Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Ciencias (Ingeniería Matemática)

Identificacíon de unidades léxicas y sintácticas de texto informal en español

Tesis

Que como parte de los requisitos para obtener el Grado de
Maestro en Ciencias en Ingeniería Matemática

Presenta
Ing. Oscar Cano Félix

Dirigido por:
M. en C. Wilfrido Jacobo Paredes García

M. en C. Wilfrido Jacobo Paredes García
Presidente

Dr. Arturo González Gutíerrez
Secretario

Dr. Eduardo Castaño Tostado
Vocal

Dr. Roberto Augusto Gómez Loenzo
Suplente

M en C. Luisa Ramírez Granados
Suplente

Centro Universitario, Querétaro, Qro.
Diciembre de 2019
México

Resumen

El procesamiento del lenguaje natural es una herramienta ampliamente utilizada hoy en día dentro de la web, así como en aplicaciones móviles entablando interacciones naturales entre humanos y computadoras. A pesar de que muchos estudios han sido publicados y el auge de las inteligencias artificiales para el entendimiento del lenguaje ha permitido desarrollar modelos que logren determinar el sentido de la oración, el análisis morfológico, etc. el procesamiento del lenguaje natural tiene mucho trabajo por delante. Un área un poco olvidada dentro del procesamiento del lenguaje natural es el preprocesamiento de los textos, ya que se supone que la entrada del texto a examinar es un texto limpio y solo con ciertos errores ya identificados, aunque esto no es cierto en el mundo real, ya que en el texto informal en el que el ser humano se comunica, tiene presente errores inherentes y diferentes, que, además de ser difíciles de identificar, estos errores evolucionan con el tiempo, por lo que suponer que el preprocesamiento de un texto informal es una tarea ya estudiada es un error. En el presente trabajo se propone una aproximación para inferir los puntos clave de una tarea del preprocesamiento del lenguaje natural que es la identificación de unidades léxicas y sintácticas en el idioma español mexicano, mediante la subtarea de la segmentación de oraciones, siendo el que se determine una propiedad de existencia o que se acople a un modelo de palabra desconocidas basado en n-gramas para la manipulación de estos. La característica primordial de la investigación se centra en el modelo de ordenamiento y selección, sin olvidar que el modelo para puntuar probabilísticamente una palabra desconocida, en si es una innovación en el lenguaje español. El modelo para el ordenamiento y selección consiste en la aplicación de algoritmos de árboles binarios basados en reglas de composición, complementado con algoritmos de optimización de creación y selección de candidatos tal como el algoritmo voraz o la propiedad de Markov.

Palabras clave: Procesamiento del lenguaje natural, N-Gramas, Segmentación, Markov, Optimización, Redes sociales, Twitter.

Abstract

Natural language processing is a tool widely use nowadays used within the web as well as mobile applications engaging natural interactions between humans and computers. Although many studies have been published and the rise of artificial intelligence for language understanding has allowed us to develop models that determine the meaning of sentences, morphological analysis, etc. Natural language processing has a lot of work ahead. A slightly forgotten area within natural language processing is the preprocessing of the texts, since it is assumed that the entry of the text to be examined is a clean text and only with certain errors already identified, although this is not true in the real world, since in the text informal in which the human being communicates, has inherent and different errors, which in addition to being difficult to identify, these errors evolve over time, so assume that the preprocessing of an informal text is a task already studied is a mistake. In this paper, an approach is proposed to infer the key points of a natural language preprocessing task, which is the identification of lexical and syntactic units in the Mexican Spanish language, through the sub-task of the segmentation of sentences, being the one determined a property of existence or that fits an unknown word model based on n-grams for the manipulation of these. The main characteristic of the research focuses on the ordering and selection model, without forgetting that the model for probabilistically scoring an unknown word, is an innovation in the Spanish language. The model for the ordering and selection consists of the application of binary tree algorithms based on composition rules, complemented with optimization algorithms for the creation and selection of candidates such as the greedy algorithm or the Markov property.

Key words: Natural language processing, N-Grams, Splitting, Markov, Optimization, Social Network, Twitter.

Natural language processing, Social network services; Machine learning; Artificial intelligence; Computers and information processing; The person-computer interaction; Morphology, social networks, Twitter.

Agradecimientos

Gracias a los miembros de mi sínodo, el M. en C. Wilfrido Jacobo Paredes García, siempre dispuesto a ayudar y cuyas valiosas enseñanzas aportaron no solo a este trabajo sino a mi formación como estudiante de Maestría en Ingeniería Matemática y a los sinodos Dr. Arturo González Gutiérrez, Dr. Eduardo Castaño Tostado, Dr. Roberto Augusto Gómez Loenzo, y a la M en C. Luisa Ramírez Granados por su apoyo durante para este proyecto.

Gracias al Consejo Nacional de Ciencia y Tecnología (CONACyT) [630504] por el apoyo recibido para realizar mis estudios y terminar esta investigación.

Índice general

Índice de tablas

Índice de figuras

Lista de Algoritmos

1 Procesamiento del lenguaje natural de texto informal en español

"Die Grenzen meiner Sprache bedeuten die Grenzen meiner Welt (Los límites de mi lenguaje son los límites de mi mundo)"

— Ludwig Wittgenstein

1.1 Introducción

En la actualidad, las redes sociales, las comunidades web, los blogs y a la interacción que existe entre los usuarios de estas, se generan una gran cantidad de datos e información que constituyen un atractivo segmento de estudio para la academia y las corporaciones. Como ejemplo tenemos a Facebook que tiene más de 2.000 millones de usuarios activos cada mes (Ríos, 2017), seguido de YouTube con 1.500 millones de usuarios mensuales, WhatsApp con 1.200 millones, el gigante asiático de la mensajería, WeChat con 889 millones (Ríos, 2017) y finalmente la plataforma de microblogging Twitter con 330 millones de usuarios mensuales. Considerando que actualmente existen aproximadamente 7.500 millones de humanos en el mundo según la Oficina del Censo de EU (Bureau, 2018), estamos diciendo que solamente Facebook, posee cerca del 20 % de la totalidad de la población humana interactuando activamente en su plataforma. De igual manera y según las previsiones de Cisco Systems, con su estudio The Visual Networking Index (Cisco, 2017), el tráfico global de Internet es mayor a 1.5 zettabytes durante este año, y se espera que se duplique en el año 2021. Esta inmensa cantidad de información es usada por las grandes compañías para tomar decisiones estratégicas de negocios, sin embargo, hasta el momento esa información solo ha sido cuantificada numéricamente, reflejando por ejemplo cuántos clics damos en una página o nuestros hábitos estadísticos de consumo. Pero los humanos somos más que eso y en Internet transmitimos sentimientos, percepciones e ideas. El manejo de toda esta información resulta un desafío enorme que solo podría ser manejado por medios computacionales. Parte del desafío consiste en que a pesar de que la información resulta muy sencilla de entender para el ser humano, es una tarea compleja para resolver por medios computacionales. Como resultado de esto,

la tarea del manejo de esta gran cantidad de información no estructurada es de vitalidad para los profesionales que buscan dar forma al entendimiento de las opiniones de millones de personas a través de la red.

Para resolver esta problemática, se han desarrollado herramientas capaces de interpretar dicha información no estructurada. Una de ellas es el procesamiento del lenguaje natural (NLP), definido como un campo interdisciplinario de la ciencia de la computación donde su investigación se refiere al refinamiento y aplicación de técnicas para resolver problemas del mundo real. Tales problemas consisten en la creación de sistemas de diálogo hablado, motores de traducción de voz a voz, minería de redes sociales para el análisis de información individual o para identificar el sentimiento hacia productos y servicios (Liu *et al.*, 2017).

Para poder realizar un completo procesamiento del lenguaje natural se pueden definir en cinco tareas básicas mostradas en la Fig. 1.1.

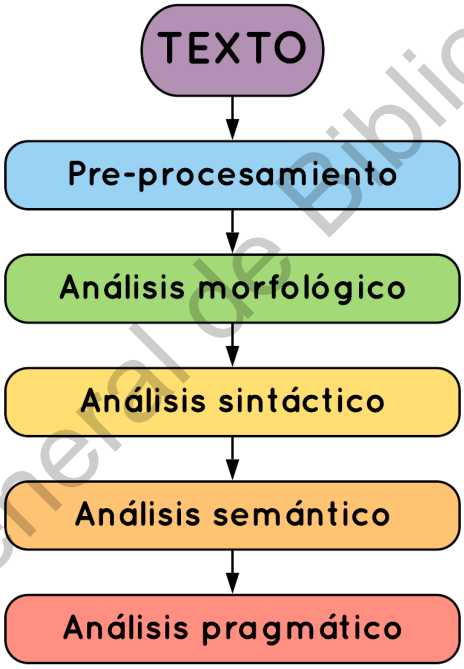

Figura 1.1: Estructura básica de un procesamiento del lenguaje natural

El **preprocesamiento** es un nombre genérico que se da para varias tareas que se consideran relativamente simples y que se ejecutan antes de las tareas más complicadas. De las cuales según Gelbukh (2009) podemos reconocer:

1. **Limpieza del texto**: es la localización y eliminación o el tratamiento adecuado de los segmentos del archivo de texto que no son propiamente texto: imágenes, marcaje del formato, fórmulas, letras mayúsculas y minúsculas, etcétera, es decir, todo aquello que ayuda a convertir el archivo dado en un texto bien formado en el sentido lingüístico.

2. **Determinación de elementos textuales (*tokenizing*)**: implica convertir el texto de una secuencia de letras a una secuencia de cadenas, aproximadamente correspondientes a palabras. Es importante recalcar que esta tarea en apariencia tan simple está en realidad interconectada con decisiones que se toman en otros niveles. Por ejemplo, en la secuencia de caracteres etc., ¿es el punto parte de la palabra o es un elemento textual aparte que indica la finalización de la oración?

3. **Reconocimiento de entidades nombradas (NER, *named entity recognition*)**: es la etapa de aglutinar varias palabras en una sola que es el nombre de un concepto u objeto, por ejemplo. Estados Unidos Mexicanos se tratará como una sola palabra que refiere al país correspondiente y es sinónima de México. Se aplican las mismas aclaraciones sobre la no trivialidad de esta tarea. El descubrimiento de tales entidades nombradas y la toma de decisiones sobre la aglutinación, o no, de los elementos textuales en un contexto dado en una entidad nombrada, han sido áreas de activa investigación en los últimos años.

4. **Separación de las oraciones (*sentence splitting*)**: es la toma de decisión acerca de dónde están las fronteras entre las oraciones. El ejemplo arriba mencionado de la secuencia «etc.» ilustra un caso no trivial de esta tarea. Un programa que efectúe esta tarea se llama en inglés *sentence splitter*.

5. **Eliminación de las palabras basura (*stopwords*)**: Para ciertas aplicaciones, las palabras sin contenido semántico, tales como las palabras funcionales o muy frecuentes, no aportan casi nada al desempeño del sistema y pueden perjudicarlo. La lista de las palabras que deben eliminarse puede variar de una aplicación a otra y en ocasiones está sujeta a decisiones inteligentes automáticas.

El **Análisis morfológico** es una tarea de índole propiamente lingüística involucra la determinación de la primera forma de la palabra y sus características gramaticales y morfosintácticas. Se puede efectuar sin contexto, es decir, sobre una palabra aislada, y en este caso proporcionará un resultado ambiguo, por ejemplo: como, nada, habla, hablamos. También se puede efectuar en el contexto, lo que usualmente permite desambiguar la palabra para dar una sola respuesta: considerarlo como tal, Juan nada bien, él habla poco, ayer hablamos de eso. El programa que efectúa la tarea completa sin contexto —dando usualmente varias variantes de la respuesta— se llama un analizador morfológico, y si lo hace en contexto — produciendo sólo una variante de la respuesta— se llama etiquetador (tagger). El programa que sólo determina la primera forma de la palabra dada (comió, comer), se llama en inglés lemmatizer. Para ciertas aplicaciones, sobre todo en la recuperación de información, es suficiente determinar sólo la base de la palabra sin generar la forma completa (comió, com).

El **Análisis sintáctico** se trata de agrupar las palabras dentro de una oración, según su relación sintáctica y descubrir así su estructura interna. Existe un número considerable de teorías y metodologías para efectuar esta tarea, las cuales difieren substancialmente en la definición de ella y la definición de la relación sintáctica. Las corrientes más comunes y distintas entre sí son la corriente de análisis sintáctico de constituyentes, y la de dependencias. En la primera las palabras se agrupan y se forma un árbol sintáctico cuyos vértices son grupos de palabras relacionadas, y las aristas son las relaciones de anidamiento entre estos conjuntos.

En la segunda aproximación las palabras se asignan una como «ayudante» de la otra, y esta relación forma un árbol cuyos vértices son palabras individuales. Por lo general el análisis de constituyentes tiene un mejor sustento matemático y es más fácil de efectuar, mientras que el análisis de dependencias proporciona información más rica y mejor organizada para el subsecuente análisis semántico. Los programas que efectúan el análisis sintáctico se llaman en inglés parsers, aunque este término más comúnmente se aplica al análisis de constituyentes.

El **Análisis semántico** es la tarea que concierne al tratamiento del significado del texto más que a su estructura, y puede involucrar varios aspectos del problema. Análisis pragmático. Mientras que el análisis semántico todavía trata de estructurar lo que directamente dice el texto, el

Y el **análisis pragmático** que proporciona metainformación sobre el texto: para qué se dice lo que se dice, cómo se organiza, qué estrategias emplea el autor para alcanzar el efecto deseado.

Particularmente en la minería de texto informal en redes sociales se trabajan sobre retos que requiere la integración de muchas tareas neurolingüísticas, incluyendo la extracción de aspecto, reconocimiento de la entidad nombrada, desambiguación de la polaridad de la palabra, etiquetado temporal, reconocimiento de la personalidad y detección de la ironía y sarcasmo Hussain & Cambria (2017).

Estas tareas neurolingüísticas han sido abordadas anteriormente por diversos investigadores, entre ellos Carvalho (Carvalho *et al.*, 2009) quienes crearon un sistema automático para detectar la ironía basándose en emoticones y signos de puntuación especiales y donde su estudio se enfocó en la detección de estilo irónico en artículos de periódicos. J. Nothman *et. al.* (Nothman *et al.*, 2013) propusieron un método de clasificación supervisada multilingüe para reconocer automáticamente la entidad nombrada en varios idiomas de Wikipedia en función del etiquetado de cada enlace. Sus resultados demuestran que este enfoque es altamente efectivo y eficiente para crear modelos de entidad nombrada en lenguajes con pocos recursos, aunque su método, inicialmente requiere la clasificación de todos los artículos de Wikipedia de forma manual.

Reyes (Reyes *et al.*, 2013) y también Barbieri (Barbieri & Saggion, 2014) han propuesto recientemente enfoques para detectar la ironía en Twitter donde propusieron un modelo basado en experimentos de clasificación, usando bases entrenadas anteriormente con minería de opiniones. Por otro lado (Veale & Hao, 2010) propusieron un algoritmo para separar los símiles irónicos de los no irónicos, detectando términos comunes usados en esta comparación irónica. Los sistemas de (González-Ibáñez *et al.*, 2011) detectan el sarcasmo con buena precisión en los *tweets* (mensajes de máximo 280 caracteres en Twitter) en inglés. (Lukin & Walker, 2017) utilizaron bootstrapping, un método estadístico de remuestreo, para mejorar el rendimiento de clasificadores de sarcasmo y maldad para la plataforma Online Dialogue. Finalmente (Riloff *et al.*, 2013) construyó un modelo para detectar el sarcasmo con un algoritmo de arranque que automáticamente aprende listas de frases de sentimientos positivos y frases de situaciones negativas de los *tweets* sarcásticos, con el fin de detectar la característica del sarcasmo, ser un contraste entre el sentimiento positivo y situación negativa (Barbieri *et al.*, 2014).

El desarrollo de la inteligencia artificial y los avances en la probabilidad moderna han influenciado la manera en la que se ha abordado la interpretación de la información en la actualidad, de manera que los enfoques para lidiar con él, las tareas neurolinguisticas se han

vuelto más sofisticados. Podemos ver que Barbieri Barbieri *et al.* (2014) no incluye patrones de palabras como características, sino más bien apunta a detectar el sarcasmo por su estructura interna con un conjunto de modelos de Markov ocultos (*Hidden Markov Models*, *HMM*) basados en texto, los cuales utilizan aumentos y agrupaciones de palabras producidas por análisis semántico latente. Por su parte, Hussain Hussain & Cambria (2017) utiliza enfoques de aprendizaje semi-supervisados, los cuales están basados en el uso combinado de escalamiento multidimensional (*MDS*) mediante proyecciones aleatorias y máquinas de vectores de soporte sesgadas (*bSVM*) para grandes análisis de datos. De tal forma el sistema se encuentra modelado sobre la capacidad intrínsecamente humana de interpretar información cognitiva y afectiva asociada con el lenguaje natural, para inferir nuevos conocimientos y tomar decisiones en relación con los valores sociales y emocionales de uno, es decir los censores e ideales. Más recientemente, Soujanya Poria Cambria *et al.* (2017) y sus colegas emplearon una red neuronal profunda para identificar el sentimiento, la emoción y las características de la personalidad para la detección del sarcasmo. En está investigación se analizan técnicas del procesamiento del lenguaje natural aplicado tanto a obras literarias como a texto proveniente de redes sociales, logrando determinar distintos factores que las entorpecen.

Los algoritmos para el procesamiento del lenguaje natural tienen un alto grado de eficiencia en el lematizado y en el etiquetado morfológico de textos formales, pero en textos informales provenientes de redes sociales, que además de faltas ortográficas, el no seguir los constructos gramaticales tradicionales están generando su propio lenguaje de Internet, por tanto, el cómo tratar estos nuevos paradigmas, deriva en tratar de enfocarse en encontrar nuevos métodos que se adapten a la evolución del lenguaje en tiempo real.

El procesamiento del lenguaje natural es dividido en cinco etapas generales, en las cuales cuatro de ellas están basados en diferentes tipos de análisis y la primera etapa está definida como el preprocesamiento del texto.

Para que el procesamiento se realice correctamente, con los analizadores que existen actualmente, es necesario que el texto se encuentre escrito de manera correcta, sin errores, es decir que se realice un correcto preprocesamiento de este texto. Existen varias tareas para el correcto preprocesamiento del lenguaje natural tales como, el tokenizado, la eliminación de palabras con bajo contenido semántico, la eliminación de palabras repetidas, la segmentación de palabras unidas, el cálculo de la frecuencia de aparición de las palabras, la identificación del idioma en que está escrito, etc.

Entre las tareas que conllevan al preprocesamiento, la segmentación de palabras, conocido como el proceso de identificar los límites entre palabras, sílabas o fonemas en las lenguas naturales, es de interés tanto por razones prácticas como teóricas. Siendo uno de los pasos primordiales para el procesamiento del lenguaje natural (NLP) de lenguajes sin límites explícitos de palabras como el chino o el japones. Así mismo algunos textos tales como, URls, o Hashtags no tienen espacios, y algunas veces los escritores cometen errores al olvidar un espacio entre palabras.

Algunos métodos para la segmentación de palabras no supervisados se basan en la observación de las transiciones entre unidades (caracteres, fonemas o sílabas) dentro de las palabras que son generalmente más predecibles que las transiciones entre los límites de las palabras según Goldwater *et al.* (2006). Otros algoritmos computacionales han optado por obtener un corpus bastante grande (billones de palabras) para poder realizar la segmentación de manera precisa, no obstante, esto los vuelve algoritmos no adecuados al querer aplicar

está metodología con diferentes lenguajes, no importando si asemejan la misma estructura gramatical o no, simplemente por el hecho no compartir las mismas palabras, manteniéndose fuera del corpus mencionado anteriormente, por ejemplo en el *Oxford English Dictionary* se definen 414,800 palabras en inglés y en el diccionario de la Real Academia Española se definen 93,000 palabras en español no concordando entre sí.

Algunos trabajos pasados como el de Lukin & Walker (2017) exploró el método de *bootstraping* aplicado en el desarrollo de clasificadores capaces de identificar tipos particulares de expresiones subjetivas en diálogos en línea. Zhang *et al.* (2018) implementó métodos bidireccionales para la segmentación de textos en el idioma chino dentro de modelos supervisados. Padró & Stanilovsky (2012) incursionó con un trabajo muy amplio, desarrollando una librería para el procesamiento de múltiples lenguajes, trabajando con objetos para la separación de objetos vectoriales. Doval & Gómez-Rodríguez (2019) trabajó con un corpus de lenguaje natural informal obtenido a partir de Twitter, comparando modelos de n-gramas, en contraste con modelos neuronales, teniendo como conclusión que los modelos basados en n-gramas resultaron ser igual de eficiente que algunas redes neuronales complejas, y finalmente Choi & Lee (2019) investigó la relación entre oraciones y cuál el cómo esto puede mejorar para los procesos en el entendimiento del lenguaje natural (NLU). La formulación de un modelo probabilístico explícito permite separar de manera clara las suposiciones sobre la entrada y las propiedades de las posibles segmentaciones de los detalles de los algoritmos utilizados para encontrar tales soluciones.

Algunos métodos para segmentar oraciones están basados en estadísticas locales y resultan bastante exitosos, sin embargo, este trabajo fue enfocado en modelos probabilísticos explícitos. Este no es el primer trabajo en proponer modelos probabilísticos explícitos de segmentación de palabras, anteriormente tres sistemas de segmentación de palabras exitosos basados en Los modelos probabilísticos explícitos fueron los de Brent (1999), Venkataraman (2001) y Goldwater *et al.* (2009).

En los trabajos citados anteriormente se encuentran el sistema de Brent llamado "*Model-Based Dynamic Programming*" (MBDP) que asume que todas las palabras son distribuidas en uni-gramas, es decir que ninguna de las palabras en la oración a segmentar tiene relación entre ellas. Mientras que Venkataraman asume que las palabras son distribuidas en n-gramas estándar en tres versiones de modelos del lenguaje, a la que nos referimos como segmentación de n-gramas (*NGS, n-grams segmentation*), y finalmente Goldwater utiliza el modelo NGS con el valor agregado de que es utilizado el "*Dirichlet Process Model*" para conocer la distribución de las variables aleatorias que se forman en los diferentes corpus analizados.

A pesar de su estructura generativa bastante diferente, las precisiones de segmentación MBDP y NGS son muy similares. Igualmente, la precisión de la segmentación de los n-gramas en el modelo NGS de Venkataraman casi no difieren, lo que sugiere que las dependencias contextuales son irrelevantes para la segmentación de palabras. Esto es claramente diferenciable con el trabajo de Goldwater en la que demuestra la que el modelo NGS usado con bi-gramas mejora al modelo de uni-gramas, demostrando la importancia de las dependencias para la segmentación de palabras.

En este trabajo se ponen a prueba las dependencias contextuales para la segmentación de palabras al comparar dos modelos probabilísticos diferenciados, suponiendo en el primero que cada una de las palabras están sujetas a la independencia entre cada una de ellas,

mientras que el segundo, supone que una palabra depende de la palabra anterior a ella, teniendo en cuenta el contexto en el que se está segmentando, donde la suposición de que la probabilidad de una palabra depende solo de la palabra anterior se denomina suposición de Markov. Siendo que los modelos de Markov son la clase de modelos probabilísticos que asumen que podemos predecir la probabilidad de alguna unidad futura sin mirar demasiado hacia el pasado. Estos modelos son enfatizados en la búsqueda de palabras conocidas y la determinación de probabilidad de ocurrencia de palabras desconocidas en este tipo de modelos, así como el cálculo probabilístico de los pares ordenados de palabras conocidos y desconocidos.

Adicional a esto, se encuentra el estudio realizado por Norvig en el libro de Halpern (2015), donde se propone un estudio basado en NGS definiendo un modelo probabilístico propio, en el cual se calculan las probabilidades de los segmentos, siendo estos conocidos basados en una bolsa de palabras o desconocidos definiendo una distribución de palabras desconocidas, estás dos formas se lograron calculando los parámetros desde la base de datos de Thorsten Brants (2006), "*corpus Web 1T 5-gram Version 1*", de Google Inc., que contiene n-gramas en inglés. Teniendo rangos de n-gramas desde uni-gramas hasta 5-gramas. Estos n-gramas fueron generados de aproximadamente un trillón de palabras de textos públicos de páginas webs. Del cual Norvig logró obtener una precisión de 98.7 en un estudio con 159 unidades léxicas.

Nuestros modelos fueron creados basándonos en las teorías mencionadas anteriormente acoplando sus modelos en uno que pudiera ser aplicado al español mexicano informal, mejorando el uso de recursos computacionales, así como construyendo el modelo probabilístico explicito para calcular la probabilidad de palabras desconocidas. Así mismo se usaron métodos estadísticos para dar valides al modelo.

Para poder trabajar en nuestros modelos distributivos aplicados en textos informales, se creó un corpus especial en el que se encuentra con inflexiones verbales, y con corrección de faltas de ortografía tratando de corregir la mayoría de los problemas que se pueden encontrar dentro del texto informal.

Finalmente se desarrolló un algoritmo optimizar la estructuración de árboles que reducen el tiempo de procesamiento de segmentación de palabras usando reglas construidas especialmente para este tipo de problemas de n-gramas. Buscando generar las segmentaciones basados en árboles n-narios, que tienden a diferenciarse en su construcción al desarrollar los modelos de segmentación dependientes del factor n con el que se esté trabajando, A pesar de que las segmentaciones que obtenemos pueden caer en óptimos locales, podemos estar seguros de que cualquier diferencia en las segmentaciones refleja diferencias en los modelos probabilísticos, es decir, que los resultados obtenidos en las segmentaciones de cualquiera de nuestros dos modelos reflejan los la relación entre las dependencias de la palabras.

1.2 Motivación

La necesidad de analizar automáticamente un lenguaje toma un significado importante a la hora de tomar decisiones personales, ya que el veredicto sobre nuestra elección en temas como la previsión financiera, la previsión política, ciber-salud, turismo electrónico, entre otros, suele ser influenciado por la opinión de los demás, expresada a través de la red. Debido al volumen de personas que utilizan medio electrónicos para expresar su opinión,

resulta simplemente imposible tener una visión general de un tema en particular. Es aquí donde existe una área de oportunidad para la computación moderna, donde podríamos pasar de categorizar temas de importancia en escalas numéricas, para dar paso a la interpretación del lenguaje, el cual reflejaría un rango más amplio para la categorización de temas, justo como lo hacen los seres humanos. Estos avances no solamente representarían una oportunidad para el usuario común de Internet, sino para las grandes corporaciones del mundo, deseosas de analizar la información de los usuarios para tomar decisiones más efectivas de negocios basadas en experiencias verificadas de usuarios. Un ejemplo muy claro es el análisis de las conversaciones de sus clientes en redes sociales mediante la identificación de contenido relevante, para así comercializar mejor sus productos y administrar su reputación (Cambria *et al.*, 2017).

Una de estas redes sociales donde las personas escriben sus reseñas y dan sus puntos de vista es Twitter, una plataforma de microblogging que permite a los usuarios publicar y leer mensajes cortos (menos de 280 caracteres) llamados *tweets*, que a menudo no siguen las reglas esperadas de la gramática. Los millones de opiniones y críticas expresadas en Twitter cada día están empezando a constituir una importante fuente de información para numerosas compañías y organizaciones, que como se mencionó, ven en este medio un lugar para sondear su área de influencia, conocer la percepción sobre productos, servicios, eventos o personalidades relevantes, así como monitorizar su reputación online. Uno de los primeros problemas a los que se enfrentan estas empresas es discriminar los mensajes pertenecientes a su ámbito de negocio en un medio tan ruidoso como Twitter. A este respecto, las funcionalidades de búsqueda de Twitter se limitan a sencillas funciones como búsqueda por palabras clave, capacidades de búsqueda por idioma o recuperación de los *tweets* de un determinado autor (Calvo *et al.*, 2014).

Particularmente el preprocesamiento del lenguaje natural en redes sociales como Twitter, es decir el lenguaje informal, además de ser desafiante para los investigadores debido a todas las inflexiones, errores ortográficos, parafraseo, errores gramaticales, etc., resulta un medio particular para estudiar ya que su uso indiscriminado tanto por la población en general como por personas con grandes influencias, reflejan un amplio espectro de estudio que pudiera arrojar resultados menos sesgados sobre temas en particular.

1.3 Hipótesis y objetivos

1.3.1 Hipótesis

El uso de bi-gramas con la función de distribución Weibull para la identificación de unidades léxicas y sintácticas, reduce la tasa de error del texto informal en español, en contraste con el uso de uni-gramas como modelo de segmentación..

1.3.2 Objetivos

Objetivo general

Determinar si el uso de bi-gramas con la función de distribución Weibull reduce la tasa de error en la identificación de unidades léxicas y sintácticas de texto

informal en español, en contraste el uso de uni-gramas como modelo de segmentación.

Objetivos particulares

1. Obtención del *corpus* de entrenamiento y validación.
2. Identificación y estimación del modelo probabilístico.
3. Calcular las probabilidades de los segmentos.
4. Diseñar el algoritmo de optimización.
5. Estimar la tasa de error de ambos modelos usando un conjunto de entrenamiento en común (*corpus*).

2
Fundamentos Teóricos

"Divide las dificultades que examines en tantas partes como sea posible, para su mejor solución."
— Rene Descartes

2.1 Datos

2.1.1 Instancia

Una instancia x representa a un objeto especifico. La instancia a menudo se representa mediante un vector de características D-dimensionales $x = (x1, ..., x_D) \in \mathbb{R}^D$ donde cada dimensión se denomina característica. La longitud D del vector de características se conoce como la dimensionalidad del vector de características.

La representación de características es una abstracción de los objetos. Básicamente ignora toda otra información no representada por las características. Por ejemplo, dos pequeños hombres verdes con el mismo peso y altura, pero con nombres diferentes, serán considerados como indistinguibles por nuestra representación característica. Tenga en cuenta que usamos $\mathbf{x}$ para denotar toda la instancia, y x_d para denotar la característica d-ésima de x. En nuestro ejemplo, una instancia es un hombrecito verde específico; el vector de características consta de $D = 2$ características: x_1 es el peso y x_2 es la altura. Las características también pueden tomar valores discretos. Cuando hay varias instancias, usaremos x_{id} para denotar la característica d-ésima de la instancia i-ésima.

2.1.2 Muestra de entrenamiento

Una muestra de entrenamiento es una colección de instancias $\{\mathbf{x}_{i=1}^n = \{\mathbf{x}_1, ..., \mathbf{x}_n\}\}$, que actúa como la entrada al proceso de aprendizaje. Suponemos que estas instancias se muestrean independientemente de una distribución subyacente $P(\mathbf{x})$, que desconocemos. Podemos denotar $\{\mathbf{x}_i\}_{i=1}^n \sim^{i.i.d.} P(\mathbf{x})$, donde i.i.d. significa independiente e idénticamente distribuido.

2.2 Árboles

En ciencias de la computación, un árbol de búsqueda es una estructura de datos de tipo árbol utilizado para localizar llaves concretas dentro de un conjunto. Para que un árbol pueda funcionar como árbol de búsqueda en cada nodo tiene que cumplirse que su llave tiene que ser más grande que cualquier llave contenida en su subárbol izquierdo y menor que cualquier llave contenida en su subárbol derecho.

La búsqueda en árboles binarios es un método de búsqueda simple, dinámico y eficiente considerado como uno de los fundamentales en Ciencia de la Computación. De toda la terminología sobre árboles,tan sólo recordar que la propiedad que define un árbol binario es que cada nodo tiene a lo más un hijo a la izquierda y uno a la derecha.Para construir los algoritmos consideraremos que cada nodo contiene un registro con un valor clave a través del cual efectuaremos las búsquedas.En las implementaciones que presentaremos sólo se considerará en cada nodo del árbol un valor del tipo tElemento aunque en un caso general ese tipo estará compuesto por dos:una clave indicando el campo por el cual se realiza la ordenación y una información asociada a dicha clave o visto de otra forma,una información que puede ser compuesta en la cual existe definido un orden.

Un árbol binario de búsqueda (ABB) es un árbol binario con la propiedad de que todos los elementos almacenados en el subárbol izquierdo de cualquier nodo x son menores que el elemento almacenado en x ,y todos los elementos almacenados en el subárbol derecho de x son mayores que el elemento almacenado en x.

2.2.1 Tipos de Árboles

Árbol de Búsqueda Binaria

Un árbol de búsqueda binaria es una estructura de datos basada en nodos donde cada nodo contiene una llave y dos subárboles, el izquierdo y el derecho. Para todos los nodos, las llaves de los nodos pertenecientes a su subárbol izquierdo deben ser menores que la llave del nodo, y las llaves de los nodos pertenecientes a su subárbol derecho deben ser mayores que la llave del nodo. Estos subárboles deben calificar también como árboles de búsqueda binarios.

La complejidad temporal del algoritmo de búsqueda en un árbol de búsqueda binaria es la altura del propio árbol, la cual es menor que O(log n) para un árbol que contiene n elementos.

B-Tree

Los B-Trees son generalizaciones de los árboles de búsqueda binaria que pueden tener un número variable de subárboles en cada nodo. Si bien los nodos hijos tienen un rango predefinido, no necesariamente contendrán datos, lo que significa que los B-Trees potencialmente pueden desperdiciar algo de espacio. La ventaja es que este tipo de árbol no necesitan ser reequilibrado con tanta frecuencia como otros árboles.

Debido al rango variable de la longitud de sus nodos, los B-trees están pensados para sistemas que leen grandes bloques de datos. También se usan comúnmente en bases de datos.

La complejidad temporal de buscar en un B-Tree es O(log n).

(a,b)-tree

Un (a,b)-tree es un árbol de búsqueda donde todas sus hojas tienen la misma profundidad. Cada nodo tiene al menos a hijos y a lo sumo b hijos, mientras que la raíz del árbol posee entre 2 y b hijos.

$$2 \leqslant a \leqslant \frac{(b+1)}{2}$$

La complejidad temporal de buscar en un (a,b)-tree es O(log n).

Árbol de búsqueda ternaria

Un árbol de búsqueda ternaria es un tipo de trie que puede tener 3 nodos: un hijo menor, un hijo igual y un hijo mayor. Cada nodo almacena un solo carácter y el árbol en sí se ordena de la misma forma que un árbol de búsqueda binaria, con la excepción de un posible tercer nodo.

La búsqueda en un árbol de búsqueda ternaria implica pasar un string para comprobar si algún camino del árbol lo contiene.

La complejidad temporal de buscar en un árbol de búsqueda ternaria equilibrado es O (log n).

2.2.2 Algoritmos de Búsqueda

Suponiendo que el árbol está ordenado, podemos tomar una llave e intentar insertarlo dentro del árbol. Los siguientes algoritmos están generalizados para árboles de búsqueda binaria, pero la misma idea puede aplicarse a otros tipos de árboles.

Recursivo

busqueda-recursiva(llave, nodo)

1. if nodo es NULL
   return ARBOL-VACIO
2. if llave $<$ nodo.llave
   return busqueda-recursiva(llave, nodo.hijo-izquierdo)
3. else if llave $>$ nodo.llave
   return busqueda-recursiva(llave, nodo.hijo-derecho)
4. else
   return nodo

Busqueda iterativa

Busqueda-iterativa(llave, nodo)

1. nodoActual := nodo
2. while nodoActual is not NULL
   if nodoActual.llave = llave
   return nodoActual
   else if nodoActual.llave > llave
   nodoActual := nodoActual.hijo-izquierdo
   else
   nodoActual := nodoActual.hijo-derecho

En un árbol ordenado, el mínimo se encuentra en el nodo más a la izquierda, mientras que el máximo se encuentra en el nodo más a la derecha

Mínimo

busca-minimo(nodo)

1. if nodo is NULL
   return ARBOLVACIO
2. minimo := nodo
3. while minimo.hijo-izquierdo is not NULL
   minimo := minimo.hijo-izquierdo
4. return minimo.llave

Máximo

busca-maximo(nodo)

1. if nodo is NULL
   return ARBOL-VACIO
2. maximo := nodo
3. while maximo.hijo-derecho is not NULL
   maximo := maximo.hijo-derecho
4. return maximo.llave

2.3 N-gramas

En Inteligencia Artificial, procesamiento de lenguaje natural, Bioinformática, o recuperación de información se llama n-grama a una subsecuencia de n elementos consecutivos en una secuencia dada. Si n=2 se denominan bigramas; n=3, trigramas; para n>=4 entonces se llaman genéricamente n-gramas o Modelos de Markov de orden (n-1).

Los n-gramas son de gran utilidad en el tratamiento de textos, la determinación del lenguaje de documentos, el procesamiento estadístico de los mismos o el descubrimiento o predicción de genes en tanto estos son subsecuencias dadas dentro del material genético.

2.3.1 Definición

Sea una secuencia S de elementos ordenados $s_1, s_2, s_3, ..., s_k, ...$ se denomina n-grama a cualquier subsecuencia $A = s_{i+1}, s_{i+2}, ..., s_{i+n}$, donde i es un valor entre 0 y $|S| - n$ para garantizar que la longitud de A sea siempre n o lo que es lo mismo $|A| = n$; $n > 1$.

Los casos particulares $n = 2$, $n = 3$ reciben los nombres de bigramas y trigramas respectivamente.

Como datos adicionales, la definición abarca el caso particular de un n-grama, pero en la práctica es preferible pensarlo como el conjunto de los mismos que componen la secuencia dada.

Debido a limitaciones computacionales y a la normalmente naturaleza abierta de los problemas (suele haber infinitos elementos posibles), se suele asumir que cada elemento solo depende de los últimos n elementos de la secuencia.

2.4 Criterios estadísticos

2.4.1 *Criterio de información de Akaike (AIC)*

El criterio de información de Akaike (AIC) es una medida de la calidad relativa de un modelo estadístico, para un conjunto dado de datos. Como tal, el AIC proporciona un medio para la selección del modelo.

AIC maneja un trade-off entre la bondad de ajuste del modelo y la complejidad del modelo. Se basa en la entropía de información: se ofrece una estimación relativa de la información perdida cuando se utiliza un modelo determinado para representar el proceso que genera los datos.

AIC no proporciona una prueba de un modelo en el sentido de probar una hipótesis nula, es decir AIC no puede decir nada acerca de la calidad del modelo en un sentido absoluto. Si todos los modelos candidatos encajan mal, AIC no dará ningún aviso de ello.

En el caso general, el AIC es: $AIC = 2\mathrm{k} - 2\ln(\mathrm{L})$ donde k es el número de parámetros en el modelo estadístico , y L es el máximo valor de la función de verosimilitud para el modelo estimado.

Dado un conjunto de modelos candidatos para los datos, el modelo preferido es el que tiene el valor mínimo en el AIC. Por lo tanto AIC no sólo recompensa la bondad de ajuste, sino también incluye una penalidad, que es una función creciente del número de parámetros estimados. Esta penalización desalienta el sobreajuste (aumentando el número de parámetros libres en el modelo mejora la bondad del ajuste, sin importar el número de parámetros libres en el proceso de generación de datos).

2.4.2 *Gráfico Cullen y Frey*

El Gráfico de cullen y Frey es un gráfico de correspondencia o relación lineal entre dos variables cuantitativas aleatorias. En palabras más simples se puede definir como un índice utilizado para medir el grado de relación que tienen dos variables, ambas cuantitativas.

Las medidas de distribución nos permiten identificar la forma en que se separan o aglomeran los valores de acuerdo a su representación gráfica. Estas medidas describen la manera como los datos tienden a reunirse de acuerdo con la frecuencia con que se hallen dentro de la información. Su utilidad radica en la posibilidad de identificar las características de la distribución sin necesidad de generar el gráfico. Sus principales medidas son la Asimetría y la Curtosis.

2.4.3 *Asimetría*

Esta medida permite identificar si los datos se distribuyen de forma uniforme alrededor del punto central (Media aritmética). La asimetría presenta tres estados diferentes Fig. 2.1, cada uno de los cuales define de forma concisa como están distribuidos los datos respecto al eje de asimetría. Se dice que la asimetría es positiva cuando la mayoría de los datos se encuentran por encima del valor de la media aritmética, la curva es Simétrica cuando se distribuyen aproximadamente la misma cantidad de valores en ambos lados de la media y

se conoce como asimetría negativa cuando la mayor cantidad de datos se aglomeran en los valores menores que la media.

Figura 2.1: Estados asimetricos

El Coeficiente de asimetría, se representa mediante la ecuación matemática, $g_1 = \frac{\frac{1}{n}\sum(X_i-\overline{X})^3 n_i}{(\frac{1}{n}\sum(X_i-\overline{X})^2 n_i)^{\frac{3}{2}}}$

Donde $(g1)$ representa el coeficiente de asimetría de Fisher, (X_i) cada uno de los valores, $(\overline{X})$ la media de la muestra y (n_i) la frecuencia de cada valor. Los resultados de esta ecuación se interpretan:

$(g1 = 0)$: Se acepta que la distribución es Simétrica, es decir, existe aproximadamente la misma cantidad de valores a los dos lados de la media. Este valor es difícil de conseguir por lo que se tiende a tomar los valores que son cercanos ya sean positivos o negativos (± 0.5).

$(g1 > 0)$: La curva es asimétricamente positiva por lo que los valores se tienden a reunir más en la parte izquierda que en la derecha de la media.

$(g1 < 0)$: La curva es asimétricamente negativa por lo que los valores se tienden a reunir más en la parte derecha de la media.

Desde luego entre mayor sea el número (Positivo o Negativo), mayor será la distancia que separa la aglomeración de los valores con respecto a la media.

2.4.4 Curtosis

Esta medida determina el grado de concentración que presentan los valores en la región central de la distribución. Por medio del Coeficiente de Curtosis, podemos identificar si existe una gran concentración de valores (Leptocúrtica), una concentración normal (Mesocúrtica) ó una baja concentración (Platicúrtica).

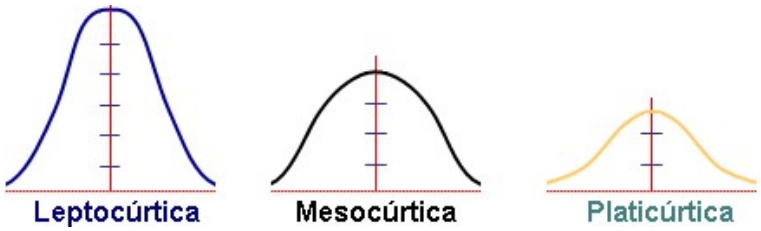

Figura 2.2: Coeficiente de curtosis

Para calcular el coeficiente de Curtosis se utiliza la ecuación $g_2 = \frac{\frac{1}{n}\sum(X_i-\overline{X})^4 n_i}{(\frac{1}{n}\sum(X_i-\overline{X})^2 n_i)^2} - 3$

Donde $(g2)$ representa el coeficiente de Curtosis, (Xi) cada uno de los valores, $(\overline{X})$ la media de la muestra y (n_i) la frecuencia de cada valor. Los resultados de esta fórmula se interpretan:

$(g2 = 0)$ la distribución es Mesocúrtica: Al igual que en la asimetría es bastante difícil encontrar un coeficiente de Curtosis de cero (0), por lo que se suelen aceptar los valores cercanos $(\pm 0.5 aprox.)$.

$(g2 > 0)$ la distribución es Leptocúrtica

$(g2 < 0)$ la distribución es Platicúrtica

3
Metodología

"Existe un lenguaje que va más allá de las palabras."

— Paulo Coelho

Para dar una solución al problema de la segmentación de palabras se realizó una aproximación seleccionando los modelos de n-gramas sobre los de redes neuronales, basándonos en el trabajo de Doval & Gómez-Rodríguez (2019). Esta aproximación está basada en separar las palabras en pequeñas secciones llamadas gramas, por ejemplo, la palabra "hola-mundo", puede ser segmentado en dos partes, de teniendo "hola" (primera grama) y "mundo" (segundo grama). Esta aproximación que fue desarrollada por norvig, si bien, logró un alto grado de precisión, algunos puntos como la determinación de las probabilidades para palabras desconocidas, se da de manera empírica, y no se demuestra el porqué de esa distribución propuesta. El desarrollo de este trabajo se explica detalladamente en los capítulos siguientes.

3.1 *Corpus de entrenamiento y validación*

***Corpus*:** definido por el grupo EAGLES (*Expert Advisory Group on Languaje Engineering Standars* 1996), como -una colección de piezas de lenguaje que se seleccionan y ordenan de acuerdo con criterios lingüísticos explícitos para ser utilizadas como una muestra del idioma-, es normalmente usado para entrenar y estimar modelos lingüísticos.

3.1.1 Selección de fuentes

Para construir nuestro *corpus* se definieron dos tipos de fuentes: formal e informal, definidas de tal manera que en la fuente formal se encuentran aquellos textos en los cuales podemos "confiar", es decir, frases en libros de escritores reconocidos a nivel nacional y noticias en revistas serias y populares, sabiendo de antemano que son una fuente de texto protocolar, revisado y editado para tener el mínimo de errores ortográficos. y la fuente informal en la que se encuentran textos escritos por usuarios populares de redes sociales (en este caso Twitter), que conforman un amplio espectro del lenguaje informal que se quiere analizar, sin que sea una jerga propia de un conjunto reducido de usuarios, y sí, la expresión

de mensajes que desean ser leídos por una mayoría, pero que pueden tener las características o "comodidades" propias del canal donde se emiten. Aunque es cierto que la temática y la homogeneidad del *corpus* es difícil de establecer, ya que cada usuario puede expresar en cada frase lo que desee (aspectos no relacionados necesariamente con frases anteriores).

Respecto de las fuentes formales, se seleccionaron algunos escritores relevantes para la sociedad mexicana. La selección de las fuentes se realizó buscando algunos de los escritores más importantes de México en sus respectivas áreas y de diferentes períodos temporales, esto con la finalidad de contar con un léxico variado, dando como resultado: Benito Taibo (Persona normal), Juan Rulfo (El llano en llamas), Laura Esquivel (Como agua para chocolate), Octavio Paz (Poemas varios) y Pablo Neruda (Poemas varios). Así como la selección de noticias, realizada mediante la obtención de noticias emitidas por el CONACYT (Consejo Nacional de Ciencia y Tecnología), esto para tener una fuente confiable de noticias en México con variedad en el uso del lenguaje.

La metodología utilizada para segmentar el área de aplicación se limitó a seleccionar entre el idioma español, específicamente la variedad del español mexicano, que es el conjunto de dialectos y sociolectos del idioma español que se habla en territorio mexicano, excluyendo el español yucateco.

Para el tipo de fuente informal los criterios de selección de estos usuarios se establecieron según el número de seguidores con el *ranking* más alto. En torno a este, están los usuarios más "famosos" (*influencers*) en sus respectivas categorías. Toledo Bastos (2014) expone una lista con 12 categorías: política, altruismo, eventos, tecnología, juegos, idiomas, música, personalidad, películas, celebridades, estilo de vida y deportes. Basándose en un conjunto de datos que contiene 1,905,989 de *tweets* y más de 14 mil millones de usuarios de Twitter. Con estas características los *influencers* fueron seleccionados delimitando el área geográfica (México), el idioma (español) y la categoría; resultando en los siguientes usuarios de Twitter: @CarlosLoret (política, eventos, celebridades), @ChumelTorres (celebridades y estilo de vida), @ElJuanpaZurita (personalidad, altruismo y estilo de vida), @werevertumorro (juegos, personalidad, estilo de vida), @yuyacst (estilo de vida y celebridades).

Para visualizar los textos analizados es posible acceder a ellos mediante el hipervínculo siguiente: (GitHub) textos informales: tweets.dat y textos formales en la carpeta librostext.

3.1.2 Captura del texto

Recursos y herramientas utilizadas

La captura de este texto se realizó de manera digital, obteniendo las fuentes formales mediante libros y noticias en documentos de texto (.txt). Las noticias se obtuvieron de la página oficial del CONACYT (`https://www.conacyt.gob.mx/index.php/comunicados`). La obtención de los mensajes publicados por usuarios de México, estos mensajes se recopilan mediante una API (*Application Programming Interface*) llamada *tweepy* que comunica la plataforma de Twitter, programando en el lenguaje de programación Python3.

La captura de los libros se realizó seleccionando las fuentes de manera digital, y capturándolas manualmente en formato "textfile" (extensión txt), y posteriormente almace-

nándolos en una base de datos en “Python” del tipo cpickle que es un algoritmo para la serialización y deserialización de un objeto estructurado de Python.

3.1.3 Pre-procesamiento

A los textos ya en formato “txt” les fue aplicado un preprocesamiento para eliminar todos los números, convertir todo el texto en minúsculas, y posteriormente separar todos los párrafos en diferentes vectores, a estos párrafos se les dio otro procesamiento para encontrar cada palabra de manera independiente (“tokenizado”) y agregada a una bolsa de palabras. La construcción del corpus se realizó en un objeto del tipo diccionario, en el cual obtuvimos las repeticiones totales de todas las fuentes (frecuencia de aparición), y la longitud de cada token (“palabra”). Este preprocesamiento se realizó con las instrucciones básicas de Python y con ayuda de la biblioteca *Regular Expression o re*, que fue usada para revisar si un *string* contiene un especifico patrón de búsqueda.

Para homogeneizar los tweets, como a los anteriores textos se realizó un filtrado de texto, donde su función principal era eliminar hipervínculos y palabras basura como RT (’retweet’), la transformación de todo el texto en letras minúsculas, la eliminación de tweets duplicados y finalmente el tokenizado de cada tweet. Esta tarea se realizó con las herramientas anteriormente mencionadas.

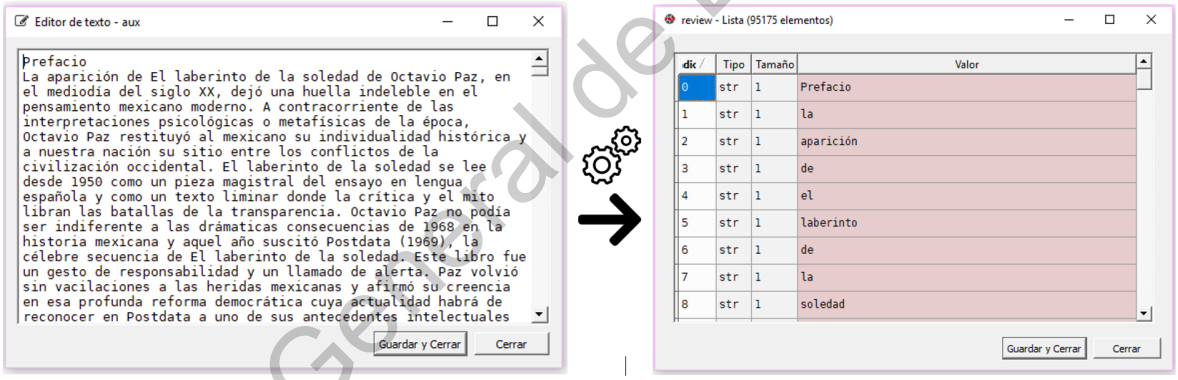

Figura 3.1: Preprocesamiento de texto plano

3.1.4 Segmentación de corpus

Para poder hacer uso del corpus en el trabajo de investigación es necesario segmentarlo en una parte de entrenamiento (la parte más grande del *corpus*) y la parte de validación (la parte que dará validez al proyecto).

La segmentación de estos *corpus* no se realizó de manera tradicional ya que se seleccionaron dos tipos de fuentes, pero solo una es candidata para validar. Por lo que el total de palabras de los textos formales se asignó a la parte de *corpus* de entrenamiento, mientras que se segmentó un 70 % de los textos informales para la parte del *corpus* de entrenamiento y un 30 % para el *corpus* de validación.

3.2 Modelo probabilístico

Se propone un modelo probabilístico en que *"la ocurrencia de una oración, puede entenderse como la probabilidad de ocurrencia de todas las palabras que la integran"*. En este sentido se tiene la siguiente expresión:

$$P[S] = P\left[\bigcap_{i=1}^{n} W_i\right] \tag{3.1}$$

Donde S es la oración, W_i es la i-ésima palabra que forma la oración y n el total de palabras en S.

Para poder entender mejor la definición anterior podemos iniciar con entender que tenemos una probabilidad condicional definida como:

$$p(y \bigcap x) = \frac{P(y|x)}{p(x)} \tag{3.2}$$

Si sabemos que *la probabilidad de que* y *y* x *sucedan juntas, es la probabilidad de* y *dado* x *sobre la probabilidad de* x.

Realizando algunas simples manipulaciones algebraicas podemos darnos cuenta de que $P(y|x) = \frac{p(y \bigcap x)}{p(x)}$ multiplicado por $p(x)$ resulta en $P(y|x)p(x) = p(y \bigcap x)$, y si utilizamos el mismo razonamiento para su análogo $P(x|y) = \frac{p(y \bigcap x)}{p(y)}$ obtenemos $P(x|y)p(y) = p(y \bigcap x)$, ahora si igualamos las ecuaciones en $p(y \bigcap x)$ y dividimos ambos lados por la expresión $p(x)$, obtenemos la ecuación clásica de Bayes.

$$P(y|x) = \frac{p(y|x)p(y)}{p(x)} \tag{3.3}$$

Sabiendo esto podemos reescribir la ecuación de tal forma que:

$$P\left[W_1 \bigcap W_2\right] = P[W_1]P[W_2|W_1] \tag{3.4}$$

Además, la probabilidad condicional se puede tratar como una probabilidad tradicional por lo tanto satisface 3 axiomas de probabilidad. El primer axioma:

$$P[W_1|W_2] = \frac{P[W_1 \bigcap W_2]}{P[W_2]} \tag{3.5}$$

y esta probabilidad condicional no es negativa y es como máximo 1 porque $P[W_1|W_2] \leqslant P[W_2]$. El segundo axioma.

$$P\left[S \bigcap W_2\right] = \frac{P[S \bigcap W_2]}{P[W_2]} \tag{3.6}$$

Donde S es el espacio muestral (la oración completa), y $P[S] = 1$, podemos notar que:

$$\frac{P\left[S \bigcap W_2\right]}{P[W_2]} = \frac{P\left[W_2\right]}{P[W_2]} = 1 \tag{3.7}$$

por lo que se cumple el segundo axioma.
Para el tercer axioma tenemos para una colección $E_i, i = 1, 2, ...$ de eventos mutuamente exclusivos donde:

$$P\left[\bigcup_{i=1}^{\infty} E_i | W_2\right] = \frac{P\left[\bigcup_{i=1}^{\infty}(E_i \bigcap W_2)\right]}{P[W_2]} \tag{3.8}$$

Para el numerador podemos usar el tercer axioma porque todos los eventos son nuevamente mutuamente excluyentes, así que:

$$P\left[\bigcup_{i=1}^{\infty} E_i | W_2\right] = \frac{P\left[\sum_{i=1}^{\infty}(E_i \bigcap W_2)\right]}{P[W_2]} = \sum_{i=1}^{\infty} \frac{P[E_i W_2]}{P[W_2]} = \sum_{i=1}^{\infty} P[E_i | W_2] \tag{3.9}$$

Esto verifica el tercer axioma de la densidad condicional. Entonces de la ecuación (3.4) podemos decir que:

$$P\left[W_1 \bigcap W_2 | E\right] = P[W_1|E]P[W_2|W_1 E] \tag{3.10}$$

Si manejamos al evento E, como los eventos de ocurrencia de las palabras segmentadas de la oración W_i, donde $i = 1, 2, ..., n$ siempre y cuando se maneje el mismo tipo de evento podemos escribir.

$$P\left[W_1 \bigcap W_2 \bigcap, ..., \bigcap W_n\right] = P\left[W_1 \bigcap \left(W_2 \bigcap W_3 \bigcap, ..., \bigcap W_n\right)\right] \tag{3.11}$$

$$= P\left[W_1\right] P\left[\left(W_2 \bigcap W_3 \bigcap, ..., \bigcap W_n\right) | W_1\right] \tag{3.12}$$

$$= P\left[W_1\right] P\left[W_2 | W_1\right] P\left[\left(W_3 \bigcap W_4 \bigcap, ..., \bigcap W_n\right) | W_2 \bigcap W_1\right] \tag{3.13}$$

Si continuamos con estos pasos podemos concluir en:

$$= P\left[W_1\right] P\left[W_2 | W_1\right] P\left[W_3 | W_2 \bigcap W_1\right] ... P\left[W_n | W_{n-1} \bigcap W_{n-2} \bigcap, ..., \bigcap W_1\right] \tag{3.14}$$

Ahora si expresamos la ecuación (3.1) de forma tal que podamos llegar a una expresión más clara tenemos que:

$$P\left[\bigcap_{i=1}^{n} W_i\right] = P\left[W_1\right] \prod_{j=2}^{n} P\left[W_j | \cap_{i=1}^{j-1} W_i\right] \tag{3.15}$$

De acuerdo a la ecuación (3.15) *la probabilidad de que en la i-ésima variable tome un valor, depende de los valores de todas las palabras que le anteceden.* Tomando en cuenta el significado de la oración, podemos darnos cuenta de que *no existe independencia entre palabras*.

Aunque este modelo se adecua a la realidad, resulta difícil de analizar, esto ya que sí n es muy grande se vuelve necesario un gran poder computacional para poder realizar el procesamiento, ya que las posibilidades de creación de oraciones se magnifica en cuanto más palabras componen la oración). Por ejemplo:

"comerpapasconsalsa" puede segmentarse de la siguiente manera:

- comerpapasconsalsa 1 palabra
- c omerpapasconsalsa 2 palabras
- co merpapasconsalsa 2 palabras
- ...
- comer papas con salsa 4 palabras
- ...
- c o m e r p a p a s c o n s a l s a 18 palabras

Para conocer todas las posibles formas de segmentar las palabras en necesario conocer las formas de acomodar todos los espacios entre cada letra para formarlas tal que, la cantidad de combinaciones posibles a formar es 2^{c-1} donde c es el número total de letras en la oración. Por lo tanto "comerpapasconsalsa" resulta tener $2^{19-1} = 262,144$ posibles combinaciones para computar. Teniendo en cuenta que el cálculo de la probabilidad de una palabra es una operación, resultaría que el número de operaciones a computar debería ser el mismo número, pero para separaciones diferentes de una palabra, cada operación de i separaciones, es necesario realizar operaciones internas de la forma $\sum_{i=1}^{c}(i^2(\sum_{j=1}^{i-1} j) + 1)$. Por ejemplo, para las oraciones con 4 caracteres tendríamos que realizar 8 operaciones agregándole 16 operaciones internas, para 5 tendríamos que realizar 16 operaciones y 57 operaciones internas, y para el ejemplo de 19 caracteres tendríamos 262,144 operaciones sumando 8,597,496,601 operaciones internas, resultando en 8,597,758,745 cálculos de probabilidades en esa oración.

Según Halpern (2015), el cálculo para 5-gramas resultaría de más de 30gb de memoria RAM, para una oración simple, por lo que no podemos escoger este modelo, por limitaciones computacionales. Basándonos en los cálculos anteriores podemos decir que es casi

imposible manejar el modelo anterior de manera completa, por lo que se proponen alternativas considerando dos modelos de lenguaje más simple.

Tomando como referencia a Norvig y a Goldwater, se propone un modelo basado en uni-gramas y uno basado en bi-gramas. El primer modelo (uni-gramas) se toma considerando que la probabilidad de existencia de cada palabra es independiente de las otras, es decir: *la probabilidad de la i-ésima palabra es independiente de todas las palabras anteriores y posteriores.*

$$P\left[\bigcap_{i=1}^{n} W_i\right] = \prod_{i=1}^{n} P[W_i] \tag{3.16}$$

En el segundo modelo (Bi-gramas) considera que la probabilidad de existencia de cada palabra es dependiente únicamente de la palabra anterior, es decir: *la probabilidad de la i-ésima palabra es dependiente de la palabra que la antecede.* La diferencia de este modelo al anterior está basado, en tomar en cuenta la el contexto de la oración en el sentido mínimo (**propiedad de Markov**), es decir que conectamos la oración completa con el enlace mínimo para reducir el proceso computacional.

Para poder definir el modelo, podemos iniciar definiendo el proceso de Markov, que es un proceso aleatorio con la propiedad de que *dado el valor actual del proceso* X_t*, los valores futuros* X_s *para* $s > t$ *son independientes de los valores pasados* X_u *para* $u < t$. Es decir, que si tenemos la información presente del proceso, saber cómo llegó al estado actual no afecta las probabilidades de pasar a otro estado en el futuro.

En el caso discreto la definición precisa es la cadena de Markov a tiempo discreto, que es una sucesión de variables aleatorias X_n, para todo n y cualesquiera estados $i_0, i_1, ..., i_n, j$ en ε, conocido como espacio de estados, y que satisface la siguiente propiedad.

$$P(X_{n+1} = j | X_0 = i_0, ..., X_{n-1} = i_{n-1}, X_n = i_n) = P(X_{n+1} = j | X_n = i_n) \tag{3.17}$$

Para todo n y cualesquiera estados $i_0, i_1, ..., i_n, j$ en ε, la propiedad recibe el nombre de **propiedad de Markov**. En general, es cómodo designar los estados de la cadena usando los enteros no-negativos $0, 1, 2, ...$ y diremos que X_n está en el estado i si $X_n = i$. La probabilidad de que X_{n+1} esté en el estado j dado que X_n está en el estado i es la probabilidad de transición en un paso de i a j y se denota como P_{ij}^{nn+1}:

$$P_{ij}^{nn+1} = P(x_{n+1} = j | X_n = i) \tag{3.18}$$

Con estas definiciones podemos reescribir el modelo (3.15) adecuado a bigramas de la forma:

$$P\left[\bigcap_{i=1}^{n} W_i\right] = \prod_{i=1}^{n} P[W_i | W_{i-1}] \tag{3.19}$$

3.3 Computo probabilístico

3.3.1 Cálculo de probabilidades de palabras

El cálculo de probabilidades de palabras se realiza teniendo en cuenta dos consideraciones, la primera es que nuestro calculo es "no sensitivo a mayúsculas o minúsculas", el segundo es que son tomados en cuenta los acentos y caracteres especiales como la "ñ" (en el caso de Norvig, esta consideración no existe ya que en el idioma inglés no se tienen acentos ni caracteres especiales).

Palabras conocidas

Para el cálculo de uni-gramas que existen dentro de nuestro corpus, utilizamos la función de distribución empírica:

Donde, $(W_1, ..., W_n)$ sea una muestra, $c_1, ...c_k$ los diferentes valores que toman los W_1. Para $h = 1, ..., k$ se denota:

$$n_h = \sum_{i=1}^{n} I_{c_h}(W_i) \tag{3.20}$$

El número de veces que el valor c_h aparece o sea el efectivo del valor c_h. La distribución empírica de la muestra es la ley de probabilidad P sobre el conjunto $\{c_1, ..., c_k\}$, tal que:

$$P(c_h) = \frac{n_h}{n} \tag{3.21}$$

De la cual podemos desarrollar para nuestro problema como:

$$P[W] = \frac{1}{N} \sum_{i=1}^{N} I(W = W_i^C) \tag{3.22}$$

donde:
I es la función indicadora
N es el total de palabras en nuestro *corpus*
W_i^C es la i-esima palabra en el corpus

La función indicadora de un subconjunto $A \subseteq X$ es una función definida en el conjunto X, y que indica la pertenencia, o no, de cada elemento de X al subconjunto A, al asignar el calor 1 a todos los elementos de A y el valor 0 a todos los elementos de $X \setminus A$ (no incluidos en A).

Sabiendo que la función indicadora del subconjunto A del conjunto X es una función:

$$1_A : X -> \{0, 1\} \tag{3.23}$$

definida como :

$$1_A(x) = \begin{cases} 1 & \text{si } x \in A \\ 0 & \text{si } x \notin A \end{cases} \tag{3.24}$$

Para al cálculo de probabilidades de bi-gramas conocidos usamos el mismo principio, simplemente cambiando los factores de la función indicadora:

$$P[W|W_{-1}] = \frac{1}{N} \sum_{i=1}^{N} I(B = B_i^C) \tag{3.25}$$

donde:
$B = (W, W_{-1})$
N es el total de bi-gramas en nuestro *corpus*
B_i^C es el i-esimo bi-grama en el corpus

Los dos cálculos anteriores, los realizamos una vez al realizar cualquier procesamiento y lo guardamos en una variable del tipo diccionario (esto para reducir el proceso computacional).

Palabras desconocidas

El cálculo de las palabras que no aparecen en nuestro corpus lo llamaremos cálculo de palabras desconocidas, Norvig propone una distribución para calcular la probabilidad de esta. En este cálculo, Norvig plantea de manera empírica para el idioma inglés la siguiente distribución:

$$P[W] = \frac{10}{N} * \frac{1}{10^{len(W)}} \tag{3.26}$$

len(W) = Longitud de la palabra
N = es el total de palabras en el *corpus*

El cálculo de probabilidad a las palabras desconocidas de Norvig se basa en una distribución (que en este trabajo llamaremos distribución de Norvig) creada de manera empírica mediante la observación de las palabras conocidas más repetidas y notando que las que más se repetían eran las de longitud 6, y tomando en cuenta el total de palabras. Así mismo se usó un compensador de valor 10, que fue escogido de manera arbitraria para que el análisis de palabras grandes no impactara en el cálculo.

Como podemos ver en la gráfica (fig.3.2) en el modelo de Norvig (3.26), las probabilidades van siendo menores entre más grande sea la palabra con una tendencia de forma exponencial decreciente, limitando el tamaño de las palabras a 20, siendo este número seleccionado por Norvig de manera arbitraria.

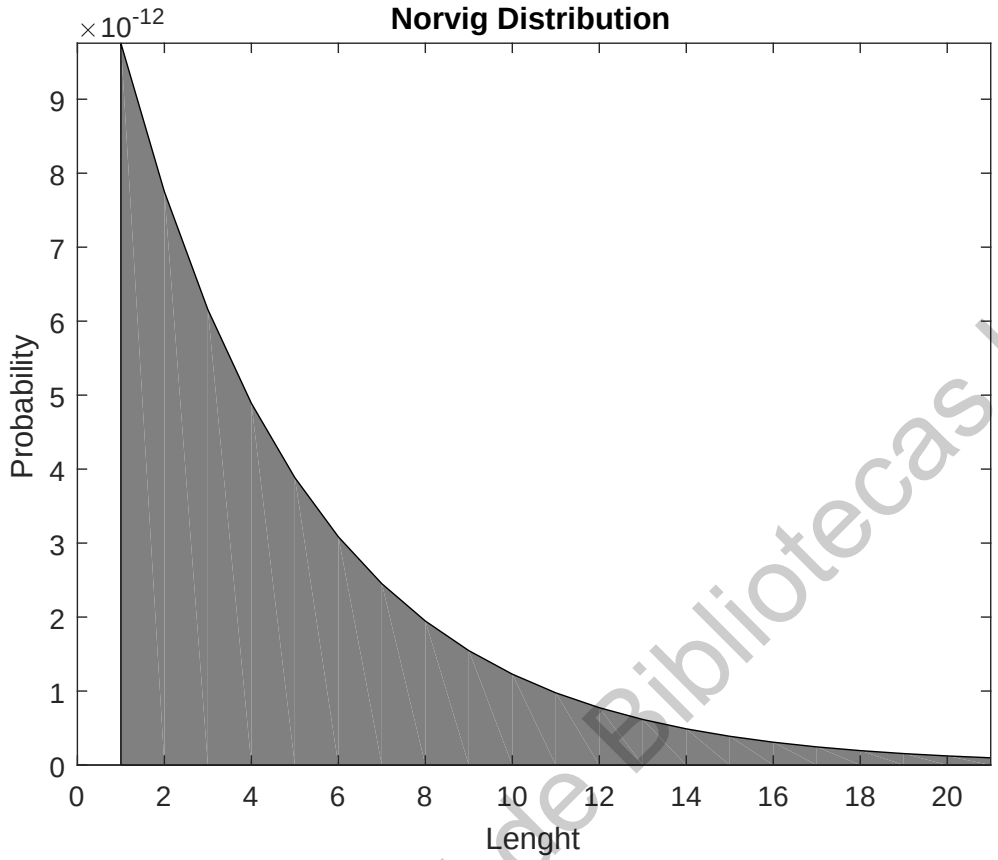

Figura 3.2: Distribución de Norvig

Este modelo desarrollado por Norvig, el menciona que el programa segmenta correctamente las palabras inusuales como "*samsa*" y "*oozy*", en un experimento de 159 oraciones realizado por él, sus resultados son buenos, aunque se presentan dos errores: "*un*","*regarded*" debería ser una palabra y "*sitdown*" deberían ser dos. A pesar de eso la precisión de Norvig, es de $157/159 = 98.7\,\%$, que es muy bueno.

Partiendo de esta distribución, se propuso una nueva distribución, desarrollada para el idioma español mexicano (pudiendo aplicarlo a cualquier idioma con la misma estructura gramatical, es decir, para idiomas como el japones o el israelí no funcionaría).

Primero se buscó la palabra más larga conocida dentro de nuestro *corpus*, que en este caso coincidía en la longitud 26, teniendo en cuenta que la palabra existente dentro de la RAE (Real Academia Española) más grande cuenta con una longitud 23, se determinó que algunas de las palabras con las que nos enfrentábamos eran compuestas. Como podemos ver en la tabla 3.1 la palabra con la longitud 26, es una palabra que cuenta con un error de formato, ya que en la noticia de donde se extrajo, se encontraba con este error, (esto nos deja sobre aviso que tendremos errores en el *corpus*, ya que no podemos evitar los errores que agregamos por la metodología usada para crearlo). Basados en esto consideramos limitar nuestra longitud de palabras a la longitud de la RAE (longitud 23).

Tabla 3.1: Palabras más largas dentro del *corpus*

Word	Length
nordihidrocapsaicina	20.00
electroencefalograma	20.00
anticonstitucionales	20.00
internacionalización	20.00
contrarrevolucionario	21.00
homodihidrocapsaicina	21.00
electroencefalografía	21.00
polihidroxialcanoatos	21.00
electroencefalográfica	22.00
seudointernacionalizado	23.00
universitariosvsaeropuerto	**26.00**

Mediante el gráfico de Cullen and Frey (fig. 3.3) se proporciona un diagrama de sesgo-curtosis para la distribución empírica. En este gráfico, los valores de las distribuciones comunes también se muestran como herramientas para ayudar a elegir las distribuciones que se ajusten a los datos. Para algunas distribuciones (normal, uniforme, logística, exponencial, por ejemplo), solo hay un valor posible para la asimetría y la curtosis (para una distribución normal, por ejemplo, asimetría = 0 y curtosis = 3), y la distribución se representa así por un punto en la trama. Para otras distribuciones, se representan áreas de posibles valores, que consisten en líneas (distribuciones gamma y log normal, por ejemplo), o áreas más grandes (distribución beta, por ejemplo). La distribución de Weibull no está representada en el gráfico, pero se indica en la leyenda que con esta distribución se pueden obtener formas cercanas a las distribuciones log normal y gamma.

Con el fin de tener en cuenta la incertidumbre de los valores estimados de curtosis y asimetría de los datos, el conjunto de datos se inicializó fijando el argumento *bootstrap* a 10. Los valores de asimetría y curtosis correspondientes a las muestras de *bootstrap* se calculan e informan en color amarillo en el diagrama de sesgo-curtosis. Así mismo exponemos los parámetros de asimetría estadística y curtosis para generar una aproximación de manera empírica de aproximaciones a nuestro modelo al igual que los datos estadísticos básicos.

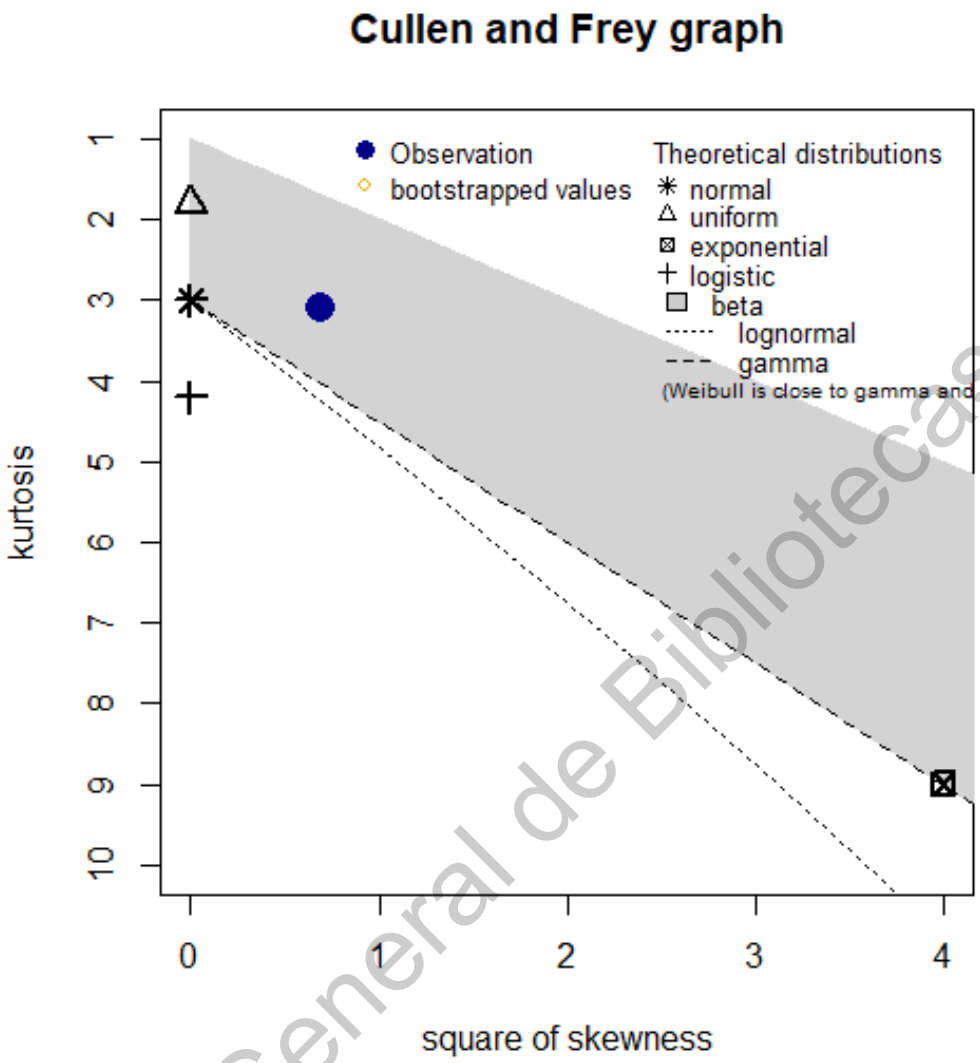

Figura 3.3: Diagrama de sesgo-curtosis (Datos sin filtrar)

Tabla 3.2: Resumen de datos estadísticos sin filtrar

Resumen estadístico	
Min:	1
Max:	26
Mediana:	4
Promedio:	4.834333
Desviación estándar estimada:	3.003821
Asimetría estimada:	0.8336918
Curtosis estimada:	3.082008

Basándonos en esto, ajustamos los modelos a nuestros datos obteniendo los siguientes resultados.

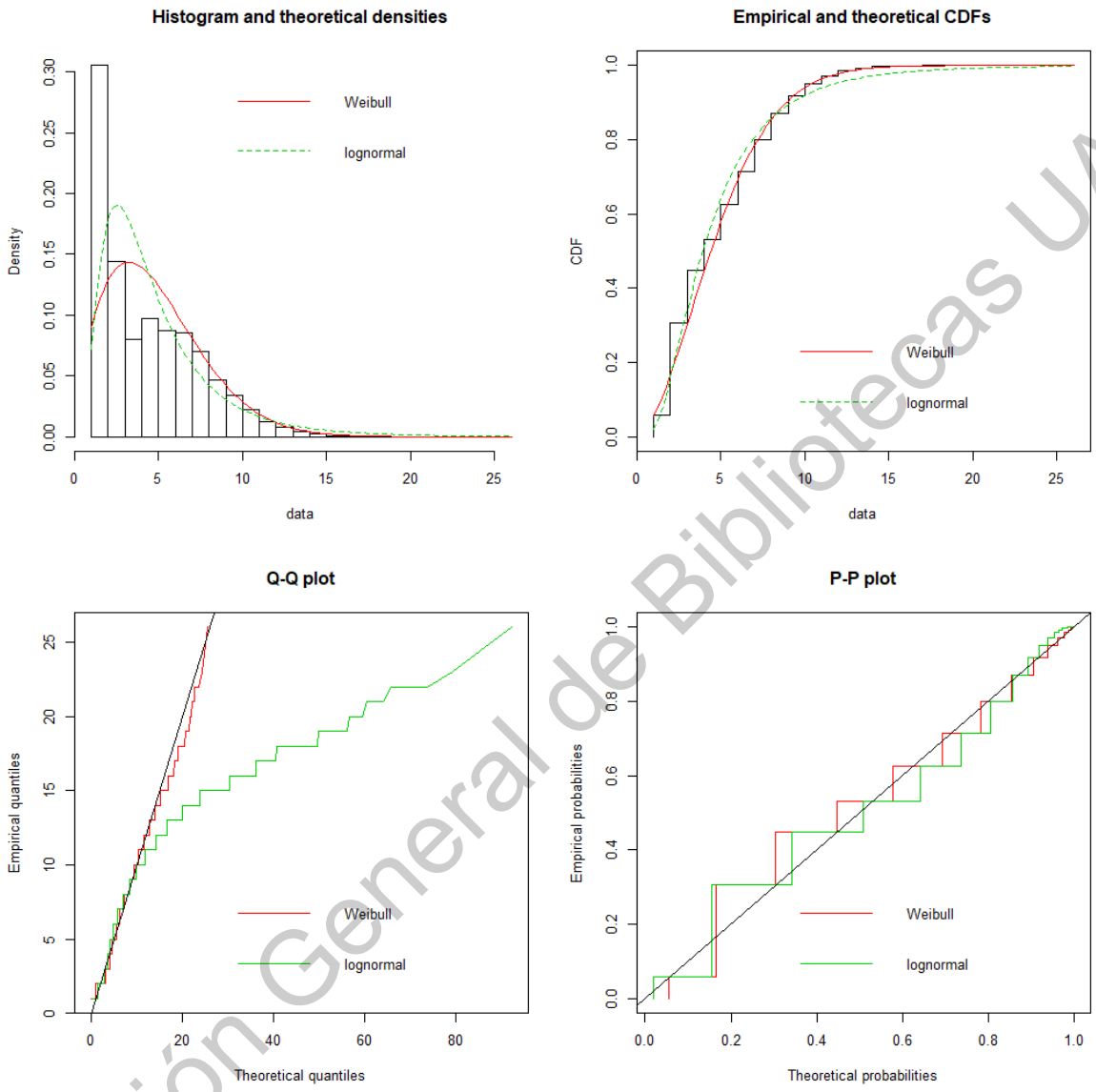

Figura 3.4: Distribuciones sin filtro

Basandonos en el grafico Q-Q, se determinó que la distribución lognormal no era la más adecuada para nuestro modelo. por lo que se decidío usar la distribución weibull para los cálculos iniciales. Basandonos en la observación de los resultados preliminares y el histograma de frecuencias relativas es notable que las palabras con longitud 1, 2 y 3, son las palabras que más se repiten, pero no son las palabras desconocidas que más se ocupan (tratando de longitudes), sin embargo, si manejamos las frecuencias relativas con un tratamiento de ajuste podemos computar mejor estos datos. En nuestro caso, el tratamiento que usamos fue el

multiplicar la frecuencia relativa por la longitud de la palabra, es decir que entre más corta la palabra más pequeña sería su probabilidad de aparecer y viceversa. Esta ecuación se ve de la forma (3.27) y Fig. 3.5:

$$\mathrm{RFT} = \mathrm{len}(Word) * \mathrm{RF} \tag{3.27}$$

donde:
RF = Frecuencia relativa
RFT = Frecuencia relativa tratada

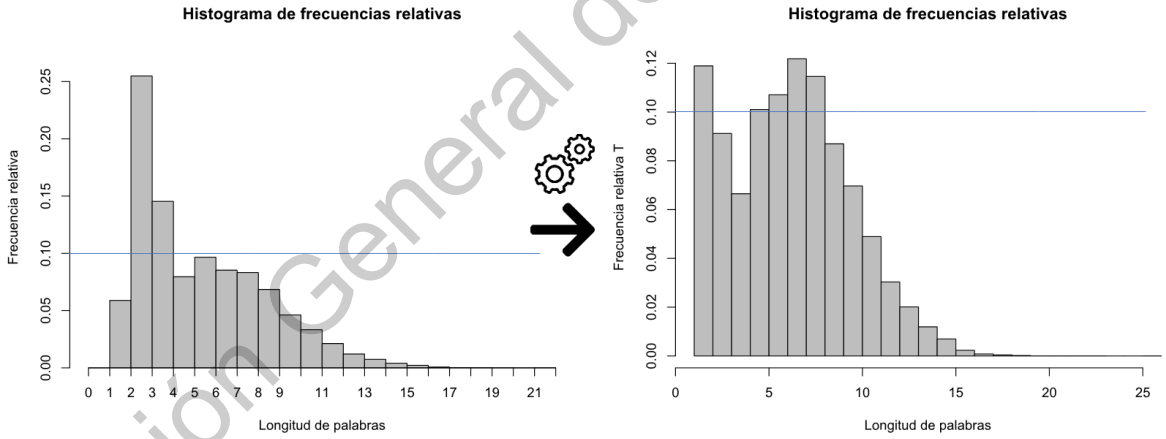

Figura 3.5: Frecuencias relativas vs Frecuencias relativas tratadas

Realizando nuevamente el diagrama de sesgo-curtosis de Cullen and Frey Fig. 3.6 tenemos los siguientes datos Tabla 3.5.

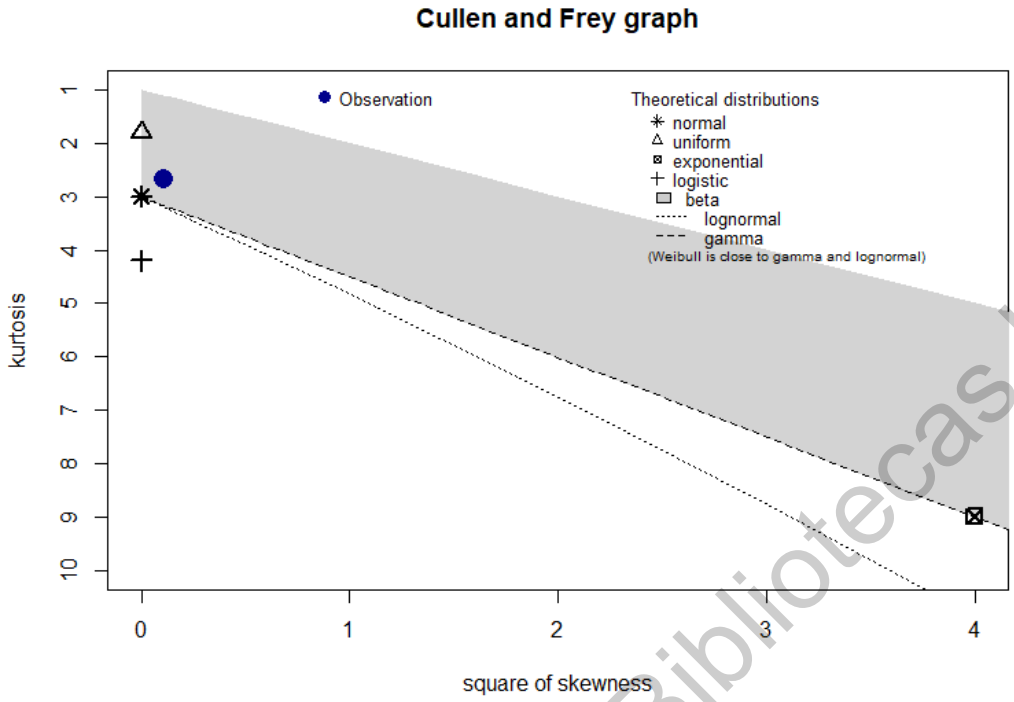

Figura 3.6: Diagrama de sesgo-curtosis (Datos filtrados)

Tabla 3.3: Resumen de datos estadísticos filtrados

Resumen estadístico	
Min:	1
Max:	26
Mediana:	7
Promedio:	6.700758
Desviación estándar estimada:	3.195838
Asimetría estimada:	0.3311568
Curtosis estimada:	2.66736

Teniendo esto en cuenta se ajustaron los modelos con los datos tratados, resultando en los siguientes.

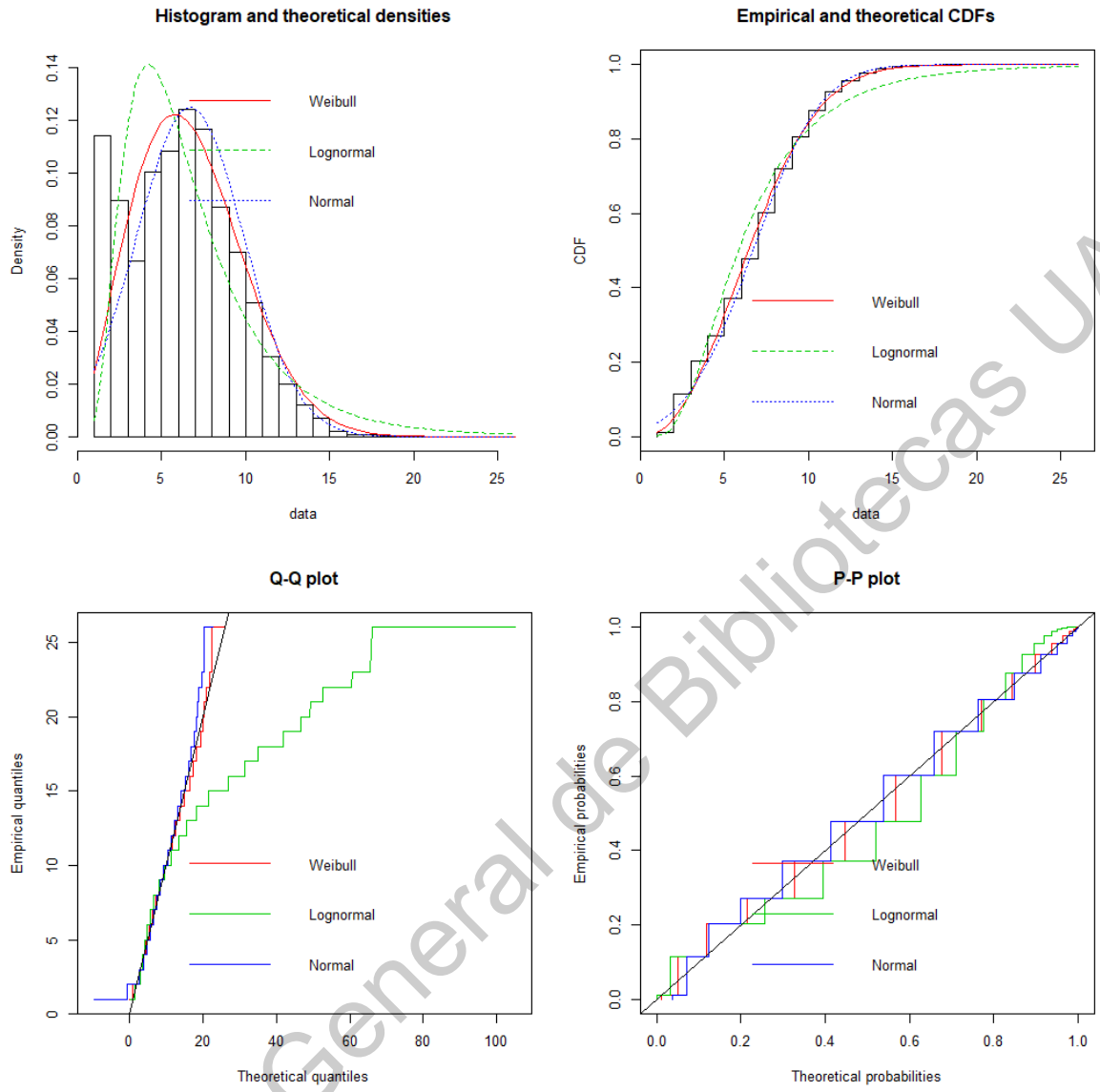

Figura 3.7: Filtered distributions

Los dos modelos que más se acercan a nuestra distribución son la Normal y la Weibull. Por lo que se realizó el ajuste para estos dos modelos teniendo los siguientes resultados.

La función de distribución acumulada (CDF) normal es definida como:

$$F(x) = \frac{1}{2}[1 + erf(\frac{x - \mu}{\sigma * \sqrt{2}})] \tag{3.28}$$

donde:

Tabla 3.4: Resumen de datos estadísticos filtrados

Resumen estadístico	
μ	7.529072 (media)
σ	2.206620 (desviación estándar)
sd μ	0.3128529
sd σ	0.2212203
aic	12070234
bic	12070259

Y la función de distribución acumulada discreta de Weibull es definida como:

$$F(x) = 1 - \exp\left[-\left(\frac{x+1}{\alpha}\right)^{\beta}\right] \tag{3.29}$$

Donde:

Tabla 3.5: Resumen de datos estadísticos filtrados

Resumen estadístico	
α	7.529072 (factor de escala)
β	2.206620 (factor de forma)
Error de forma	0.0836146
Error de escala	0.1713985
aic	11904410
bic	11904436

Observando los resultados de las distribuciones acumuladas (CDF) y mediante el criterio de información Akaike (aic) que es una medida de la calidad relativa de un modelo estadístico, se decidió tomar la distribución Weibull discreta como la distribución mejor ajustada a nuestro modelo, teniendo en cuenta que dado el conjunto de modelos candidatos para los datos, el modelo preferido es el que tiene el valor mínimo en el AIC, el modelo seleccionado para representar a nuestra distribución fue la CFD Weibull discreta.

Este modelo se ajustó a los datos del histograma de frecuencias relativas tratadas obteniendo los datos de la CDF de Weibull discreta de la figura 3.7.

3.4 Algoritmo de optimización

Para resolver el problema de encontrar el mejor candidato con la mayor probabilidad, se creó y enumeró a los candidatos planteando dos estructuras de segmentación. Una basada en arboles binarios simples para minimizar el tiempo de procesamiento, y la segunda en arboles binarios con reglas de composición.

3.4.1 Árbol binario

Los árboles son estructuras de datos muy similares a las listas doblemente enlazadas, en el sentido que tienen punteros que apuntan a otros elementos, pero no tienen una estructura lógica de tipo lineal o secuencial como aquellas, sino ramificada. Su estudio desde el punto de vista matemático pertenece a la teoría de grafos; desde el punto de vista informático son estructuras de datos, lo que significa que cada elemento, denominado nodo u hoja, contiene un valor. Su estudio corresponde a la teoría de bases de datos, y en esta terminología, los nodos que dependen de otros se denominan hijos. Cada hoja puede tener un máximo de hijos, si no tiene ninguno se dice que es un nodo terminal. Un árbol es una estructura de datos no lineal en la que cada nodo puede apuntar a uno o varios nodos. También se suele dar una definición recursiva: un árbol es una estructura compuesta por un dato y varios árboles. Una representación gráfica de los árboles se puede visualizar en la Fig. 3.8.

Un árbol consiste en un nodo (r, denominado nodo raíz) y una lista o conjunto de subárboles $(A_1, A_2, ..., A_k)$.

Si el orden de los subárboles importa, entonces forman una lista, y se denomina árbol ordenado (por defecto un árbol se supone que es ordenado). En caso contrario los subárboles forman un conjunto, y se denomina árbol no ordenado.

Se definen como nodos hijos de r a los nodos raíces de los subárboles $(A_1, A_2, ..., A_k)$.

Si b es un nodo hijo de a entonces a es el nodo padre de b.

Un nodo puede tener cero o más hijos, y uno o ningún padre. El único nodo que no tiene padre es el nodo raíz del árbol.

Un nodo sin hijos se denomina nodo hoja o externo. En caso contrario se denomina nodo interno.

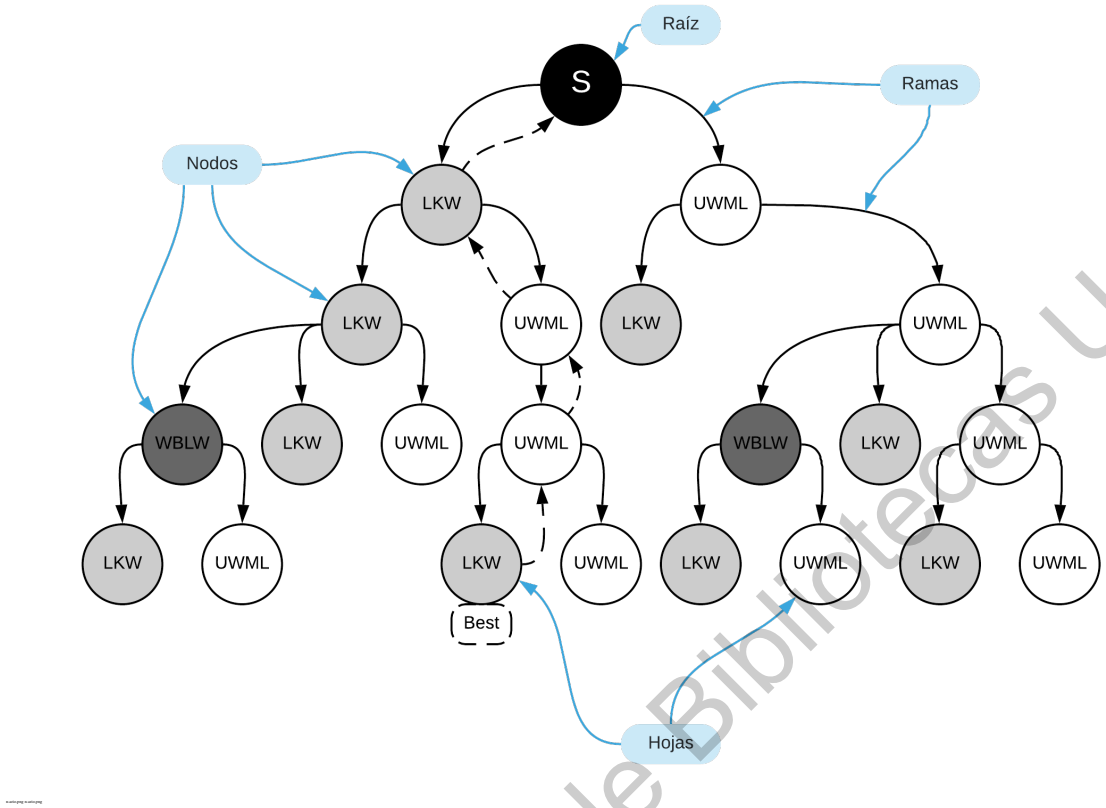

Figura 3.8: Estructura básica de árbol

Un árbol binario es un árbol que o bien está vacío (sin contenido) o bien consta de un nodo raíz con dos subárboles binarios, denominados izquierdo y derecho.

La existencia de árboles vacíos es una convención para que no exista ambigüedad al identificar el subárbol izquierdo y derecho. Se representa por un cuadrado. La altura de un árbol vacío es -1. Cada nodo puede tener cero hijos (subárbol izquierdo y derecho vacíos), un hijo (algún subárbol vacío) o dos hijos.

3.4.2 *Árbol Binario Simple*

Este árbol binario fue estructurado bajo dos reglas simples mostradas en la fig. 3.9, la hoja derecha será para asignar palabras desconocidas con la mayor probabilidad de aparecer (UWML), es decir utilizando la distribución que calculamos con anterioridad, y la hoja izquierda será para asignar la palabra conocida con la mayor longitud (LKW) utilizando un algoritmo voraz (*greedy*) maximizando la función de factibilidad.

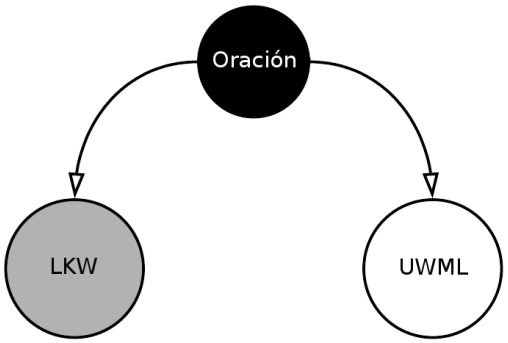

Figura 3.9: Estructura básica de árbol binario simple

El algoritmo voraz es un esquema simple para resolver problemas de optimización, en la que se deben cumplir algunos criterios.

Debe existir una entrada de tamaño n que son los candidatos para formar parte de la solución. Debe existir un subconjunto de esos n candidatos que satisface ciertas restricciones: se llama solución factible. Hay que obtener la solución factible que maximice o minimice la función objetivo y se llama solución óptima.

El esquema voraz procede de la siguiente manera:

1. El conjunto de candidatos escogido es vacío
2. En cada paso, se intenta añadir al conjunto de los escogidos "el mejor" de los no escogidos (sin pensar en el futuro), utilizando una función de selección basada en algún criterio de optimización (puede ser o no ser la función objetivo)
3. Tras cada paso, hay que ver si el conjunto seleccionado se puede completar (i.e., si añadiendo más candidatos se puede llegar a una solución);

   Si el conjunto no se puede completar, se rechaza el último candidato elegido y no se vuelve a considerar en el futuro;

   Si se puede completar, se incorpora al conjunto de escogidos y permanece siempre en él;
4. Tras cada incorporación se comprueba si el conjunto resultante es una solución;
5. El algoritmo termina cuando se obtiene una solución;
6. el algoritmo es correcto si la solución encontrada es siempre óptima;

3.4.3 Árbol Binario Compuesto

El árbol binario compuesto se creó bajo una estructura parecida al anterior, agregando un conjunto de reglas para el manejo de bigramas, logrando que nuestro árbol binario

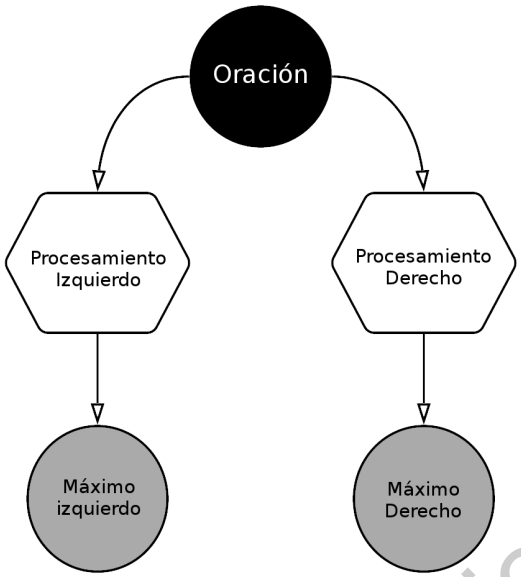

Figura 3.10: Estructura básica de árbol binario compuesto

se comporte de manera simple en algunas ramas y complejo en otras, siempre y cuando se cumplan las reglas prestablecidas.

Procesamiento Los bloques de procesamiento izquierdo y derecho se definieron de la siguiente manera:

Procesamiento izquierdo.

1. Entrada: Oración a procesar (S)
2. Inicializar $\mathtt{tmp} = []$.
3. Buscar palabra más larga conocida (LKW), si existe:
   - Guardar LKW en arreglo tmp.
   - Buscar bigrama conocido (LBK), si existe:
     Guardar LBW en arreglo tmp.
   - Buscar palabra conocida anterior (LKW-1), si existe,
     Guardar (LKW-1) en arreglo tmp.
4. $\mathrm{Máx\_izq} = \mathtt{argmax}_s \in (\mathrm{P}[\mathtt{tmp}])$
5. $\mathrm{S} = \mathrm{S} - \mathrm{Máx\_izq}$
6. Si $\mathrm{S} \neq 0$ repetir hasta que S converja a 0

Procesamiento derecho.

1. Entrada: Oración a procesar (S)
2. Inicializar $\mathtt{tmp} = []$.
3. Calcular la palabra desconocida más probable basados en la CDF Weibull (LUW) y guardarla en arreglo tmp.
4. Buscar la palabra mas larga conocida (sin símbolos, ni acentos) (LKWW), si existe:
   - Guardar LKWW en arreglo tmp.
5. Ejecutar función de búsqueda $\mathrm{Máx\_der} = \mathtt{argmax}_s \in (\mathrm{P}[\mathtt{tmp}])$
6. $\mathrm{S} = \mathrm{S} - \mathrm{Máx\_izq}$
7. Si $\mathrm{S} \neq 0$ repetir hasta que S converja a 0

Con los árboles de oraciones creados, el procedimiento para formar los candidatos es dividido en dos tipos, el primero, para uni-gramas multiplicando la probabilidad de la hoja por la rama que la generó, hasta que se llega a la raíz del árbol, y de ahí se selecciona la mejor oración según cual sea la de probabilidad más alta.

Y el segundo para crear los candidatos de los bigramas, tenemos que considerar que en algunas ocasiones se encontraran bigramas y en otras solamente uni-gramas. En el caso de que existan bigramas se dividirá la probabilidad de la hoja actual entre la probabilidad de la palabra anterior, esto para poder agregar el bigrama sin agregar otra multiplicación, seguido

del paso anterior se multiplica la probabilidad actual por la probabilidad de ocurrencia del bigrama y se activa un marcador de bigrama, para que en el caso de que ocurra otro bigrama después, esté no se borre por el paso de eliminar la probabilidad de la palabra anterior.

3.4.4 Selección del mejor candidato

El mejor candidato simplemente es aquella hoja que cuente con la mayor probabilidad, y posteriormente realizar un corrimiento hacia atrás con el algoritmo recursivo previamente implementado, como se muestra en las figuras 3.11 y 3.12.

$$\text{Best} = \text{argmax}_S \in \text{candidates } P[c] \tag{3.30}$$

Las siguientes gráficas muestran un ejemplo de un árbol binario y las probabilidades que podrían resultar de una oración segmentada, así como la elección del mejor candidato.

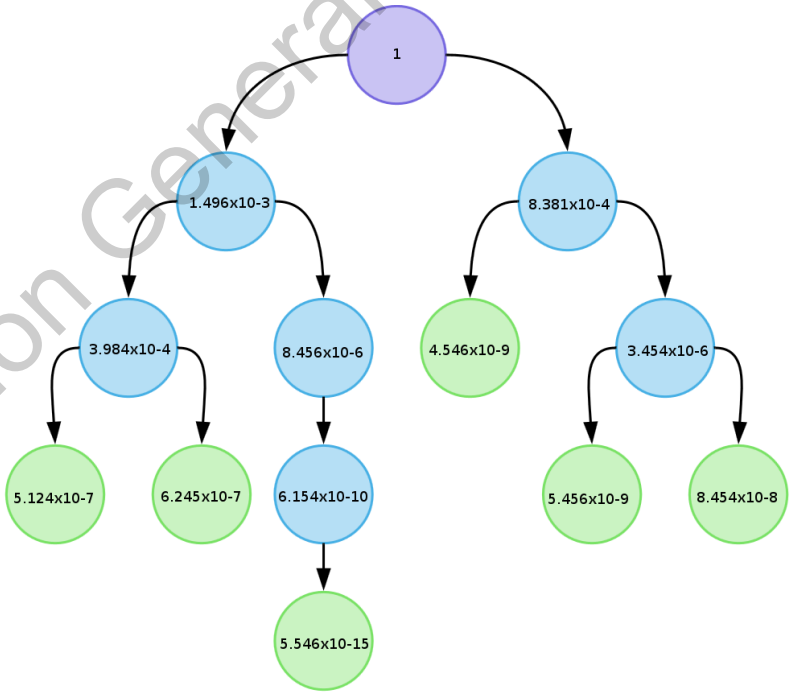

Figura 3.11: Ejemplo de candidatos en árbol binario

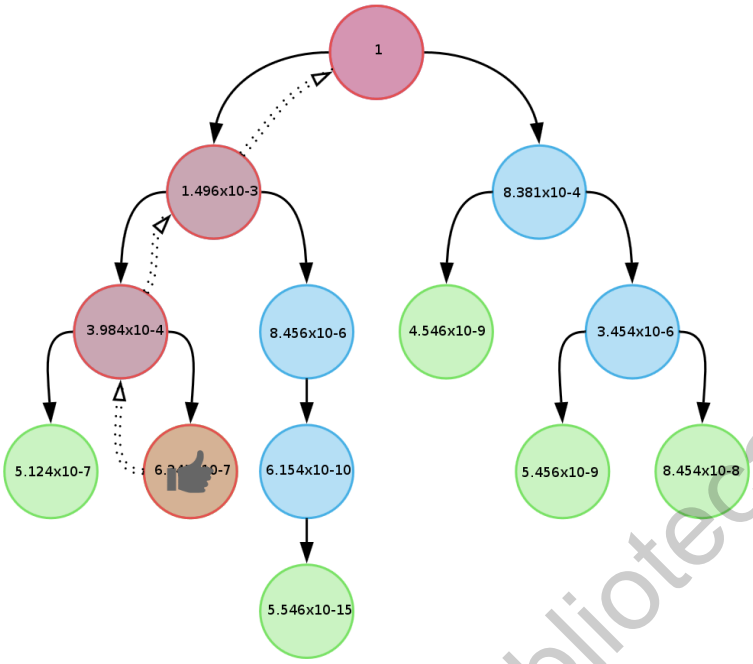

Figura 3.12: Ejemplo de selección del mejor candidato en árbol binario

3.5 Notas y consideraciones

Para poder realizar la experimentación sobre texto informal tenemos que hacer consideraciones para que todo pueda funcionar correctamente, teniéndolas listadas de la siguiente manera:

1. La metodología utilizada para obtener texto se limitó a seleccionar entre el idioma español, la variedad del español mexicano, que es el conjunto de dialectos y sociolectos del idioma español que se habla en territorio mexicano, excluyendo el español yucateco.

2. Las noticias solo fueron obtenidas de una fuente (CONACYT), lo que pudo sesgar la investigación a un tipo de escritura más formal que lo que podemos encontrar en redes sociales.

3. La parte de los tweets que se aplicó para segmentar palabras solo fueron los *hashtags*, ya que la propiedad que los caracteriza es que un *hashtag* solo puede ser escrito sin espacios.

4. El cálculo es no sensitivo a mayúsculas y minúsculas (cosa que podría ayudar a segmentar velozmente las oraciones que se encuentran escritas de esta forma).

5. En caso de existir palabras desconocidas de longitud menor a 8 en la parte final de la oración, esas palabras se considerarán como las más probables.

6. Si la probabilidad de una palabra menor (LKW-1) es al menos 1000 veces mayor que la palabra mayor (LKW), la prioridad será de esa palabra.

7. Las palabras sin acentos se comparan con las palabras con acentos para verificar que no exista un error de ortografía y se agregan a la rama de palabras desconocidas.

8. Se eliminaron los bigramas de una letra con repeticiones menores a 4 veces dentro del corpus de bigramas, esto para evitar errores con números romanos o con siglas.

9. El algoritmo se programó en Python mediante lógica estructurada, para ahorrar tiempo, pero sería recomendable el usar otro tipo de enfoque al programar el algoritmo, ya que resulta muy ineficiente con el manejo de recursos.

4
Resultados

“La lengua no es la envoltura del pensamiento sino el pensamiento mismo.”

— Miguel de Unamuno

Se configuró una prueba del algoritmo de tal manera que pudiéramos obtener un total de 150 oraciones (*Hashtags*) procesadas. El *script* para obtener las oraciones basado en phyton realizó iteraciones de tal manera que con 1580 tweets, se encontraron 279 *hashtags*, de los cuales solo 157 eran *hashtags* “originales” (no repetitivos). El *script* se puede revisar en el anexo.

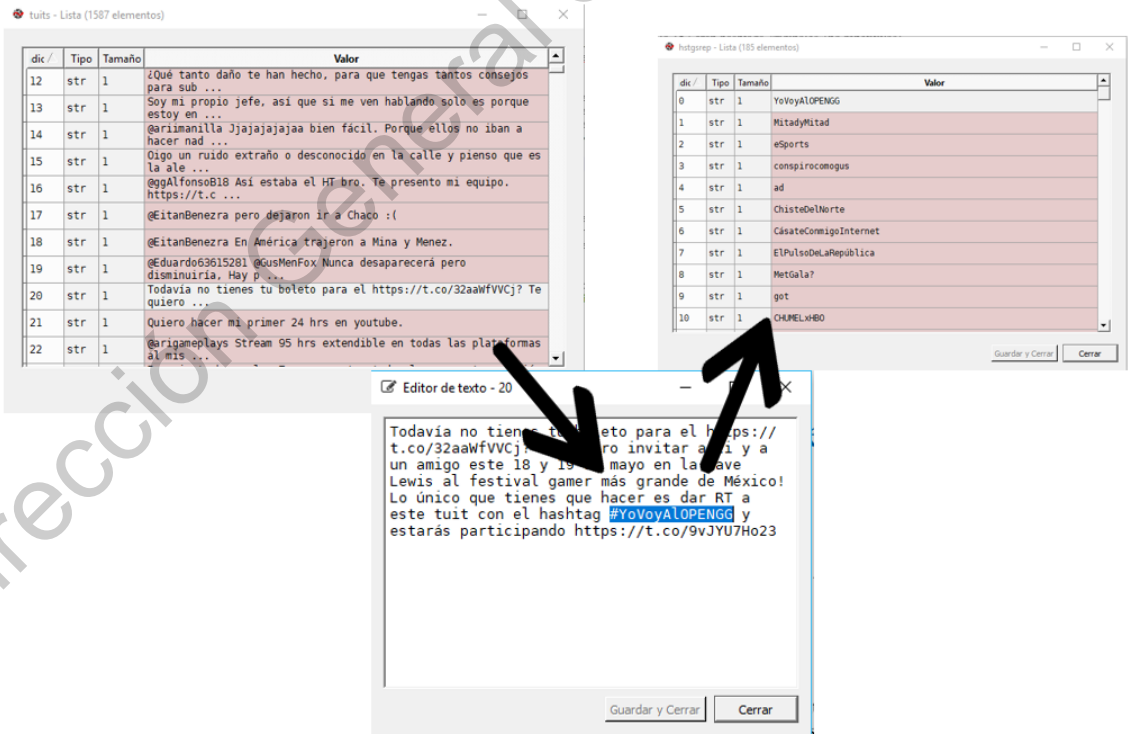

Figura 4.1: Ejemplo de obtención de oraciones

Posteriormente se ejecutaron dos *scripts* para la segmentación de las oraciones anteriormente obtenidas. El script construyó los árboles como se muestra en la siguiente figura. La oración de ejemplo que se muestra a continuación es: **cdmxsinplástico**, que tiene muchas de las características con las que los árboles toman la decisión. Por ejemplo: cdmxsinplástico, cuenta con un acento en la palabra plástico, cuenta con una contracción de las palabras ciudad de México, expresada como cdmx y de una palabra simple común con altas probabilidades de aparecer como palabra conocida (sin).

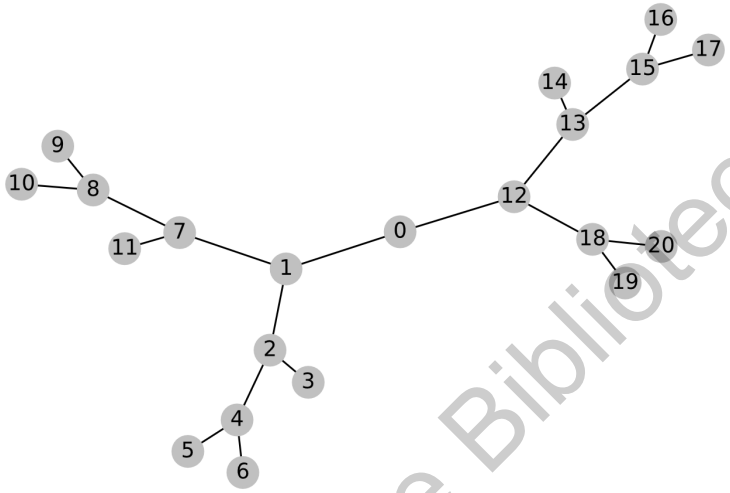

Figura 4.2: Ejemplo de árbol de candidatos “cdmxsinplástico”

Tabla 4.1: Ejemplo de candidatos para la oración “cdmxsinplástico”

ID	W1	W2	W2	W4	Probabilidad	Ramificación
1	**cdmx**	**sin**	**plástico**		**6.329441120464345e-13**	**[1, 2, 3]**
2	cdmx	sin	plásti	co	1.4294362480014421e-19	[1, 2, 4, 5]
3	cdmx	sin	plásti	co	1.4294362480014421e-19	[1, 2, 4, 6]
4	cdmx	sinplá	st	ico	4.144094263630002e-23	[1, 7, 8, 9]
5	cdmx	sinplá	st	ico	4.144094263630002e-23	[1, 7, 8, 10]
6	cdmx	sinplá	stico		8.012734421368534e-19	[1, 7, 11]
7	cdmxsi	n	plástico		3.8421379431812997e-16	[12, 13, 14]
8	cdmxsi	n	plásti	co	8.677055590339609e-23	[12, 13, 15, 16]
9	cdmxsi	n	plásti	co	8.677055590339609e-23	[12, 13, 15, 17]
10	cdmxsi	nplást	ico		1.4037763750253668e-19	[12, 18, 19]
11	cdmxsi	nplást	ico		1.4037763750253668e-19	[12, 18, 20]

Con el algoritmo se generaron 11 candidatos en la oración cdmxsinplástico, de las cuales se puede notar que los candidatos 1, 6 y 7 son únicos, y que los demás candidatos se

repiten debido al algoritmo de búsqueda de caracteres especiales. Así mismo podemos notar que el mejor candidato es el primero, al tener la probabilidad más alta dentro del árbol de segmentación.

Otro caso que se presenta, es el de la oración “síaldesarmesíalapaz” en la que podemos ver, primero que al ser la oración más grande es más complejo el árbol binario Fig. 4.3, el mejor candidato de la Tabla 4.2 es el tercer candidato a pesar de no ser la forma correcta de segmentar la oración (sí al desarme sí a la paz).

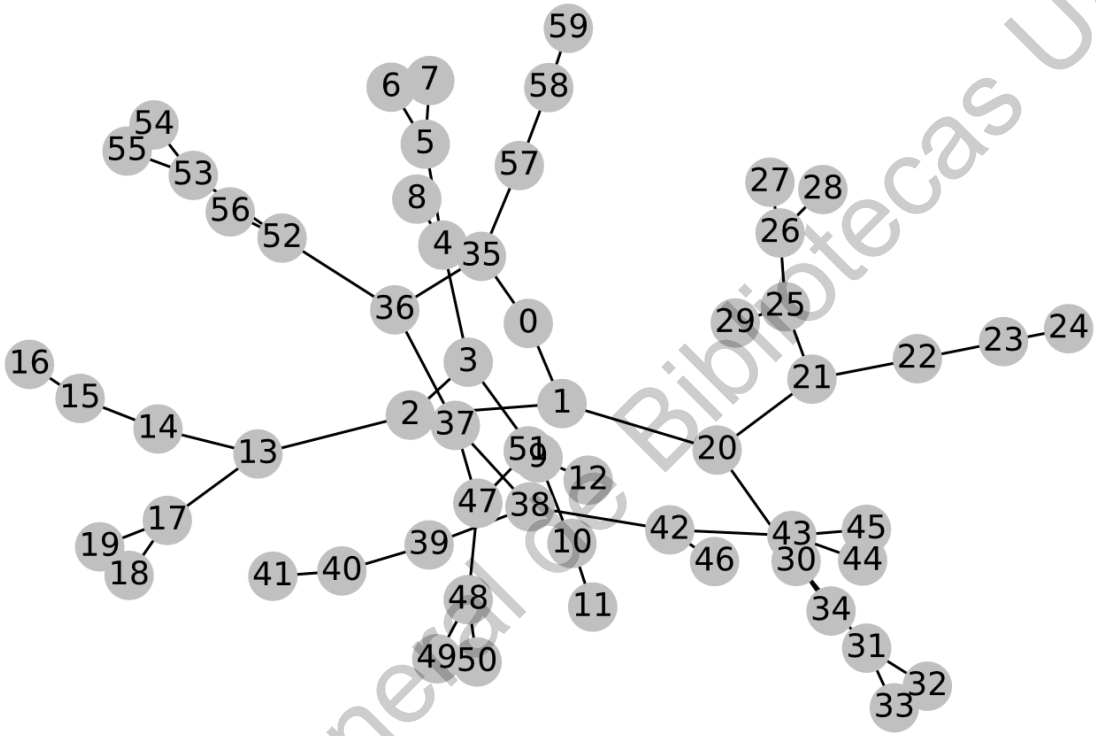

Figura 4.3: Ejemplo de árbol de candidatos “síaldesarmesíalapaz”

Si analizamos la oración de forma más minuciosa, nos podemos encontrar con lo siguiente.

Tabla 4.3: Comparativa de métodos de segmentación de la oración “síaldesarmesíalapaz”

ID	W1	W2	W2	W4	W5	W6	W7	Probabilidad	Ramificación	Método
1	sí	al	desarme	sí	alapaz			3.181014397456624e-21	[1, 2, 3, 4, 7]	Uni-gramas
2	sí	al	desarme	sí	alapaz			3.181014397456624e-21	[1, 2, 3, 4, 8]	Bi-gramas
3	sí	al	desarme	sí	a	la	paz	1.118666454831667e-21	Manual	Uni-gramas
4	sí	al	desarme	sí	a	la	paz	2.2800550590253927e-20	Manual	Bi-gramas

En la Tabla 4.3, podemos observar cuatro metodologías diferentes para segmentar la oración, la primera y la segunda ponen a prueba el algoritmo desarrollado para esta tarea, teniendo un valor de probabilidad segmentación de la oración de $3.18e^{21}$, siendo esté más alta

Tabla 4.2: Ejemplo de candidatos para la oración “síaldesarmesíalapaz”

ID	W1	W2	W2	W4	W5	W6	W7	Probabilidad	Ramificación
1	sí	al	desarme	sí	ala	paz		5.884830056249312e-23	[1, 2, 3, 4, 5, 6]
2	sí	al	desarme	sí	ala	paz		5.884830056249312e-23	[1, 2, 3, 4, 5, 7]
3	**sí**	**al**	**desarme**	**sí**	**alapaz**			**3.181014397456624e-21**	**[1, 2, 3, 4, 8]**
4	sí	al	desarme	síalap	a	z		1.491048757242457e-25	[1, 2, 3, 9, 10, 11]
5	sí	al	desarme	síalap	az			2.942177984712772e-25	[1, 2, 3, 9, 12]
6	sí	al	desarm	es	íalapa	z		7.254812030275359e-27	[1, 2, 13, 14, 15, 16]
7	sí	al	desarm	esíala	paz			3.9926897522847177e-23	[1, 2, 13, 17, 18]
8	sí	al	desarm	esíala	paz			3.9926897522847177e-23	[1, 2, 13, 17, 19]
9	sí	aldesa	r	mes	íalapa	z		3.3718806821904354e-31	[1, 20, 21, 22, 23, 24]
10	sí	aldesa	r	mesíal	a	paz		3.5334078163702e-27	[1, 20, 21, 25, 26, 27]
11	sí	aldesa	r	mesíal	a	paz		3.5334078163702e-27	[1, 20, 21, 25, 26, 28]
12	sí	aldesa	r	mesíal	apaz			3.484103901909892e-28	[1, 20, 21, 25, 29]
13	sí	aldesa	rmesía	la	paz			2.744441846062106e-22	[1, 20, 30, 31, 32]
14	sí	aldesa	rmesía	la	paz			2.744441846062106e-22	[1, 20, 30, 31, 33]
15	sí	aldesa	rmesía	lapaz				1.6425508757998997e-23	[1, 20, 30, 34]
16	síalde	s	a	r	mes	íalapa	z	3.6351827143474724e-34	[35, 36, 37, 38, 39, 40, 41]
17	síalde	s	a	r	mesíal	a	paz	3.80932311295936e-30	[35, 36, 37, 38, 42, 43, 44]
18	síalde	s	a	r	mesíal	a	paz	3.80932311295936e-30	[35, 36, 37, 38, 42, 43, 45]
19	síalde	s	a	r	mesíal	apaz		3.756169174700979e-31	[35, 36, 37, 38, 42, 46]
20	síalde	s	a	rmesía	la	paz		2.958748692393198e-25	[35, 36, 37, 47, 48, 49]
21	síalde	s	a	rmesía	la	paz		2.958748692393198e-25	[35, 36, 37, 47, 48, 50]
22	síalde	s	a	rmesía	lapaz			1.7708137131546555e-26	[35, 36, 37, 47, 51]
23	síalde	s	armesí	ala	paz			1.5564722763143816e-26	[35, 36, 52, 53, 54]
24	síalde	s	armesí	ala	paz			1.5564722763143816e-26	[35, 36, 52, 53, 55]
25	síalde	s	armesí	alapaz				8.41343024840678e-25	[35, 36, 52, 56]
26	síalde	sarmes	íalapa	z				3.658013151481208e-26	[35, 57, 58, 59]

que la probabilidad de el tercer caso con una probabilidad de $1.11e^{21}$, que es la forma correcta de segmentar la oración, con la condición de que cada una de las palabras es independiente (3.16) de todas las palabras anteriores y posteriores.

Con esta condición podemos darnos cuenta de que los casos uno y dos son más grandes que el caso tres ya que al tener menos números racionales entre el cero y el uno para multiplicar, la probabilidad resultante tiende a ser más grande que el caso tres con siete números racionales a multiplicar.

La metodología de interés es la cuatro ya que la probabilidad resulta ser mayor que los casos anteriores, a pesar de haber segmentado en siete partes la oración, y esto se da, debido a que en este caso en específico el bigrama [la, paz], es un bigrama conocido dentro de nuestra bolsa de palabras, y a pesar de que sigue teniendo más segmentos que los casos uno y dos, la probabilidad del bigrama es más alto que la probabilidad de la palabra desconocida basada en la distribución de Weibull (alapaz), teniendo el caso de la mejor segmentación con la probabilidad más alta.

El por qué el algoritmo no determino el caso cuatro que se menciona anterior es debido a las reglas de procesamiento de los árboles, que no determinan todos los casos posibles de la segmentación, si no, los mejores casos con un bajo procesamiento computacional, es decir los casos óptimos.

Tabla 4.4: Comparativa de separación de palabras mediante uni-gramas y bi-gramas.

Original	Tratamiento	Tratada										
ubersintarjeta	Uni-grama	uber	sin	tarjeta								
	Bi-grama	uber	sin	tarjeta								
yamecansédedecirlesqueesmásdelomismo	Uni-grama	yameca	n	sé	de	decirles	que	es	más	del	o	mismo
	Bi-grama	ya	me	cansé	de	decirles	que	es	más	de	lo	mismo
lovearmymexico	Uni-grama	lovear	my	mexico								
	Bi-grama	lo	vearmy	mexico								
votanathcampos	Uni-grama	votan	athcam	pos								
	Bi-grama	votan	athcam	pos								
reescribeméxico	Uni-grama	re	escribe	méxico								
	Bi-grama	re	escribe	méxico								
embajadoraeroméxico	Uni-grama	embajadora	eroméx	ico								
	Bi-grama	embajadora	eroméx	ico								
envivo	Uni-grama	en	vivo									
	Bi-grama	en	vivo									
ciudadsegura	Uni-grama	ciudad	segura									
	Bi-grama	ciudad	segura									
audienciaspúblicas	Uni-grama	audiencias	públicas									
	Bi-grama	audiencias	públicas									
ciudadquebaila	Uni-grama	ciudad	que	baila								
	Bi-grama	ciudad	que	baila								
capitalculturaldeamérica	Uni-grama	capital	cultural	dea	mérica							
	Bi-grama	capital	cultural	de	américa							
sábadodetequio	Uni-grama	sábado	de	tequio								
	Bi-grama	sábado	de	tequio								
innovaciónyderechos	Uni-grama	innovación	y	derechos								
	Bi-grama	innovación	y	derechos								
viernesdekaraoke	Uni-grama	viernes	de	karaok	e							
	Bi-grama	viernes	de	karaoke								
Hacemuuuuuchotiempoquequeríaescribiracercadeestetema	Uni-grama	hace	mumuuuuu	cho	tiempo	que	quería	escribir	acerca	de	este	tema
	Bi-grama	hace	mumuuuuu	cho	tiempo	que	quería	escribir	acerca	de	este	tema

4.1 Resultados de segmentación

La comprobación de las oraciones correctamente segmentadas se calificó manualmente de manera independiente, teniendo como parámetro de calificación, asignar el valor de uno en el caso de que la segmentación se realice de manera correcta, si y solo si, se encontrara perfectamente segmentado, y cero siempre que se tuviera al menos un error al segmentar la oración.

Al realizar la experimentación con uni-gramas y bigramas obtenidos de Twitter para probar los modelos, se obtuvieron tablas de calificación de oraciones segmentadas como la Tabla 4.4 en la que se muestra una sección de las mismas:

Teniendo en cuenta que es estudio actual está enfocado a la tasa de error, el método de validación será definido por(James *et al.*, 2013):

$$\text{Tasa de error} = \frac{\text{Total de fracasos}}{\text{Total de intentos}} \tag{4.1}$$

La tasa de error resultante fue:

Tabla 4.5:

Método	**Tasa de error**
Uni-grama	0,5031847134
Bi-grama	0,3949044586

5
Discusión y Conclusiones

5.1 Conclusiones

Es posible la segmentación de palabras mediante el uso de bi-gramas y uni-gramas con la CFD Weibull.

El CFD Weibull resultó ser el modelo más adecuado para la estimación de probabilidad de palabras desconocidas en español mexicano de acuerdo al análisis en el Capítulo 4.

Los modelos de las ecuaciones (3.16) y (3.19) resultaron correctos y factibles para el cálculo de las probabilidades de las oraciones segmentadas.

La estructura de árboles binarios en conjunto con el algoritmo *greedy*, resolvieron el problema de optimización de creación y búsqueda de candidatos, dando buenos resultados, con la consideración que al manejar un algoritmo voraz, podemos tener óptimos locales y que debido a nuestro procesamiento de ramificación de árboles binarios compuestos, en algunas ocasiones se creaban candidatos duplicados con diferentes caminos de ramificación volviendo un poco más pesado el procesamiento de los mismos.

La tasa de error de los algoritmos resultó ser muy baja, teniendo en cuenta que la muestra de prueba se dio en textos informales quedando demostrado que el modelo de bigramas resulta ser más eficiente qal eliminar errores como el mencionado en el Capítulo 3.

La distribución encontrada que ejemplifica el uso del español mexicano para este tipo de fuentes demostró que es posible modelar la distribución del uso del lenguaje informal basado en longitud de palabras, para predecir la aparición de palabras desconocidas en nuevas entradas.

Se determinó que el uso de bi-gramas con la función de distribución Weibull reduce la tasa de error en la identificación de unidades léxicas y sintácticas de texto informal en español, en contraste con el uso de uni-gramas como modelo de segmentación.

En trabajos futuros se recomienda experimentar con métodos de inteligencia artificial, específicamente con redes neuronales convolucionales en la determinación de unidades léxicas y sintácticas ya que en trabajos actuales como resultaron en un mejor entendimiento del contexto ya que el problema de los modelos tratados en este trabajo computacionalmente resulta muy exhaustivos ya que manejan grandes dependencias como en la estructura de un árbol, diferente a las redes neuronales convolucionales, donde el número de pasos de procedimiento depende de la posición en que la palabra es una entrada.

5.2 Discusión

Los dos algoritmos funcionan con un gran porcentaje de eficacia (teniendo en consideración que se está calificando en texto informal). Aun así, se presentan algunos errores mencionados a continuación:

- La complejidad del problema a resolver es del tipo 2^n por lo que cuando la longitud de la palabra a segmentar es mayor a 50 caracteres (donde $caracteres = n$), el cálculo se empieza a hacer demasiado grande, es decir, que debido a la recursividad del algoritmo creador de árboles se genera una acumulación en la memoria RAM del pc (en una computadora con 16GB de RAM, un procesador Core i7 de 8va generación y una tarjeta GPU Quadro K620).

- Si la palabra a segmentar tiende a separar en muchas secciones la oración el algoritmo tenderá a escoger las palabras desconocidas de la longitud dominante debido que, al multiplicar varias probabilidades entre sí, la probabilidad más grande será la que menos segmentos tenga. Por ejemplo:
  vive tu centro 1.3655477660231285e-11
  vive tucent r o 2.1376895849509005e-18
  vive tucent ro 2.333675145427596e-18
  vivetu centro **9.912162625543935e-11**
  donde "vivetu centro" resultó ser más probable que "vive tu centro" que es la opción correcta.

- Debido a que solo se toma la palabra anterior conocida a la palabra conocida con mayor longitud, es posible cometer el error de no seleccionar la mejor oración segmentada, por ejemplo en la palabra "alapaz", el algoritmo segmentará la palabra "ala paz", "al a paz" y "alapaz", sin que se llegue a la oración correctamente segmentada, "a la paz".

- Debido a que el algoritmo solo "lee" el texto de izquierda a derecha puede generar errores debido a palabras desconocidas que alteran los resultados de palabras conocidas, por ejemplo "cianuroumiel", se segmenta de la forma "cianur oymiel", ya que la palabra "cianuro" no se encontraba dentro del diccionario por lo que la seleccionó la palabra desconocida optima generando un error en las palabras "y miel".

- Los modelos basados en bi-gramas disminuye el error, pero en algunos casos provoca errores que en los modelos de uni-gramas no existían.

6 Anexos

6.1 Anexo 1 (Respuesta de Bigramas Tratados)

Tabla 6.1: Respuesta de Bigramas tratados

Original	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	Resultado
yovoyalopengg	['yovoya'	lopengg']										0
mitadymitad	['mitad'	y'	mitad']									1
esports	['esports']											1
conspirocomogus	['con'	spiroc'	omogus']									0
ad	['ad']											1
chistedelnorte	['chiste'	del'	norte']									1
cásateconmigointernet	['cásatec'	onmigo'	internet']									0
elpulsodelarepública	['el'	pulso'	del'	a'	república']							0
metgala	['metgala']											1
got	['got']											1
chumelxhbo	['chumel'	xhbo']										0
ubersintarjeta	['uber'	sin'	tarjeta']									1
yamecansédedecirlesqueesmásdelomismo	['ya'	me'	cansé'	de'	decirles'	que'	es'	más'	de'	lo'	mismo']	1
lovearmymexico	['lo'	vearmy'	mexico']									0
pijamadazuricata	['pijama'	dazuri'	cata']									0
newprofilepic	['newpro'	filepic']										0
askzuritips	['askzur'	itips']										0
votanathcampos	['votan'	athcam'	pos']									0
kca	['kca']											1
reescribeméxico	['re'	escribe'	méxico']									0
embajadoraeroméxico	['embajadora'	eroméx'	ico']									0
envivo	['en'	vivo']										1
ciudadsegura	['ciudad'	segura']										1
audienciaspúblicas	['audiencias'	públicas']										1
ciudadquebaila	['ciudad'	que'	baila']									1
capitalculturaldeamérica	['capital'	cultural'	de'	américa']								1
sábadodetequio	['sábado'	de'	tequio']									1
innovaciónyderechos	['innovación'	y'	derechos']									1
sabadodetequio	['sabado'	de'	tequio']									1
viernesdekaraoke	['viernes'	de'	karaoke']									1
capitalculturaldeamerica	['capital'	cultural'	de'	america']								1
síaldesarmesíalapaz	['sí'	al'	desarme'	sí'	alapaz']							0
ciudaddederechos	['ciudad'	de'	derechos']									1
centrohistórico	['centro'	histórico']										1
vivetucentro	['vivetu'	centro']										0
pilares	['pilares']											1
senderoseguro	['senderos'	eguro']										0
conferenciamatutina	['conferencia'	matuti'	na']									0

sábadosdetequio	['sábados'	de'	tequio']									1
díadeltrabajo	['día'	del'	trabajo']									1
igualdadyderechos	['igualdad'	y'	derechos']									1
díadelaniñayelniño	['día'	de'	la'	niña'	yelniño']							0
mejorescuela	['mejores'	cuela']										0
díadelasniñasylosniños	['día'	de'	las'	niñas'	y'	los'	niños']					1
tierrabeat	['tierra'	beat']										1
conamloporlapaz	['con'	amlopo'	rlapaz']									0
sábadodedanzón	['sábado'	de'	danzón']									1
másymejormovilidad	['más'	y'	mejor'	movilidad']								1
sijuárezviviera	['si'	juárez'	viviera']									1
elmañanero	['elmaña'	nero']										0
precedentes	['precedentes']											1
entreotrascosas	['entre'	otras'	cosas']									1
paraaplicarse	['para'	aplicarse']										1
elveraniego	['elvera'	niego']										0
gansoficio	['gansof'	icio']										0
megamocos	['me'	gamocos']										0
tómala	['tómala']											1
faltancredenciales	['faltan'	credenciales']										1
entrechayoterosymaiceados	['entre'	chayot'	eros'	y'	maicea'	dos']						0
ayotzinapa	['ayotzinapa']											1
mocos	['mocos']											1
videocharla	['video'	charla']										1
libertaddeprensa	['libertad'	de'	prensa']									1
bancadaanimalista	['bancada'	animal'	ista']									0
cdmxsinplástico	['cdmx'	sin'	plástico']									1
sylviaderbez	['sylvia'	derbez']										1
juntosesmejor	['juntos'	es'	mejor']									1
maythefourthbewithyou	['maythe'	fourth'	bewith'	you']								0
starwarscelebration	['starwa'	rscele'	bration']									0
starwarsday2019	['star'	warsday'	2'	0'	1'	9']						0
angrybirdsmovie2	['angryb'	ir'	dsmovie'	2']								0
angrybirds2	['angryb'	irds'	2']									0
fionaderbez	['fionad'	erbez']										0
domingo	['domingo']											1
bulldogs	['bull'	dogs']										1
sundayvibes	['sunday'	vibes']										1
ampnche	['ampnche']											1
snowball	['s'	nowball']										0
lavidasecretadetusmascotas2	['lavida'	secreta'	de'	tus'	mascotas'	2']						0
mala	['mala']											1
malabares	['mala'	bares']										0
vadhirderbez	['vadhir'	derbez']										1
tbt	['tbt']											1
cadenadeperdonchallenge	['cadena'	de'	per'	doncha'	llenge']							0
doramovie	['do'	ramovie']										0
kidschoiceawards	['kidsch'	oiceaw'	ards']									0
kidschoiceawards2019	['kidsch'	oiceaw'	ards'	2'	0'	1'	9']					0
nomanchesfrida2	['no'	manche'	sfrida'	2']								0
tolucasintauromaquia	['tolucas'	intaur'	omaquia']									1
elcomplotmongol	['el'	complot'	mongol']									1
fashionforward	['fashion'	forward']										1
barbie60	['barbie'	6'	0']									1
familyfirst	['family'	first']										1
familytime	['family'	time']										0
makingrememories	['making'	rem'	emories']									0
diainternacionaldelamujer	['di'	a'	internacional'	de'	la'	mujer']						0
internationalwomensday	['international'	womens'	da'	y']								0
cenaconamigos	['cena'	con'	amigos']									1
lososcares	['los'	oscares']										1

daytona500	['daytona'	5'	0'	0']									1
complotmongol	['complot'	mongol']											1
fernandolujan	['fernando'	lujan']											1
gabrielaestrada	['gabriela'	estrada']											1
orgullosamentemexicanos	['orgullosamente'	mexicanos']											1
hats	['hats']												1
sombreros	['sombreros']												1
lolmx	['lolmx']												0
familygoals	['family'	goals']											1
familyphotography	['family'	photography']											1
christmas	['christ'	mas']											0
amazonprime	['amazon'	pri'	me']										0
americavscruzazul	['america'	vscruz'	azul']										0
finalapertura2018	['final'	apertura'	2'	0'	1'	8']							1
domingofamiliar	['domingo'	familiar']											1
finalligamx	['final'	ligamx']											0
cruzazulvsamerica	['cruza'	zulvsa'	merica']										0
comoserunlatinlover	['como'	ser'	unlati'	nlover']									0
brunovaldez	['bruno'	valdez']											1
marcone	['marcone']												1
cruzazulvsamérica	['cruza'	zulvsa'	mérica']										0
grinch	['grinch']												1
mariodomm	['mario'	domm']											1
amigos	['amigos']												1
familia	['familia']												1
enmodogrinch	['en'	modo'	grinch']										1
thegrinch	['the'	grinch']											1
cianuroymiel	['cianur'	oymiel']											0
tipspurina	['tipspu'	rina']											0
elgrinch	['el'	grinch']											1
daddysgirl	['daddys'	girl']											1
overboard	['overbo'	ard']											0
aitana	['aitana']												1
momentsforlife	['moments'	forlife']											0
thewheelofbrisbane	['thewhe'	elofbr'	isbane']										0
14dediciembre	['1'	4'	de'	diciembre']									1
lol	['lo'	l']											0
nuevazeland	['nueva'	zeland']											1
familytrip	['family'	trip']											1
cruzazul	['cruza'	zul']											0
oscar	['oscar']												1
madmaxfury	['madmax'	fury']											1
apoyaelbuencinemexicano	['apoya'	el'	buen'	cine'	mexicano']								1
disney	['disney']												1
dreamscometrue	['dreams'	come'	true']										1
disneypixar	['disney'	pixar']											1
sundayfunday	['sunday'	funda'	y']										0
díainternacionaldelamiña	['día'	internacional'	de'	la'	niña']								1
dayofthegirl	['dayoft'	hegirl']											0
kai	['kai']												1
jdwason	['jdwason']												1
nuevafotodeperfil	['nueva'	foto'	de'	perfil']									1
esunaloca	['es'	unaloca']											0
jdpantojaenhoy	['jdpant'	o'	jaenhoy']										0
1	['1']												1
katunga	['katunga']												1
borrachos	['borrachos']												1
pantojaresponde	['pantoja'	responde']											1

Correctas	**95**
Total	**157**
Tasa de error	**0.394904459**

6.2 Anexo 2 (Respuesta de Unigramas Tratados)

Tabla 6.2: Respuesta de Unigramas tratados

Original	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	Resultado
yovoyalopengg	['yovoya'	lopengg']										1
mitadymitad	['mitad'	y'	mitad']									1
esports	['esports']											0
conspirocomogus	['con'	spiroc'	omogus']									0
ad	['ad']											1
chistedelnorte	['chiste'	del'	norte']									1
cásateconmigointernet	['cásatec'	onmigo'	internet']									1
elpulsodelarepública	['el'	pulso'	del'	a'	república']							0
metgala	['metgala']											0
got	['got']											1
chumelxhbo	['chumel'	xhbo']										0
ubersintarjeta	['uber'	sin'	tarjeta']									1
yamecansédedecirlesqueesmásdelomismo	['ya'	me'	cansé'	de'	decirles'	que'	es'	más'	de'	lo'	mismo']	0
lovearmymexico	['lo'	vearmy'	mexico']									0
pijamadazuricata	['pijama'	dazuri'	cata']									0
newprofilepic	['newpro'	filepic']										0
askzuritips	['askzur'	itips']										0
votanathcampos	['votan'	athcam'	pos']									0
kca	['kca']											1
reescribeméxico	['re'	escribe'	méxico']									0
embajadoraeroméxico	['embajadora'	eroméx'	ico']									0
envivo	['en'	vivo']										1
ciudadsegura	['ciudad'	segura']										1
audienciaspúblicas	['audiencias'	públicas']										1
ciudadquebaila	['ciudad'	que'	baila']									1
capitalculturaldeamérica	['capital'	cultural'	de'	américa']								0
sábadodetequio	['sábado'	de'	tequio']									1
innovaciónyderechos	['innovación'	y'	derechos']									1
sabadodetequio	['sabado'	de'	tequio']									1
viernesdekaraoke	['viernes'	de'	karaoke']									0
capitalculturaldeamerica	['capital'	cultural'	de'	america']								0
síaldesarmesíalapaz	['sí'	al'	desarme'	sí'	alapaz']							0
ciudaddederechos	['ciudad'	de'	derechos']									1
centrohistórico	['centro'	histórico']										1
vivetucentro	['vivetu'	centro']										0
pilares	['pilares']											1
senderoseguro	['senderos'	eguro']										0
conferenciamatutina	['conferencia'	matuti'	na']									0
sábadosdetequio	['sábados'	de'	tequio']									1
díadeltrabajo	['día'	del'	trabajo']									1
igualdadyderechos	['igualdad'	y'	derechos']									1
díadelaniñayelniño	['día'	de'	la'	niña'	yelniño']							0
mejorescuela	['mejores'	cuela']										0
díadelasniñasylosniños	['día'	de'	las'	niñas'	y'	los'	niños']					0
tierrabeat	['tierra'	beat']										1
conamloporlapaz	['con'	amlopo'	rlapaz']									0
sábadodedanzón	['sábado'	de'	danzón']									1
másymejormovilidad	['más'	y'	mejor'	movilidad']								1
sijuárezviviera	['si'	juárez'	viviera']									1
elmañanero	['elmaña'	nero']										0
precedentes	['precedentes']											1
entreotrascosas	['entre'	otras'	cosas']									1
paraaplicarse	['para'	aplicarse']										1
elveraniego	['elvera'	niego']										0
gansoficio	['gansof'	icio']										0
megamocos	['me'	gamocos']										0
tómala	['tómala']											1

faltancredenciales	['faltan'	credenciales']										1
entrechayoterosymaiceados	['entre'	chayot'	eros'	y'	maicea'	dos']						0
ayotzinapa	['ayotzinapa']											1
mocos	['mocos']											1
videocharla	['video'	charla']										1
libertaddeprensa	['libertad'	de'	prensa']									1
bancadaanimalista	['bancada'	animal'	ista']									0
cdmxsinplástico	['cdmx'	sin'	plástico']									1
sylviaderbez	['sylvia'	derbez']										1
juntosesmejor	['juntos'	es'	mejor']									1
maythefourthbewithyou	['maythe'	fourth'	bewith'	you']								0
starwarscelebration	['starwa'	rscele'	bration']									0
starwarsday2019	['star'	warsday'	2'	0'	1'	9']						0
angrybirdsmovie2	['angryb'	ir'	dsmovie'	2']								0
angrybirds2	['angryb'	irds'	2']									0
fionaderbez	['fionad'	erbez']										0
domingo	['domingo']											1
bulldogs	['bull'	dogs']										0
sundayvibes	['sunday'	vibes']										1
ampnche	['ampnche']											0
snowball	['s'	nowball']										0
lavidasecretadetusmascotas2	['lavida'	secreta'	de'	tus'	mascotas'	2']						0
mala	['mala']											1
malabares	['mala'	bares']										0
vadhirderbez	['vadhir'	derbez']										0
tbt	['tbt']											1
cadenadeperdonchallenge	['cadena'	de'	per'	doncha'	llenge']							0
doramovie	['do'	ramovie']										0
kidschoiceawards	['kidsch'	oiceaw'	ards']									0
kidschoiceawards2019	['kidsch'	oiceaw'	ards'	2'	0'	1'	9']					0
nomanchesfrida2	['no'	manche'	sfrida'	2']								0
tolucasintauromaquia	['tolucas'	intaur'	omaquia']									0
elcomplotmongol	['el'	complot'	mongol']									1
fashionforward	['fashion'	forward']										0
barbie60	['barbie'	6'	0']									1
familyfirst	['family'	first']										1
familytime	['family'	time']										1
makingrememories	['making'	rem'	emories']									0
diainternacionaldelamujer	['di'	a'	internacional'	de'	la'	mujer']						0
internationalwomensday	['international'	womens'	da'	y']								0
cenaconamigos	['cena'	con'	amigos']									0
lososcares	['los'	oscares']										0
daytona500	['daytona'	5'	0'	0']								0
complotmongol	['complot'	mongol']										1
fernandolujan	['fernando'	lujan']										1
gabrielaestrada	['gabriela'	estrada']										1
orgullosamentemexicanos	['orgullosamente'	mexicanos']										1
hats	['hats']											1
sombreros	['sombreros']											1
lolmx	['lolmx']											0
familygoals	['family'	goals']										1
familyphotography	['family'	photography']										0
christmas	['christ'	mas']										0
amazonprime	['amazon'	pri'	me']									0
americavscruzazul	['america'	vscruz'	azul']									0
finalapertura2018	['final'	apertura'	2'	0'	1'	8']						1
domingofamiliar	['domingo'	familiar']										1
finalligamx	['final'	ligamx']										0
cruzazulvsamerica	['cruza'	zulvsa'	merica']									0
comoserunlatinlover	['como'	ser'	unlati'	nlover']								0
brunovaldez	['bruno'	valdez']										1

marcone	['marcone']												0
cruzazulvsamérica	['cruza'	zulvsa'	mérica']										0
grinch	['grinch']												1
mariodomm	['mario'	domm']											1
amigos	['amigos']												1
familia	['familia']												1
enmodogrinch	['en'	modo'	grinch']										1
thegrinch	['the'	grinch']											1
cianuroymiel	['cianur'	oymiel']											0
tipspurina	['tipspu'	rina']											0
elgrinch	['el'	grinch']											1
daddysgirl	['daddys'	girl']											1
overboard	['overbo'	ard']											0
aitana	['aitana']												1
momentsforlife	['moments'	forlife']											0
thewheelofbrisbane	['thewhe'	elofbr'	isbane']										0
14dediciembre	['1'	4'	de'	diciembre']									1
lol	['lo'	l']											0
nuevazeland	['nueva'	zeland']											1
familytrip	['family'	trip']											1
cruzazul	['cruza'	zul']											0
oscar	['oscar']												1
madmaxfury	['madmax'	fury']											1
apoyaelbuencinemexicano	['apoya'	el'	buen'	cine'	mexicano']								1
disney	['disney']												1
dreamscometrue	['dreams'	come'	true']										1
disneypixar	['disney'	pixar']											1
sundayfunday	['sunday'	funda'	y']										0
díainternacionaldelaniña	['día'	internacional'	de'	la'	niña']								0
dayofthegirl	['dayoft'	hegirl']											0
kai	['kai']												1
jdwason	['jdwason']												0
nuevafotodeperfil	['nueva'	foto'	de'	perfil']									1
esunaloca	['es'	unaloca']											0
jdpantojaenhoy	['jdpant'	o'	jaenhoy']										0
1	['1']												1
katunga	['katunga']												0
borrachos	['borrachos']												1
pantojaresponde	['pantoja'	responde']											1
												Correctas	**78**
												Total	**157**
												Tasa de error	**0.503184713**

6.3 Anexo 3 (Código Fuente Unigramas)

```
# -*- coding: utf-8 -*-

#Created on Tue Oct 16 15:39:34 2018
#Procesador texto
#@author: Oscar Cano Felix

import _pickle as cPickle
import math as mt
import matplotlib.pyplot as plt
import networkx as nx
import re
##variables globales
dbname = "tweets"
#####################################################################
    funciones
#Load data
def cargar_datos(datos):
    try:
        with open(datos + ".dat", "rb") as f:
            return cPickle.load(f)
    except (OSError, IOError) as e:
        print (e)
        return dict()

#Calcula probabilidad de palabras desconocidas
def unk_probabilidad(palabra):
    alpha = 2.211243
    beta = 7.267729
    aux = len(palabra)
    return ((mt.exp(-((aux-1)/beta)**alpha) - mt.exp(-(aux/beta)**alpha))
    /len(corpus))#Se divide entre coprpus para escalar el factor

#calculation of probabilities
def probW (word):
     aux = diccionario.get(word)
     if aux == None:
         prob = unk_probabilidad(word)
         flag = 0
         return prob,flag
     else:
          prob = aux
          flag = 1
     return prob,flag

################################################
#Arbol de probabilidad
class Arbol:
    def __init__(self, elemento, prob):
        self.hijos = []
        self.elemento = elemento
        self.prob = prob
```

```
50
51 def agregarElemento(arbol, elemento, prob, elementoPadre):
52     subarbol = buscarSubarbol(arbol, elementoPadre);
53     subarbol.hijos.append(Arbol(elemento, prob))
54
55 def buscarSubarbol(arbol, elemento):
56     if arbol.elemento == elemento:
57         return arbol
58     for subarbol in reversed(arbol.hijos):
59         arbolBuscado = buscarSubarbol(subarbol, elemento)
60         if (arbolBuscado != None):
61             return arbolBuscado
62     return None
63     #printfrase(aux,wrd,prb,g)
64 def printfrase(i,wrd,prb,g):
65      short = nx.shortest_path(g,0,i)
66      short.pop(0)#quitar la raiz del tratamiento
67      frase = []
68      for j in short:
69           frase.append(wrd[j])
70 #      print (frase, prb[i])
71      return frase,short
72
73 def printElement(element,prob):
74      print (element, prob)
75
76 def is_number(s):
77     try:
78         float(s)
79         return True
80     except ValueError:
81         pass
82
83     try:
84         import unicodedata
85         unicodedata.numeric(s)
86         return True
87     except (TypeError, ValueError):
88         pass
89     return False
90
91 def encontrar_num(f):
92      valores = []
93      i = 0
94      for c in f:
95           i+=1
96           numero = is_number(c)
97           if numero == True and i != 1:
98                cut = f[0:i-1]
99                f = f[i-1:len(f)]
100               cut2 = f[0:1]
101               f = f[1:len(f)]
102               valores.append(cut)
103               valores.append(cut2)
```

```
                i=0
            elif numero == True and i == 1:
                cut = f[0:i]
                f = f[i:len(f)]
                valores.append(cut)
                i=0
        if len(f)>0:
            valores.append(f)
        return valores


def makefraseok (frase,arbol,ram,hoja,hojas,g):
        if len(frase) <= 0:
            hojas.append(hoja-1)
            return hoja,hojas
        else:
            palabra = ""
            aux = 0
            izq = None
            der = None
            izqp = 0
            tmp = 0
            flag = 0
            ramtmp = ram
            for c in frase:
                palabra = palabra + c
                prob,flag = probW(palabra)
                if flag == 1:
                    izq = palabra
                    izqp = prob
                if prob > aux and prob != izqp:#
                    aux = prob
                    der = palabra
                    derp = prob
            if izq != None:
                subarbol = buscarSubarbol(arbol, arbol.elemento)
                ram,hoja = graficarnx (izq,arbol.elemento,izqp,subarbol.
    prob,ram,hoja,g)#agregar rama al arbol ploteado
                agregarElemento(arbol, izq, izqp * subarbol.prob, subarbol
    .elemento)
                palCor = frase.replace(izq,"",1)
                #hojatemp = makefraseok (palCor,arbol.hijos[tmp],ram,hoja)
                hoja,hojas = makefraseok (palCor,arbol.hijos[tmp],ram,hoja
    ,hojas,g)
                tmp += 1
            if der != None:
                subarbol = buscarSubarbol(arbol, arbol.elemento)
                ram,hoja = graficarnx (der,arbol.elemento,derp,subarbol.
    prob,ramtmp,hoja,g)#agregar rama al arbol ploteado#agregar rama al
    arbol ploteado
                agregarElemento(arbol, der, derp*subarbol.prob, subarbol.
    elemento)
                palCor = frase.replace(der,"",1)
                #hojatemp = makefraseok (palCor,arbol.hijos[tmp],ram,hoja)
```

```
152                 hoja,hojas = makefraseok (palCor,arbol.hijos[tmp],ram,hoja
    ,hojas,g)
153       return hoja,hojas
154
155
156 def graficarnx (hijo,padre,probizq,prob,rama,hoja,g):
157       pnode = probizq * prob
158       g.add_node(hoja, word=hijo,prob = pnode)
159       g.add_edge(rama,hoja)
160       rama = hoja
161       hoja += 1
162       return rama,hoja
163
164 #graficador de arbol
165 def grphbol (g,name):
166       nx.draw(g,with_labels = True,node_color="silver")#pos=nx.
    circular_layout(g)
167       plt.savefig('OraProc/'+name + '.svg')
168       plt.clf()
169
170
171 def save_orc (posi,name,prba,nume):
172     name = re.sub('[^a-zA-Z]', ' ', name)
173     file = open('OraProc/' + name + ".txt","w")
174     for s in range(0,len(posi)):
175         temp = str(posi[s])
176         temp = re.sub('[^a-zA-Z1234567890]', ' ', temp)
177         file.write(temp + "\t\t" + str(prba[s]) + "\t\t" + str(nume[s]) +
     "\n")
178     file.close()
179
180 #procesador de tweets
181 def procesa_tuits(tuit):
182     limpias=[]
183     gato = "#"
184     contador=0
185     contadores=[]
186     wrds=tuit.split()
187     for palabra in wrds:
188         contador+=1
189         if palabra[0] == gato:
190             token='token'+str(contador)
191             contadores.append(token)
192             limpia=palabra.strip("#")
193             limpias.append(limpia)
194             tuit=tuit.replace(palabra,token)
195     return (limpias,contadores,tuit)
196
197
198 #funcion de llamada principal
199 def generadora(frase):
200       if frase == None:
201           return
202     #iniciadores de arbol
```

```
203       g = nx.Graph()
204       g.clear();
205       g.add_node(0, word = " ",prob = "1")
206       arbol = Arbol(" ", 1)
207       hojas = []
208       ########################
209       frase = str(frase)
210       frase = frase.lower()#pretratamiento
211       frase = re.sub('[^a-zA-Z0123456789]', '', frase)
212       frase.strip()
213       frase.replace(' ', '');
214       makefraseok(frase,arbol,0,1,hojas,g)#analizadora de frases
215       #algoritmo para encontrar la ramificacion y plotearla
216       wrd = nx.get_node_attributes(g,'word')
217       prb = nx.get_node_attributes(g,'prob')
218       #################
219       aux = len(prb)-1
220       posibles = []
221       prpos = []
222       num = []
223       for i in hojas:
224           if prb[i] > prb[aux]:
225               aux = i
226       ############descomentar para imprimir todas las oraciones posibles
227           fp,shrt = printfrase(i,wrd,prb,g) #descomentar para imprimir
      todas las oraciones posibles
228           num.append(shrt)
229           posibles.append(fp)
230           prpos.append(prb[i])
231       save_orc(posibles,frase,prpos,num) #descomentar para gardar txt con
      posibles frases y sus prob
232       ############
233 #      print("Mejor frase: ")
234       fp,shrt = printfrase(aux,wrd,prb,g)
235 #      print (fp)
236       #################Graficas
237       if len(frase)<20:
238            grphbol(g,frase)#graficar arbol -- Comentar para hacer mas
      rapido el procesamiento
239       g.clear();
240       return (fp)
241
242 ######################### main
243
244 diccionario = cargar_datos("diccionario_relativo")
245 corpus = cargar_datos("corpus")
246
247 ##funcion de llamado a arbol
248 tuits=cargar_datos(dbname)
249 tuitsproc=[]
250 hstgsrep = []
251 results = {}
252 #Limpiar hashtags
253 contador = 0
```

```
for i in tuits:
    tmp=[]
    contador += 1
    if contador == 5000:
        break;
    if i != None:
        #print(i)   tweet
        hasht, ubicaciones, tweHs = procesa_tuits(i)
        for h in hasht:
            if h not in hstgsrep:
                hstgsrep.append(h)
                print(h)
                htsp = []
                h = str(h)
                h = h.lower()#pretratamiento
                h = re.sub('[^a-zA-Z0123456789]', '', h)
                h.strip()
                h.replace(' ', '');
                frases = encontrar_num(h)
                for fr in frases:
                     htsp_tmp = generadora(fr)
                     for tmpp in htsp_tmp:
                          htsp.append(tmpp)
                print(htsp)
                print("")
                results[h] = htsp
                htspconc=''
                for t in htsp:
                    htspconc=htspconc + ' ' + t
                tmp.append(htspconc)
#         for u in range(0,len(ubicaciones)):
#             tweHs=tweHs.replace(ubicaciones[u],tmp[u])
        tuitsproc.append(tweHs)
#         print (tweHs)     #tweet procesado


with open('ResultsUni.csv', 'w') as f:
    for key in results.keys():
        f.write("%s,%s\n"%(key,results[key]))
```

6.4 Anexo 4 (Código Fuente Bigramas)

```
# -*- coding: utf-8 -*-

#Created on Tue Oct 16 15:39:34 2018
#Procesador texto
#@author: Oscar Cano F lix

import _pickle as cPickle
import math as mt
import matplotlib.pyplot as plt
import networkx as nx
import re
import unicodedata
##variables globales
dbname = "tweets"
#####################################################################
    funciones
#Load data
def cargar_datos(datos):
    try:
        with open(datos + ".dat", "rb") as f:
            return cPickle.load(f)
    except (OSError, IOError) as e:
        print (e)
        return dict()

#Calcula probabilidad de palabras desconocidas
def unk_probabilidad(palabra):
    alpha = 2.211243
    beta = 7.267729
    aux = len(palabra)
    return ((mt.exp(-((aux-1)/beta)**alpha) - mt.exp(-(aux/beta)**alpha))
    /len(corpus))#Se divide entre coprpus para escalar el factor

#calculation of probabilities of unigrams
def probW (word):
     aux = diccionario.get(word)
     if aux == None:
          if word == '':
               prob = 0.00000000000000000001
          else:
               prob = unk_probabilidad(word)
          flag = 0
          return prob,flag
     else:
          prob = aux
          flag = 1
     return prob,flag

def probWHT (word):
     aux = diccionarioW.get(word)
     if aux == None:
```

```
50            if word == '':
51                 prob = 0.000000000000000000001
52            else:
53                 prob = unk_probabilidad(word)
54            flag = 0
55            return prob,flag
56       else:
57            prob = aux
58            flag = 1
59       return prob,flag
60
61  #calculation of probabilities of bigrams
62  def probB (word):
63       aux = diccionario_big.get(word)
64       if aux == None:
65            prob = 0.000000000000000000001
66            flag = 0
67            return prob,flag
68       else:
69            prob = aux
70            flag = 1
71       return prob,flag
72  ################################################
73  #Arbol de probabilidad
74  class Arbol:
75      def __init__(self, elemento, prob):
76          self.hijos = []
77          self.elemento = elemento
78          self.prob = prob
79
80  def agregarElemento(arbol, elemento, prob, elementoPadre):
81      subarbol = buscarSubarbol(arbol, elementoPadre);
82      subarbol.hijos.append(Arbol(elemento, prob))
83
84  def buscarSubarbol(arbol, elemento):
85      if arbol.elemento == elemento:
86          return arbol
87      for subarbol in reversed(arbol.hijos):
88          arbolBuscado = buscarSubarbol(subarbol, elemento)
89          if (arbolBuscado != None):
90              return arbolBuscado
91      return None
92      #printfrase(aux,wrd,prb,g)
93  def printfrase(i,wrd,prb,g):
94       short = nx.shortest_path(g,0,i)
95       short.pop(0)#quitar la raiz del tratamiento
96       frase = []
97       for j in short:
98            frase.append(wrd[j])
99  #     print (frase, prb[i])
100      return frase,short
101
102 #para quitar acentos y simbolos manteniendo la
103 def withoutSimb (s):
```

```
104     s1 = s.replace("  ", "#").replace("  ", "%")
105     s2 = unicodedata.normalize("NFKD", s1)\
106          .encode("ascii","ignore").decode("ascii")\
107          .replace("#", "  ").replace("%", "  ")
108     return s2
109
110 def printElement(element,prob):
111      print (element, prob)
112
113 def is_number(s):
114     try:
115         float(s)
116         return True
117     except ValueError:
118         pass
119
120     try:
121         import unicodedata
122         unicodedata.numeric(s)
123         return True
124     except (TypeError, ValueError):
125         pass
126     return False
127
128 def encontrar_num(f):
129      valores = []
130      i = 0
131      for c in f:
132           i+=1
133           numero = is_number(c)
134           if numero == True and i != 1:
135                cut = f[0:i-1]
136                f = f[i-1:len(f)]
137                cut2 = f[0:1]
138                f = f[1:len(f)]
139                valores.append(cut)
140                valores.append(cut2)
141                i=0
142           elif numero == True and i == 1:
143                cut = f[0:i]
144                f = f[i:len(f)]
145                valores.append(cut)
146                i=0
147      if len(f)>0:
148           valores.append(f)
149      return valores
150
151
152 def makefraseok (frase,arbol,ram,hoja,hojas,g):
153      if len(frase) <= 0:
154           hojas.append(hoja-1)
155           return hoja,hojas
156      else:
157           palabra = ""
```

```
158           palabra_ant = ""
159           palabra_post = ""
160           izq_m2 = None
161 #          replaceb = ""
162           unkbig = 0
163           knobig = 0
164           aux = 0
165           izq_m = None
166           izq = None
167           der = None
168           izqp = 0
169           izq_m12p = 0
170           tmp = 0
171           flag = 0
172           b_flag = 0
173           ramtmp = ram
174           for c in frase:
175                palabra_ant = izq
176                prob_ant = izqp
177                palabra = palabra + c
178                prob,flag = probW(palabra)
179                palabraWTH = withoutSimb(palabra) #palabras sin acentos
180                probWTH,flagWHT = probWHT(palabraWTH) #palabras sin
     acentos
181                #-------
182                final = frase.replace(palabra,"",1)
183                if len (frase) != 1 and palabra_ant != None:
184                     if len(final) == 0:
185                          unkbig = 1
186                          if palabra_ant != "" and (len(palabra)-len(
     palabra_ant) < 2):
187                               knobig = 1
188                #-----
189                if flag == 1:
190                   izq = palabra
191                   izqp = prob
192                   izq_m = palabra_ant
193                   izq_mp = prob_ant
194                if (prob > aux and prob != izqp) or (unkbig == 1 and len(
     palabra) < 8) or (knobig == 1):
195                     aux = prob
196                     der = palabra
197                     derp = prob
198                elif flagWHT == 1 and flag != 1:
199                     aux = probWTH
200                     der = palabraWTH
201                     derp = probWTH
202           if izq_m != None:
203                palCor = frase.replace(izq_m,"",1)
204                for c in palCor:
205                     palabra_post = palabra_post + c
206                     m_prob,m_flag = probW(palabra_post) #aqui
     probabilidad de palabr avacia no se por que
207                     if m_flag == 1:
```

```
208                         izq_m2 = palabra_post
209                 if izq_m2 != None:
210                     izq_m12 = izq_m + " " + izq_m2
211                     izq_m12p,b_flag = probB(izq_m12) # pprobabilidad de
    bigramas
212                     if izq_mp > izqp*1000:
213                         b_flag = 1
214                 if izq_mp > izqp and izq_mp > izq_m12p and b_flag == 1:
215                     izq = izq_m
216                     izqp = izq_mp
217                 elif izq_m12p > izqp and izq_m12p > izq_mp and b_flag ==
    1:
218                     izq = izq_m
219                     izqp = izq_mp #revisar si es proponer probabilidad de
     bigrama
220             if izq != None:
221                 subarbol = buscarSubarbol(arbol, arbol.elemento)
222                 ram,hoja = graficarnx (izq,arbol.elemento,izqp,subarbol.
    prob,ram,hoja,g)#agregar rama al arbol ploteado
223                 agregarElemento(arbol, izq, izqp * subarbol.prob, subarbol
    .elemento)
224                 palCor = frase.replace(izq,"",1)
225                 #hojatemp = makefraseok (palCor,arbol.hijos[tmp],ram,hoja)
226                 hoja,hojas = makefraseok (palCor,arbol.hijos[tmp],ram,hoja
    ,hojas,g)
227                 tmp += 1
228             if der != None:
229                 subarbol = buscarSubarbol(arbol, arbol.elemento)
230                 ram,hoja = graficarnx (der,arbol.elemento,derp,subarbol.
    prob,ramtmp,hoja,g)#agregar rama al arbol ploteado#agregar rama al
    arbol ploteado
231                 agregarElemento(arbol, der, derp*subarbol.prob, subarbol.
    elemento)
232                 palCor = frase.replace(der,"",1)
233                 #hojatemp = makefraseok (palCor,arbol.hijos[tmp],ram,hoja)
234                 hoja,hojas = makefraseok (palCor,arbol.hijos[tmp],ram,hoja
    ,hojas,g)
235             return hoja,hojas
236
237
238 def graficarnx (hijo,padre,probizq,prob,rama,hoja,g):
239         pnode = probizq * prob
240         g.add_node(hoja, word=hijo,prob = pnode)
241         g.add_edge(rama,hoja)
242         rama = hoja
243         hoja += 1
244         return rama,hoja
245
246 #graficador de arbol
247 def grphbol (g,name):
248         nx.draw(g,with_labels = True,node_color="silver")#pos=nx.
    circular_layout(g)
249         plt.savefig('OraProcBi/'+name + '.svg')
250         plt.clf()
```

```
251
252
253 def save_orc (posi,name,prba,nume):
254     name = re.sub('[^a-zA-Z                          ]', ' ', name)
255     file = open('OraProcBi/' + name + ".txt","w")
256     for s in range(0,len(posi)):
257         temp = str(posi[s])
258         temp = re.sub('[^a-zA-Z                    1 2 3 4 5 6 7 8 9 0 ]', '
    ', temp)
259         file.write(temp + "\t\t" + str(prba[s]) + "\t\t" + str(nume[s]) +
    "\n")
260     file.close()
261
262 #procesador de tweets
263 def procesa_tuits(tuit):
264     limpias=[]
265     gato = "#"
266     contador=0
267     contadores=[]
268     wrds=tuit.split()
269     for palabra in wrds:
270         contador+=1
271         if palabra[0] == gato:
272             token='token'+str(contador)
273             contadores.append(token)
274             limpia=palabra.strip("#")
275             limpias.append(limpia)
276             tuit=tuit.replace(palabra,token)
277     return (limpias,contadores,tuit)
278
279 #funcion de llamada principal
280 def generadora(frase):
281      if frase == None:
282          return
283     #iniciadores de arbol
284      g = nx.Graph()
285      g.clear();
286      g.add_node(0, word = " ",prob = "1")
287      arbol = Arbol(" ", 1)
288      hojas = []
289      ########################
290      frase = str(frase)
291      frase = frase.lower()#pretratamiento
292      frase = re.sub('[^a-zA-Z                    0 1 2 3 4 5 6 7 8 9 ]', '',
    frase)
293      frase.strip()
294      frase.replace(' ', '');
295      makefraseok(frase,arbol,0,1,hojas,g)#analizadora de frases
296      #algoritmo para encontrar la ramificacion y plotearla
297      wrd = nx.get_node_attributes(g,'word')
298      prb = nx.get_node_attributes(g,'prob')
299      #################
300      aux = len(prb)-1
301      posibles = []
```

```
302       prpos = []
303       num = []
304       for i in hojas:
305           if prb[i] > prb[aux]:
306               aux = i
307       ############descomentar para imprimir todas las oraciones posibles
308           fp,shrt = printfrase(i,wrd,prb,g) #descomentar para imprimir
      todas las oraciones posibles
309           num.append(shrt)
310           posibles.append(fp)
311           prpos.append(prb[i])
312       save_orc(posibles,frase,prpos,num) #descomentar para gardar txt con
      posibles frases y sus prob
313       ############
314 #      print("Mejor frase: ")
315       fp,shrt = printfrase(aux,wrd,prb,g)
316       #################Graficas
317       if len(frase)<20:
318            grphbol(g,frase)#graficar arbol -- Comentar para hacer m s
      rapido el procesamiento
319       g.clear();
320       return (fp)
321
322 ########################## main
      ########################################
323 #
      ##############################################################################


324
325
326 diccionario = cargar_datos("diccionario_relativo")
327 corpus = cargar_datos("corpus")
328 diccionario_big = cargar_datos("diccionario_relativo_big")
329 corpus_big = cargar_datos("corpus_big")
330
331 diccionarioW = cargar_datos("diccionario_relativo_w")
332 corpusW = cargar_datos("corpus_w")
333 #diccionario_bigW = cargar_datos("diccionario_relativo_big")
334 #corpus_bigW = cargar_datos("corpus_big")
335
336
337 ##funcion de llamado a arbol
338 tuits=cargar_datos(dbname)
339 tuitsproc=[]
340 hstgsrep = []
341 results = {}
342 #Limpiar hashtags
343 contador = 0
344 cnt = 0
345 for i in tuits:
346     tmp=[]
347     contador += 1
348     if contador == 5000:
349         break;
```

```
350     if i != None:
351         #print(i)   tweet
352         hasht, ubicaciones, tweHs = procesa_tuits(i)
353         for h in hasht:
354             cnt += 1
355             if h not in hstgsrep:
356                 hstgsrep.append(h)
357                 print(h)
358                 htsp = []
359                 h = str(h)
360                 h = h.lower()#pretratamiento
361                 h = re.sub('[^a-zA-Z                    0 1 2 3 4 5 6 7 8 9 ]
    ', '', h)
362                 h.strip()
363                 h.replace(' ', '');
364                 frases = encontrar_num(h)
365                 for fr in frases:
366                      htsp_tmp = generadora(fr)
367                      for tmpp in htsp_tmp:
368                           htsp.append(tmpp)
369                 print(htsp)
370                 print("")
371                 results[h] = htsp
372                 htspconc=''
373                 for t in htsp:
374                     htspconc=htspconc + ' ' + t
375                 tmp.append(htspconc)
376 #         for u in range(0,len(ubicaciones)):
377 #             tweHs=tweHs.replace(ubicaciones[u],tmp[u])
378 #         tuitsproc.append(tweHs)
379         #print (tweHs)     tweet procesado
380
381
382 with open('ResultsBi.csv', 'w') as f:
383     for key in results.keys():
384         f.write("%s, %s\n"%(key,results[key]))
385
386 #input("Press Enter to continue...")
```

6.5 Anexo 5 (Código Fuente Diccionario Unigramas)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Oct 11 12:34:32 2018

@author: wilfridovich
@revisor: Oscar Cano-Felix
"""

import _pickle as cPickle
import re
import collections
import unicodedata

#Libros
libros = ['librostxt/El laberinto de la soledad - Octavio Paz.txt',
          'librostxt/Gringo_Viejo.txt',
          'librostxt/Juan_Salvador_Gaviota.txt',
          'librostxt/videojuegos-la-normalizacio-n-de-la-violencia.txt',
          'librostxt/la-lucha-por-abatir-el-desperdicio-de-alimentos-en-me-xico.txt',
          'librostxt/produccio-n-de-ganado-en-condiciones-de-sequi-a.txt',
          'librostxt/promueve-fenaci-proyectos-cienti-ficos-y-tecnolo-gicos-en-quere-taro.txt',
          'librostxt/triunfa-mexicana-en-concurso-internacional-de-leyes-de-harvard.txt',
          'librostxt/adio-s-al-tlcan.txt',
          'librostxt/agua-de-reu-so-la-solucio-n-para-la-escasez-en-valle-de-guadalupe.txt',
          'librostxt/aumento-de-temperatura-para-baja-california-en-las-pro-ximas-de-cadas.txt',
          'librostxt/cerveza-artesanal-una-industria-competitiva.txt',
          'librostxt/chile-habanero-de-yucata-n-u-nico-en-el-mundo.txt',
          'librostxt/ciencia-por-mi-ciudad-juventud-e-innovacio-n.txt',
          'librostxt/co-mo-hacer-un-transge-nico.txt',
          'librostxt/co-mo-se-adaptan-los-peces-arrecifales-al-cambio-clima-tico.txt',
          'librostxt/cuando-la-inteligencia-artificial-nos-supere.txt',
          'librostxt/disen-o-color-y-simetri-a-a-lo-bestia.txt',
          'librostxt/do-nde-verter-los-materiales-del-dragado-portuario.txt',
          'librostxt/el-desafi-o-de-la-donacio-n-de-o-rganos.txt',
          'librostxt/El-planeta-solo-tiene-hasta-el-2030.txt',
          'librostxt/en-busca-del-superamaranto.txt',
          'librostxt/escenarios-de-riesgos-por-sismos-e-inundaciones.txt',
          'librostxt/factores-socioecono-micos-del-sobrepeso-y-obesidad-en-mexicanos.txt',
          'librostxt/fi-sica-y-ficcio-n-literatura-que-roza-la-realidad.txt',
```

```
41          'librostxt/impulsan-programa-nacional-de-monitoreo-costero.txt'
    ,
42          'librostxt/ingenieri-a-mexicana-en-accio-n.txt',
43          'librostxt/intervencio-n-lingu-i-stica-para-fortalecer-
    identidad-indi-gena-final.txt',
44          'librostxt/la-ciencia-detra-s-de-los-suen-os.txt',
45          'librostxt/maten-al-leon-de-ibarguengoitia-user16.txt',
46          'librostxt/[Paz_Octavio]_Mascaras_Mexicanas(BookFi).txt',
47          'librostxt/[Pealosa_Joaquin_Antonio]_Vida_Pasion_Y_Muerte_D(
    BookFi).txt',
48          'librostxt/Ibarguengoitia, Jorge - Estas ruinas que ves.txt',
49          'librostxt/ibarguengoitia-jorge-los-relampagos-de-agosto.txt',
50          'librostxt/LOS PASOS DE L PEZ, DE JORGE IBARG ENGOITIA.txt',
51          'librostxt/noticia.txt',
52          'librostxt/   Coco    hace crecer el consumo de comida mexicana
     en China.txt',
53          'librostxt/7-habilidades-ba-sicas-para-los-empleos-del-futuro.
    txt',
54          'librostxt/advierten-sobre-toxicidad-en-peli-culas-para-
    envolver-alimentos.txt',
55          'librostxt/Alista L pez Obrador propuestas para Defensa y
    Marina.txt',
56          'librostxt/al-rescate-del-oso-negro-americano.txt',
57          'librostxt/Amazon-patenta-tecnolog a-que-sabra-si-estas-
    enfermo.txt',
58          'librostxt/AMLO no obligar traslados de trabajadores
    federales.txt',
59          'librostxt/AMLO se reunir con el gobernador que le declaro la
     guerra.txt',
60          'librostxt/Ampl a Issste numero de salas de hemodinamia a
    nivel nacional.txt',
61          'librostxt/Anal tica de datos, o c mo los Astros ganaron la
    Serie Mundial.txt',
62          'librostxt/aprendizaje-computacional-y-algoritmos-heuri-sticos-
    para-la-resolucio-n-de-problemas.txt',
63          'librostxt/As  de complicado y est pido es el racismo
    mexicano.txt',
64          'librostxt/Bancomer y Santander perder n millones en Argentina
     por hiperinflaci n.txt',
65          'librostxt/calentamiento-global-principal-problema-para-la-
    agenda-poli-tica-ambiental.txt',
66          'librostxt/Campesinos bloquean Constituyentes frente a SHCP.txt
    ',
67          'librostxt/ce-sar-burgos-el-investigador-de-los-narcocorridos.
    txt',
68          'librostxt/cesa-rea-o-parto-natural.txt',
69          'librostxt/cli-nica-del-duelo.txt',
70          'librostxt/C mo Facebook, Google y otras empresas
    tecnol gicas buscan acabar con la adicci n al m vil.txt',
71          'librostxt/co-mo-disminuir-la-contaminacio-n-de-ladrilleras.txt
    ',
72          'librostxt/Comunidad estudiantil convoca a una consulta
    universitaria sobre NAICM.txt',
73          'librostxt/Con llenos totales, Les Ballets de Montecarlo
```

```
triunfa en el Cervantino.txt',
74          'librostxt/conacyt-y-el-gobierno-de-guanajuato-apoyan-proyectos
    -cienti-ficos-tecnolo-gicos-y-de-innovacio-n.txt',
75          'librostxt/conductas-alimentarias-y-estilo-de-vida-de-jo-venes-
    mexicanos-y-de-latinoame-rica.txt',
76          'librostxt/Conservar tu empleo puede depender m s de tus
    habilidades blandas que de las t cnicas.txt',
77          'librostxt/Considera Trump nuevo plan de separaci n de
    familias en la frontera.txt',
78          'librostxt/consolida-tecnm-en-celaya-posgrado-de-competencia-
    internacional.txt',
79          'librostxt/Convivir con tecnolog a y actualizarse.txt',
80          'librostxt/Coparmex pide legislaci n laboral correcta para
    evitar quejas ante OIT.txt',
81          'librostxt/Coreas acuerdan mejorar ferrocarriles y carreteras
    en frontera.txt',
82          'librostxt/Crea EU fuerza especial contra     crteles     y MS
    -13.txt',
83          'librostxt/Dan 225 a os de prisi n a integrante de Los Zetas.
    txt',
84          'librostxt/desarrollan-sistema-que-diagnostica-problemas-de-
    columna.txt',
85          'librostxt/Descentralizaci n de bur cratas sindicalizados,
    s lo de forma voluntaria,FSTSE.txt',
86          'librostxt/Desconoce UNAM presunto ataque contra alumna de CCH
    Naucalpan.txt',
87          'librostxt/Desde 2004 cerraron m s de mil 800 peri dicos en
    EU.txt',
88          'librostxt/Destruye PGR m s de 1 ton de productos marinos en
    veda.txt',
89          'librostxt/Detienen en EU a presunto secuestrador de hija de
    Nelson Vargas.txt',
90          'librostxt/Digitalizaci n sustituir  hasta 180 millones de
    empleos ocupados por mujeres.txt',
91          'librostxt/Disney cede para reducir presi n antimonopolios
    tras compra de Fox.txt',
92          'librostxt/divulgacio-n-cienti-fica-con-mucha-cafei-na.txt',
93          'librostxt/educacio-n-financiera-para-nin-os.txt',
94          'librostxt/El colapso del volc n Etna.txt',
95          'librostxt/El FMI nombr  a representante en Argentina para
    reabrir su oficina.txt',
96          'librostxt/El lunes entregar n concesiones de agua a
    particulares hasta por 30 a os.txt',
97          'librostxt/El ultim tum de la embajada de Ecuador a Julian
    Assange para que cuide de su gato.txt',
98          'librostxt/el-discurso-visual-de-mujeres-artistas-en-chiapas.
    txt',
99          'librostxt/el-impacto-de-los-mexicanos-en-el-cern.txt',
100         'librostxt/el-impacto-social-del-ecoturismo2018-08-17-19-50-01.
    txt',
101         'librostxt/el-uso-de-la-computadora-su-relevancia-en-la-economi
    -a-y-el-trabajo.txt',
102         'librostxt/Empleados de Rappi buscan crear el primer sindicato
    de plataformas digitales del mundo.txt',
```

```
103         'librostxt/En auge, el mercado del tatuaje.txt',
104         'librostxt/entender-la-mente-humana-para-comprender-la-economi-
    a2018-08-30-20-34-36.txt',
105         'librostxt/Entrega hoy Pe a segunda etapa de carretera en
    Michoac n.txt',
106         'librostxt/Es constitucional que el Gobierno de la CdMx suprima
     programas sociales.txt',
107         'librostxt/Exige AI liberaci n inmediata de l deres catalanes
    .txt',
108         'librostxt/Expo Capital Humano pone el foco en tecnolog a,
    competencias y seguridad.txt',
109         'librostxt/fa-rmaco-econo-mico-y-accesible-para-virus-de-
    influenza.txt',
110         'librostxt/forman-investigadoras-con-perspectiva-de-ge-nero.txt
    ',
111         'librostxt/fragilidad-y-maltrato-en-adultos-mayores.txt',
112         'librostxt/gastronomi-a-maya-y-arqueologi-a-de-los-sentidos.txt
    ',
113         'librostxt/Generar nuevas audiencias, propuesta de la Muestra
    Nacional de Teatro.txt',
114         'librostxt/Gobierno desperdicia 656 millones de pesos en
    hospitales bajo esquema APP.txt',
115         'librostxt/Gobierno, obligado a consultar megaproyectos a
    ind genas.txt',
116         'librostxt/Google desaf a a Apple con inteligencia artificial.
    txt',
117         'librostxt/Google+ cerrar en agosto de 2019.txt',
118         'librostxt/Guatemala niega visas a miembros de comisi n contra
     impunidad.txt',
119         'librostxt/Guatemala,No habr  m s bolsas de pl stico.txt',
120         'librostxt/Hallan fosa clandestina con 10 cuerpos en Jalisco.
    txt',
121         'librostxt/Han participado 22.5 millones de personas en
    jornadas de salud.txt',
122         'librostxt/Impugnan proyecto de la Corte sobre empleadas
    dom sticas.txt',
123         'librostxt/Incremento salarial estar  por encima de la
    inflaci n.txt',
124         'librostxt/infancia-media-y-la-concepcio-n-del-suicidio.txt',
125         'librostxt/innovacio-n-mexicana-desde-el-oce-ano-a-rtico.txt',
126         'librostxt/inteligencia-artificial-el-psico-logo-del-futuro.txt
    ',
127         'librostxt/investigador-de-quere-taro-nuevo-miembro-de-la-amc.
    txt',
128         'librostxt/J venes en Canad  estrenan especialidad
    universitaria en mariguana.txt',
129         'librostxt/Jubilados de la Universidad Michoacana exigen pago
    de pensiones.txt',
130         'librostxt/La consulta sobre el nuevo aeropuerto s  es
    representativa, afirma AMLO.txt',
131         'librostxt/La IA podr a ampliar la brecha entre pa ses,
    empresas y trabajadores.txt',
132         'librostxt/La movilidad a debate.txt',
133         'librostxt/La OIT presentar  en 2019 una nueva iniciativa por
```

```
el futuro del trabajo.txt',
134         'librostxt/La  soluci n  del  Infonavit  para  que  t    y  tu  pareja
    compren  casa.txt',
135         'librostxt/la-ciencia-entre-la-incertidumbre-y-la-certeza.txt',
136         'librostxt/la-ciudad-de-los-archivos-resguarda-memoria-histo-
    rica-de-oaxaca.txt',
137         'librostxt/la-importancia-del-periodista-cienti-fico.txt',
138         'librostxt/Las criptomonedas son inviables para sustituir al
    euro.txt',
139         'librostxt/la-salud-en-la-frontera-sur-de-me-xico.txt',
140         'librostxt/la-santeri-a-una-visio-n-antropolo-gica.txt',
141         'librostxt/Lideres-se-aprueban-bonos.txt',
142         'librostxt/Llama AMLO a elevar  producci n  petrolera  del  pa s.
    txt',
143         'librostxt/Logran crear vida artificial  cu ntica  con una
    supercomputadora por primera vez en la historia.txt',
144         'librostxt/Los errores de Salinas, Fox, Calder n y  Pe a,
    seg n  Slim.txt',
145         'librostxt/Los mercados  burs tiles  frenan  sus  p rdidas.txt',
146         'librostxt/Los postes de CFE.txt',
147         'librostxt/Los-pobres-no-comen-gasolina.txt',
148         'librostxt/lucha-contra-el-ca-ncer-de-mama-desde-la-inmunologi-
    a - copia.txt',
149         'librostxt/ma-s-apoyo-a-la-ciencia-de-frontera-mari-a-elena-a-
    lvarez-buylla.txt',
150         'librostxt/matema-ticas-ciencia-divertida-y-viva - copia.txt',
151         'librostxt/mecate-propuestas-tecnolo-gicas-para-la-
    transparencia.txt',
152         'librostxt/M xico  debe  erradicar  el  trabajo  infantil  en  7
    a os,  OIT.txt',
153         'librostxt/miguel-gonza-lez-mendoza-de-la-inteligencia-
    artificial-especi-fica-a-la-superinteligencia - copia.txt',
154         'librostxt/Morenista presenta ley para prevenir el suicidio.txt
    ',
155         'librostxt/Mujeres rurales padecen mayor  marginaci n.txt',
156         'librostxt/Mujer-fallece-durante-cirug a-est tica.txt',
157         'librostxt/murcie-lagos-los-polinizadores-naturales-del-noreste
    .txt',
158         'librostxt/narcocorrido-el-ge-nero-musical-que-distingue-a-me-
    xico.txt',
159         'librostxt/Narcotr fico en M xico, la misteriosa vida de las
    familias de los capos de la droga.txt',
160         'librostxt/nin-os-jornaleros.txt',
161         'librostxt/No  habr  amnist a a quienes da aron  erario,  dice
    S nchez  Cordero.txt',
162         'librostxt/noe-torres-en-el-pasado-de-las-religiones.txt',
163         'librostxt/nopal-una-fuente-viable-de-gas-metano.txt',
164         'librostxt/nuevas-propuestas-para-entender-el-cambio-clima-tico
    .txt',
165         'librostxt/Odebrecht, la ruta de la corrupci n.txt',
166         'librostxt/Orange se  al a  con Google.txt',
167         'librostxt/OXXO  realizar  entregas a domicilio por medio de
    una app en 2019.txt',
168         'librostxt/Pareja-de-Ecatepec-es-imputada-por-delito-de-
```

```
feminicidio.txt',
169         'librostxt/pensamos-mucho-en-sexenios-y-poco-en-el-futuro-
    enrique-cabrero.txt',
170         'librostxt/periodismo-e-IA.txt',
171         'librostxt/PGR destruye m s de tres mil armas en Tamaulipas.
    txt',
172         'librostxt/Pide senador ir tras contribuyentes que falsifican
    facturas digitales.txt',
173         'librostxt/pla-sticos-redes-fantasmas-y-altas-temperaturas-
    amenazas-de-las-tortugas-marinas.txt',
174         'librostxt/Por qu  dormir deber a ser la prioridad de todos
    los estudiantes.txt',
175         'librostxt/Por qu  McDonalds cre  una hamburguesa sin
    carne que solo vende en Suecia.txt',
176         'librostxt/preservacio-n-de-tortugas-marinas-en-santander-
    veracruz.txt',
177         'librostxt/prevencio-n-primaria-la-clave-para-la-salud-
    cardiovascular-en-me-xico.txt',
178         'librostxt/Programas de aprendices deben ser semilleros de
    talento,.txt',
179         'librostxt/promueven-reciclaje-de-aguas-industriales-en-pymes-a
    -nivel-iberoame-rica.txt',
180         'librostxt/Propone Morena hasta 60 a os de c rcel por
    feminicidio en CDMX.txt',
181         'librostxt/Qu  es el impuesto digital propuesto por
    legisladores en M xico.txt',
182         'librostxt/Qu  es la ameba come cerebros, y c mo puedes
    evitar que te contagie.txt',
183         'librostxt/que-suen-an-las-mujeres-con-depresio-n2018
    -09-10-14-41-30.txt',
184         'librostxt/que-ven-los-nin-os-en-los-medios-de-comunicacio-n.
    txt',
185         'librostxt/Qui n creo el lenguaje C.txt',
186         'librostxt/quieres-donar-tu-cuerpo-a-la-ciencia.txt',
187         'librostxt/radiografi-a-del-sicario-mexicano.txt',
188         'librostxt/Razones por las que no tienes que ver Made in
    M xico.txt',
189         'librostxt/reconoce-premio-lore-al-unesco-2018-talento-de-
    cienti-ficas-mexicanas.txt',
190         'librostxt/Recuerdan desaparici n forzada de activista en
    Guerrero, ocurrida en 1976.txt',
191         'librostxt/recuperacio-n-asistida-de-petro-leo.txt',
192         'librostxt/Reitera la OACI que Texcoco es la mejor opci n para
     el NAIM.txt',
193         'librostxt/Retinopat a di betica, primera causa de ceguera en
      pa s.txt',
194         'librostxt/Se espera temporal de lluvias en gran parte de
    M xico.txt',
195         'librostxt/Se revocar n concesiones de agua, dice Monreal.txt'
    ,
196         'librostxt/Sears se declara en quiebra afectada por grandes
    deudas y p rdidas.txt',
197         'librostxt/se-buscan-cienti-ficos-emprendedores.txt',
198         'librostxt/Seis cosas que podemos hacer para evitar una
```

```
    cat strofe clim tica.txt',
199         'librostxt/Siete de cada 10 que se tat an, se arrepienten,
    especialistas.txt',
200         'librostxt/sistema-para-evitar-la-somnolencia-al-conducir.txt',
201         'librostxt/sobrestimada-la-obesidad-en-adultos-mayores-
    mexicanos.txt',
202         'librostxt/Suman ocho las v ctimas por derrumbe en centro
    comercial de NL.txt',
203         'librostxt/tecnologi-a-para-atender-problemas-nacionales.txt',
204         'librostxt/tecnologi-as-4-0-para-la-industria-automotriz.txt',
205         'librostxt/Tesla-solicit -una-patente-para-comercializar-la-
    marca-Telasquila.txt',
206         'librostxt/tiene-valor-moral-un-cada-ver-humano.txt',
207         'librostxt/Transformar al INEE, en lugar de eliminarlo.txt',
208         'librostxt/Trayectoria y consecuencias los huracanes que
    golpean Estados Unidos y M xico.txt',
209         'librostxt/Trump revira y admite la existencia del cambio
    clim tico.txt',
210         'librostxt/Un mensaje de aliento desde el 5 de octubre.txt',
211         'librostxt/Violencia en M xico durante 2017 cost  $4.72
    billones.txt',
212         'librostxt/violencia-y-vi-ctimas-hacia-un-nuevo-periodismo-
    judicial.txt',
213         'librostxt/0palabrasagregadas.txt']
214
215
216
217
218 #Guarda diccionario
219 def guardar_datos(dic,nombre):
220     print("Guardando diccionario...")
221     with open(nombre + ".dat", "wb") as f:
222         cPickle.dump(dic, f)
223         print("Diccionario guardado :)")
224
225 #Crea diccionario
226 def crea_diccionario(tokens):
227     cuenta = collections.defaultdict(int)
228     for w in tokens:
229         cuenta[w] +=1
230     return cuenta
231
232 #Crea diccionario relativo
233 def f_distribucion(dicc):
234     frecuencias = dicc.values()
235     aux = sum(frecuencias)
236     frecuencias = [x/aux for x in frecuencias]
237     return dict(zip(dicc.keys(),frecuencias))
238
239 #para quitar acentos y simbolos manteniendo la
240 def withoutSimb (s):
241     s1 = s.replace("  ", "#").replace("  ", "%")
242     s2 = unicodedata.normalize("NFKD", s1)\
243          .encode("ascii","ignore").decode("ascii")\
```

```
244          .replace("#", "   ").replace("%", "   ")
245     return s2
246
247 #Carga libros
248 corpus = []
249 corpusW = []
250 count =  0
251 for i in libros:
252      count += 1
253      lib = open(i,"r",encoding="utf8")
254      aux = lib.read()
255      lib.close()
256      #Crea corpus, falta bucle
257      aux = aux.rstrip()
258      review = re.sub('[^a-zA- Z                                    ]', ' ', aux)
259      review = review.lower()
260      review = review.split()
261      corpus += review
262      for rev in review:
263          corpusW.append(withoutSimb (rev))
264
265
266 #Crea el diccionario
267 dicc = crea_diccionario(corpus)
268 diccW = crea_diccionario(corpusW)
269 #Crea el diccionario relativo
270 dicc2 = f_distribucion(dicc)
271 dicc2W = f_distribucion(diccW)
272
273 #Se guardan ambos diccionarios
274 guardar_datos(dicc,"diccionario_absoluto")
275 guardar_datos(dicc2,"diccionario_relativo")
276 guardar_datos(corpus,"corpus")
277
278 #Se guardan ambos diccionarios sin acentos
279 guardar_datos(diccW,"diccionario_absoluto_w")
280 guardar_datos(dicc2W,"diccionario_relativo_w")
281 guardar_datos(corpusW,"corpus_w")
282
283 #Para exportancion y estimaci n en R
284 longitudes = [len(x) for x in dicc.keys()]
285 tam = open("longitudes.csv","w")
286 for x,y in zip(longitudes,dicc.values()):
287     tam.write("{:0d},{:0d}\n".format(x,y))
288 tam.close()
289
290
291 #borrar palabras que no existen
292 def guardar_datos2(dic):
293     print("Guardando diccionario...")
294     with open("diccionario_relativo.dat", "wb") as f:
295         cPickle.dump(dic, f)
296         print("Diccionario guardado :)")
297
```

```
298 prohibited=['der','lac','c satec','casatec']
299 for k in prohibited:
300      if dicc2.get(k) != None:
301           del dicc2[k]
302 guardar_datos2(dicc2)
```

6.6 Anexo 6 (Código Fuente Diccionario Bigramas)

```
# -*- coding:
"""
Created on Sat Nov 10 13:43:30 2018
@author: Andrea Monserrat y Oscar Cano
Mail: oscar.cfel@hotmail.com
"""
import _pickle as cPickle
import re
import nltk
import unicodedata

#Libros
libros = ['librostxt/0palabrasagregadas.txt',
          'librostxt/El laberinto de la soledad - Octavio Paz.txt',
          'librostxt/Gringo_Viejo.txt',
          'librostxt/Juan_Salvador_Gaviota.txt',
          'librostxt/videojuegos-la-normalizacio-n-de-la-violencia.txt',
          'librostxt/la-lucha-por-abatir-el-desperdicio-de-alimentos-en-me-xico.txt',
          'librostxt/produccio-n-de-ganado-en-condiciones-de-sequi-a.txt',
          'librostxt/promueve-fenaci-proyectos-cienti-ficos-y-tecnolo-gicos-en-quere-taro.txt',
          'librostxt/triunfa-mexicana-en-concurso-internacional-de-leyes-de-harvard.txt',
          'librostxt/adio-s-al-tlcan.txt',
          'librostxt/agua-de-reu-so-la-solucio-n-para-la-escasez-en-valle-de-guadalupe.txt',
          'librostxt/aumento-de-temperatura-para-baja-california-en-las-pro-ximas-de-cadas.txt',
          'librostxt/cerveza-artesanal-una-industria-competitiva.txt',
          'librostxt/chile-habanero-de-yucata-n-u-nico-en-el-mundo.txt',
          'librostxt/ciencia-por-mi-ciudad-juventud-e-innovacio-n.txt',
          'librostxt/co-mo-hacer-un-transge-nico.txt',
          'librostxt/co-mo-se-adaptan-los-peces-arrecifales-al-cambio-clima-tico.txt',
          'librostxt/cuando-la-inteligencia-artificial-nos-supere.txt',
          'librostxt/disen-o-color-y-simetri-a-a-lo-bestia.txt',
          'librostxt/do-nde-verter-los-materiales-del-dragado-portuario.txt',
          'librostxt/el-desafi-o-de-la-donacio-n-de-o-rganos.txt',
          'librostxt/El-planeta-solo-tiene-hasta-el-2030.txt',
          'librostxt/en-busca-del-superamaranto.txt',
          'librostxt/escenarios-de-riesgos-por-sismos-e-inundaciones.txt',
          'librostxt/factores-socioecono-micos-del-sobrepeso-y-obesidad-en-mexicanos.txt',
          'librostxt/fi-sica-y-ficcio-n-literatura-que-roza-la-realidad.txt',
          'librostxt/impulsan-programa-nacional-de-monitoreo-costero.txt',
```

```
        'librostxt/ingenieri-a-mexicana-en-accio-n.txt',
        'librostxt/intervencio-n-lingu-i-stica-para-fortalecer-
identidad-indi-gena-final.txt',
        'librostxt/la-ciencia-detra-s-de-los-suen-os.txt',
        'librostxt/maten-al-leon-de-ibarguengoitia-user16.txt',
        'librostxt/[Paz_Octavio]_Mascaras_Mexicanas(BookFi).txt',
        'librostxt/[Pealosa_Joaquin_Antonio]_Vida_Pasion_Y_Muerte_D(
BookFi).txt',
        'librostxt/Ibarguengoitia, Jorge - Estas ruinas que ves.txt',
        'librostxt/ibarguengoitia-jorge-los-relampagos-de-agosto.txt',
        'librostxt/LOS PASOS DE L PEZ, DE JORGE IBARG ENGOITIA.txt',
        'librostxt/noticia.txt',
        'librostxt/   Coco    hace crecer el consumo de comida mexicana
 en China.txt',
        'librostxt/7-habilidades-ba-sicas-para-los-empleos-del-futuro.
txt',
        'librostxt/advierten-sobre-toxicidad-en-peli-culas-para-
envolver-alimentos.txt',
        'librostxt/Alista L pez Obrador propuestas para Defensa y
Marina.txt',
        'librostxt/al-rescate-del-oso-negro-americano.txt',
        'librostxt/Amazon-patenta-tecnolog a-que-sabra-si-estas-
enfermo.txt',
        'librostxt/AMLO no obligar  traslados de trabajadores
federales.txt',
        'librostxt/AMLO se reunir  con el gobernador que le declaro la
 guerra.txt',
        'librostxt/Ampl a Issste numero de salas de hemodinamia a
nivel nacional.txt',
        'librostxt/Anal tica de datos, o c mo los Astros ganaron la
Serie Mundial.txt',
        'librostxt/aprendizaje-computacional-y-algoritmos-heuri-sticos-
para-la-resolucio-n-de-problemas.txt',
        'librostxt/As  de complicado y est pido es el racismo
mexicano.txt',
        'librostxt/Bancomer y Santander perder n millones en Argentina
 por hiperinflaci n.txt',
        'librostxt/calentamiento-global-principal-problema-para-la-
agenda-poli-tica-ambiental.txt',
        'librostxt/Campesinos bloquean Constituyentes frente a SHCP.txt
',
        'librostxt/ce-sar-burgos-el-investigador-de-los-narcocorridos.
txt',
        'librostxt/cesa-rea-o-parto-natural.txt',
        'librostxt/cli-nica-del-duelo.txt',
        'librostxt/C mo Facebook, Google y otras empresas
tecnol gicas buscan acabar con la adicci n al m vil.txt',
        'librostxt/co-mo-disminuir-la-contaminacio-n-de-ladrilleras.txt
',
        'librostxt/Comunidad estudiantil convoca a una consulta
universitaria sobre NAICM.txt',
        'librostxt/Con llenos totales, Les Ballets de Montecarlo
triunfa en el Cervantino.txt',
        'librostxt/conacyt-y-el-gobierno-de-guanajuato-apoyan-proyectos
```

```
-cienti-ficos-tecnolo-gicos-y-de-innovacio-n.txt',
73          'librostxt/conductas-alimentarias-y-estilo-de-vida-de-jo-venes-
    mexicanos-y-de-latinoame-rica.txt',
74          'librostxt/Conservar tu empleo puede depender m s de tus
    habilidades blandas que de las t cnicas.txt',
75          'librostxt/Considera Trump nuevo plan de separaci n de
    familias en la frontera.txt',
76          'librostxt/consolida-tecnm-en-celaya-posgrado-de-competencia-
    internacional.txt',
77          'librostxt/Convivir con tecnolog a y actualizarse.txt',
78          'librostxt/Coparmex pide legislaci n laboral correcta para
    evitar quejas ante OIT.txt',
79          'librostxt/Coreas acuerdan mejorar ferrocarriles y carreteras
    en frontera.txt',
80          'librostxt/Crea EU fuerza especial contra     crteles     y MS
    -13.txt',
81          'librostxt/Dan 225 a os de prisi n a integrante de Los Zetas.
    txt',
82          'librostxt/desarrollan-sistema-que-diagnostica-problemas-de-
    columna.txt',
83          'librostxt/Descentralizaci n de bur cratas sindicalizados,
    s lo de forma voluntaria,FSTSE.txt',
84          'librostxt/Desconoce UNAM presunto ataque contra alumna de CCH
    Naucalpan.txt',
85          'librostxt/Desde 2004 cerraron m s de mil 800 peri dicos en
    EU.txt',
86          'librostxt/Destruye PGR m s de 1 ton de productos marinos en
    veda.txt',
87          'librostxt/Detienen en EU a presunto secuestrador de hija de
    Nelson Vargas.txt',
88          'librostxt/Digitalizaci n sustituir  hasta 180 millones de
    empleos ocupados por mujeres.txt',
89          'librostxt/Disney cede para reducir presi n antimonopolios
    tras compra de Fox.txt',
90          'librostxt/divulgacio-n-cienti-fica-con-mucha-cafei-na.txt',
91          'librostxt/educacio-n-financiera-para-nin-os.txt',
92          'librostxt/El colapso del volc n Etna.txt',
93          'librostxt/El FMI nombr  a representante en Argentina para
    reabrir su oficina.txt',
94          'librostxt/El lunes entregar n concesiones de agua a
    particulares hasta por 30 a os.txt',
95          'librostxt/El ultim tum de la embajada de Ecuador a Julian
    Assange para que cuide de su gato.txt',
96          'librostxt/el-discurso-visual-de-mujeres-artistas-en-chiapas.
    txt',
97          'librostxt/el-impacto-de-los-mexicanos-en-el-cern.txt',
98          'librostxt/el-impacto-social-del-ecoturismo2018-08-17-19-50-01.
    txt',
99          'librostxt/el-uso-de-la-computadora-su-relevancia-en-la-economi
    -a-y-el-trabajo.txt',
100         'librostxt/Empleados de Rappi buscan crear el primer sindicato
    de plataformas digitales del mundo.txt',
101         'librostxt/En auge, el mercado del tatuaje.txt',
102         'librostxt/entender-la-mente-humana-para-comprender-la-economi-
```

```
a2018-08-30-20-34-36.txt',
103         'librostxt/Entrega hoy Pe a segunda etapa de carretera en
    Michoac n.txt',
104         'librostxt/Es constitucional que el Gobierno de la CdMx suprima
     programas sociales.txt',
105         'librostxt/Exige AI liberaci n inmediata de l deres catalanes
    .txt',
106         'librostxt/Expo Capital Humano pone el foco en tecnolog a,
    competencias y seguridad.txt',
107         'librostxt/fa-rmaco-econo-mico-y-accesible-para-virus-de-
    influenza.txt',
108         'librostxt/forman-investigadoras-con-perspectiva-de-ge-nero.txt
    ',
109         'librostxt/fragilidad-y-maltrato-en-adultos-mayores.txt',
110         'librostxt/gastronomi-a-maya-y-arqueologi-a-de-los-sentidos.txt
    ',
111         'librostxt/Generar nuevas audiencias, propuesta de la Muestra
    Nacional de Teatro.txt',
112         'librostxt/Gobierno desperdicia 656 millones de pesos en
    hospitales bajo esquema APP.txt',
113         'librostxt/Gobierno, obligado a consultar megaproyectos a
    ind genas.txt',
114         'librostxt/Google desaf a a Apple con inteligencia artificial.
    txt',
115         'librostxt/Google+ cerrar  en agosto de 2019.txt',
116         'librostxt/Guatemala niega visas a miembros de comisi n contra
     impunidad.txt',
117         'librostxt/Guatemala,No habr  m s bolsas de pl stico.txt',
118         'librostxt/Hallan fosa clandestina con 10 cuerpos en Jalisco.
    txt',
119         'librostxt/Han participado 22.5 millones de personas en
    jornadas de salud.txt',
120         'librostxt/Impugnan proyecto de la Corte sobre empleadas
    dom sticas.txt',
121         'librostxt/Incremento salarial estar  por encima de la
    inflaci n.txt',
122         'librostxt/infancia-media-y-la-concepcio-n-del-suicidio.txt',
123         'librostxt/innovacio-n-mexicana-desde-el-oce-ano-a-rtico.txt',
124         'librostxt/inteligencia-artificial-el-psico-logo-del-futuro.txt
    ',
125         'librostxt/investigador-de-quere-taro-nuevo-miembro-de-la-amc.
    txt',
126         'librostxt/J venes en Canad  estrenan especialidad
    universitaria en mariguana.txt',
127         'librostxt/Jubilados de la Universidad Michoacana exigen pago
    de pensiones.txt',
128         'librostxt/La consulta sobre el nuevo aeropuerto s  es
    representativa, afirma AMLO.txt',
129         'librostxt/La IA podr a ampliar la brecha entre pa ses,
    empresas y trabajadores.txt',
130         'librostxt/La movilidad a debate.txt',
131         'librostxt/La OIT presentar  en 2019 una nueva iniciativa por
    el futuro del trabajo.txt',
132         'librostxt/La soluci n del Infonavit para que t  y tu pareja
```

```
compren casa.txt',
133         'librostxt/la-ciencia-entre-la-incertidumbre-y-la-certeza.txt',
134         'librostxt/la-ciudad-de-los-archivos-resguarda-memoria-histo-
rica-de-oaxaca.txt',
135         'librostxt/la-importancia-del-periodista-cienti-fico.txt',
136         'librostxt/Las criptomonedas son inviables para sustituir al
euro.txt',
137         'librostxt/la-salud-en-la-frontera-sur-de-me-xico.txt',
138         'librostxt/la-santeri-a-una-visio-n-antropolo-gica.txt',
139         'librostxt/Lideres-se-aprueban-bonos.txt',
140         'librostxt/Llama AMLO a elevar producci n petrolera del pa s.
txt',
141         'librostxt/Logran crear vida artificial cu ntica con una
supercomputadora por primera vez en la historia.txt',
142         'librostxt/Los errores de Salinas, Fox, Calder n y Pe a,
seg n Slim.txt',
143         'librostxt/Los mercados burs tiles frenan sus p rdidas.txt',
144         'librostxt/Los postes de CFE.txt',
145         'librostxt/Los-pobres-no-comen-gasolina.txt',
146         'librostxt/lucha-contra-el-ca-ncer-de-mama-desde-la-inmunologi-
a - copia.txt',
147         'librostxt/ma-s-apoyo-a-la-ciencia-de-frontera-mari-a-elena-a-
lvarez-buylla.txt',
148         'librostxt/matema-ticas-ciencia-divertida-y-viva - copia.txt',
149         'librostxt/mecate-propuestas-tecnolo-gicas-para-la-
transparencia.txt',
150         'librostxt/M xico debe erradicar el trabajo infantil en 7
a os, OIT.txt',
151         'librostxt/miguel-gonza lez-mendoza-de-la-inteligencia-
artificial-especi-fica-a-la-superinteligencia - copia.txt',
152         'librostxt/Morenista presenta ley para prevenir el suicidio.txt
',
153         'librostxt/Mujeres rurales padecen mayor marginaci n.txt',
154         'librostxt/Mujer-fallece-durante-cirug a-est tica.txt',
155         'librostxt/murcie-lagos-los-polinizadores-naturales-del-noreste
.txt',
156         'librostxt/narcocorrido-el-ge-nero-musical-que-distingue-a-me-
xico.txt',
157         'librostxt/Narcotr fico en M xico, la misteriosa vida de las
familias de los capos de la droga.txt',
158         'librostxt/nin-os-jornaleros.txt',
159         'librostxt/No habr  amnist a a quienes da aron erario, dice
S nchez Cordero.txt',
160         'librostxt/noe-torres-en-el-pasado-de-las-religiones.txt',
161         'librostxt/nopal-una-fuente-viable-de-gas-metano.txt',
162         'librostxt/nuevas-propuestas-para-entender-el-cambio-clima-tico
.txt',
163         'librostxt/Odebrecht, la ruta de la corrupci n.txt',
164         'librostxt/Orange se al a con Google.txt',
165         'librostxt/OXXO realizar  entregas a domicilio por medio de
una app en 2019.txt',
166         'librostxt/Pareja-de-Ecatepec-es-imputada-por-delito-de-
feminicidio.txt',
167         'librostxt/pensamos-mucho-en-sexenios-y-poco-en-el-futuro-
```

```
enrique-cabrero.txt',
168         'librostxt/periodismo-e-IA.txt',
169         'librostxt/PGR destruye m s de tres mil armas en Tamaulipas.
    txt',
170         'librostxt/Pide senador ir tras contribuyentes que falsifican
    facturas digitales.txt',
171         'librostxt/pla-sticos-redes-fantasmas-y-altas-temperaturas-
    amenazas-de-las-tortugas-marinas.txt',
172         'librostxt/Por qu  dormir deber a ser la prioridad de todos
    los estudiantes.txt',
173         'librostxt/Por qu  McDonalds cre  una hamburguesa sin
    carne que solo vende en Suecia.txt',
174         'librostxt/preservacio-n-de-tortugas-marinas-en-santander-
    veracruz.txt',
175         'librostxt/prevencio-n-primaria-la-clave-para-la-salud-
    cardiovascular-en-me-xico.txt',
176         'librostxt/Programas de aprendices deben ser semilleros de
    talento,.txt',
177         'librostxt/promueven-reciclaje-de-aguas-industriales-en-pymes-a
    -nivel-iberoame-rica.txt',
178         'librostxt/Propone Morena hasta 60 a os de c rcel por
    feminicidio en CDMX.txt',
179         'librostxt/Qu  es el impuesto digital propuesto por
    legisladores en M xico.txt',
180         'librostxt/Qu  es la ameba come cerebros, y c mo puedes
    evitar que te contagie.txt',
181         'librostxt/que-suen-an-las-mujeres-con-depresio-n2018
    -09-10-14-41-30.txt',
182         'librostxt/que-ven-los-nin-os-en-los-medios-de-comunicacio-n.
    txt',
183         'librostxt/Qui n creo el lenguaje C.txt',
184         'librostxt/quieres-donar-tu-cuerpo-a-la-ciencia.txt',
185         'librostxt/radiografi-a-del-sicario-mexicano.txt',
186         'librostxt/Razones por las que no tienes que ver Made in
    M xico.txt',
187         'librostxt/reconoce-premio-lore-al-unesco-2018-talento-de-
    cienti-ficas-mexicanas.txt',
188         'librostxt/Recuerdan desaparici n forzada de activista en
    Guerrero, ocurrida en 1976.txt',
189         'librostxt/recuperacio-n-asistida-de-petro-leo.txt',
190         'librostxt/Reitera la OACI que Texcoco es la mejor opci n para
     el NAIM.txt',
191         'librostxt/Retinopat a di betica, primera causa de ceguera en
     pa s.txt',
192         'librostxt/Se espera temporal de lluvias en gran parte de
    M xico.txt',
193         'librostxt/Se revocar n concesiones de agua, dice Monreal.txt'
    ,
194         'librostxt/Sears se declara en quiebra afectada por grandes
    deudas y p rdidas.txt',
195         'librostxt/se-buscan-cienti-ficos-emprendedores.txt',
196         'librostxt/Seis cosas que podemos hacer para evitar una
    cat strofe clim tica.txt',
197         'librostxt/Siete de cada 10 que se tat an, se arrepienten,
```

```
especialistas.txt',
198         'librostxt/sistema-para-evitar-la-somnolencia-al-conducir.txt',
199         'librostxt/sobrestimada-la-obesidad-en-adultos-mayores-
    mexicanos.txt',
200         'librostxt/Suman ocho las v ctimas por derrumbe en centro
    comercial de NL.txt',
201         'librostxt/tecnologi-a-para-atender-problemas-nacionales.txt',
202         'librostxt/tecnologi-as-4-0-para-la-industria-automotriz.txt',
203         'librostxt/Tesla-solicit -una-patente-para-comercializar-la-
    marca-Telasquila.txt',
204         'librostxt/tiene-valor-moral-un-cada-ver-humano.txt',
205         'librostxt/Transformar al INEE, en lugar de eliminarlo.txt',
206         'librostxt/Trayectoria y consecuencias los huracanes que
    golpean Estados Unidos y M xico.txt',
207         'librostxt/Trump revira y admite la existencia del cambio
    clim tico.txt',
208         'librostxt/Un mensaje de aliento desde el 5 de octubre.txt',
209         'librostxt/Violencia en M xico durante 2017 cost  $4.72
    billones.txt',
210         'librostxt/violencia-y-vi-ctimas-hacia-un-nuevo-periodismo-
    judicial.txt']
211
212 #Separar oraciones por signo de puntuaci n:
213 def procesa_simbolos(cadena):
214      puntuacion = [".",",",":",";","!","?","...", ")","]","-"]#los
    simbolos que estoy buscando
215      contador = 0
216      for k in cadena:#cada letra en mi oraci n
217           contador +=1
218           for j in puntuacion:#cada signo de puntuaci n
219                if k == j: #si coincide mi caracter de oraci n con alg n
     signo de puntuaci n
220                     return contador,j#posici n, puntuacion encontrada
221      return 0,""
222
223 #Crea diccionario relativo
224 def f_distribucion(dicc):
225     frecuencias = dicc.values()
226     aux = sum(frecuencias)
227     frecuencias = [x/aux for x in frecuencias]
228     return dict(zip(dicc.keys(),frecuencias))
229
230 #Guarda diccionario
231 def guardar_datos(dic,nombre):
232     print("Guardando diccionario...")
233     with open(nombre + ".dat", "wb") as f:
234         cPickle.dump(dic, f)
235         print("Diccionario guardado :)")
236
237 #para quitar acentos y simbolos manteniendo la
238 def withoutSimb (s):
239     s1 = s.replace("  ", "#").replace("  ", "%")
240     s2 = unicodedata.normalize("NFKD", s1)\
241          .encode("ascii","ignore").decode("ascii")\
```

```
242          .replace("#", "  ").replace("%", "  ")
243     return s2
244
245 #Carga libros
246 oraciones=[]#guarda las oraciones en una lista
247 corpus=[]
248 count=0
249 for i in libros:
250      count += 1
251      with  open(i,"r",encoding="utf8") as lib:
252       parrafo= [(columna.rstrip()) for columna in lib]
253       corpus += parrafo #todos los parrafos de los libros totales
254 for texto in corpus:
255     buff = ""
256     while True:
257         buff = texto
258         pos,sim = procesa_simbolos(texto)#encuentra la posici n y el
    simbolo
259         pos_tx=texto[:pos]#para que me pueda determinar mi posici n
260         pos_txa=texto[:pos-1]#es mi oracion hasta el primer signo de
    puntuaci n sin el
261         pos_txa = re.sub('[^a-zA- Z                              ]', ' ',
    pos_txa)
262         oraciones.append([pos_txa])
263         texto=texto.replace(pos_tx,"",1)
264         if texto == buff:
265              break
266         if texto=="":
267              break
268
269 ##creador de bigramass
270 bigramas=[]
271 for oracion in oraciones:
272      oracionToken=oracion[0].split()
273      for palabra in range(0,len(oracionToken)):#indices de cada palabra
274           if palabra !=0 : #identifica a la primer palabra de un renglon
275                bigramas.append([oracionToken[palabra-1],oracionToken[
    palabra]])
276
277 #Creador de diccionario con bigramass:
278 dicbig= {}
279 biguna = []
280 bigdos = []
281 for bigrama in bigramas:
282      accpt = 1
283      KeyDic= (bigrama[0]+ " "+ bigrama[1]).lower()#LAs llaves son las
    palabras
284      if len(bigrama[0]) == 1 and len(bigrama[1]) == 1:
285           biguna.append(KeyDic)
286           accpt = 0
287           if KeyDic == 'y a' or KeyDic == 'o a':
288                accpt = 1
289      if (len(bigrama[0]) == 2 and len(bigrama[1]) == 1) or (len(bigrama
    [0]) == 1 and len(bigrama[1]) == 2):
```

```
290             bigdos.append(KeyDic)
291             accpt = 0
292             if KeyDic == 'y a' or KeyDic == 'o a':
293                 accpt = 1
294       if dicbig.get(KeyDic) == None and accpt == 1:
295         dicbig[KeyDic]=1
296       elif dicbig.get(KeyDic) != None and accpt == 1:
297         dicbig[KeyDic]=dicbig.get(KeyDic)+1#agrega la frecuencia de la
      misma llave
298
299 keys = []
300 for key, value in dicbig.items():
301       if value < 4:
302             keys.append(key)
303 for key in keys:
304       del dicbig[key]
305
306 dicc2 = f_distribucion(dicbig)
307
308 freq = nltk.FreqDist(biguna)
309 freq.plot(20,cumulative=False,title="Frecuencias de Bigramas de una letra
      ")
310 freq2 = nltk.FreqDist(bigdos)
311 freq2.plot(50,cumulative=False,title="Frecuencias de Bigramas de una
      letra")
312
313
314 guardar_datos(dicbig,"diccionario_absoluto_big")
315 guardar_datos(dicc2,"diccionario_relativo_big")
316 guardar_datos(bigramas,"corpus_big")
```

6.7 Anexo 7 (Código Fuente Obtención Tweets)

```
# -*- coding: utf-8 -*-
"""
Created on Tue Nov 20 15:42:40 2018
@author: Andrea Monserrat
@editor: Oscar Cano F lix
"""

import tweepy
import tweepy.api
import _pickle as cPickle
##########################################
consumer_name = "SeptTags"
API_key = "yOFrLYJZ9hyTJtQT3NxbT2yUK" #APIkey
API_secret_key = "0RbjOnqEBnMbKjGfgDYeWulqBhipShe35i4q3S55Nq2J5TBvIF"#
    Apisecrectkey
access_token="1064994930404274178-DbqZy5BwUJCshvLuuSFaYE7GHpiYbA"
access_token_secret = "68k5WeFwKBzCm2dsnVodR3Z1JTEC1wnugDCWOph1hwh6o"
Acceso= tweepy.OAuthHandler (API_key, API_secret_key)
Acceso.set_access_token (access_token, access_token_secret)
##########################################

##################FUnciones
##Funcion para obtener y limpiar haghstag
def hashtagfun(usrs):
     print("hashtagfun")
     tuits = []
     for i in usrs:
         t=api.user_timeline(screen_name = i, count = 2000, include_rts =
     False, tweet_mode = 'extended')
         for l in t:
              tuits.append(l.full_text)
              print(l.full_text)
     return (tuits)

#########################
##########Coneccion twiter
api = tweepy.API (Acceso)
usuario=api.me()
usuarios=["werevertumorro","ChumelTorres","eljuanpazurita","
    LuisitoComunica","yuyacst","Claudiashein","brozoxmiswebs","
    EugenioDerbez","Juandedios_P"]
a = hashtagfun(usuarios)

print("Guardando tweets...")
with open("tweets.dat", "wb") as f:
    cPickle.dump(a, f)
    print("...")
#with open("tweetsmeta.dat", "wb") as f:
#    cPickle.dump(b, f)
    print("Diccionario guardado :)")
```

Bibliografía

Abdi, Asad, Shamsuddin, Siti Mariyam, Hasan, Shafaatunnur, & Piran, Jalil. 2018. Machine learning-based multi-documents sentiment-oriented summarization using linguistic treatment. *Expert Systems with Applications*, **109**, 66–85.

Ackland, Robert, & Graham, Timothy. 2017. Text Analysis in R. *ACSPRI Summer Program 2017*, **11**(4), 5.

Arora, Sarita. 2017. *Python Natural Language Toolkit*.

Atserias, Jordi, Casas, Bernardino, Comelles, Elisabet, González, Meritxell, Padró, Lluís, & Padró, Muntsa. 2006 (may). FreeLing 1.3: Syntactic and semantic services in an open-source NLP library. *In: Proceedings of the fifth international conference on Language Resources and Evaluation (LREC 2006)*. ELRA, Genoa, Italy.

Barbieri, Fancesco, & Saggion, Horacio. 2014. Modelling Irony in Twitter (p. 56). *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, **53**(9), 56–64.

Barbieri, Francesco, Saggion, Horacio, & Ronzano, Francesco. 2014. Modelling sarcasm in twitter, a novel approach. *Workshop on computational approaches to subjectivity, sentiment and social media analysis*, 50–58.

Belgiu, Mariana, & Drăgu, Lucian. 2016. Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, **114**, 24–31.

Blais, Brian. 2011. Statistical inference for everyone.

Boussaïd, Ilhem, Lepagnot, Julien, & Siarry, Patrick. 2013. A survey on optimization metaheuristics. *Information Sciences*, **237**(November), 82–117.

Brent, Michael R. 1999. An Efficient, Probabilistically Sound Algorithm for Segmentation and Word Discovery. **105**, 71–105.

Brockler, Frank, Singh, Vatsala, Hutkowski, Paul, Ptucha, Raymond, Petroski Such, Felipe, & Pillai, Suhas. 2018. Intelligent character recognition using fully convolutional neural networks. *Pattern Recognition*, **88**, 604–613.

Bureau, U.S. Census. 2018. *U.S. Census Bureau*.

Calvo, David Vilares, Pardo, Miguel a Alonso, & Gómez Rodríguez, Carlos. 2014. *Análisis de contenidos en Twitter: clasificación de mensajes e identificación de la tendencia política de los usuarios*. Ph.D. thesis, Universidad de Coruña.

Cambria, Erik, Ebrahimi, Monireh, Hossein Yazdavar, Amir, Sheth, Amit, & Center, Knoesis. 2017. Challenges of sentiment analysis for dynamic events. *IEEE Intelligent Systems*, **32**(5), 70 – 75.

Carreras, Xavier, Chao, Isaac, Padró, Lluís, & Padró, Muntsa. 2004. FreeLing: An Open-Source Suite of Language Analyzers. *In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*.

Carvalho, Paula, Sarmento, Luís, Silva, Mário J., & de Oliveira, Eugénio. 2009. Clues for detecting irony in user-generated contents. *Proceeding of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion - TSA '09*, 53.

Castillo-Ron, Enrique. 1988. Estadística de Valores Extremos: Distribuciones Asintóticas. *Estadistica Española*, **116**, 5–34.

Choi, HongSeok, & Lee, Hyunju. 2019. Multitask learning approach for understanding the relationship between two sentences. *Information Sciences*, **485**, 413–426.

Chou, Wu (Avaya Labs Research, Avaya Inc.), Goel, Vaibhava (Ibm), Bryne, William (Johns Hopkins University), Huo, Qiang (University of Hong Kong), Katagiri, Shigeru (Ntt Communication Science Laboratories), Gauvain, Jean-Luc (Limsi), Lamel, Lori (Limsi), Furui, Sadaoki (Tokyo Institute of Technology), Li, Qi (Bell Labs), Juang, Biing-Hwang (Avaya Labs Research), Schwartz, Richard M. (BBN Technologies, Verizon), Makhoul, John (BBN Technologies, Verizon), Bellegarda, Jerome R. (Apple Computer, Inc.), Gorin, a.L. (At&T Laboratories), Abella, a. (At&T Laboratories), Alonso, T. (At&T Laboratories), Riccardi, G. (At&T Laboratories), Wright, J.H. (At&T Laboratories), Ney, Herman (Aachen University of Technology), Och, F.J. (Aachen University of Technology), & Allan, James (University of Massachusetts). 2003. Pattern Recognition in Speech and Language Processing. *New York*.

Chrupała, Grzegorz, Dinu, Georgiana, & van Genabith, Josef. 2008. Learning morphology with morfette. *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC '08*, 2362–2367.

Cisco. 2017. *Cisco Visual Networking Index: Forecast and Methodology, 2016–2021*.

Doval, Yerai, & Gómez-Rodríguez, Carlos. 2019. Comparing neural- and N-gram-based language models for word segmentation. *Journal of the Association for Information Science and Technology*, **70**(2), 187–197.

Du, Ke-Lin, & Swamy, M. N. S. 2014. *Neural Networks and Statistical Learning*.

Evans, Merran, Hastings, Nicholas, & Peacock, Brian. 2001. Statistical Distributions, Third Edition. *Measurement Science and Technology*, **12**(1), 117.

Fernández-Gavilanes, Milagros, Juncal-Martńnez, Jonathan, García-Méndez, Silvia, Costa-Montenegro, Enrique, & Javier González-Castaño, Francisco. 2019. Differentiating users by language and location estimation in sentiment analisys of informal text during major public events. *Expert Systems with Applications*, **117**, 15–28.

Gattani, Abhishek, Doan, AnHai, Lamba, Digvijay S., Garera, Nikesh, Tiwari, Mitul, Chai, Xiaoyong, Das, Sanjib, Subramaniam, Sri, Rajaraman, Anand, & Harinarayan, Venky. 2013. Entity extraction, linking, classification, and tagging for social media. *Proceedings of the VLDB Endowment*, **6**(11), 1126–1137.

Gelbukh, Alexander. 2009. Procesamiento del lenguaje natural: estado de la investigación. *I Simposio Internacional sobre Organización del Conocimiento: Bibliotecología y Terminología*.

Goldwater, S, Griffiths, T, & Johnson, M. 2006. Interpolating between types and tokens by estimating power-law generators. *Nips*, **18**, 459.

Goldwater, Sharon, Griffiths, Thomas L., & Johnson, Mark. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, **112**(1), 21–54.

González-Ibáñez, Roberto, Muresan, Smaranda, & Wacholder, Nina. 2011. Identifying Sarcasm in Twitter: A Closer Look. *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 581–586.

Halpern, Orit. 2015. *Beautiful Data*. Duke University Press.

Hohendahl, Andrés; Zelasco, Josñe. Flexionador y Estimador Idiomático , usando algoritmos eficientes y compactos para idiomas muy ricos en formas como el español. *Aires, Buenos Universitario, Campus Seco, Paraje Arroyo*, 1729–1739.

Hualde, Jose Ignacio, Olarrea, Antxon, Escobar, Anna María, & Travis, Catherine E. 2010. *Introducción a la Linguística Hispánica*.

Hussain, Amir, & Cambria, Erik. 2017. Semi-supervised learning for big social data analysis. *Neurocomputing*, **275**, 1662–1673.

James, Gareth, Witten, Daniela, Hastie, Trevor, & Tibshirani, Robert. 2000. *An introduction to Statistical Learning*. Vol. 7.

James, Gareth, Witten, Daniela, Hastie, Trevor, & Tibshirani, Robert. 2013. *An Introduction to Statistical Learning*. Springer Texts in Statistics, vol. 103, nos. 9–12. New York, NY: Springer New York.

Karoui, Jihen, Benamara, Farah, Patti, Viviana, Bosco, Cristina, & Aussenac-gilles, Nathalie. 2017. Exploring the Impact of Pragmatic Phenomena on Irony Detection in Tweets : A Multilingual Corpus Study. **1**(2), 262–272.

Kiusalaas, Jaan. 2013. *Numerical Methods in Engineering with Python 3*.

Le Roux, Joseph, Sagot, Benoit, & Seddah, Djamé. 2012. Statistical Parsing of Spanish and Data Driven Lemmatization. *Proceedings of the ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages*, 55–61.

Lee, Kathy, Palsetia, Diana, Narayanan, Ramanathan, Patwary, Md Mostofa Ali, Agrawal, Ankit, & Choudhary, Alok. 2011. Twitter trending topic classification. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 251–258.

Lenci, Alessandro. 2008. Distributional approaches in linguistic and cognitive research. *Italian Journal of Linguistics*, **20**(May), 1–31.

Liu, Dapeng, Li, Yan, & Thomas, Manoj A. 2017. A Roadmap for Natural Language Processing Research in Information Systems. *Pages 1112–1121 of: Proceedings of the 50th Hawaii International Conference on System Sciences*.

López, Víctor Ignacio, & Ramos, Rogelio. 2007. Una introducción a los diseños óptimos. *Revista Colombiana de Estadistica*, **30**(1), 37–51.

Lukin, Stephanie, & Walker, Marilyn. 2017. Really? Well. Apparently Bootstrapping Improves the Performance of Sarcasm and Nastiness Classifiers for Online Dialogue. **1**(Lasm), 30–40.

Mukherjee, Prasenjit, & Chakraborty, Baisakhi. 2016. Automated knowledge provider system with natural language query processing. *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, **33**(5), 525–538.

Munoz, Alfonso, & Arguelles Alvarez, Irina. 2010. Análisis del discurso en redes sociales. Twitter un caso bajo estudio. *Analizar datos > Describir variación [Recurso electrónico]: Analysing data > Describing variation*, **1**(April 2010), 64.

Nothman, Joel, Ringland, Nicky, Radford, Will, Murphy, Tara, & Curran, James R. 2013. Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, **194**, 151–175.

Olbertz, Hella. 1998. *Verbal Periphrases in a Functional Grammar of Spanish*.

Padró, Lluís, & Stanilovsky, Evgeny. 2012 (may). FreeLing 3.0: Towards Wider Multilinguality. *In: Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*. ELRA, Istanbul, Turkey.

Padró, Lluís, Collado, Miquel, Reese, Samuel, Lloberes, Marina, & Castellón, Irene. 2010a. FreeLing 2.1: Five years of open-source language processing tools. *Proceedings of the 7th International Conference on Language Resources and Evaluation*, 931–936.

Padró, Lluís, Collado, Miquel, Reese, Samuel, Lloberes, Marina, & Castellón, Irene. 2010b. FreeLing 2.1: Five years of open-source language processing tools. *Pages 931–936 of:* Calzolari, Nicoletta, Choukri, Khalid, Maegaard, Bente, Mariani, Joseph, Odijk, Jan, Piperidis, Stelios, Rosner, Mike, & Tapias, Daniel (eds), *Proceedings of the 7th International Conference on Language Resources and Evaluation*. European Language Resources Association (ELRA).

Reyes, Antonio, Rosso, Paolo, & Veale, Tony. 2013. A multidimensional approach for detecting irony in Twitter. *Language Resources and Evaluation*, **47**(1), 239–268.

Riloff, Ellen, Qadir, Ashequl, Surve, Prafulla, Silva, Lalindra De, Gilbert, Nathan, & Huang, Ruihong. 2013. Sarcasm as Contrast between a Positive Sentiment and Negative Situation. *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 704–714.

Ríos, Valeria. 2017. *Facebook supera los 2.000 millones de usuarios activos al mes*.

Saggion, Horacio. 2010. Procesamiento De Lenguaje Natural Para El Análisis De Lenguaje Subjetivo. *Subjetividad y Procesos Cognitivos*, **14**, 247–259.

Sánchez, Belém Priego, & Pinto, David. 2015. Identification of verbal phraseological units in Mexican news stories. *Computacion y Sistemas*, **19**(4), 713–720.

Sengupta, P, & Chaudhuri, B B. 1993. Natural Language Processing in an Indian Language (Bengali)-I: Verb Phrase Analysis. *IETE Technical Review*, **10**(1), 27–41.

Sicilia, Rosa, Lo, Stella, Pei, Yulong, Pechenizkiy, Mykola, & Soda, Paolo. 2018. Twitter rumour detection in the health domain. *Expert Systems with Applications*, **110**, 33–40.

Tellez, Eric S., Miranda-Jiménez, Sabino, Graff, Mario, Moctezuma, Daniela, Siordia, Oscar S., & Villaseñor, Elio A. 2017. A case study of Spanish text transformations for twitter sentiment analysis. *Expert Systems with Applications*, **81**(sep), 457–471.

Thorsten Brants, Alex Franz. 2006. *Web 1T 5-gram Version 1*.

Toledo Bastos, Marco Travitzki, Rodrigo Puschmann. 2014. What Sticks With Whom? Twitter Follower-Followee Networks and News Classification. *Review of International Political Economy*, **0**(0), 1–23.

Triefenbach, Fabian. 2008. Design of experiments: the D-optimal approach and its implementation as a computer algorithm. *Bachelor's Thesis in Information and Communication* ....

Tullio, Di, De, Manual, Buenos, Autónoma De, & Waldhuter, Aires. 2005. *Manual de gramática del español*.

Veale, Tony, & Hao, Yanfen. 2010. Detecting ironic intent in creative comparisons. *Frontiers in Artificial Intelligence and Applications*, **215**, 765–770.

Venkataraman, Anand. 2001. A Statistical Model for Word Discovery in Transcribed Speech.

Vilares, David, Alonso, Miguel A., & Gómez-Rodríguez, Carlos. 2015. A syntactic approach for opinion mining on Spanish reviews. *Natural Language Engineering*, **21**(1), 139–163.

Wang, Yining, & Singh, Aarti. 2016. Minimax Subsampling for Estimation and Prediction in Low-Dimensional Linear Regression.

Wasserman, Larry. 2011. *All of statistics*.

Winograd, Terry. 1972. Understanding natural language. *Cognitive Psychology*, **3**(1), 1–191.

Zhang, Qi, Liu, Xiaoyu, & Fu, Jinlan. 2018. Neural Networks Incorporating Dictionaries for Chinese Word Segmentation. *Aaai*, 5682–5689.