir de este trabajo.

Los valores de estas gráficas están agrupados en función creciente de la nota y muestran como, a partir de una calificación real de 0.6, la diferencia entre ambos métodos se torna cada vez mayor en líneas generales. Esta indagación, analizando meticulosamente los datos y sus valores, es lógica, pues únicamente tres estudiantes superan el 0.75 con el sistema de puntuación B. Se comprueba que para los 36 primeros estudiantes se representan los valores de las notas más próximas entre ellas. No obstante, existe una evidente diferencia entre ambas, salvo en las notas cuya calificación real fue de cero. En todas las notas A con valor de cero, la nota B fue la misma, exceptuando un único caso, que bien podría considerarse como anómalo.

Las conclusiones que se puedan extraer de estas gráficas se apoyarán también en las imágenes representadas en la Subsección II.I (Anexos), que muestran la diferencia de notas de los alumnos por cada estudiante, esta vez sin orden alguno en cuanto a nota.

5.4. Error Absoluto entre las notas

Con los resultados anteriores, se evidencia la disparidad existente entre los métodos. Para obtener la diferencia entre ambos, con mayor precisión y claridad, se utilizó como métrica la medida del error absoluto entre los valores de las notas A y B (Anexos, Subsección II.III). En la Figura 5.4, se muestra la desviación existente entre muestras, obtenida a partir del cálculo del mencionado error absoluto. Para considerar las muestras similares, la mayor parte de éstas debería estar situada cerca del cero, de forma que la gaussiana generada tuviera poca varianza. Sin embargo, se observa que la distribución de las muestras es achatada y está desplazada a la derecha, con un máximo formado alrededor del 0.5 de desviación, lo que significa que el error obtenido es muy considerable, pues supone no ten-

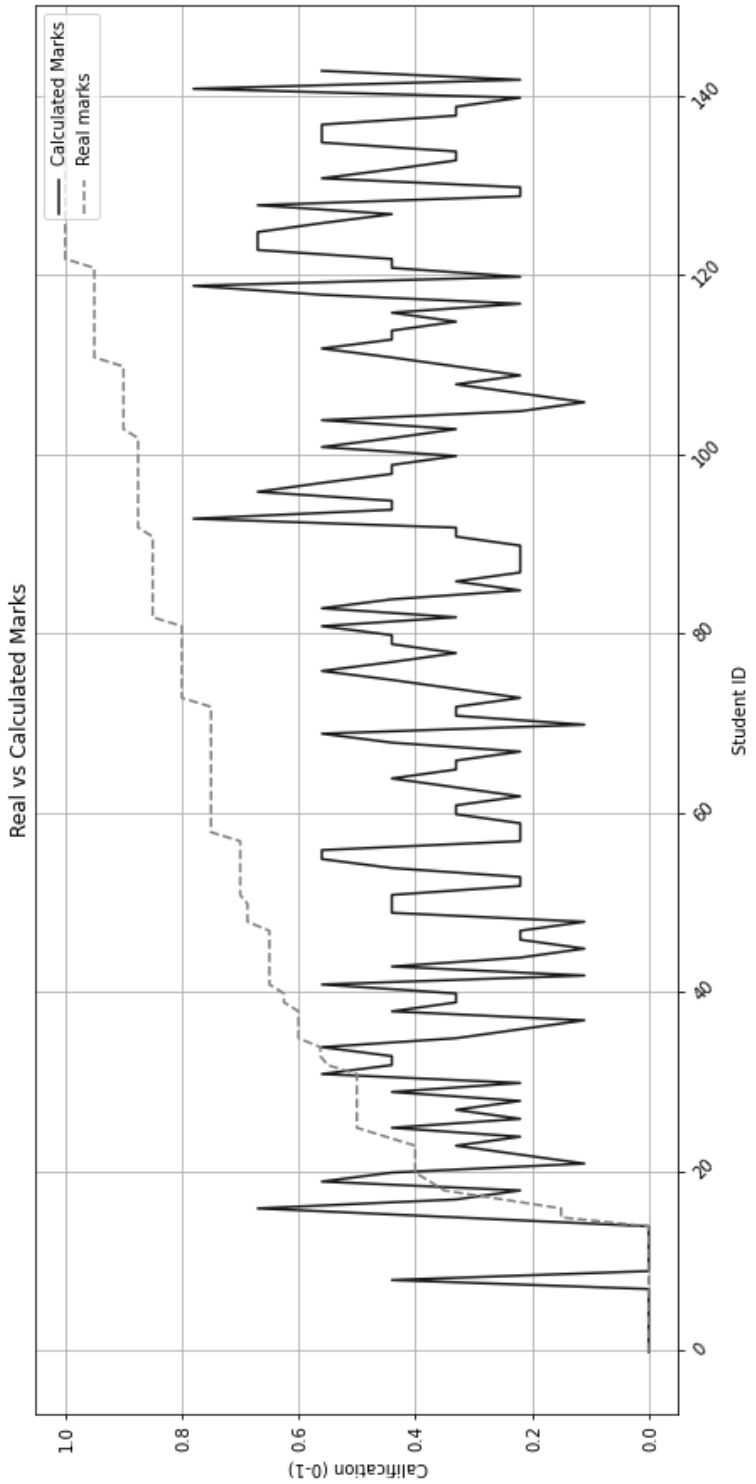


Figura 5.3: Relación entre la nota calculada y la nota real (ordenadas de menor a mayor nota real)

er la capacidad si quiera de identificar aprobados o suspensos. Se observa, también, que generalmente la nota B es inferior a la A porque la desviación obtenida es positiva y no negativa.

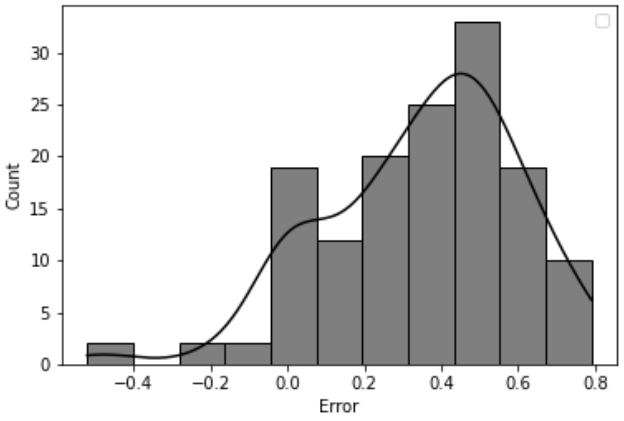


Figura 5.4: Desviación obtenida entre las notas

5.5. Relación entre el número de *keywords* y la nota

En esta sección, se mostrará el número de *keywords* identificados por cada estudiante, ordenados por nota real creciente. Con este análisis, se pretende profundizar la comparación anterior, pues el número de palabras clave es la base tomada para la calificación de las notas B. Si existiera relación entre ambos métodos, se observaría un aumento creciente en la frecuencia de aparición de los *keyowords* conforme aumentara la nota. Es evidente que este comportamiento no se dará (si se tiene en cuenta lo visto en las anteriores gráficas), sin embargo, podría servir como referencia a partir de la que extraer alguna conclusión que hubiera quedado enmascarada por la forma de evaluación asignada.

Las gráficas se deben leer de dos en dos; es decir, juzgando la Figura 5.5 en comparación con la Figura 5.6. En ellas, se observa la arbitrariedad del número total de *keywords* acertados en relación a las notas similares entre sí, lo que explica que el método de calificación desarrollado no tenga relación con el método real. Tampoco se observa una aparente relación entre el número de palabras clave y el aprobado o suspenso, existiendo calificaciones con notas bajas que tienen más aciertos que otras notas más altas.

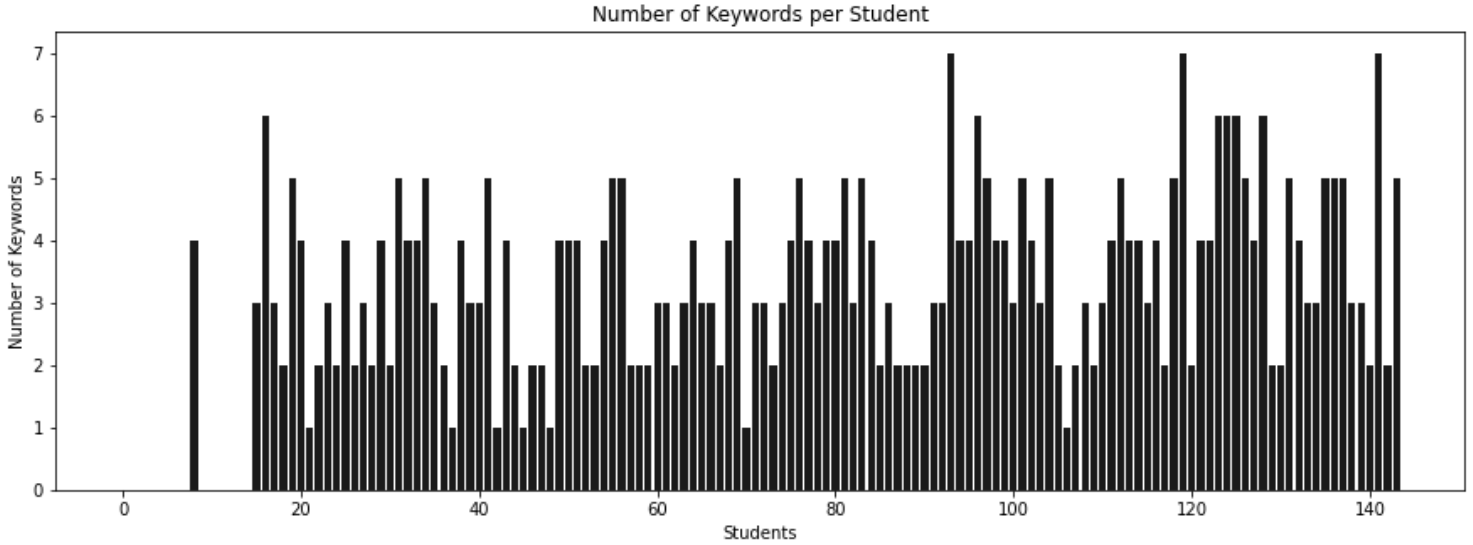


Figura 5.5: Número de *keywords* identificados por estudiante

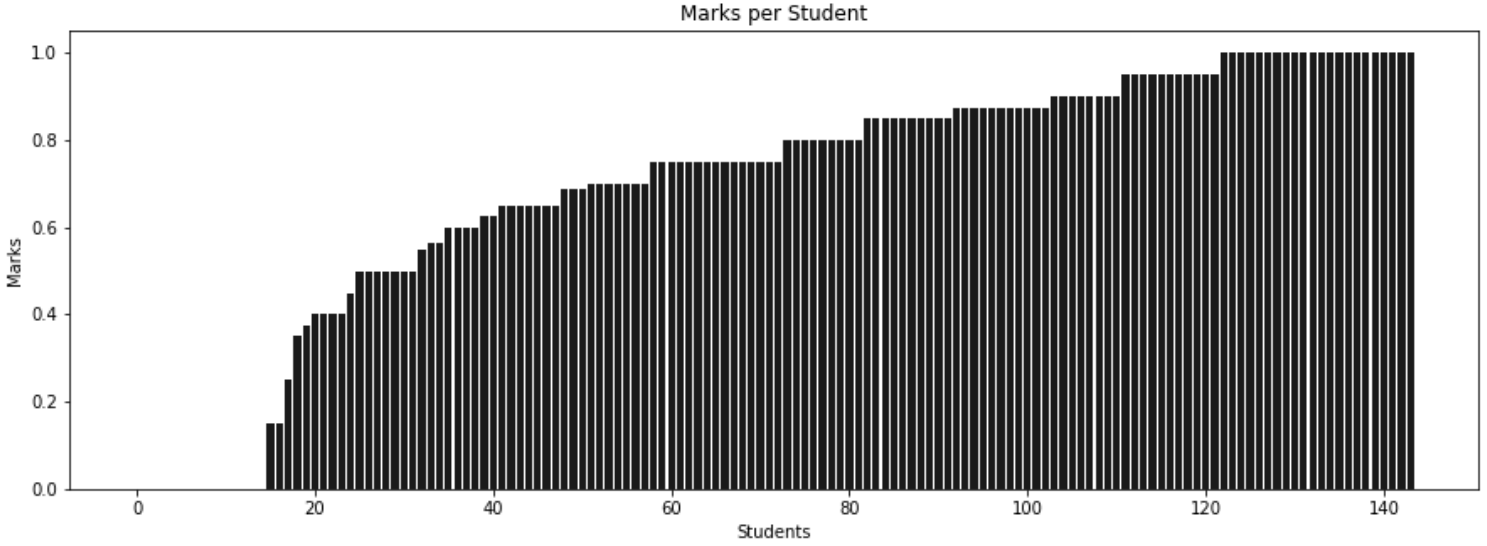
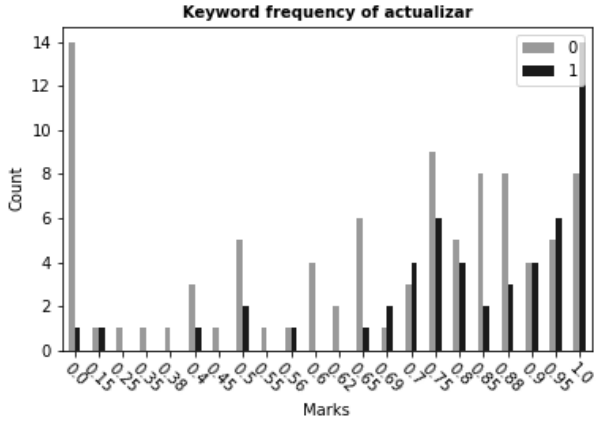


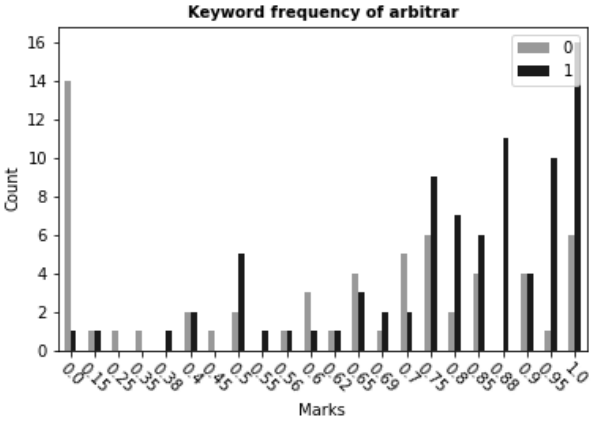
Figura 5.6: Calificación real por estudiante

5.6. Influencia de cada *keyword* en la nota

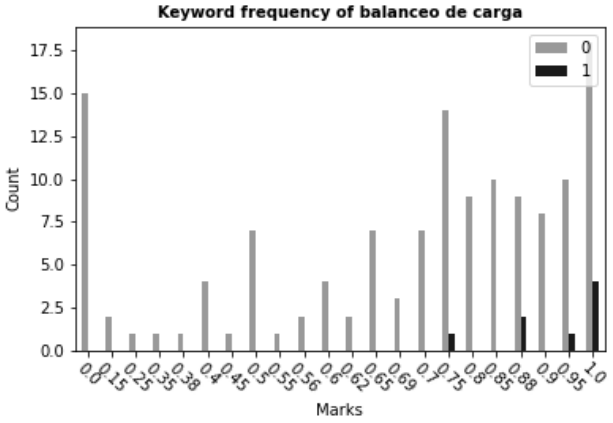
Existe la posibilidad de que el método de calificación propuesto no sea el más adecuado, pues algunos de los *keywords* del listado podrían tener más importancia para el profesor que otros. Esta hipótesis podría justificar el error obtenido hasta el momento, por este motivo, se estudiará el impacto de cada palabra clave representando la frecuencia de aparición de cada *keyword* por estudiante y nota. Esta información se complementa con la Tabla I.1 mostrada en la Subsección II.II de los Anexos, donde se muestra el número de apariciones de cada *keyword* por cada nota.



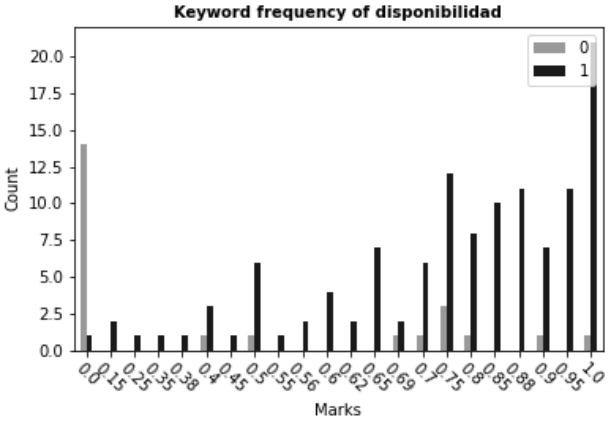
(a) Frecuencia de aparición con respecto a cada nota del *keyword* actualizar



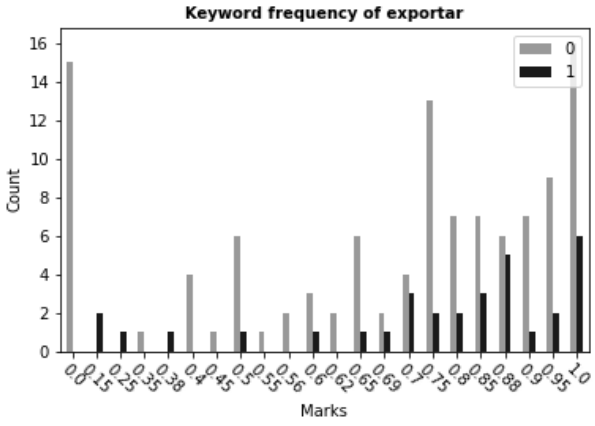
(b) Frecuencia de aparición con respecto a cada nota del *keyword* arbitrar



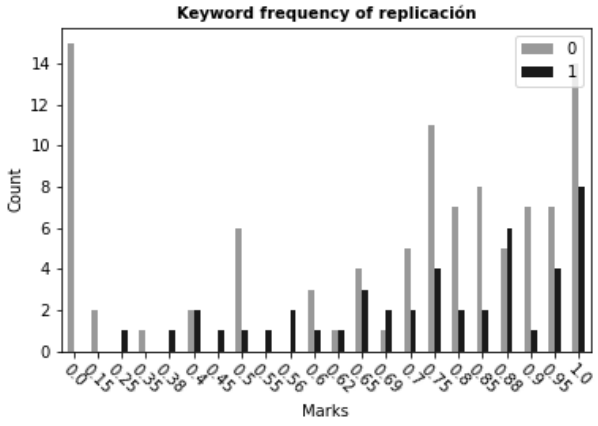
(c) Frecuencia de aparición con respecto a cada nota del *keyword* balanceo de carga



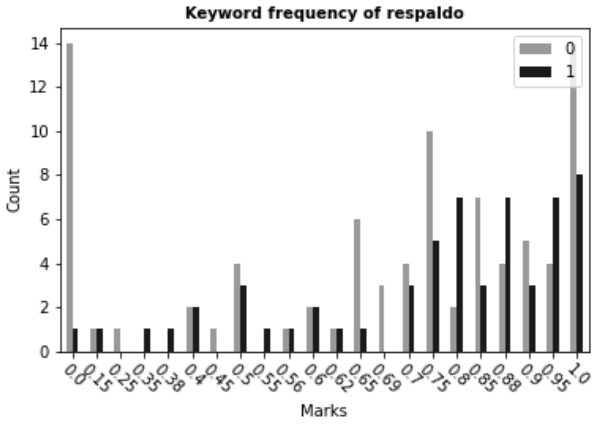
(d) Frecuencia de aparición con respecto a cada nota del *keyword* disponibilidad



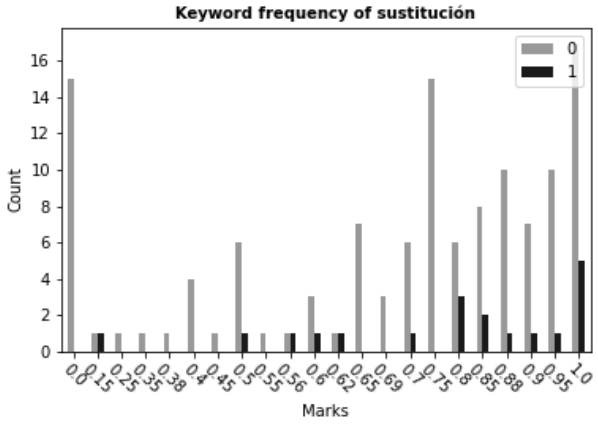
(e) Frecuencia de aparición con respecto a cada nota del *keyword* exportar



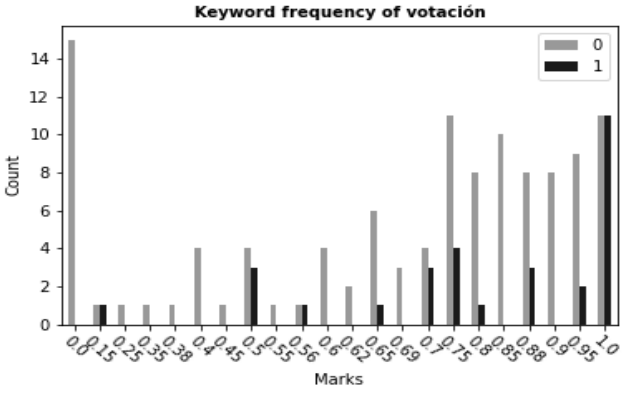
(f) Frecuencia de aparición con respecto a cada nota del *keyword* replicación



(g) Frecuencia de aparición con respecto a cada nota del *keyword* respaldo



(h) Frecuencia de aparición con respecto a cada nota del *keyword* sustitución



(i) Frecuencia de aparición con respecto a cada nota del *keyword* votación

Figura 5.11: Gráficas para mostrar la aparición de los *keywords* por nota

Se observa que los *keywords* respaldo, actualizar, arbitrar y disponibilidad son aquellos que mayor influencia parecen tener, pues su frecuencia de aparición aumenta conforme se incrementa la nota. De entre todos ellos, el *keyword* "disponibilidad" parece ser el más importante con respecto a la nota (seguido por el de "arbitrar"), pues es el que mayor número de veces se identifica de notable hacia arriba y, además, o bien es el más utilizado por los estudiantes o bien es el más detectado por el método desarrollado. Por otra parte, se observa que la cantidad de apariciones relativa a cada *keyword* aumenta a partir de 0.5. Sin embargo, esto podría no ser del todo significativo (en cuanto a influencia real en la nota), pues la cantidad de muestras disponibles en un rango de 0 a 0.5 es muy inferior a la que se dispone de 0.5 a 1.

Para complementar el análisis, se consideró importante representar, también, la influencia de cada par de *keywords*, entre sí, con respecto a la nota (Anexos, Subsección II.IV). Se pretendía comprobar, de esta manera, la existencia de una posible relación vinculante entre ellos. Es decir, si la aparición en solitario de una palabra clave influía o no en la nota antes y después de complementarse con otro de los *keywords*. Como se puede observar en la Figura 5.12, que resume el contenido mostrado en la Subsección II.IV de los Anexos, las dos palabras clave más relacionadas entre sí y, a su vez, más relacionadas con la nota son "disponibilidad" y "arbitrar".

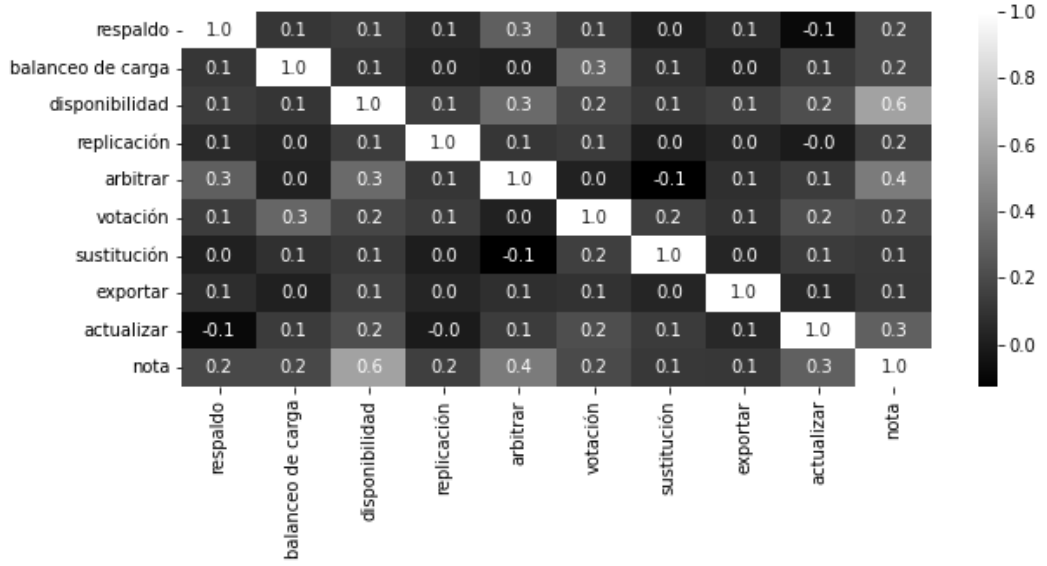


Figura 5.12: Matriz de correlación de las variables trabajadas

Sin embargo, no se puede extraer de las gráficas ninguna evidencia que afirme que la relación sea suficiente significativa. Los valores obtenidos en la matriz de correlación mostrada, tras considerar las variables categóricas como numéricas, muestran, de hecho, la debilidad de dichas relaciones. Esta afirmación, se fundamenta, también, en la Tabla I.2 mostrada en la Subsección II.IV de los Anexos, donde se observan resultados análogos utilizando otra métrica: el coeficiente de correlación de Pearson de cada par de *keywords*.

Capítulo 6

Conclusiones y Trabajo Futuro

Los resultados muestran, con clara evidencia, que no existe relación alguna entre el número de *keywords* encontrados y la nota final que recibe el estudiante. Por ejemplo, hay alumnos con nota de sobresaliente que acertaron menos palabras clave que algunos que suspendieron. Sin embargo, se observa que, salvo en un caso en particular, todas las calificaciones con un 0 por el método de evaluación de este proyecto también lo fueron por el real. Esto permitiría tener la capacidad de advertir al estudiante, sirviendo como alerta de detección temprana de contenido insuficiente, y posibilitando que éste pudiera valorar si le conviene efectuar una reentrega.

Se descarta también la existencia de algún tipo de relación entre los distintos *keywords* con respecto a la nota, a pesar de haber observado una cierta tendencia de los estudiantes a incluir, con mayor frecuencia, ciertos pares de palabras clave (como "disponibilidad" y "arbitrar"). Dicha frecuencia de aparición parece estar más correspondida a la forma de redacción de los estudiantes y a la relación de los mismos en el contexto de la prueba que a la calificación que se asignó. De hecho, en la respuesta del estudiante con ID "cc74af2802f44d4dd32ff43c7035e086" (Subsección 5.2.3) se identificaron 4 *keywords*: arbitrar, disponibilidad, respaldo y actualizar, que son precisamente los *keywords* considerados como más importantes. Sin embargo, la nota real calculada para este alumno fue de cero.

Por otra parte, analizando la forma de aparición de los *keywords* en las distintas oraciones, se puede determinar que la distribución de las palabras clave en la oración no es un factor determinante para el cálculo de la nota. Los *keywords* no aparecen mayormente concentrados en ninguna frase en concreto, sino que se distribuyen de forma natural por la respuesta, lo que indica que los alumnos utilizan dichas palabras como herramienta para

apoyar parte del contenido aprendido, y no como forma de comprimir la respuesta.

Si el número de *keywords* no tiene peso en la nota, consecuentemente, el método diseñado para calcular la calificación del estudiante es erróneo o, por lo menos, no significativo. Existe, por tanto, una nueva variable no contemplada a la hora de calificar que influye en la diferencia del criterio de ambas técnicas. Si bien este método se ha demostrado que es ineficiente por sí mismo, podría no serlo en conjunción con otras técnicas. Es decir, esta aproximación es solo una sección de una rúbrica, por lo que la diferencia de criterios podría basarse, también, en la información que sí proporcionarían los demás métodos de ésta. Al no disponer de medios para identificar el origen de esta diferencia, se postulan una serie de hipótesis que deberán ser refutadas en trabajos futuros:

- La primera de ellas, se basa en considerar que sí existe una relación entre el número de palabras clave y el criterio del profesor. Se justificaría, de esta manera, la falta de relación obtenida entre los dos métodos debido a la selección del listado de palabras clave. Es decir, es posible que los *keywords* que fueron seleccionados, al no pertenecer a un dominio muy concreto (casi todas son palabras del ámbito y uso general), no sean muy representativos. Por tanto, el profesor que califica el examen podría ignorar su aparición en el proceso de evaluación de la respuesta, pues estaría buscando otro tipo de palabras cuyo significado sí que se adecuara más a aquello que espera.
- La segunda, está basada en sostener que los *keywords* sí son representativos del dominio, pero no fueron igual de significativos para todo el conjunto de examinadores que llevó a cabo la tarea de corrección. Se introduciría, de esta manera, un factor de subjetividad a la hora de corregir la pregunta. De esta manera, un listado idéntico de palabras clave podría ajustarse, en mayor medida, a lo que busca un profesor pero no a lo que busca otro. Esto podría explicar las calificaciones desproporcionadas al número de *keywords*, tanto en aprobados como en suspendidos. Se sabe, además, que las calificaciones reales pudieron estar sometidas a criterios de corrección distintos, un dato que unido a la subjetividad que introduce cada profesor podría ocultar, perfectamente, la relación entre la nota y las palabras clave de los exámenes
- La tercera se basa en considerar que el método trabajado no permite obtener información suficiente por sí mismo a partir de palabras clave. Es decir, al analizar únicamente determinadas palabras se podría estar perdiendo la esencia de la respuesta, que sería aquello que más valora el profesor. La forma del escrito, la relación de

conceptos y el uso de lenguaje descriptivo son recursos muy importantes a la hora de evaluar y que pasan completamente desapercibidos con esta aproximación. De esta manera, podría no existir una relación directa con una palabra clave como tal pero sí con su significado o con lo que representa en el temario. La búsqueda de palabras clave, por tanto, debería complementarse, pues el listado de *keywords* se contemplaría como un conjunto de índices abstractos que el estudiante no debería citar como tal. Aquello que debería hacer sería explicar su significado con otras palabras o expresar esa idea aplicada a otro punto en concreto del dominio.

- La última, se basaría en la combinación de todas o algunas de las hipótesis mencionadas.

Sea como fuere, la inexistencia de relación entre métodos de evaluación no significa que el programa realizado sea incorrecto. De hecho, los resultados obtenidos de la extracción de palabras clave son muy positivos. Esto significa que los objetivos que componen el trabajo, desde la construcción de los diccionarios hasta la retroalimentación y estrategia de comparación de palabras, han sido alcanzados (en mayor o menor medida).

Gran parte de este éxito se encuentra relacionado con el funcionamiento del generador de *feedback*. Esto se debe a que la identificación de *keywords* es muy robusta, consiguiendo reconocerlos en sus múltiples formas gramaticales debido a la eficiencia del etiquetador morfosintáctico y a la estrategia de lematización implementada. Así, se consigue particularizar el *feedback* a cada situación, de forma que las sentencias creadas son tan claras, precisas y sencillas, que consiguen informar sin agobiar al lector, lo que las hace idóneas para ser consultadas rápidamente durante la realización de un examen o tarea.

Por último, en relación al etiquetador, se ha comprobado que su eficiencia y adaptabilidad a palabras nuevas es bastante alta. Es cierto que los "*tokenseliminados*", en ocasiones, no se gestionan correctamente. Sin embargo, en este trabajo únicamente interesa desambiguar las palabras clave. De esta forma, se asumirá un cierto error en palabras complementarias siempre que se garantice, como es el caso, el correcto etiquetado de las palabras recogidas en el diccionario. Se observa, por otra parte, como apenas aparecen antónimos, no porque no se logren identificar, sino porque probablemente los estudiantes prefieran utilizar otras unidades de significado más próximo al buscado.

En resumen, tras analizar los resultados del presente trabajo se consigue llegar a las siguientes conclusiones:

- El etiquetador morfosintáctico realizado funciona adecuadamente y consigue desambiguar las situaciones en las que una misma palabra se puede expresar con varias categorías gramaticales.
- Se ha conseguido elaborar un procedimiento para crear varios instrumentos de control terminológico con los que recoger, de forma adaptativa, los sinónimos y antónimos de las palabras clave del examen que se desee procesar.
- La estrategia de identificación de palabras elaborada permite la correcta extracción de los *keywords* de la respuesta del alumno.
- El *feedback* generado es suficientemente claro y particular como para permitir entender el motivo por el que una palabra clave ha sido considerada o no.
- El método de evaluación diseñado no es, en este caso, adecuado para proporcionar información útil sobre la nota del alumno pero sí lo es como posible instrumento indicativo de la falta de un mínimo de nivel en la respuesta.

Asimismo, se considera que las siguientes tareas podrían ser interesantes de cara a ampliar, en un futuro, el trabajo presentado en este documento:

- Mejorar la función de segmentación de las oraciones en frases. El análisis sintáctico o morfosintáctico tiene la limitación que, cuanto mayor es la frase a analizar, mayor probabilidad hay de que no se ejecute correctamente. A pesar de haber incluido técnicas para acortar las oraciones, podría ser interesante mejorar el divisor para que segmente oraciones subordinadas o frases muy largas que no incluyan puntos ni comas.
- Ampliar el corpus para integrar no sólo más expresiones de habla española, sino también de América latina u otras variantes del español.
- Mejorar la aproximación que seguir en caso de encontrar palabras no existentes en el diccionario. En este trabajo, dada la forma en la que se creó el etiquetador morfosintáctico, se condicionó la categoría de estas palabras a sustantivo/adjetivo.

Sin embargo, se podrían implementar otro tipo de técnicas que podrían ser más precisas. Por ejemplo, se podría calcular la distancia de levenshtein (que es una técnica adecuada para la lengua española) entre la palabra desconocida y el conjunto de palabras del diccionario, asignando el significado de la palabra más próxima.

- Incorporar un sistema, basado en PLN, capaz de interpretar el *feedback* conseguido en este trabajo e interactuar con el estudiante/profesor para comunicarle la información de manera natural y poder resolver todas las cuestiones que se puedan suscitar.
- Valorar otras opciones de extracción de significado del texto para no depender únicamente de la aparición exacta de los términos o sus sinónimos/antónimos
- Establecer técnicas más objetivas para la detección y comparación del significado de los antónimos negados con el significado real de la palabra clave en cuestión.
- Analizar los casos en los que convenga ampliar los niveles de relación de sinonimia y antonimia y operar con éstos utilizando técnicas de WSD, independientemente de que sean palabras simples o compuestas.
- Ampliar la tarea de WSD con la incorporación de otras técnicas que complementen la desambiguación que puede ofrecer un etiquetador morfosintáctico.
- Estudiar con más ejemplos si existe, en alguno de éstos, relación entre el criterio de corrección del profesor y la identificación de un listado de palabras consideradas como clave.

Referencias

Adimen. (s.f.). *Mcr using wordnet 3.0 as ili consulted using the web eurowordnet interface (consult mode).* Recuperado de: https://adimen.si.ehu.es/web/MCR/ el día 21/09/2021.

Ambróz, N. (2019). *Natural language processing: A short introduction to get you started.* Recuperado de: https://aliz.ai/natural-language-processing-a-short-introduction-to-get-you-started/ el día 21/09/2021. Aliz.

Amores, M., Arco, L., Nadal, A., y Santana, K. (2016). Desambiguación del sentido de las palabras: una aproximación no supervisada. *XVI Convención y feria internacional*, Cuba.

ASPgems. (2019). *Metodología de desarrollo de software (iii) - modelo en espiral.* Recuperado de: https://aspgems.com/metodologia-de-desarrollo-de-software-iii-modelo-en-espiral/ el día 21/09/2021.

Brame, C. J. (2019). Rubrics: Tools to make grading more fair and efficient. *Science Teaching Essentials, Academic Press*, 175–184.

Brill, E. (1995). Transformation–based Error–driven Learning and Natural Language Processing: A Case Study in Part–of–speech Tagging. *Computational Linguistics*, *21*(4), 543–565.

Chen, B., Chang, Y.-H., Ouyang, F., y Zhou, W. (2018). Fostering student engagement in online discussion through social learning analytics. *Internet High. Educ.*, *37*, 21-30. doi: https://doi.org/10.1016/j.iheduc.2017.12.002

Cárdenas, V. (2010). La relación entre semántica y sintaxis desde la perspectiva de la producción de lenguaje escrito. *Tópicos del Seminario*, 241 - 289. Descargado de http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1665-12002010000100008&lng=es&tlng=es

Dascalu, M. (2014). *Readerbench (1) - cohesion-based discourse analysis and dialogism*

(Vol. 534).

Deline, G., Lin, F., Wen, D., Gaševic, D., y Kinshu. (2010). A case study of ontology-driven development of intelligent educational systems. *International Journal of Web-Based Learning and Teaching Technologies*, *4*(1), 66-81. doi: 10.4018/jwltt.2009010105

Diamantini, C., Mircoli, A., Potena, D., y Storti, E. (2015). Desambiguación semántica en un sistema de descubrimiento de información social. *Conferencia internacional de 2015 sobre tecnologías y sistemas de colaboración (CTS)*, 326–333.

Elkan, C., y Greiner, R. (1993). Building large knowledge-based systems: Representation and inference in the cyc project: D.b. lenat and r.v. guha. *Artificial Intelligence*, *61*(1), 41-52. doi: https://doi.org/10.1016/0004-3702(93)90092-P

Fellbaum, C. (1997). Análisis de una tarea de escritura a mano. *Proc. del taller ANLP-97 sobre etiquetado de texto con semántica léxica: ¿por qué, qué y cómo?*.

Frost, R., y McCray, A. (2012). Markov chain ontology analysis (mcoa). *BMC Bioinformatics*, *13, 23*. doi: https://doi.org/10.1186/1471-2105-13-23

Galo, R. (2011). Modelo espiral de un proyecto de desarrollo de software. UNEMI (Universidad estatal de Milagro).

Gil, I., y Rodríguez, J. V. (1996). El procesamiento del lenguaje natural aplicado al análisis del contenido de los documentos. *Revista General de Información y Documentación*, *6*(2), 205-218.

Graesser, A. C., McNamara, D. S., y Kulikowich, J. M. (2011). Coh-metrix. *Educ. Res.*, *40*(5), 223–234. doi: https://doi.org/10.3102/0013189X11413260

Hernández-Lara, A. B., Perera-Lluna, A., y Serradell-López, E. (2019). Applying learning analytics to students' interaction in business simulation games. the usefulness of learning analytics to know what students really learn. *Comput. Human Behav.*, *92*, 600–612. doi: 10.1016/j.chb.2018.03.001

Jurafsky, D., y Martin, J. (2008). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (Vol. 2).

Kalita, P., y Barman, A. K. (2015). Word sense disambiguation: A survey. *J. Control Theory Comput. Model*, *4*(5), 11743–11748.

Karlsson, F., Voutilainen, A., Heikkilä, J., y Anttila, A. (1995). Constraint grammar: a language-independent system for parsing unrestricted text. *Mouton de Gruyter*. Descargado de https://www.researchgate.net/publication/230876180_Constraint_Grammar_A_Language-Independent_System_for

_Parsing_Unrestricted_Text

Leacock, C., y Chodorow, M. (2003). C-rater: Automated scoring of short-answer questions. *Comput. Hum.*, *37*(4), 389–405. Descargado de https://doi.org/10.1023/A:1025779619903

Lesk, M. (1986). Desambiguación del sentido de las palabras: una aproximación no supervisada. *En Proc. de SIGDOC-86: 5ª Conferencia Internacional sobre Documentación de Sistemas*, Toronto, Canadá.

Maldonado, L. F., Londoño, O. L., y Gómez, J. P. (2017). Sistemas ontológicos en el aprendizaje significativo: estado del arte. *Actualidades Investigativas en Educación*, *17*(2), 1-18.

McNamara, D. S., Crossley, S. A., Roscoe, R. D., Allen, L. K., y Dai, J. (2015). A hierarchical classification approach to automated essay scoring. *Assess. Writ.*, *23*, 35–59. doi: http://dx.doi.org/10.1016/j.asw.2014.09.002

McNamara, D. S., Levinstein, I. B., y Boonthum, C. (2004). istart: Interactive strategy training for active reading and thinking. *Behav. Res. Methods, Instruments, Comput.*, *36*(2), 222–233. doi: 10.3758/bf03195567

Nye, B. D., Graesser, A. C., y Hu, X. (2014). Autotutor and family: A review of 17 years of natural language tutoring. *Int. J. Artif. Intell. Educ.*, *24*(4), 427-469. doi: https://doi.org/10.1007/s40593-014-0029-5

Panaite, M., Dascalu, M., Johnson, A., Balyan, R., Dai, J., McNamara, D., y Trausan-Matu., S. (2018). Bring it on! challenges encountered while building a comprehensive tutoring system using readerbench. *Springer, Cham*, 409–419. doi: 10.1007/978-3-319-93843-1_30

Ramineni, C. (2013). Automated essay scoring: Psychometric guidelines and practices. *Assess. Writ.*, *18*(1), 25–39. doi: https://www.learntechlib.org/p/92345/

Real Academia Española, R. (s.f.). *Diccionario de la lengua española.* Recuperado de: https://dle.rae.es/diccionario/ el día 21/09/2021.

Reese, S., Boleda, G., Cuadros, M., Padró, L., y Rigau, G. (2010). Wikicorpus: un corpus de wikipedia multilingüe sin ambigüedades del sentido de la palabra. *En Actas de la Séptima Conferencia de Evaluación y Recursos Lingüísticos (LREC'10).*

Rodrigues, F., y Oliveira, P. (2014). A system for formative assessment and monitoring of students' progress. *Comput. Educ.*, *76*(4), 30-41. Descargado de https://www.learntechlib.org/p/201601/.

Rodríguez, H. (2007). Técnicas básicas en el tratamiento informático de la lengua. *Quark: Ciencia, medicina, comunicación y cultura*, *2000*(19).

Rus, V., D'Mello, S., Hu, X., y Graesser, A. (2013). Recent advances in conversational intelligent tutoring systems. *AI Mag.*, *34*(3), 42-54. doi: https://doi.org/10.1609/aimag.v34i3.2485

Schütze, H. (1998). Discriminación automática del sentido de las palabras. *Lingüística computacional*, *24*(1), 97-123.

Snyder, B., y Palmer, M. (2004). La tarea de todas las palabras en inglés. *En Proc. del 3er Taller Internacional de Evaluación de Sistemas para el Análisis Semántico de Texto (Senseval-3).*

Torrijos, C. (2020). *Procesamiento del lenguaje natural en el ámbito biomédico.* Recuperado de: https://www.iic.uam.es/lasalud/procesamiento-del-lenguaje-natural-ambito-biomedico/ el día 21/09/2021. Instituto de ingeniería del conocimiento.

Vilches-Blázquez, B. (2019). Construcción de ontologías a partir de tesauros. *Semántica Espacial y descubrimiento de conocimientos para desarrollo sostenible*, 59–78.

Westera, W., Dascalu, M., Kurvers, H., Ruseti, S., y Trausan-Matu, S. (2018). Automated essay scoring in applied games: Reducing the teacher bandwidth problem in online training. *Comput. Educ.*, *123*, 212–224. doi: https://doi.org/10.1016/j.compedu.2018.05.010

WordReference. (s.f.). *Diccionario de sinónimos y antónimos.* Recuperado de: https://www.wordreference.com/sinonimos/ el día 21/09/2021.

Xie, K., Tosto, G. D., Lu, L., y Cho, Y. S. (2018). Detecting leadership in peer-moderated online collaborative learning through text mining and social network analysis. *Internet High. Educ.*, *38*, 9-17. doi: https://doi.org/10.1016/j.iheduc.2018.04.002

Zhou, L. (2007). Ontology learning: state of the art and open issues. *Information Technology and Management*, *8*(3), 241-252. doi: 10.1007/s10799-007-0019-5

Zouaq, A., Gasevic, D., y Hatalar, M. (2011). Towards open ontology learning and filtering. *Inf. Syst.*, *36*(7), 1064–1081. doi: 10.1016/j.is.2011.03.005

Apéndice I

Anexos

I. Formato del fichero con la respuesta del alumno

```
{
"respuesta": "La arquitectura Replica-Set es la respuesta de MongoDB
    para dotar al sistema de una alta disponibilidad y robustez
   frente a posibles fallos en el sistema hardware. Este permite
   llevar a cabo una replicación en tiempo real de las bases de
   datos del sistema. \nLa arquitectura básica mínima se compone
   de tres nodos, un nodo primario otro secundario y un tercero
   que desarrollar las funciones de arbitro a la hora de
   seleccionar el nodo primario. El sistema de elección del nodo
   primario no es un sistema estático, sino que en función de la
   disponibilidad y estado de los nodos se procederá a la
   selección del mismo.\nLas instancias creadas sobre el nodo
   primario se replican sobre el nodo secundario. Esto implica que
    el sistema no es robusto frente a escrituras erróneas en la
   base de datos, aunque existe la posibilidad de activar un
   retraso en el proceso de replicación entre el nodo primario y
   secundario. \nPor último, la principal diferencia con otras
   aplicaciones como mongodump, es que mongodump se utiliza para
   realizar copias de seguridad offline, es decir bajo petición o
   programación previa cuando la arquitectura replica-set es de
   alta disponibilidad gracias a replicación en tiempo real entre
   nodos",
"metadata": {
```

```
4         "enunciado": "Describe, en menos de 200 palabras, en qué
              consiste una arquitectura en Replica-Set en MongoDB. Tu
              respuesta, que debe ser una redacción y no un listado, debe
               contener la respuesta a las siguientes preguntas: ¿cuáles
              son es la función de un replica-set?, ¿cuántos nodos son
              necesarios para una configuración mínima?, ¿qué ocurre
              cuando el nodo primario deja de tener disponibilidad?,
              ¿cuál es la principal diferencia con mongodump?",
5         "minipreguntas": [
6                 {
7                         "minipregunta": "¿cuáles son es la función de un
                              replica-set?",
8                         "minirespuesta": "La principal función de un
                              replica-set es la de ofrecer respaldo de
                              los datos. Además, aporta alta
                              disponibilidad y posibilita el balanceo de
                              carga."
9                 },
10                {
11                        "minipregunta": "¿cuántos nodos son necesarios
                              para una configuración mínima?",
12                        "minirespuesta": "La configuración mínima es de
                              tres servidores: un primario, un secundario
                               normal y un secundario de tipo árbitro."
13                },
14                {
15                        "minipregunta": "¿qué ocurre cuando el nodo
                              primario deja de tener disponibilidad?",
16                        "minirespuesta": "Cuando el nodo primario deja de
                               tener disponibilidad, los nodos
                              secundarios dejan de recibir el heartbeat o
                               latido. Entonces, inician una votación
                              para elegir un nuevo nodo primario. Tas la
                              elección, el nodo elegido asume
                              inmediatamente las funciones del primario.
                              En caso de reconexión, el nodo que antes
                              era primario toma el rol de secundario."
```

```
                },
                {
                        "minipregunta": "¿cuál es la principal diferencia
                             con mongodump?",
                        "minirespuesta": "La principal diferencia de
                            replica-set con mongodump es que replica-
                            set hace una copia instantánea y
                            actualizada de los datos, mientras que
                            mongodump exporta la base de datos en un
                            momento concreto y no captura las nuevas
                            actualizaciones."
                }
        ],
        "keywords": [
                "respaldo",
                "balanceo de carga",
                "disponibilidad",
                "replicación",
                "arbitro",
                "votación",
                "sustitución",
                "exporta",
                "actualizada"
        ]
    },
    "hashed_id": "1056c39c8300ac26304ad7463f5ea8ce",
    "nota": 1
    }
```

II. Gráficas y datos obtenidos

II.I. Relación entre la nota calculada y la real

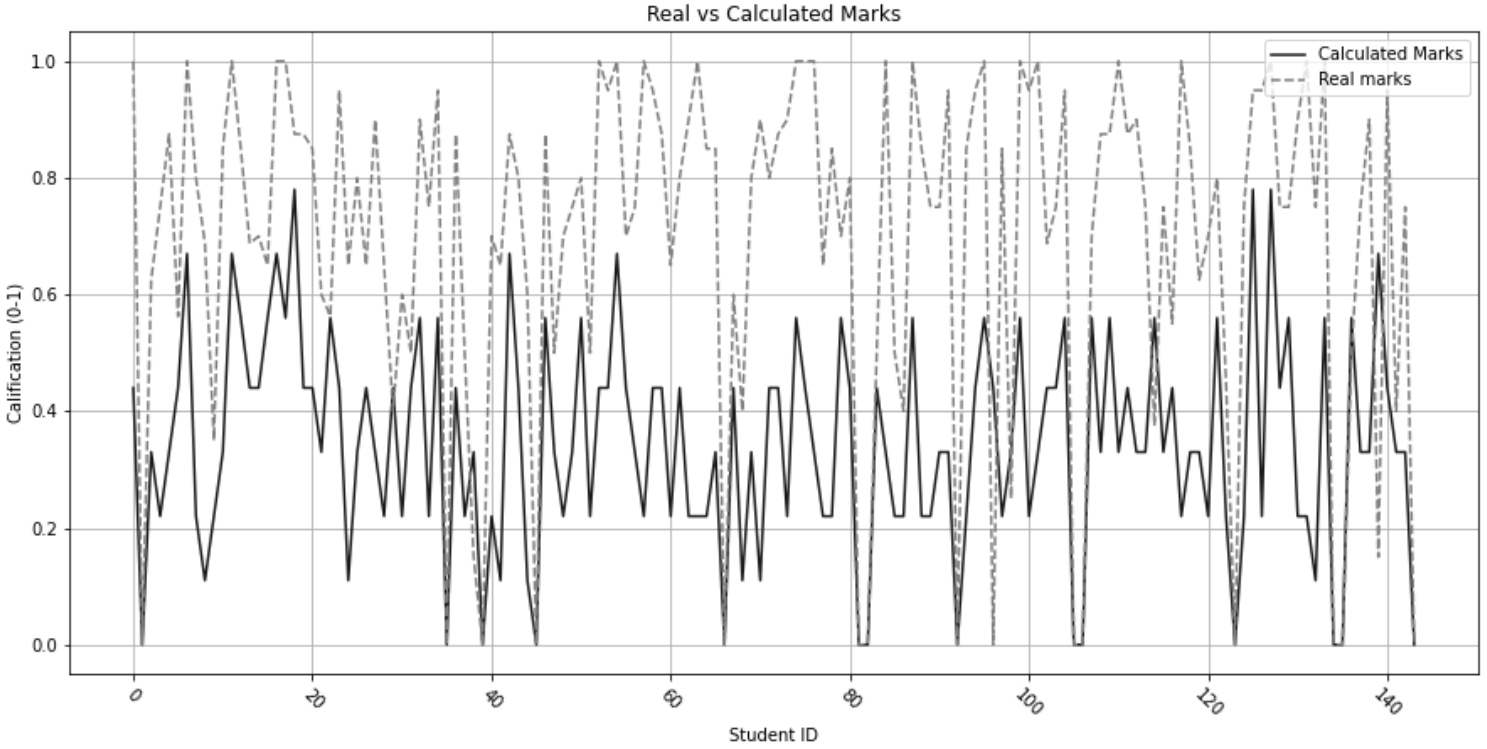


Figura I.1: Relación entre la nota calculada y la nota real de los estudiantes

Como se puede observar, la diferencia entre las calificaciones obtenidas es bastante considerable. No obstante, a primera vista parece posible la existencia de una mínima relación entre ambas, pues en muchos casos cuando la nota A de un estudiante incrementa con respecto al anterior, sucede lo mismo con la nota B, aunque no con la misma pendiente. Este fenómeno también se produce cuando la nota A decrementa. Por ejemplo, en la Figura I.1, el comportamiento de ambas describe una línea similar entre los estudiantes 36 y 55.

Sin embargo, existen demasiados puntos de discordancia entre ambas señales como para justificar la existencia de una relación real entre ambas. Además, dada la cantidad existente de dichos puntos, se descarta la posibilidad de que pudieran ser valores anómalos con los que poder defender esta teoría.

II.II. Número de *keywords* identificados por nota

	respaldo	balanceo de carga	disponibilidad	replicación	arbitrar	votación	sustitución	exportar	actualizar
0.0	1	0	1	0	1	0	0	0	1
0.15	1	0	2	0	1	1	1	2	1
0.25	0	0	1	1	0	0	0	1	0
0.35	1	0	1	0	0	0	0	0	0
0.375	1	0	1	1	1	0	0	1	0
0.4	2	0	3	2	2	0	0	0	1
0.45	0	0	1	1	0	0	0	0	0
0.5	3	0	6	1	5	3	1	1	2
0.55	1	0	1	1	1	0	0	0	0
0.5625	1	0	2	2	1	1	1	0	1
0.6	2	0	4	1	1	0	1	1	0
0.625	1	0	2	1	1	0	1	0	0
0.65	1	0	7	3	3	1	0	1	1
0.6875	0	0	2	2	2	0	0	1	2
0.7	3	0	6	2	2	3	1	3	4
0.75	5	1	12	4	9	4	0	2	6
0.8	7	0	8	2	7	1	3	2	4
0.85	3	0	10	2	6	0	2	3	2
0.875	7	2	11	6	11	3	1	5	3
0.9	3	0	7	1	4	0	1	1	4
0.95	7	1	11	4	10	2	1	2	6
1.0	8	4	21	8	16	11	5	6	14

Tabla I.1: Número de identificaciones de cada *keyword* por cada nota

II.III. Error Absoluto entre las notas

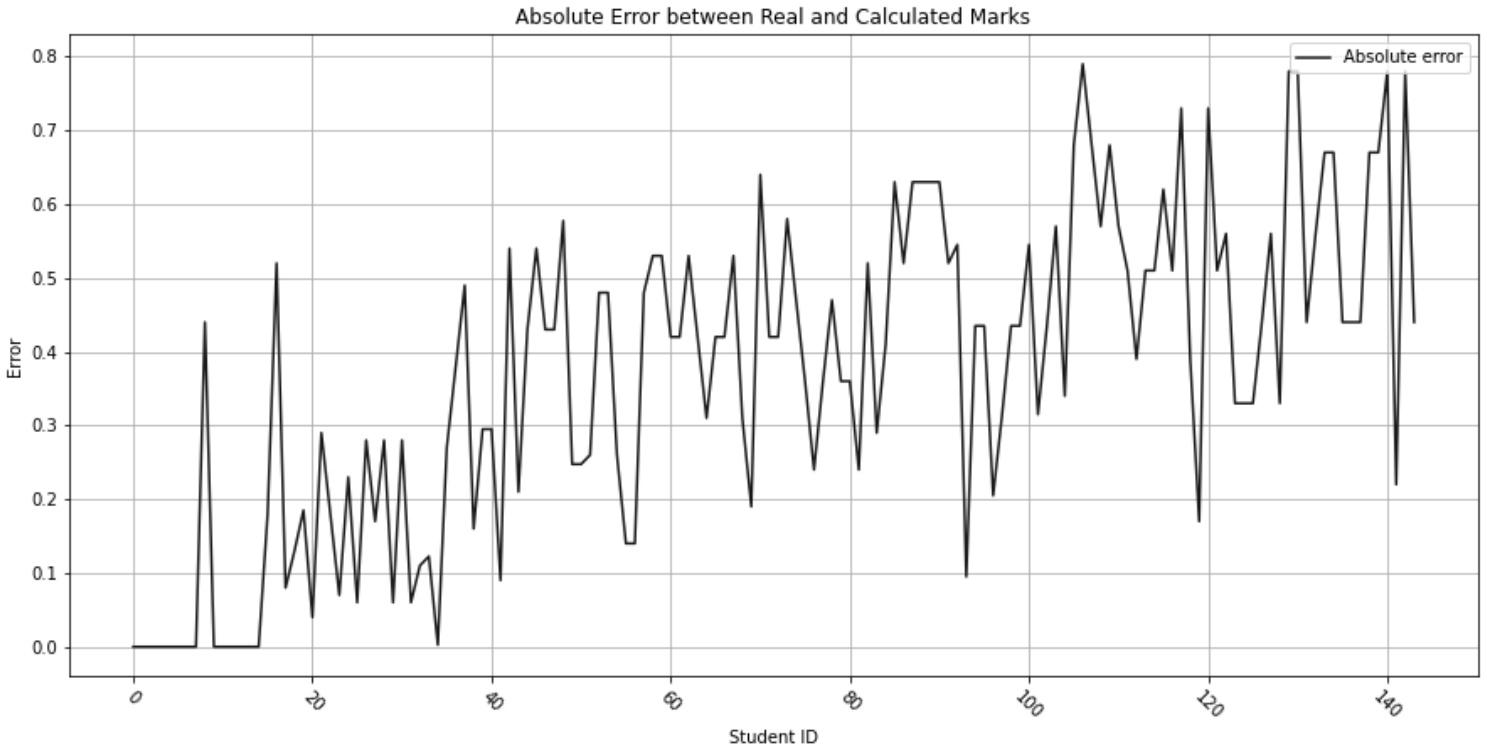


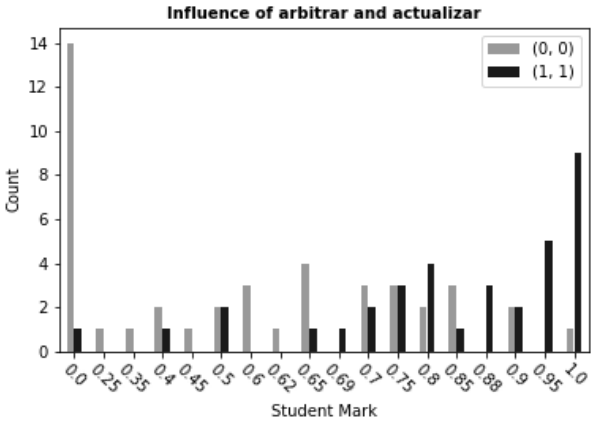
Figura I.2: Error Absoluto entre las dos notas de los estudiantes

II.IV. Influencia conjunta de los *keywords* en la nota

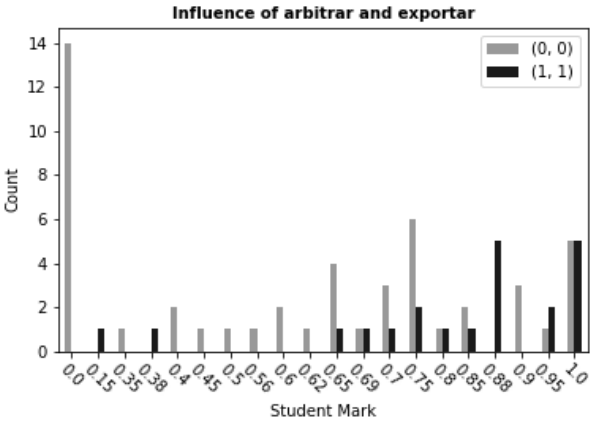
Palabra clave 1	Palabra clave 2	Coeficiente de Pearson
balanceo de carga	respaldo	0.10989104295939639
disponibilidad	respaldo	0.1393074864530851
replicación	respaldo	0.05727652847757037
arbitrar	respaldo	0.2919863748102095
votación	respaldo	0.13655349765454547
sustitución	respaldo	0.014526967835589932
exportar	respaldo	0.0718995811991051
actualizar	respaldo	-0.08679746646064457
disponibilidad	balanceo de carga	0.10846522890932807
replicación	balanceo de carga	0.032703497008386456
arbitrar	balanceo de carga	0.02049800154226967
votación	balanceo de carga	0.32348719116573565
sustitución	balanceo de carga	0.0846043423048847
exportar	balanceo de carga	0.01620509308880412
actualizar	balanceo de carga	0.13324887481766265
replicación	disponibilidad	0.14070529413628968
arbitrar	disponibilidad	0.3401680257083045
votación	disponibilidad	0.18353258709644937
sustitución	disponibilidad	0.11929618161269212
exportar	disponibilidad	0.14940357616679922
actualizar	disponibilidad	0.21983575504754546
arbitrar	replicación	0.11396057645963795
votación	replicación	0.13373146301410777
sustitución	replicación	0.0027668578554642907
exportar	replicación	0.03603749850782237
actualizar	replicación	-0.007797996951841548
votación	arbitrar	0.017342199390482396
sustitución	arbitrar	-0.12833227548956977
exportar	arbitrar	0.07905694150420951

Palabra clave 1	Palabra clave 2	Coeficiente de Pearson
actualizar	arbitrar	0.13685429109080416
sustitución	votación	0.15368421052631576
exportar	votación	0.09597148699373931
actualizar	votación	0.21953528232743563
exportar	sustitución	0.03838859479749572
actualizar	sustitución	0.13409452380000125
actualizar	exportar	0.05023237925348388

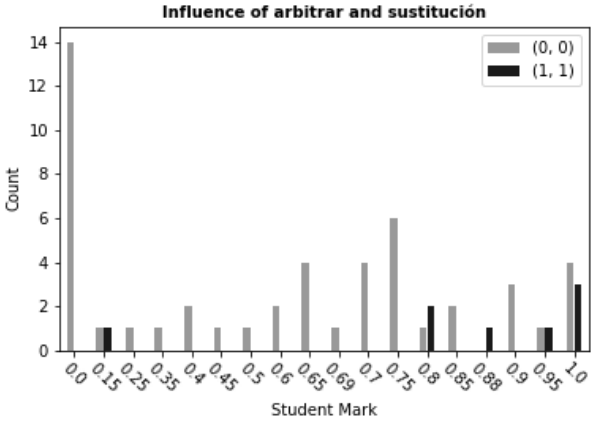
Tabla I.2: Coeficiente de correlación de Pearson para cada par de *keywords*



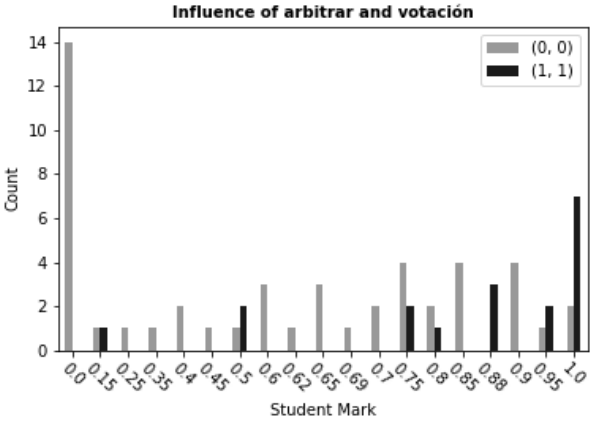
(1) Influencia de la aparición conjunta de los *keywords* arbitrar y actualizar



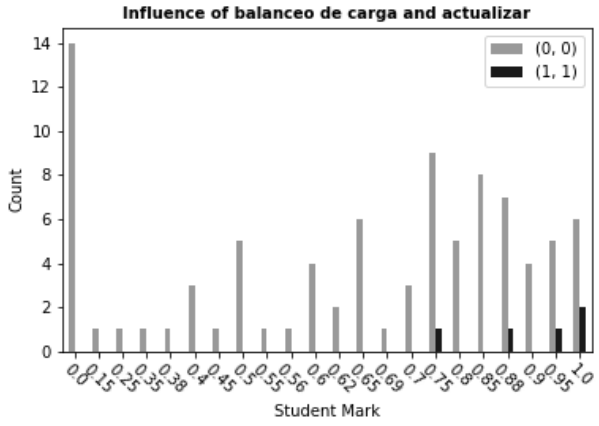
(2) Influencia de la aparición conjunta de los *keywords* arbitrar y exportar



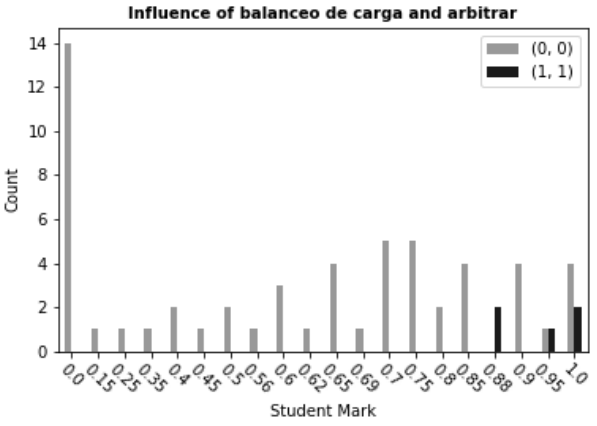
(3) Influencia de la aparición conjunta de los *keywords* arbitrar y sustitución



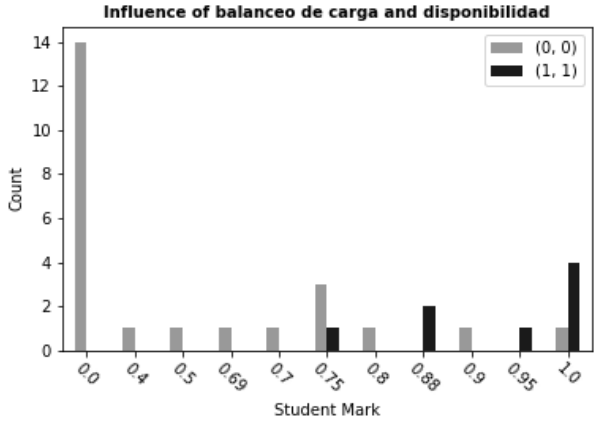
(4) Influencia de la aparición conjunta de los *keywords* arbitrar y votación



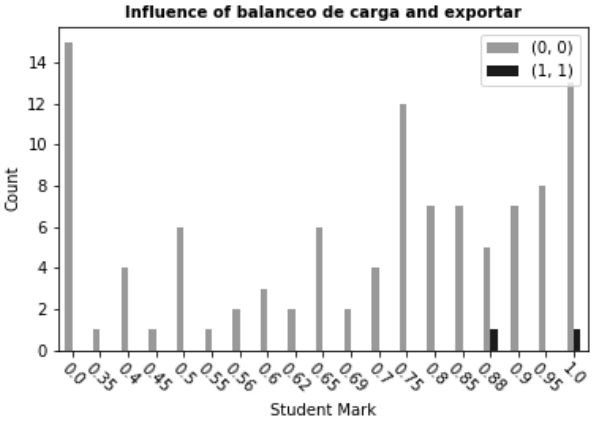
(5) Influencia de la aparición conjunta de los *keywords* balanceo de carga y actualizar



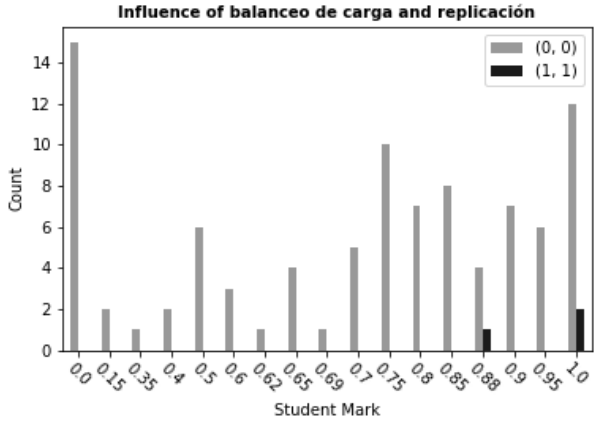
(6) Influencia de la aparición conjunta de los *keywords* balanceo de carga y arbitrar



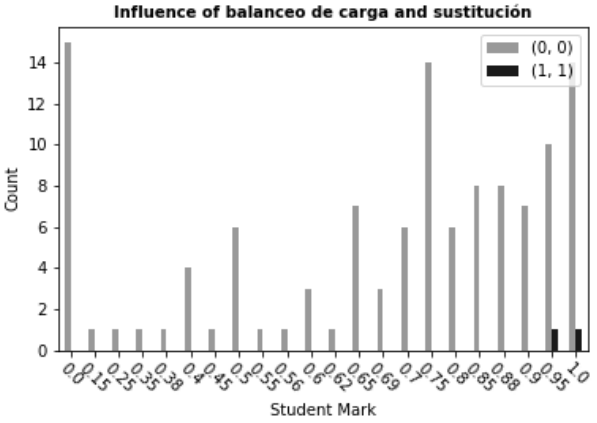
(7) Influencia de la aparición conjunta de los *keywords* balanceo de carga y disponibilidad



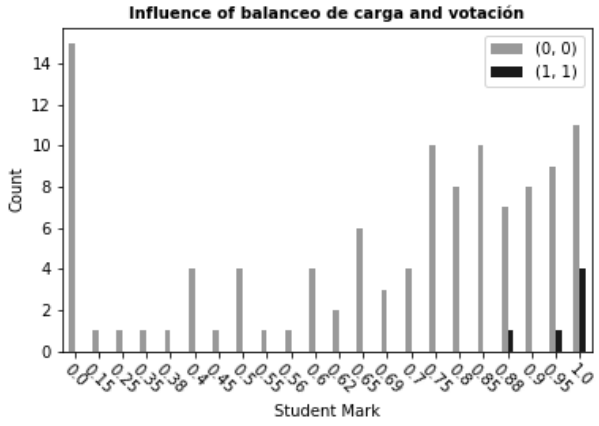
(8) Influencia de la aparición conjunta de los *keywords* balanceo de carga y exportar



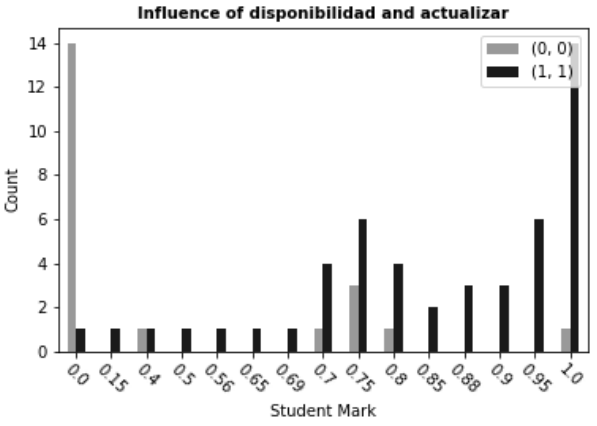
(9) Influencia de la aparición conjunta de los *keywords* balanceo de carga y replicación



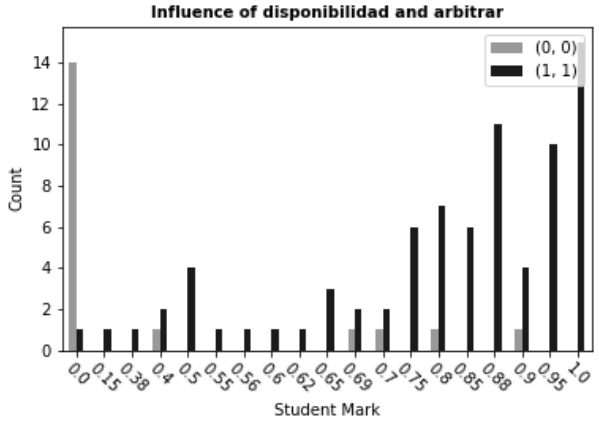
(10) Influencia de la aparición conjunta de los *keywords* balanceo de carga y sustitución



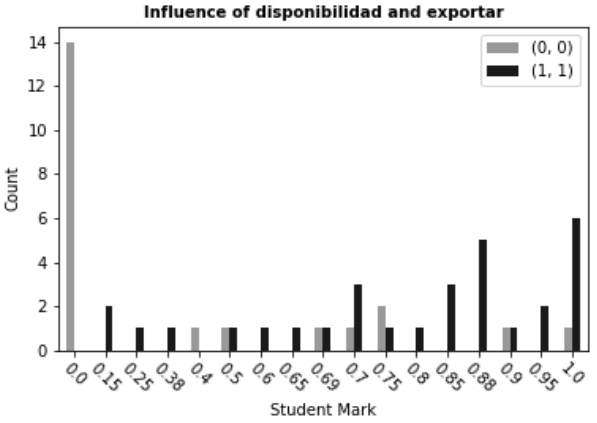
(11) Influencia de la aparición conjunta de los *keywords* balanceo de carga y votación



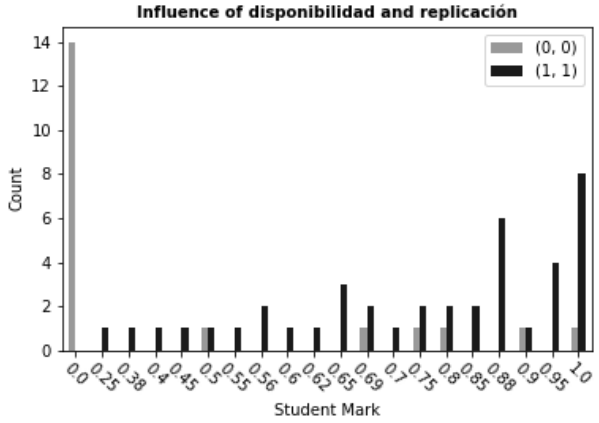
(12) Influencia de la aparición conjunta de los *keywords* disponibilidad y actualizar



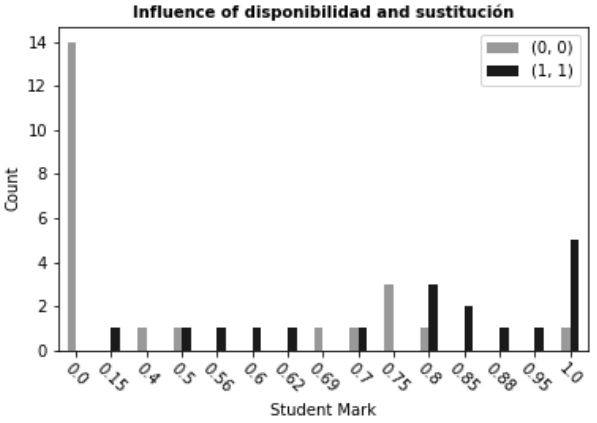
(13) Influencia de la aparición conjunta de los *keywords* disponibilidad y arbitrar



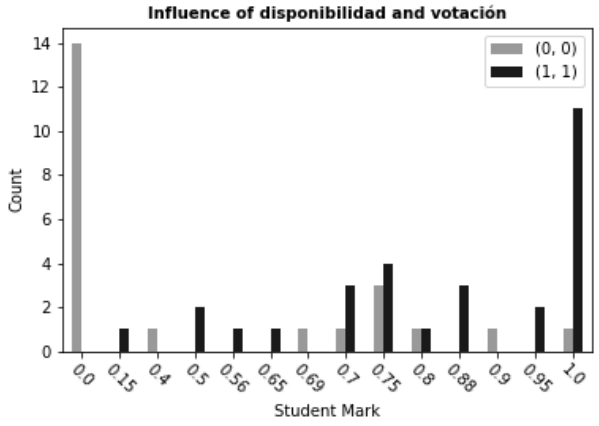
(14) Influencia de la aparición conjunta de los *keywords* disponibilidad y exportar



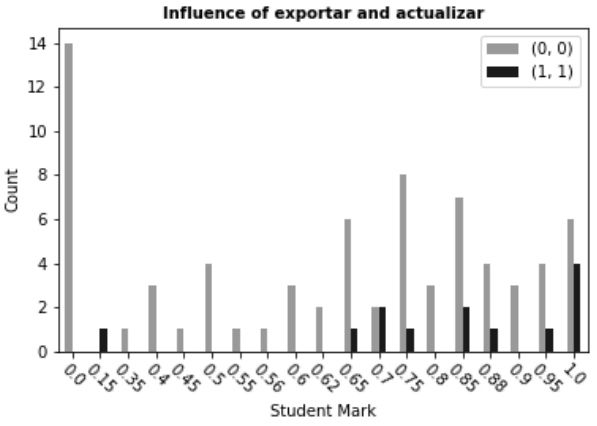
(15) Influencia de la aparición conjunta de los *keywords* disponibilidad y replicación



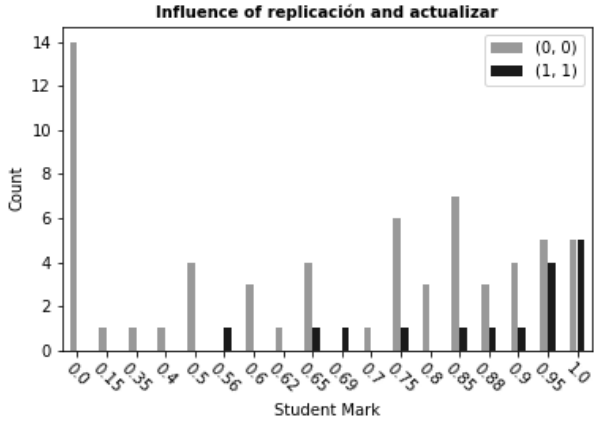
(16) Influencia de la aparición conjunta de los *keywords* disponibilidad y sustitución



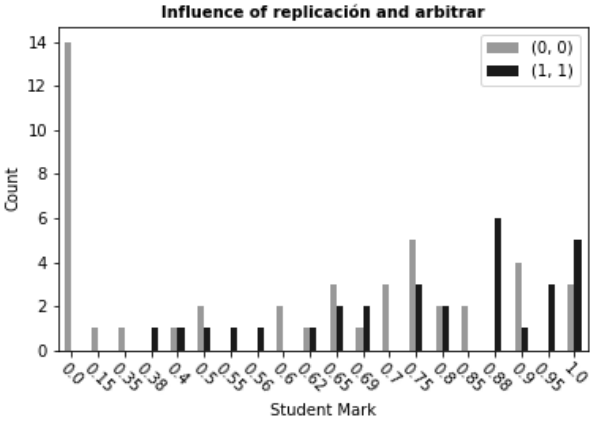
(17) Influencia de la aparición conjunta de los *keywords* disponibilidad y votación



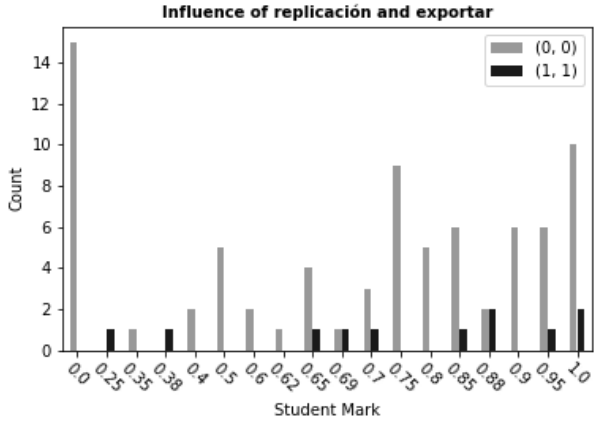
(18) Influencia de la aparición conjunta de los *keywords* exportar y actualizar



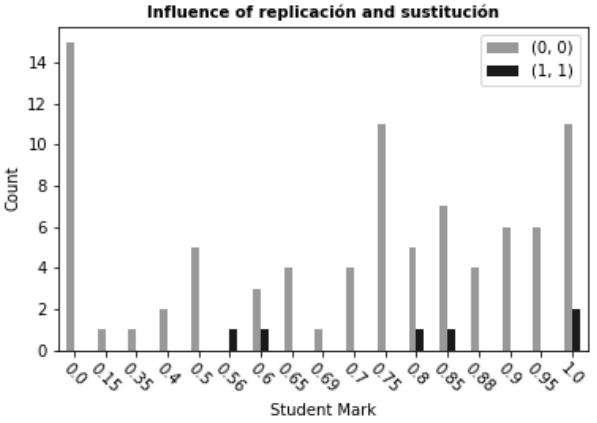
(19) Influencia de la aparición conjunta de los *keywords* exportar y actualizar



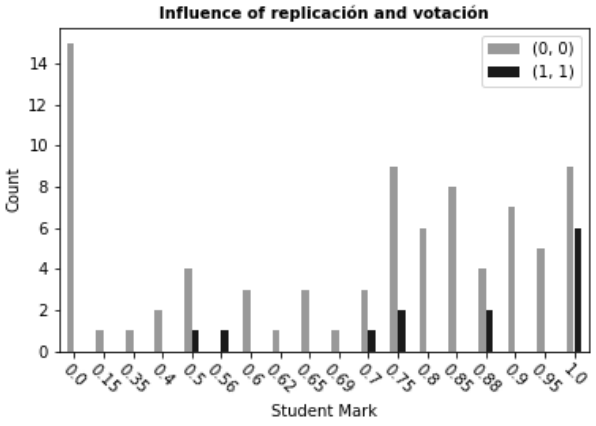
(20) Influencia de la aparición conjunta de los *keywords* replicación y arbitrar



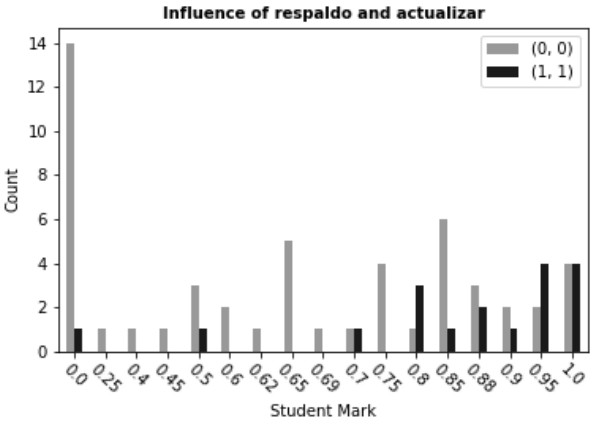
(21) Influencia de la aparición conjunta de los *keywords* replicación y exportar



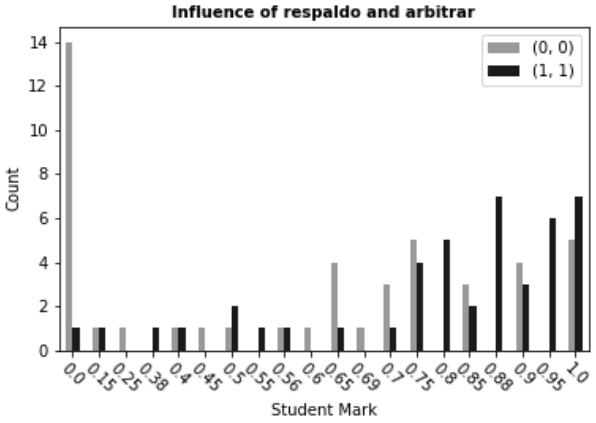
(22) Influencia de la aparición conjunta de los *keywords* replicación y sustitución



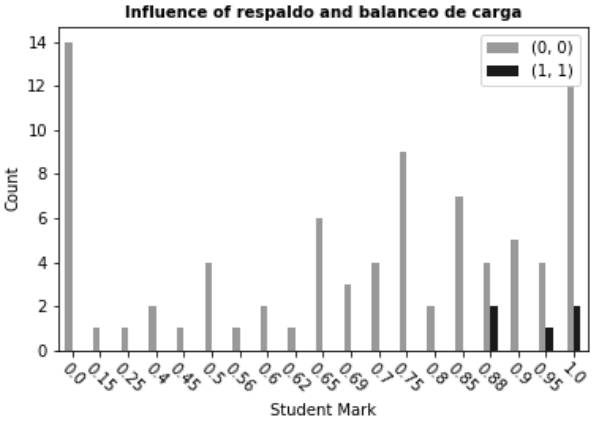
(23) Influencia de la aparición conjunta de los *keywords* replicación y votación



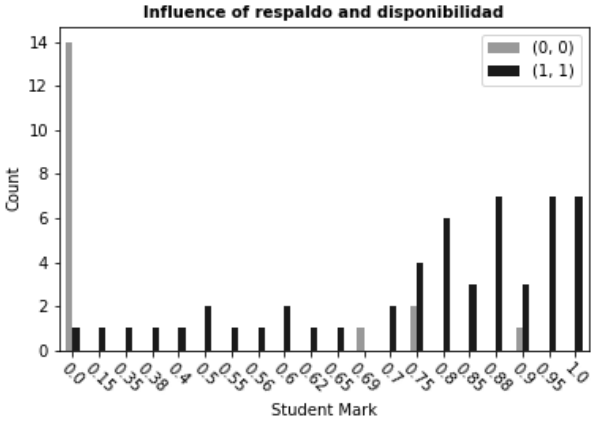
(24) Influencia de la aparición conjunta de los *keywords* respaldo y actualizar



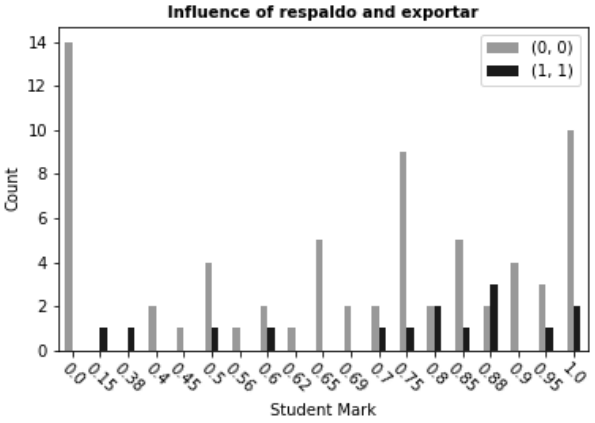
(25) Influencia de la aparición conjunta de los *keywords* respaldo y arbitrar



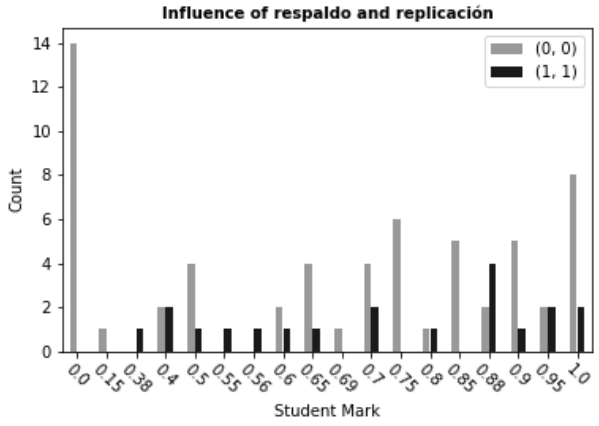
(26) Influencia de la aparición conjunta de los *keywords* respaldo y balanceo de carga



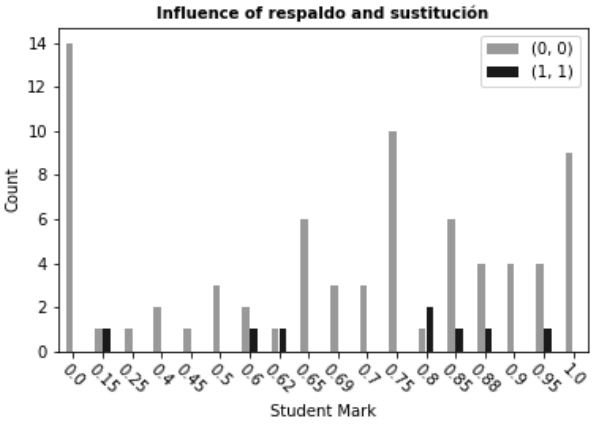
(27) Influencia de la aparición conjunta de los *keywords* respaldo y disponibilidad



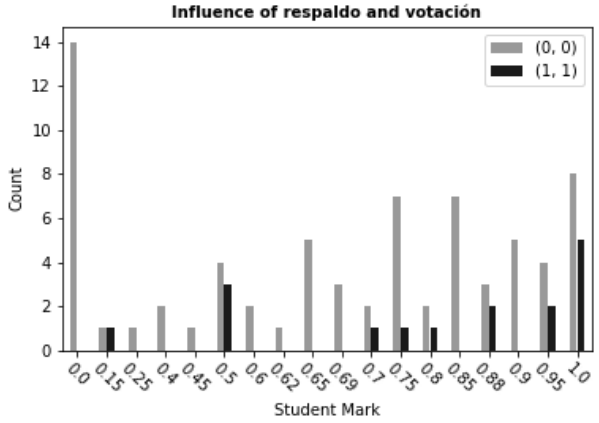
(28) Influencia de la aparición conjunta de los *keywords* respaldo y exportar



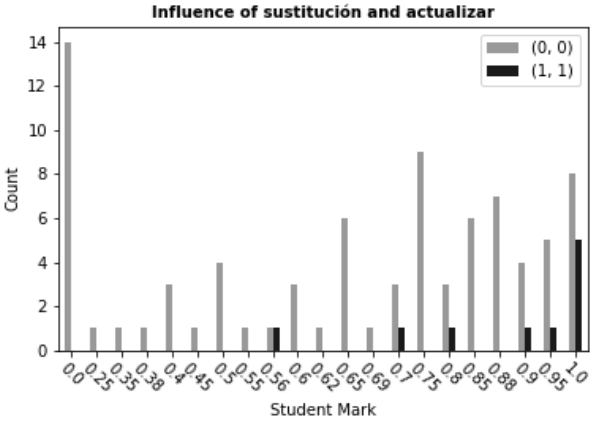
(29) Influencia de la aparición conjunta de los *keywords* respaldo y replicación



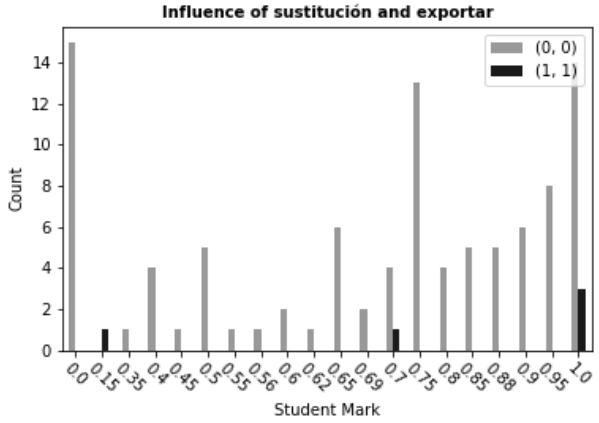
(30) Influencia de la aparición conjunta de los *keywords* respaldo y sustitución



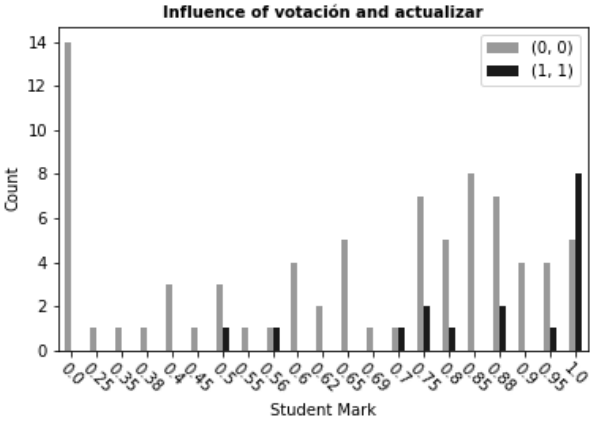
(31) Influencia de la aparición conjunta de los *keywords* respaldo y votación



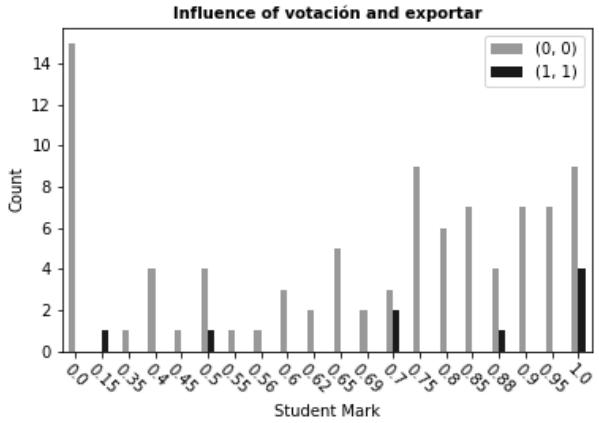
(32) Influencia de la aparición conjunta de los *keywords* sustitución y actualizar



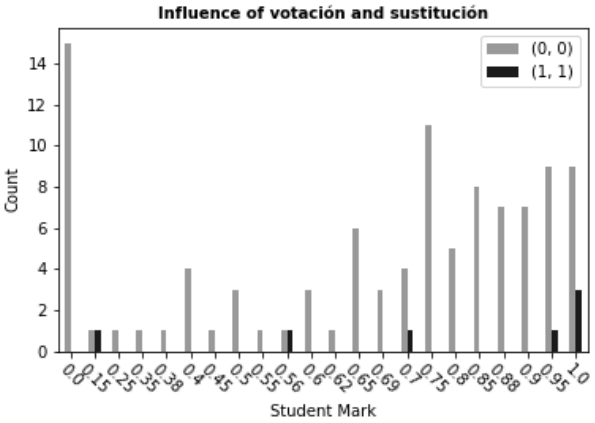
(33) Influencia de la aparición conjunta de los *keywords* sustitución y exportar



(34) Influencia de la aparición conjunta de los *keywords* votación y actualizar



(35) Influencia de la aparición conjunta de los *keywords* votación y exportar



(36) Influencia de la aparición conjunta de los *keywords* votación y sustitución

Figura I.20: Gráficas para mostrar la influencia de aparición conjunta de los *keywords*

III. Archivos *Jupyter Notebook*

Los archivos de programa diseñados y generados para este proyecto se encuentran disponibles en *GitHub*, a partir del siguiente enlace: https://github.com/jasanfa/TFM_JAVIER_SANZ_FAYOS. Además, los programas se encuentran comentados a fin de facilitar su comprensión y, a su vez, el repositorio de *GitHub* dispone de un fichero *ReadMe* donde se facilitan una serie de directrices para entender la función de cada fichero subido.

Apéndice II

Artículo

Procesamiento del Lenguaje Natural para determinar el grado de acierto o validez de una respuesta corta basado en palabras clave

Javier Sanz Fayos

Universidad Internacional de La Rioja, Logroño (España)

15/09/2021

RESUMEN

En ocasiones, el contenido y calidad de un examen no se adecúa a los objetivos marcados para la prueba. Por eso, se pretende desarrollar una herramienta capaz de reconocer las respuestas de contenido más pobre, facilitando *feedback* al estudiante y/o al profesor. En este trabajo, se estudia la influencia del reconocimiento de palabras clave como método de evaluación semiautomático en relación a la calificación final de una respuesta. Para ello, se elaboran técnicas de segmentación, lematización y desambiguación del sentido de las palabras mediante un corpus seleccionado. Además de analizar palabras clave en la respuesta pre-procesada del alumno, se estudian sus sinónimos y antónimos recogidos en dos tesauros creados automáticamente. Los resultados obtenidos avalan el funcionamiento del método desarrollado pero indican una falta de relación entre la técnica de evaluación desarrollada y la real, evidenciando la falta de impacto de las palabras clave sobre la nota.

unir
LA UNIVERSIDAD EN INTERNET

PALABRAS CLAVE

Evaluación semiautomática, palabras clave, Procesamiento del Lenguaje Natural, retroalimentación, tesauro.

I. INTRODUCCIÓN

La corrección de exámenes y trabajos es una de las tareas más importantes en el ámbito educativo. Se trata de una tarea exigente y susceptible a errores aleatorios en el juicio de las preguntas. Concretamente, las de respuesta abierta son aquellas que más comprometen la precisión de la evaluación pues están más ligadas a factores de subjetividad, debido a la diferencia de criterios de corrección del profesorado [1] [2]. Esta variabilidad en la nota, aunque se trate de mitigar con el uso de rúbricas, podría penalizar o beneficiar a ciertos estudiantes [3].

Por este motivo, se pretende investigar y desarrollar una aplicación software, basada en Procesamiento del Lenguaje Natural (PLN), que contribuya al desarrollo y mejoría de las aproximaciones existentes en la actualidad para la evaluación automática/semiautomática de contenido. Para ello, se pretenderá dar respuesta a estas preguntas de investigación:

- ¿En qué medida es válida la evaluación de una respuesta basándose en un conjunto de palabras clave (o *keywords*, en inglés) para determinar la calidad de una respuesta?
- ¿Existe correlación entre el criterio utilizado por los examinadores y el que emplea este método?
- ¿Se puede generar *feedback* competen-

te con este método para justificarle al alumno sus aciertos y errores de contenido en su respuesta?

Este trabajo desarrollará una sección del plano semántico de un proyecto investigativo de mayor calibre. Éste pretende, a partir de la combinación de las salidas de múltiples métodos que evalúan diferentes aspectos de la respuesta del estudiante, diseñar un formato de evaluación basado en la rúbrica ponderada de los diferentes planos. Se busca conseguir, así, una calificación general y argumentada de dicha respuesta.

II. ESTADO DEL ARTE

En el campo de la educación, el PLN busca extraer, modular y transformar a nivel computacional el conocimiento de un experto en un determinado campo. Este escenario propicia la posibilidad de mejorar la educación en línea actual [4]. *Assisted Study*, por ejemplo, es una herramienta que ayuda a la corrección sintáctica y semántica de exámenes de respuesta abierta, proporcionando *feedback* con el que argumentar la calificación calculada [2].

Existen, también, sistemas que proporcionan una serie de índices analizables que sirven para obtener métricas con las que evaluar de mejor manera al estudiante en cuestión. Westera et al. [5] consiguen generar hasta 200 índices para evaluar el aprobado o suspenso del alumno. Otra solución es ReaderBench [6], que permite evaluar la complejidad de textos, resúmenes y explicaciones, integrando los índices provistos por los sistemas *E-rater* [7], *iSTART* [8] y *CohMetrix* [9].

Sistemas, como el propuesto por Panaite et al. [10], se nutren del conocimiento que proporcionan este tipo de índices para efectuar la calificación de respuestas cortas. Otro sistema similar, pero más centrado en el entrenamiento del alumno en la escritura de ensayos, es *Writing Pal*, que utiliza PLN para valorar y proporcionar *feedback* de la calidad del escrito del alumno, incluyendo índices de evaluación para identificar la complejidad, cohesión, retórica y lingüística del texto. Por otra parte, también se podrían utilizar motores de puntuación automatizados, como *C-rater* [11]

En general, se busca dar soporte al estudiante, no sólo a partir de la evaluación. *AutoTutor*, por ejemplo, es un sistema de tutoría de lenguaje natural capaz de dialogar y orientar al alumno construyendo un modelo de éste que se va perfeccionando a partir de la interacción con el sistema [12]. Otros ejemplos de tutores son: *Why2*, *CIRCSIM-Tutor*, *GuruTutor*, *DeepTutor* o *MetaTutor*. Éstos, están también basados en mecanismos de aprendizaje adaptativo, y ofrecen garantías de efectividad similares a las de una persona humana [13].

En la actualidad se están promoviendo métodos de abstracción automática del conocimiento experto. Sin embargo, falta todavía investigar más en esta área para mejorar algoritmos, sobre todo de PLN, que perfeccionen las técnicas de identificación y extracción de los componentes semánticos actuales para construir mejores modelos de conocimiento y de evaluación de los estudiantes [14].

III. OBJETIVOS Y METODOLOGÍA

A. Objetivo

Investigar la posibilidad de hacer una evaluación semiautomatizada de la respuesta de un alumno a una pregunta concreta de un examen, basándose en el conjunto de palabras clave que el profesor espera que conteste.

B. Metodología

Se formará, primero, un corpus en español que sea representativo, aleatorio, equilibrado y no demasiado específico; es decir, que no esté compuesto principalmente por palabras de un determinado dominio. Se consigue, así, extraer información de la lengua en su totalidad, una

característica que es útil para desambiguar los sentidos de las palabras (WSD, en inglés).

Para trabajar el problema, se dispone de un *dataset* con 144 respuestas a una misma pregunta de un examen real. Cada contestación está formada por la pregunta del examen, las palabras clave que deben incluirse, la respuesta del alumno, la nota y un identificador anonimizado. A partir de los *keywords*, se formará un tesauro con éstos y con sus sinónimos y antónimos, que serán obtenidos mediante procedimientos de *web scraping*. Este instrumento será recorrido para identificar si la respuesta del alumno incluye palabras clave o cercanas a éstas en cuanto a significado.

El conjunto de operaciones realizadas será programado en el entorno *Jupyter Notebook* de *Google Colab*. Las respuestas serán preprocesadas y segmentadas en frases más simples para facilitar el uso de determinadas técnicas de PLN y para comprobar la distribución de aparición de las palabras clave a lo largo de la respuesta. A cada una de éstas, se le aplicará también un proceso de *tokenización* seguido de un proceso de lematización, que será robusto a los posibles sentidos de las diferentes palabras.

En función de la relación entre el número total de palabras clave definidas por el tutor y el número de coincidencias independientes en la respuesta del alumno, se ajustará la calificación del método y se comparará con la real. Todos los *keywords* ponderarán equitativamente, independientemente de la forma en la que aparezcan (de forma literal o mediante sinónimos/antónimos). El sistema de *feedback*, a su vez, estará basado en una agrupación informativa de sentencias que servirán para informar la adecuación de éstos en la oración.

IV. CONTRIBUCIÓN

El corpus se extrajo de *Wikicorpus* y se formó con 32 textos y 60 880 634 palabras. A partir de su procesamiento, es posible crear un diccionario de lemas que servirá como motor de lematización, combinándose con el *pipeline* en español de *Stanza* (que permite también calcular la categoría gramatical de las palabras). Dicho diccionario, se estructurará por niveles (palabra, categoría gramatical y lema) para facilitar la tarea de WSD.

Existen palabras que se deberán reducir a una forma canónica u otra en función de su categoría gramatical en la oración. Para la cuestión de WSD, se optó por utilizar un modelo oculto de *Márkov* (HMM, en inglés) que se decodificará utilizando el algoritmo de *Viterbi*. Durante el cálculo de las tablas de probabilidad, dado el vasto conjunto de datos manejados, el programa se interrumpía. Para solucionarlo, se trabajó con cada fichero por separado, asumiendo igual representatividad y calidad en cada uno de ellos, de forma que las tablas de probabilidad finales se obtuvieron a partir de la media relativa al número de apariciones de cada *token*.

Se filtraron, también, las fechas, números y nombres propios. A diferencia de los dos primeros, la eliminación de nombres propios sí conlleva la pérdida de información significativa, por eso, se creó un nuevo *token* (denominado *tokenseliminados*) para recoger y almacenar la mencionada información descartada, como si de un *buffer* se tratara. De esta forma, si al utilizar *Viterbi* aparece alguna palabra nueva será considerada como "*tokenseliminados*". Se aprovechan, así, los valores de probabilidad de este *token* (combinados con una penalización de categorías menos probables) para etiquetar la palabra.

Como instrumento para construir los tesauros de sinónimos y antónimos se utilizó *WordReference* [15]. Para proporcionar información complementaria se añadió un nivel más en los dos tesauros. Uno de los índices de dicho nivel indica si la palabra es clave en sí misma o es una unidad significativa de un *keyword* compuesto. Los índices restantes, permiten acceder al tesauro en cuestión y a la categoría gramatical del *keyword* a estudiar.

Por último, la estrategia desarrollada consiste en recorrer la frase para comprobar si el

n-grama de tamaño variable, coincide con alguna palabra clave. Este proceso se repetirá por cada *keyword* disponible, por cada tamaño de ventana y por cada *set* de estudio (*keywords*-sinónimos-antónimos).

V. RESULTADOS

A. Funcionamiento del programa

Se mostrará el funcionamiento del etiquetador morfosintáctico elaborado, incluyendo el correspondiente etiquetado de una frase seleccionada como ejemplo. Dicha oración, contiene la palabra "1996" (categorizada como "*tokenseliminados*") y la palabra ambigua vino.

El invitado vino con el vino tinto de la cosecha de 1996.

el / Determiner,
invitado / Noun,
vino / Verb,
con / Adposition,
el(2) / Determiner,
vino(2) / Noun,
tinto / Adjective,
de / Adposition,
la / Determiner,
cosecha / Noun,
de(2) / Adposition,
1996 / Noun.

Como se puede observar, el etiquetador es robusto a palabras nuevas. Sin embargo, las palabras consideradas como "*tokenseliminados*", debido a su proceso de formación, están bastante condicionadas a ser sustantivo, lo que en ocasiones provoca que el análisis no sea del todo correcto. Se comprueba, además, que la palabra vino ha sido etiquetada correctamente dependiendo de su acepción en la oración, por lo que sí se ha conseguido desambiguar.

Por otra parte, se muestra un ejemplo de *feedback* generado al analizar una determinada frase de la respuesta de un estudiante.

Frase de la respuesta:

"- arbitro: encargado de nombrar al nodo secundario que comienza a actuar en caso de que el primario deje de tener disponibilidad.",

***Feedback* generado**:

"La palabra clave arbitrar aparece en el texto, pero no con la categoría gramatical buscada, por lo que su sentido podría ser inadecuado", "La palabra clave disponibilidad aparece correctamente utilizada en el texto", "En el texto aparece la palabra actuar, que es sinónima del lema de la palabra clave arbitrar".

Como se puede observar, se consiguen identificar con precisión tanto los *keywords* como sus sinónimos/antónimos. La estrategia de filtrado y lematización implementada consigue que la identificación sea robusta a las posibles formas de escritura de las palabras. Además, permite procesar correctamente la palabra arbitro, que tiene adherido el símbolo de dos puntos y que podría no ser identificado en una comparación de palabra a palabra. Por otra parte, el *feedback* generado describe con claridad cada situación encontrada, lo que facilita el entendimiento de la decisión del algoritmo.

B. Relación entre la nota calculada y la real

Se mostrará la comparativa entre la nota real que obtuvo el estudiante en su examen y la calificación calculada en este trabajo a partir de las palabras clave (Figura 1). Se representará, en el eje de las ordenadas (Y), la calificación de 0 a 1 conseguida por ambos métodos y, por otra parte, cada unidad del eje de las abscisas (X) representará a un estudiante distinto. Los valores de estas gráficas, están agrupados en función creciente de la nota, y muestran como, a partir de una calificación real de 0.6, la diferencia entre ambos métodos se torna cada vez mayor en líneas generales.

En la Figura 2, se muestra la desviación existente entre muestras, obtenida a partir del cálculo del error absoluto entre éstas. Para considerar las muestras similares, la mayor parte de éstas debería estar situada cerca del cero, de

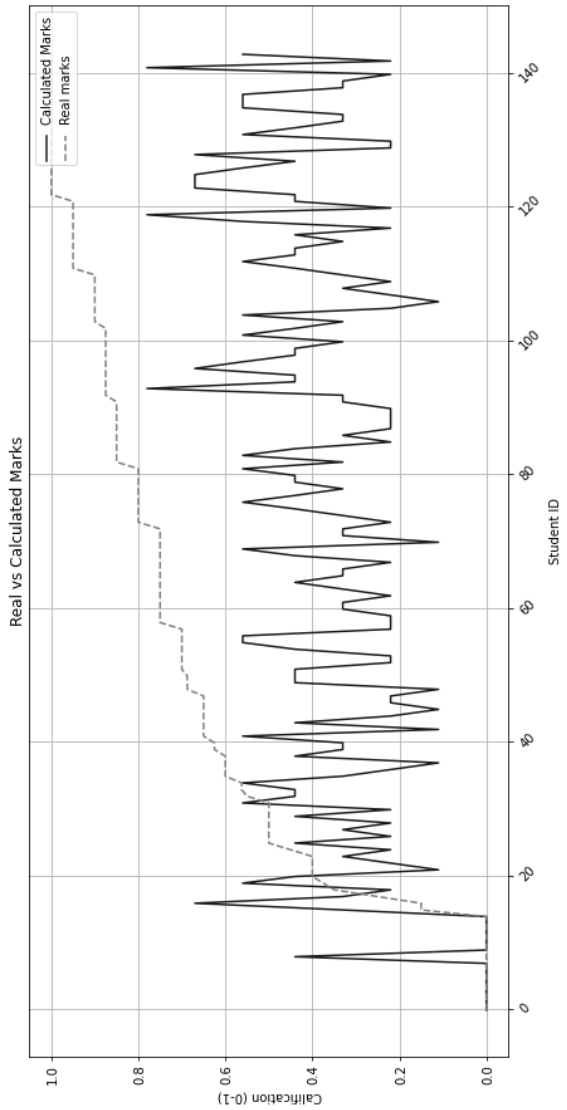


Figura 1: Relación entre nota calculada y real (ordenadas de menor a mayor nota real)

forma que la gaussiana generada tuviera poca varianza. Sin embargo, la distribución de las muestras es achatada y está desplazada a la derecha, con un máximo formado alrededor del 0.5 de desviación, lo que significa que el error obtenido es muy considerable, pues supone no tener la capacidad si quiera de identificar aprobados o suspensos. Se observa, también, que generalmente la nota calculada es inferior a la real porque la desviación obtenida es positiva y no negativa.

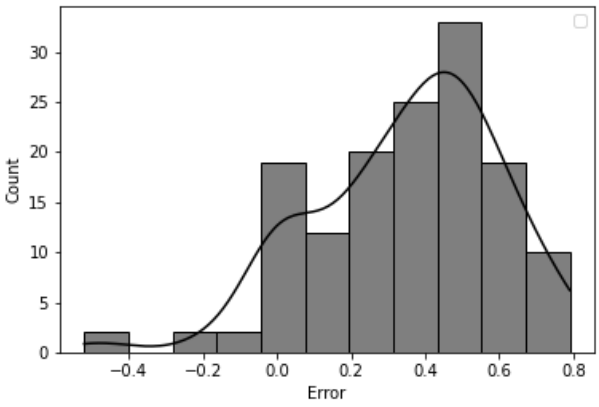


Figura 2: Desviación (línea) obtenida del error absoluto de las notas (barras)

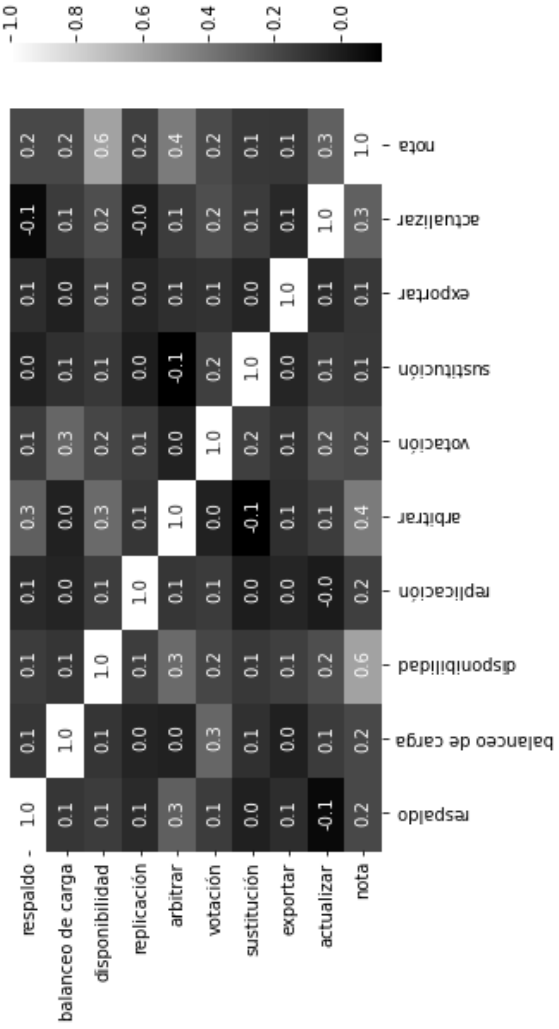


Figura 3: Matriz de correlación de las variables trabajadas

C. Influencia de cada *keyword* en la nota

Para complementar el análisis, se consideró importante representar, también, la influencia de cada par de *keywords*, entre sí, con respecto a la nota. Se pretendía comprobar, de esta manera, la existencia de una posible relación vinculante entre ellos. Como se puede observar en la Figura 3, las dos palabras clave más relacionadas entre sí y, a su vez, más relacionadas con la nota son "disponibilidad" y "arbitrar". Sin embargo, a causa de la debilidad de dichas relaciones, no se puede extraer ninguna evidencia que afirme que la relación sea suficiente significativa.

VI. DISCUSIÓN

Los resultados muestran, con clara evidencia, que no existe relación alguna entre el número de *keywords* encontrados y la nota final que recibe el estudiante. Por ejemplo, hay alumnos con nota de sobresaliente que acertaron menos palabras clave que algunos que suspendieron. Sin embargo, se observa que, salvo en un caso en particular, todas las calificaciones con un 0 por el método de evaluación de este proyecto también lo fueron por el real. Esto permitiría tener la capacidad de advertir al estudiante, sirviendo como alerta de detección temprana de contenido insuficiente, y posibilitando que éste pudiera valorar si le conviene efectuar una reentrega.

Se descarta también la existencia de algún tipo de relación entre los distintos *keywords* con respecto a la nota, a pesar de haber observado una cierta tendencia de los estudiantes a incluir, con mayor frecuencia, ciertos pares de palabras clave (como "disponibilidad" y "arbitrar"). Dicha frecuencia de aparición, parece estar más correspondida a la forma de redacción de los estudiantes y a la relación de los mismos en el contexto de la prueba, que a la calificación que se asignó. De hecho, en la respuesta de un estudiante se identificaron 4 *keywords*: arbitrar, disponibilidad, respaldo y actualizar, que son precisamente los *keywords* considerados como más importantes. Sin embargo, la nota real calculada para este alumno fue de cero.

Por otra parte, analizando la forma de aparición de los *keywords* en las distintas oraciones, se puede determinar que la distribución de las palabras clave en la oración no es un factor determinante para el cálculo de la nota. Los *keywords* no aparecen mayormente concentrados en ninguna frase en concreto, sino que se distribuyen de forma natural por la respuesta, lo que indica que los alumnos utilizan dichas palabras como herramienta para apoyar parte del contenido aprendido, y no como forma de comprimir la respuesta.

Si el número de *keywords* no tiene peso en la nota, consecuentemente, el método diseñado para calcular la calificación del estudiante es erróneo o, por lo menos, no significativo. Existe, por tanto, una nueva variable no contemplada a la hora de calificar, que influye en la diferencia del criterio de ambas técnicas. Si bien este método se ha demostrado que es ineficiente por sí mismo, podría no serlo en conjunción con otras técnicas. Es decir, esta aproximación es solo una sección de una rúbrica, por lo que la diferencia de criterios podría basarse, también, en la información que sí proporcionarían los demás métodos de ésta. Al no disponer de medios para identificar el origen de esta diferencia, se postulan una serie de hipótesis que deberán ser refutadas en trabajos futuros:

La primera de ellas, se basa en considerar que sí existe una relación entre el número de palabras clave y el criterio del profesor. Se justificaría, de esta manera, la falta de relación obtenida entre los dos métodos debido a la selección del listado de palabras clave. Es decir, es posible que los *keywords* que fueron seleccionados, al no pertenecer a un dominio muy concreto (casi todas son palabras del ámbito y uso general), no sean muy representativos. Por tanto, el profesor que califica el examen podría ignorar su aparición en el proceso de evaluación de la respuesta, pues estaría buscando otro tipo de palabras cuyo significado sí que se adecuara más a aquello esperado por el profesor.

La segunda, está basada en sostener que los *keywords* sí son representativos del dominio, pero no fueron igual de significativos para todo el conjunto de examinadores que llevó a cabo la tarea de corrección. Se introduciría, de esta manera, un factor de subjetividad a la hora de corregir la pregunta. De esta manera, un listado idéntico de palabras clave podría ajustarse, en mayor medida, a lo que busca un profesor pero no a lo que busca otro. Esto podría explicar las calificaciones desproporcionadas al número de *keywords*, tanto en aprobados como en suspendidos. Se sabe, además, que las calificaciones reales pudieron estar sometidas a criterios de corrección distintos, un dato que unido a la subjetividad que introduce cada profesor podría ocultar, perfectamente, la relación entre la nota y las palabras clave de los exámenes.

La tercera se basa en considerar que el método trabajado no permite obtener información suficiente por sí mismo a partir de palabras clave. Es decir, al analizar únicamente determinadas palabras se podría estar perdiendo la esencia de la respuesta, que sería aquello que más valora el profesor. La forma del escrito, la relación de conceptos y el uso de lenguaje descriptivo son recursos muy importantes a la hora de evaluar y que pasan completamente desapercibidos con esta aproximación. De esta manera, podría no existir una relación directa con una palabra clave como tal pero sí con su significado o con lo que representa en el temario. La búsqueda de palabras clave, por tanto, debería complementarse, pues el listado de *keywords* se contemplaría como un conjunto de índices abstractos que el estudiante no debería citar como tal. Aquello que debería hacer sería explicar su significado con otras palabras o expresar esa idea aplicada a otro punto en concreto del dominio.

La última, se basaría en la combinación de las hipótesis mencionadas. Sea como fuere, la inexistencia de relación entre métodos de evaluación no significa que el programa realizado sea incorrecto. De hecho, los resultados obtenidos de la extracción de palabras clave son muy positivos. Esto significa que los objetivos que componen el trabajo, desde la construcción de los diccionarios hasta la retroalimentación y estrategia de comparación de palabras, han sido alcanzados (en mayor o menor medida).

Gran parte de este éxito se encuentra relacionado con el funcionamiento del generador de *feedback*. Esto se debe a que la identificación de *keywords* es muy robusta, consiguiendo reconocerlos en sus múltiples formas gramaticales debido a la eficiencia del etiquetador morfosintáctico y a la estrategia de lematización implementada. Así, se consigue particularizar el *feedback* a cada situación, de forma que las sentencias creadas son tan claras, precisas y sencillas, que consiguen informar sin agobiar al lector, lo que las hace idóneas para ser consultadas rápidamente durante la realización de un examen o tarea.

Por último, en relación al etiquetador, se ha comprobado que su eficiencia y adaptabilidad a palabras nuevas es bastante alta. Es cierto que los "*tokenseliminados*", en ocasiones, no se gestionan correctamente. Sin embargo, en este trabajo únicamente interesa desambiguar las palabras clave. De esta forma, se asumirá un cierto error en palabras complementarias siempre que se garantice, como es el caso, el correcto etiquetado de las palabras recogidas en el diccionario. Se observa, por otra parte, como apenas aparecen antónimos, no porque no se logren identificar, sino porque probablemente los estudiantes prefieran utilizar otras unidades de significado más próximo al buscado.

VII. CONCLUSIONES

- El etiquetador morfosintáctico realizado funciona adecuadamente y consigue desambiguar las situaciones en las que una misma palabra se puede expresar con varias categorías gramaticales.
- Se ha conseguido elaborar un procedimiento para crear varios instrumentos de control terminológico con los que recoger, de forma adaptativa, los sinónimos y antónimos de las palabras clave del examen que se desee procesar.

- La estrategia de identificación de palabras elaborada permite la correcta extracción de los *keywords* de la respuesta del alumno.
- El *feedback* generado es suficientemente claro y particular como para permitir entender el motivo por el que una palabra clave ha sido considerada o no.
- El método de evaluación diseñado no es, en este caso, adecuado para proporcionar información útil sobre la nota del alumno pero sí lo es como posible instrumento indicativo de la falta de un mínimo nivel en la respuesta.

Referencias

[1] D. S. McNamara, S. A. Crossley, R. D. Roscoe, L. K. Allen, and J. Dai. A hierarchical classification approach to automated essay scoring. *Assess. Writ.*, 23:3559, 2015.

[2] F. Rodrigues and P. Oliveira. A system for formative assessment and monitoring of students progress. *Comput. Educ.*, 76(4):30–41, 2014.

[3] C. J. Brame. Rubrics: Tools to make grading more fair and efficient. *Science Teaching Essentials, Academic Press*, page 175184, 2019.

[4] Luis Facundo Maldonado, Olga Lucía Londoño, and Jeimmy Patricia Gómez. Sistemas ontológicos en el aprendizaje significativo: estado del arte. *Actualidades Investigativas en Educación*, 17(2):1–18, 2017.

[5] W. Westera, M. Dascalu, H. Kurvers, S. Ruseti, and S. Trausan-Matu. Automated essay scoring in applied games: Reducing the teacher bandwidth problem in online training. *Comput. Educ.*, 123:212224, 2018.

[6] M. Dascalu. *ReaderBench (1) - Cohesion-Based Discourse Analysis and Dialogism.* 2014.

[7] C. Ramineni. Automated essay scoring: Psychometric guidelines and practices. *Assess. Writ.*, 18(1):2539, 2013.

[8] D. S. McNamara, I. B. Levinstein, and C. Boonthum. istart: Interactive strategy training for active reading and thinking. *Behav. Res. Methods, Instruments, Comput.*, 36(2):222233, 2004.

[9] A. C. Graesser, D. S. McNamara, and J. M. Kulikowich. Coh-metrix. *Educ. Res.*, 40(5):223234, 2011.

[10] M. Panaite, M. Dascalu, A. Johnson, R. Balyan, J. Dai, D. McNamara, and S. Trausan-Matu. Bring it on! challenges encountered while building a comprehensive tutoring system using readerbench. *Springer, Cham*, page 409419, 2018.

[11] C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Comput. Hum.*, 37(4):389405, 2003.

[12] B. D. Nye, A. C. Graesser, and X. Hu. Autotutor and family: A review of 17 years of natural language tutoring. *Int. J. Artif. Intell. Educ.*, 24(4):427–469, 2014.

[13] V. Rus, S. DMello, X. Hu, and A. Graesser. Recent advances in conversational intelligent tutoring systems. *AI Mag.*, 34(3):42–54, 2013.

[14] A. Zouaq, D. Gasevic, and M. Hatalar. Towards open ontology learning and filtering. *Inf. Syst.*, 36(7):10641081, 2011.

[15] WordReference. Diccionario de sinónimos y antónimos. Recuperado de: https://www.wordreference.com/sinonimos/, 2021.