ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

MINERÍA DE TEXTO PARA CONSTRUIR LA FILOGENIA DE LOS INSECTOS VECTORES DE LA ENFERMEDAD DE CHAGAS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN

BRITO MORALES DANNI ANDRÉ
danni.brito@epn.edu.ec

LEMA AUZ BRYAN GERMÁN
bryan.lema@epn.edu.ec

DIRECTOR: HALLO CARRASCO MARÍA ASUNCIÓN
maria.hallo@epn.edu.ec

CODIRECTOR: CARRERA IZURIETA IVÁN MARCELO
ivan.carrera@epn.edu.ec

Quito, noviembre 2019

AVAL

Certificamos que el presente trabajo fue desarrollado por Brito Morales Danni André y Lema Auz Bryan Germán, bajo nuestra supervisión.

Hallo Carrasco María Asunción
Director

Carrera Izurieta Iván Marcelo
Codirector

DECLARACIÓN DE AUTORÍA

Nosotros, Brito Morales Danni André y Lema Auz Bryan Germán, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

Brito Morales Danni André	Lema Auz Bryan Germán

DEDICATORIA

Este proyecto está dedicado de todo corazón a mis amados padres, Jimmy y Amparo, quienes han sido mi fuente de inspiración y quienes me han dado la fortaleza de seguir adelante cuando he dudado en lo que hago, son ellos quienes me han dado incondicionalmente su apoyo moral, espiritual, emocional y financiero.

A mis hermanas, abuelos, tíos, primos, y demás familiares quienes me han hecho crecer personalmente al compartir conmigo palabras y experiencias a lo largo de mi vida universitaria.

Danni Brito

Dedico este proyecto, y cada uno de mis logros académicos y personales, a mis padres, Germán y Lucía, y a mis hermanas, Guisella y Paula, quienes han sido mi apoyo incondicional, en los mejores y peores momentos. Han sabido levantarme en las situaciones más difíciles, cuando mi mente y cuerpo se negaba a continuar, y me han acompañado en cada uno de mis logros y derrotas.

No he de olvidar a cada una de las personas que han pasado por mi vida, durante este periodo universitario, dejando pequeñas marcas que han ayudado, en parte, a convertirme en quien soy y espero poder seguir siendo.

Y como no, dejar un mensaje para las personas que han esperado que se celebre este gran momento. Mejor tarde que nunca.

Bryan Lema

AGRADECIMIENTO

Es con inmensa gratitud que reconozco el apoyo y soporte de mis amigos, compañeros, profesores y demás personas que han compartido conmigo, aunque sea un instante en mi desarrollo académico.

Danni Brito

De todo corazón, a mi familia y profesores, me han ayudado a cumplir con este gran sueño. También, expreso mi respeto y admiración por su paciencia.

Bryan Lema

ÍNDICE DE CONTENIDO

RESUMEN

A lo largo de la historia, naturistas, científicos y biólogos han publicado información morfológica de plantas y animales en medios impresos, como revistas y libros. Sin embargo, solo una pequeña fracción de esa información está disponible y correctamente estructurada para llevar a cabo análisis filogenéticos. Es necesario desarrollar herramientas de software para extraer, integrar y publicar esta información.

En este trabajo, se desarrolló una metodología para obtener descripciones de especies en documentos separados de un libro sobre los insectos vectores de la enfermedad de Chagas (*Triatominae kissing bugs*), y también para construir el árbol filogenético de estos insectos. Se han obtenido 131 documentos de diferentes especies, con sus características y valores. Estos datos obtenidos se utilizaron para inferir la filogenia de este importante grupo de insectos.

PALABRAS CLAVE: Extracción de información, Bioinformática, Enfermedad de Chagas, Filogenética, Clusterización.

ABSTRACT

Throughout history, naturists, scientists and biologists have published morphological information of plants and animals in printed media, such as journals and books. However, only a small fraction of that information is available and properly structured to conduct further phylogenetic analyzes. It is necessary to develop software tools to extract, integrate and publish this information.

In this work, a methodology was developed to obtain descriptions of species in separate documents from a book on insect vectors of Chagas disease (Triatominae kissing bugs), and to build the phylogenetic tree of these insects. 131 documents of different species have been obtained, with their characteristics and values. These data obtained were used to infer the phylogeny of this important group of insects.

KEYWORDS: Information extraction, Bioinformatics, Chagas disease, Phylogenetics, Clustering.

1. INTRODUCCIÓN

La información morfológica sobre animales y plantas ha sido publicada en libros y revistas. Esta información sintetiza descripciones morfológicas y observaciones hechas por taxónomos; contiene declaraciones que describen estructuras, subestructuras, caracteres, estados, valores y relaciones entre estructuras [1]. Solamente una pequeña fracción de esta información está fácilmente disponible y estructurada para construir filogenias (árboles de la vida), una filogenia o un árbol filogenético es un diagrama de árbol bifurcado que representa relaciones evolutivas [2], en la **Figura 1** podemos observar un ejemplo de un árbol filogenético, en el que se puede diferenciar como cada especie diverge desde una raíz común. El procesamiento manual de la información morfológica para obtener las matrices de caracteres necesarias para realizar el análisis filogenético es una tarea compleja.

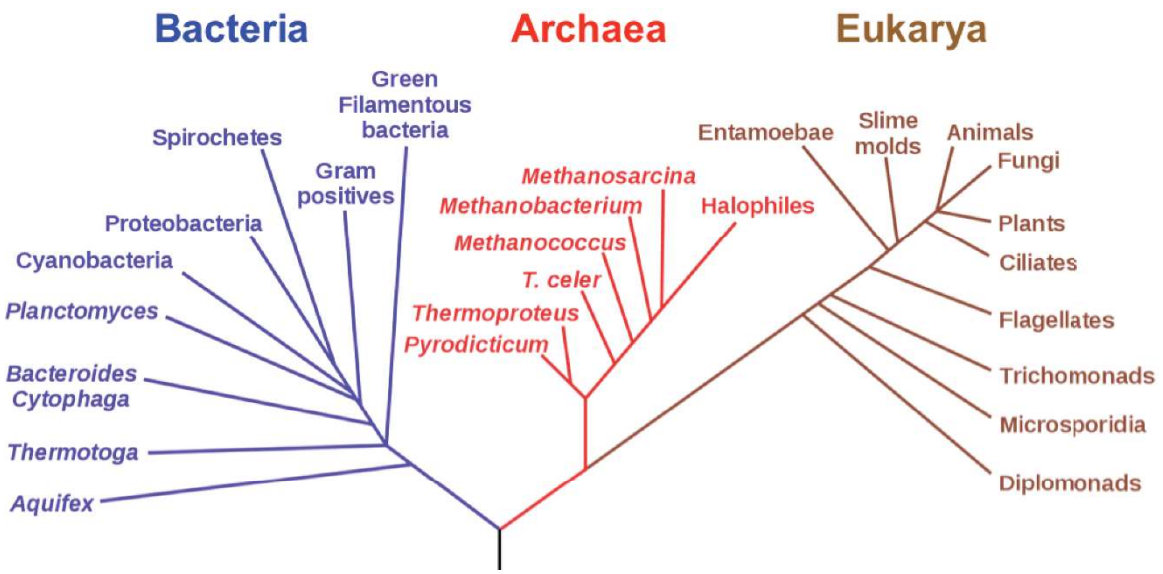


Figura 1. Ejemplo de árbol filogenético

La enfermedad de Chagas es causada por el parásito *Trypanosoma cruzi*, y se transmite por varias especies de insectos Triatominos *(Kissing bugs)*, que son insectos que pertenecen al orden de los hemípteros conocidos como insectos verdaderos [3]. La enfermedad de Chagas es endémica en América Latina, y se estima que de 5 a 18 millones de personas están infectadas por *Trypanosoma cruzi*, lo que ocasiona más de 10,000 muertes por año, principalmente debido a insuficiencia cardíaca [4]; además, se estima que el costo total de la enfermedad es de $ 7.9 billones por año [5]. Si bien el estudio de todos los componentes de la enfermedad de Chagas (parásitos, hospedadores de mamíferos y vectores) es importante, particularmente aquellos aspectos relacionados con el control y

posibles curas de la enfermedad, varios aspectos biológicos de los vectores no se conocen bien. Uno de estos aspectos es la relación filogenética de todas las especies de insectos *Triatominae*. Las filogenias que se han publicado de este grupo (como las publicadas en [6] y [7]) están incompletas porque incluyen solo una fracción de las especies de este grupo, por lo que se necesita una filogenia que incluya todas las especies de *Triatominae*.

En este estudio, hemos desarrollado un proceso semiautomático basado en enfoques de minería de texto para:

a) Obtener descripciones de especies en documentos separados de un texto base que detalla las características morfológicas de los insectos *Triatominae*, los vectores de la enfermedad de Chagas [3].
b) Extraer características morfológicas de esos vectores y con ellas inferir la filogenia de los insectos.

La separación de las descripciones de especies en varios documentos ayuda a utilizar herramientas de aprendizaje automático para extraer y comparar los caracteres de diferentes especies. Los datos se utilizarán para inferir las filogenias, incluidas todas las especies descritas de este importante grupo de vectores de enfermedades.

1.1. Pregunta de Investigación

Para el presente trabajo se define la siguiente pregunta de investigación:

¿Es posible generar un árbol filogenético a partir de información morfológica hallada en fuentes bibliográficas?

1.2. Objetivo General

Construir un árbol filogenético a partir de las características morfológicas de los insectos vectores de la enfermedad de Chagas obtenidas utilizando minería de texto.

1.3. Objetivos Específicos

a) Analizar, combinar e implementar procesos y técnicas de extracción de información aplicables a este caso de estudio.
b) Desarrollar software que extraiga y codifique la información de los diferentes insectos.
c) Procesar la información codificada para la elaborar un árbol filogenético.

1.4. Alcance

El presente proyecto de investigación tiene como base fundamental de estudio el texto de Lent y Wygodzinski [3], considerado como la base del estudio de los Triatominos, por lo cual el árbol filogenético o "árbol de vida" generado en este trabajo únicamente toma en cuenta las 131 especies descritas por los autores en su texto. Se implementan técnicas y procesos de extracción de la información según la metodología CRISP, ya que brinda una estructura útil para obtener resultados ágiles y óptimos en la minería de datos.

1.5. Marco Teórico

Esta investigación está relacionada con las áreas de extracción de información y minería de texto, y tiene como objetivo extraer y codificar, de forma semiautomática, los caracteres morfológicos de las especies de insectos descritos en inglés, utilizando la información de las mismas descripciones.

La minería de texto es el descubrimiento de información nueva mediante la extracción automática en diferentes recursos escritos. A veces está precedida por un preprocesamiento de la fuente, necesario para corregir posibles deficiencias en los datos debido a errores de la fuente, o para preparar los datos para procesos adicionales en las etapas de clasificación o codificación [8].

En los últimos años se ha realizado una importante cantidad de investigaciones sobre la extracción de información taxonómica de la literatura; algunas de ellas se han orientado a estructurar el contenido completo de la descripción morfológica de la especie [9]. Mora [1] presenta un algoritmo basado en técnicas de lingüística computacional para extraer información estructurada de descripciones morfológicas escritas en español; utiliza un análisis semántico, ontologías y un repositorio de conocimiento adquirido de las mismas descripciones, cada objeto extraído está asociado con atributos. Balhoff et al [10] concluyen que la anotación de datos fenotípicos utilizando ontologías e identificadores taxonómicos únicos a nivel mundial permitirá a los biólogos integrar datos fenotípicos de diferentes organismos y estudios, simplificando el trabajo en una morfología sistemática y comparativa.

Deans et al [11] propusieron la aplicación de anotaciones ontológicas a las descripciones taxonómicas, para aplicar técnicas de integración de datos, búsqueda y razonamiento automatizado a estos datos, aumentando el valor del trabajo taxonómico y promoviendo su reutilización.

Balhoff [11] presenta una revisión de la fauna de avispas (*Hymenoptera: Evaniidae*) de Nueva Caledonia utilizando un nuevo modelo para la descripción de especies. Las matrices descriptivas, los datos de muestras y la nomenclatura taxonómica se reúnen en una aplicación web unificada, luego se exportan para el tratamiento taxonómico tradicional y las declaraciones semánticas utilizando el OWL (*Web Ontology Language*). Zhou [12] se centra en el desarrollo de la ontología de la morfología de insectos y taxonomía para proporcionar los recursos web de la red "comprensible por la máquina". Thien Huu Nguyen [13] propone utilizar el aprendizaje profundo para la extracción de entidades y relaciones del texto. Estas técnicas podrían ser utilizadas para extraer estructuras de insectos. Pannavat Terdchanakul [14] propone un modelo de clasificación de informes de error con N-gram IDF (Frecuencia de documentos inversa), una extensión teórica de IDF para el manejo de palabras y frases de cualquier longitud. N-gram IDF nos permite extraer términos clave de textos de cualquier longitud, estos términos clave se pueden usar como características para clasificar los informes de errores.

Teniendo en cuenta los datos de entrada necesarios para la generación del árbol filogenético, hemos utilizado el enfoque IDF de N-gram para identificar los valores de los caracteres y luego usar técnicas de agrupamiento para codificar los caracteres similares de diferentes especies.

1.5.1. Cross-industry standard process for data mining (CRISP-DM)

El proceso estándar intersectorial para la minería de datos, conocido como CRISP-DM [15], es un modelo de proceso estándar abierto que describe los enfoques más comunes utilizados por los expertos en minería de datos. Es el modelo analítico más utilizado [16].

CRISP-DM divide el proceso de minería de datos en seis fases principales [17]. La secuencia de las fases no es estricta y se mueve hacia adelante y hacia atrás entre las diferentes fases, siempre que sea necesario. Las flechas en el diagrama de proceso, descrito en la **Figura 2**, indican las dependencias más importantes y frecuentes entre las fases. El círculo exterior en el diagrama simboliza la naturaleza cíclica de la minería de datos en sí. Un proceso de minería de datos continúa después de que se haya implementado una solución. Las lecciones aprendidas durante el proceso pueden desencadenar nuevas preguntas comerciales, a menudo más enfocadas, y los procesos subsiguientes de extracción de datos se beneficiarán de las experiencias de los anteriores.

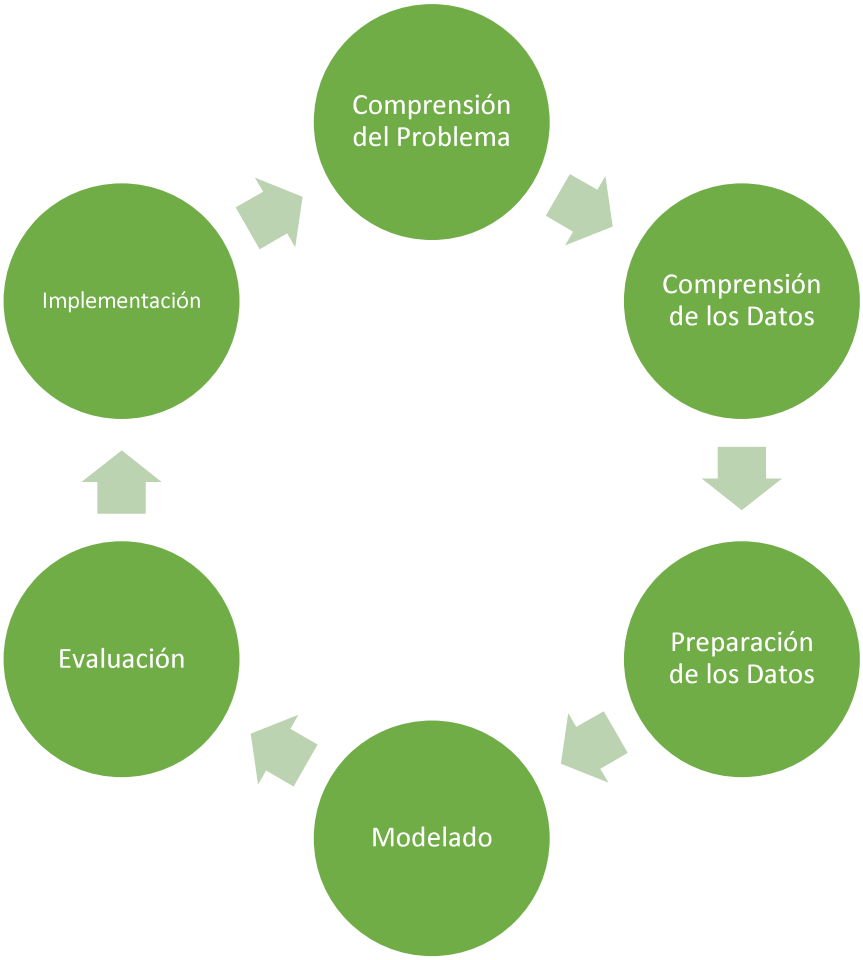


Figura 2. Diagrama del proceso CRISP-DM

La primera etapa del proceso CRISP-DM es comprender lo que se desea obtener; su objetivo es descubrir factores importantes que podrían influir en el resultado del proyecto. Ignorar este paso puede significar que se pone mucho esfuerzo en producir respuestas correctas a las preguntas incorrectas.

La segunda etapa del proceso CRISP-DM requiere que se recolecten todos los datos e información disponible en el desarrollo del proyecto e inventariar estos datos.

En la tercera etapa se seleccionan los datos a utilizar para el análisis. Los criterios para tomar esta decisión incluyen, entre otros, la relevancia de los datos para sus objetivos, la calidad de los datos y también las limitaciones técnicas, como los límites en el volumen o los tipos de los datos. En esta etapa los datos son seleccionados y, previo a la extracción, se les realiza una limpieza.

En la cuarta etapa se decide la técnica de extracción de información y se la modela para que se oriente con los objetivos definidos de nuestro problema.

En la quinta y sexta etapa se evalúan los resultados preliminares y posteriormente con un análisis de estos resultados se implementa el modelo sobre los datos.

1.5.2. Lenguaje R

R es un lenguaje y entorno para computación estadística y gráficos [18]. Se trata de un proyecto GNU[1] que es similar al lenguaje y entorno S[2], su software se distribuye en forma de código fuente con licencia GNU GPL. Está disponible en una amplia variedad de plataformas UNIX, Windows y MacOS. Este lenguaje de programación proporciona técnicas estadísticas y técnicas gráficas, útiles para la investigación en metodología estadística [18].

1.5.3. N-Gram Inverse Document Frecuency

Es una extensión teórica de IDF propuesta para el manejo de palabras y frases de cualquier longitud. Al comparar pesos entre *n-gramas*, en el que *n* puede tomar cualquier valor mayor a uno, se puede determinar *n-gramas* dominantes para superponer y extraer términos clave de cualquier longitud de los textos sin utilizar ninguna técnica clásica [19].

1.5.4. Phylogenetic Analysis Using Parsimony (PAUP*)

Análisis Filogenético Usando Parsimonia es un programa computacional de filogenética para inferir árboles evolutivos (filogenias o árboles de vida), escrito por David L. Swofford [20].

Originalmente, como su nombre lo indica, PAUP solo implementó parsimonia para la inferencia de árboles; pero, a partir de la versión 4.0 (cuando el programa se conoció como PAUP*), también admite la utilización de los métodos que implican matrices de distancia o probabilidad.

Debido a que la influencia del análisis computacional de datos moleculares, morfológicos y / o de comportamiento para inferir relaciones filogenéticas, se ha expandido mucho más allá de su papel central en la biología evolutiva, que abarca aplicaciones en áreas como la biología de la conservación, la ecología y los estudios forenses, han hecho que PAUP*, junto con el programa MacClade [21] (con el que comparte el formato de datos NEXUS [22]), sean los software filogenéticos elegidos por muchos filogenetistas y aficionados [23].

La versión 3.0 se ejecutó en computadoras Macintosh y admitía por primera vez una interfaz gráfica. La versión 4.0 agregó soporte para las plataformas Windows y Unix, y representa una gran mejora con respecto a sus predecesores, la velocidad del algoritmo

[1] GNU es un sistema operativo de software libre.
[2] S es un lenguaje de programación estadístico.

de bifurcación y límite se ha mejorado y se han agregado una serie de nuevas características, desde subárboles de “acuerdo” hasta pruebas de combinabilidad y permutación de datos.

El programa, a partir de su versión 4.0, muestra el título autorreferencial de PAUP* (Phylogenetic Analysis Using PAUP*) debido a que ya no solo utiliza parsimonia.

1.5.5. FigTree

FigTree [24] es un paquete de software, diseñado por Andrew Rambaut, como un visor gráfico de árboles filogenéticos y como un programa para exportar dichos árboles en formato pdf, png y jpg.

1.5.6. K-Means

El agrupamiento por clústeres con *k-means* (*K-Means Clustering*) es un método de cuantificación vectorial, para el análisis de clústeres en la minería de datos. *K-Means clustering* tiene como objetivo dividir n observaciones en k clústeres en los que cada observación pertenece al clúster con la media más cercana. En la **Figura 3** podemos observar como ejemplo que las observaciones han sido divididas en 3 clústeres.

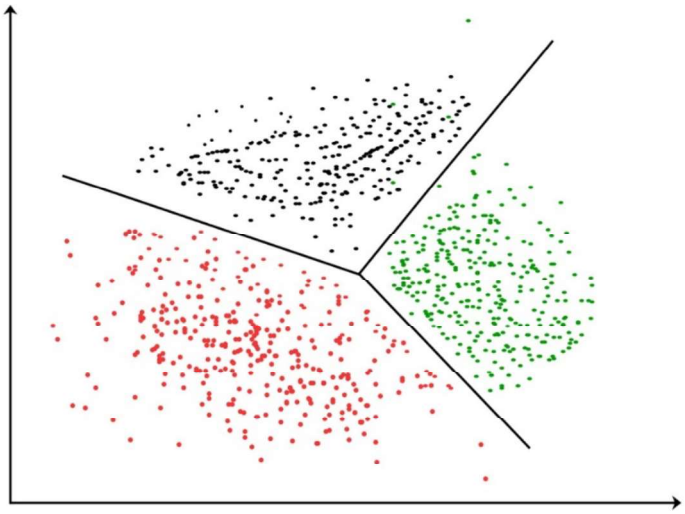


Figura 3. Representación gráfica de clústeres en un espacio bidimensional

El algoritmo utiliza una técnica de refinamiento iterativo. Dado un conjunto inicial de k-means: m1, …, mk, el algoritmo procede alternando entre dos pasos [25]:

a) Paso de asignación: Asigna cada observación al grupo cuya media tenga la mínima distancia euclidiana al cuadrado, esta es intuitivamente la media "más cercana".
b) Paso de actualización: Calcula los nuevos medios (centroides) de las observaciones en los nuevos grupos.

El algoritmo ha convergido cuando las asignaciones ya no cambian. El algoritmo no garantiza encontrar las clasificaciones óptimas [26].

1.5.7. Organización del presente documento

A continuación de la introducción se organiza el documento con la siguiente información: pregunta de investigación, el objetivo general, los objetivos específicos, el alcance y el marco teórico. Posteriormente, en el capítulo 2 se presenta la metodología, en donde se explica lo realizado en cada una de las fases para la obtención del árbol filogenético. En el capítulo 3, se muestran los resultados y discusión. Por último, en el capítulo 4, se presentan las conclusiones del presente proyecto de investigación.

2. METODOLOGÍA

La presente investigación tuvo un enfoque exploratorio, por el que se determinaron las características clave del problema para realizar la búsqueda, análisis y uso de herramientas y lenguajes de programación que mejor se adapten a las necesidades técnicas encontradas. Este proyecto de investigación se ha realizado utilizando el proceso Cross-industry standard process for data mining (CRISP-DM) adaptando parte de su metodología y combinándola con las metodologías de minería de texto más comunes. Además, se utilizaron técnicas de extracción de información [8] en un documento existente para construir vectores de características morfológicas y obtener así un árbol filogenético con un software especializado [24].

A continuación, se describen las fases del CRISP-DM (**Figura 2**) y cómo fueron estas utilizadas en el desarrollo del presente proyecto de investigación. Es importante mencionar que la metodología, permite moverse hacia adelante o atrás entre las distintas etapas, aunque en el presente proyecto no fue necesario retroceder entre etapas, ya que las etapas fueron bien desarrolladas, con los resultados esperados.

2.1. Comprensión del Problema

En la primera etapa se comprende lo que desea lograr en el proyecto. El objetivo de esta etapa del proceso es descubrir factores importantes que podrían influir en el resultado del proyecto. Esta etapa es importante ya que define los objetivos del proyecto.

Para el presente proyecto, esta etapa es considerada en la búsqueda de la fuente bibliográfica en formato digital, de preferencia archivos pdf (Portable Document Format), en bibliotecas académicas virtuales. Tomando en cuenta que debe contener información fiable y precisa para ser extraída y procesada; y con esto cumplir el objetivo general del proyecto. También, esta etapa es considerada, durante el análisis de los posibles lenguajes de programación a utilizar, para la extracción y procesamiento de la información, ya que es importante tener en cuenta que no todos los lenguajes de programación cuentan con las librerías necesarias para el procesamiento de grandes cantidades de datos.

2.2. Comprensión de los Datos

Esta etapa requiere que se adquieran todos los datos y recursos del proyecto, si estos se adquieren de múltiples fuentes, se debe considerar cómo y cuándo se los va a integrar.

Para el presente proyecto, esta etapa es fundamental, ya que es importante obtener y analizar la fuente bibliográfica, con las características descritas en el paso anterior. Con el análisis de la fuente bibliográfica, se logran asimilar los términos biológicos necesarios para

realizar el análisis y extracción de los datos necesarios para el procesamiento y construcción del árbol filogenético. También, se considera la información relevante, teniendo como resultado que esta es aquella que describe la morfología de los insectos en estudio.

2.3. Preparación de los Datos

En esta etapa se decide los datos que van a utilizar para el análisis. Los criterios que se pueden usar para tomar esta decisión incluyen la relevancia de los datos para sus objetivos de minería de datos, la calidad de los datos y también las limitaciones técnicas, como los límites en el volumen de datos o los tipos de datos.

Es importante tomar en cuenta que, en esta etapa, se analiza la calidad de información que brinda la fuente bibliográfica, el formato en el que se encuentra y posibles errores de lectura, durante la extracción de la información, debido al formato y la librería utilizada para esta tarea. El resultado de esta etapa es la obtención de las características morfológicas de los insectos, a través del preprocesamiento de la información extraída.

Una vez obtenidas las características morfológicas de los insectos, siguiendo el proceso especificado en el paso anterior, se logra agrupar y codificar la información a través de técnicas de *clustering*. El resultado de esta etapa es la obtención de una matriz codificada de datos que servirá como entrada para la herramienta que ha sido analizada y seleccionada, conforme se especifica en 2.1.

Las etapas realizadas fueron:

1. Preprocesamiento: Para obtener un texto limpio y formateado.
2. Análsis en R: Para obtener las características morfológicas.
3. Extracción de características: Para obtener los documentos de las descripciones de las características.
4. Clusterización: Para obtener la matriz codificada.

2.4. Modelado

En esta etapa se seleccionó la herramienta Morphobank la cual utiliza un algoritmo de parsimonia para la generación del árbol filogenético como se describió en la sección 1.5.4; se seleccionó esta opción con ayuda de un experto en el dominio del problema.

2.5. Implementación

CRISP-DM, define la sexta etapa del proceso, de la siguiente manera: la etapa de implementación, se tomarán los resultados de la evaluación, en caso de haberlos obtenido, y se determinará una estrategia para su implementación.

Para el presente proyecto, esta etapa es considerada en la construcción del árbol filogenético, utilizando la herramienta de software analizada y seleccionada en 2.1. Para la construcción del árbol filogenético, es necesario utilizar como entrada la matriz de datos codificados, obtenida en 2.4 y se obtiene, como resultado final, el árbol filogenético de los insectos clasificados por sus características morfológicas.

2.6. Evaluación

CRISP-DM, define la quinta etapa del proceso, de la siguiente manera: durante esta etapa, se evaluará el grado en que el modelo cumple con sus objetivos comerciales y buscará determinar si hay alguna razón por la cual este modelo es deficiente. La fase de evaluación también implica evaluar cualquier otro resultado de minería de datos que se haya generado. Los resultados de la minería de datos involucran modelos que están relacionados con los objetivos originales y todos los otros hallazgos que no necesariamente relacionados, pero que también pueden revelar desafíos, información o sugerencias adicionales para futuras direcciones.

En el presente proyecto, esta etapa es considerada en la evaluación de los resultados, es decir, comparar la efectividad de los datos, obtenidos después del procesamiento y codificación, con otros datos con similar procesamiento, pero con diferentes fuentes de información y técnicas de extracción. Esta etapa está considerada como un trabajo futuro a realizar, ya que no se encuentra dentro del alcance del proyecto.

3. RESULTADOS Y DISCUSIÓN

Una vez culminada las etapas de comprensión del problema y comprensión de los datos, se procedió a la etapa de preparación de datos definiendo las siguientes fases:

a) Preprocesamiento. Se procesó el texto de la bibliografía base, seleccionada en 2.1, para obtener una lista de insectos con sus respectivas descripciones y para crear, por cada uno de ellos, un documento que contenga su descripción.
b) Análisis en R. Se realizó un procesamiento de todas las descripciones de los insectos para obtener un conjunto de N-Gramas, con su frecuencia en cada documento.
c) Extracción de características. Los documentos generados en la fase anterior se utilizaron para buscar las características morfológicas de cada insecto basándose en frecuencia de N-Gramas; una vez obtenida esta lista de características morfológicas, se procede a la extracción de dichas características en documentos separados.
d) Clusterización. Al ser un aprendizaje de máquina no supervisado, es necesario asociar los datos obtenidos, utilizando los documentos de las características extraídas. Se analizaron los datos usando técnicas de aprendizaje automático, a través de la comparación de las cadenas de las características morfológicas de cada insecto, se obtuvieron grupos de datos y estos fueron codificados y asociados dentro de clústeres y así obtener una matriz codificada de características.

Para extraer las características morfológicas de un libro y codificarlas, algunas partes de la extracción de características se desarrollaron utilizando algoritmos de aprendizaje de máquina no supervisado. En la Figura 4 se muestra una vista general del proceso seguido para generar el árbol filogenético.

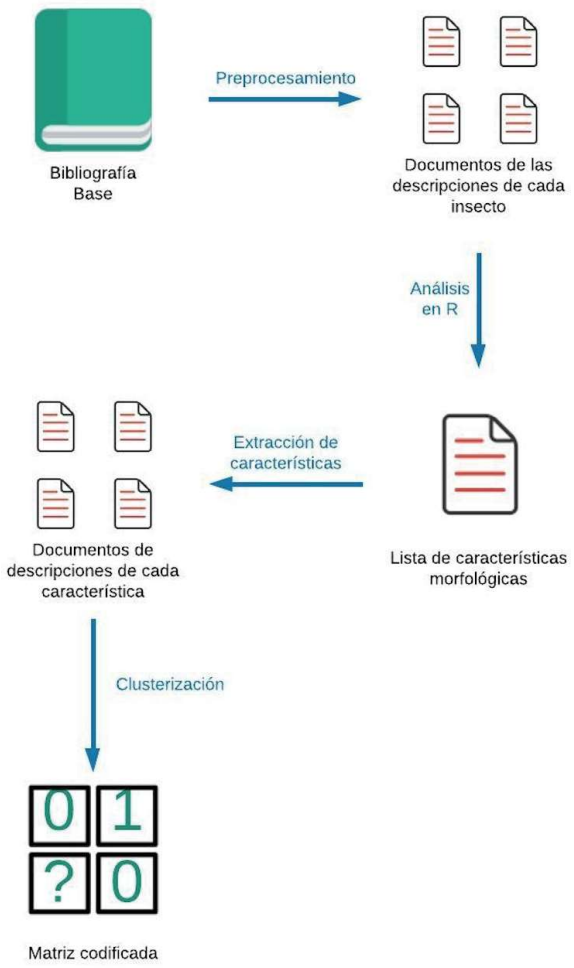


Figura 4. Vista general de la preparación de datos

3.1. Preprocesamiento

Debido a que el texto presenta una redacción en lenguaje natural y este no está estructurado, se tuvo que realizar un proceso de preparación de datos que permita obtener un texto limpio, plano y fácil de manipular, para ello se deben seguir una serie de pasos (manuales y automáticos) para modificar el archivo en formato Portable Document Format (.pdf) y su contenido. Para lograr esto, se utilizó el lenguaje de programación Python en conjunto con la librería FITZ.

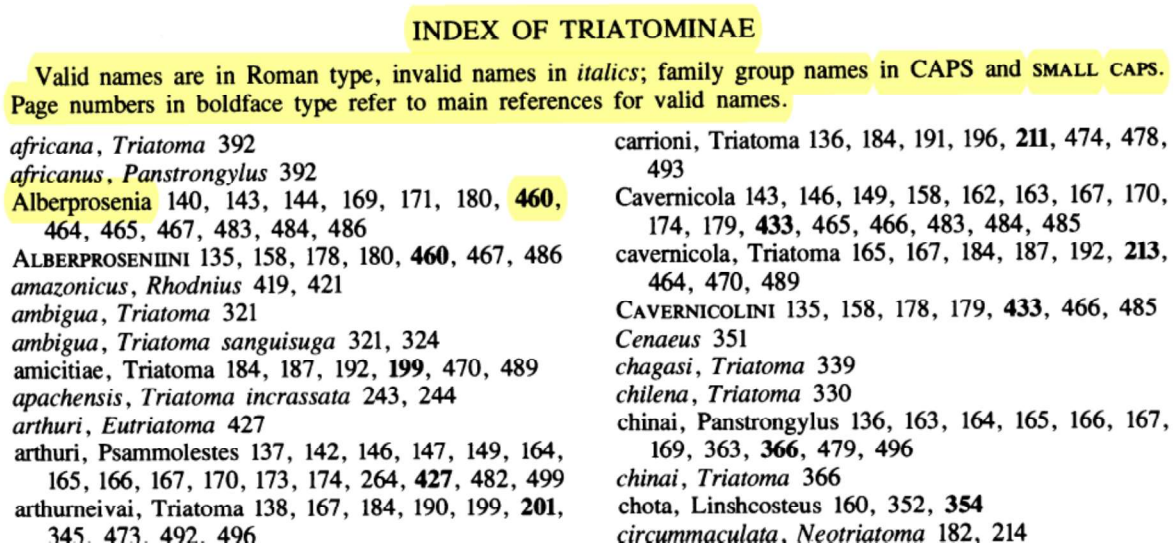
En la **Tabla 1** se muestra un ejemplo del lenguaje natural empleado por los autores en la descripción de la misma característica morfológica, “Abdomen”, en dos especies de triatominos distintos.

Tabla 1. Comparación de la característica morfológica "Abdomen" entre 2 especies

Especie	Descripción de Abdomen
Rhodnius Robustus	Abdomen in some specimens rather uniformly dark brown ventrally, but in most brown speckled with yellow; longitudinal percurrent dark line along center, widened slightly at middle of each segment.
Cavernicola Barber	Abdomen convex below.

3.1.1. Extracción del Índice

Para extraer el texto que describe detalladamente a cada uno de los insectos, se realiza un análisis del índice contenido en la fuente bibliográfica. En la Figura 5 se muestra en un fragmento del índice contenido en la fuente bibliográfica. A partir de este análisis, se obtiene un archivo de texto que contiene un índice, en el que se clasifican las especies que se encuentran descritas detalladamente. Cada especie clasificada ocupa una línea, en el archivo de texto, en el que se especifica su nombre y la página donde se ubica esta descripción en la fuente bibliográfica.



INDEX OF TRIATOMINAE

Valid names are in Roman type, invalid names in *italics*; family group names in CAPS and SMALL CAPS. Page numbers in boldface type refer to main references for valid names.

africana, *Triatoma* 392
africanus, *Panstrongylus* 392
Alberprosenia 140, 143, 144, 169, 171, 180, **460**, 464, 465, 467, 483, 484, 486
ALBERPROSENIINI 135, 158, 178, 180, **460**, 467, 486
amazonicus, *Rhodnius* 419, 421
ambigua, *Triatoma* 321
ambigua, *Triatoma sanguisuga* 321, 324
amicitiae, Triatoma 184, 187, 192, **199**, 470, 489
apachensis, *Triatoma incrassata* 243, 244
arthuri, *Eutriatoma* 427
arthuri, Psammolestes 137, 142, 146, 147, 149, 164, 165, 166, 167, 170, 173, 174, 264, **427**, 482, 499
arthurneivai, Triatoma 138, 167, 184, 190, 199, **201**, 345, 473, 492, 496
carrioni, Triatoma 136, 184, 191, 196, **211**, 474, 478, 493
Cavernicola 143, 146, 149, 158, 162, 163, 167, 170, 174, 179, **433**, 465, 466, 483, 484, 485
cavernicola, Triatoma 165, 167, 184, 187, 192, **213**, 464, 470, 489
CAVERNICOLINI 135, 158, 178, 179, **433**, 466, 485
Cenaeus 351
chagasi, *Triatoma* 339
chilena, *Triatoma* 330
chinai, Panstrongylus 136, 163, 164, 165, 166, 167, 169, 363, **366**, 479, 496
chinai, *Triatoma* 366
chota, Linshcosteus 160, 352, **354**
circummaculata, *Neotriatoma* 182, 214

Figura 5. Fragmento del Índice contenido en la fuente bibliográfica

Para obtener las especies clasificadas contenidas en este archivo de texto, partiendo del índice contenido en la fuente bibliográfica, se seleccionan los nombres de las especies y sus respectivas páginas. Para lograr la selección de los nombres de los insectos, como se advierte en la descripción del índice, se toman los nombres válidos identificados por su formato de letra en tipo romano y se descartan aquellos que se encuentran en formato itálico. Cada nombre viene acompañado con un conjunto de páginas que ubican la

descripción o alguna referencia. Se toma las páginas que se encuentran en negritas y se descarta el resto, ya que se trata de referencias, y en aquellas páginas no se cuenta con una descripción del insecto indicado.

En la **Figura 6** se presenta un fragmento del contenido del archivo de texto con las especies clasificadas. Se realiza manualmente esta clasificación para descartar los insectos que solamente se mencionen como referencia y no cuenten con una descripción que ayude en la construcción de la filogenia.

460:Alberprosenia,Martinez and Carcavallo
460:ALBERPROSENIINI,MARTINEZ AND CARCAVALLO
199:Triatoma,amicitiae
427:Psammolestes,arthuri
201:Triatoma,arthurneivai
203:Triatoma,Barberi
440:BELMINUS,STAL
438:Bolbodera,Valdes
437:BOLBODERINI,USINGER
457:Microtriatoma,borbai

Figura 6. Fragmento del contenido del archivo de texto con las especies clasificadas.

3.1.2. Construcción de la clase “insecto”

Para organizar la información extraída se construyen los objetos que representarán a cada insecto que será parte del análisis para la construcción del árbol filogenético. En la **Tabla 2** se presenta el modelo de los objetos construidos, denominados “insect”, y sus atributos.

Tabla 2. Representación del objeto "Insect" con sus atributos

Atributo	Descripción
Nombre	Almacenará el nombre de cada insecto
Página Inicial	Almacenará la página en donde inicia la descripción del insecto según el índice extraído manualmente
Página Final	Almacenará la página en donde termina la descripción de cada insecto
Descripción	Almacenará la descripción del insecto

Utilizando el índice obtenido manualmente del paso anterior (**Figura 6**), cada objeto “insect” fue añadido secuencialmente en una estructura de datos tipo arreglo, para luego ser ordenado de acuerdo con la página inicial de cada insecto.

Luego, se realiza una iteración sobre cada elemento de la estructura de datos que contiene los insectos, para colocar dentro de sus atributos la página en la que termina su descripción, y esta página es simplemente la página inicial del siguiente insecto, excepto por el último, ya que no existe más insectos, al final se ubicó como página final la 464 luego de una observación manual.

En la **Tabla 3** se muestra un extracto de la lista de objetos *Insect* luego de las operaciones descritas anteriormente.

Tabla 3. Ejemplo de objetos "Insect" luego de ordenarlos y poner su página final

Nombre del Insecto	Página Inicial	Página Final
Triatomini Jeannel	180	181
Triatoma Laporte	181	199
Triatoma Amicitiae	199	201
Triatoma Arthurneivai	201	203
Triatoma Barberi	203	204

3.1.3. Eliminación de encabezados, pies de página e imágenes

Debido que la fuente bibliográfica presenta su contenido escaneado, no es posible diferenciar un encabezado o una referencia del resto de texto relevante. Durante el procesamiento del texto, se presentaron dificultades para eliminar automáticamente los encabezados, pies de página, referencias y figuras, por lo que, fue necesario eliminar manualmente cada uno de estos elementos. En la **Figura 7** se presenta un ejemplo de la eliminación de encabezados, referencias, pie de página y figuras en una página de la fuente bibliográfica.

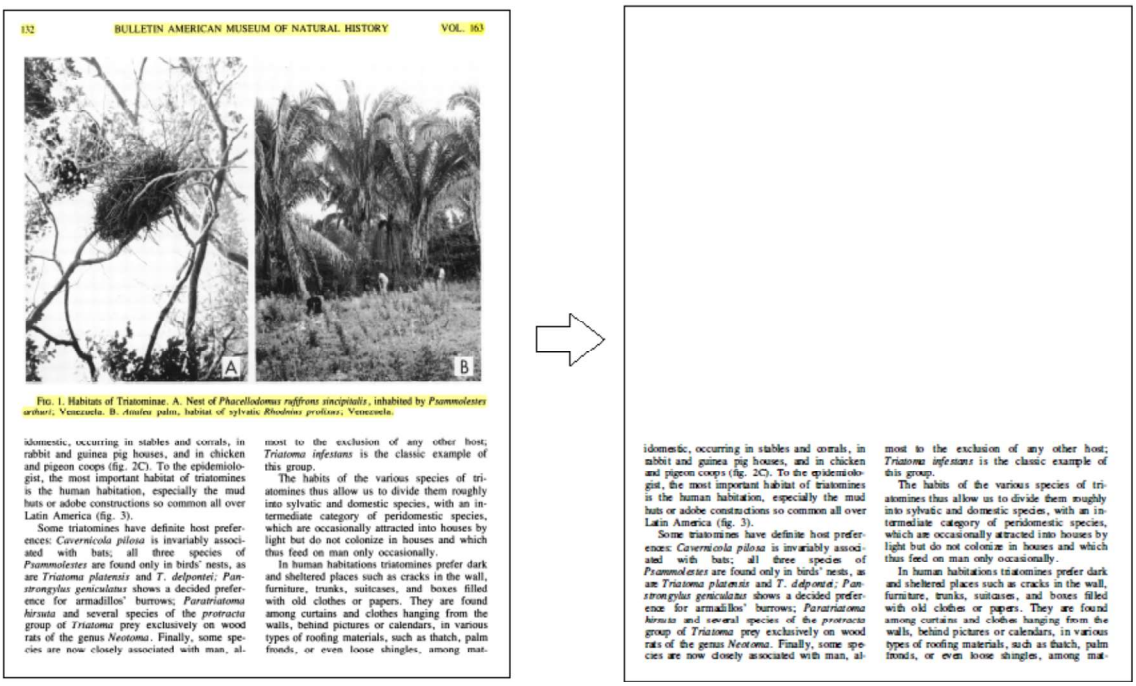


Figura 7. Ejemplo de eliminación manual de encabezados, referencias, pie de página y figuras en las páginas de la fuente bibliográfica.

3.1.4. Elección de la librería

Para la extracción del texto de la fuente bibliográfica, es necesario elegir una librería que extraiga contenido de tal manera que pueda ser identificada la división en columnas que presenta cada página. Además, el texto no debe ser desordenado ni alterado, en el momento en el que se utilice la librería para extraer el texto, para que la información resulte útil durante el análisis.

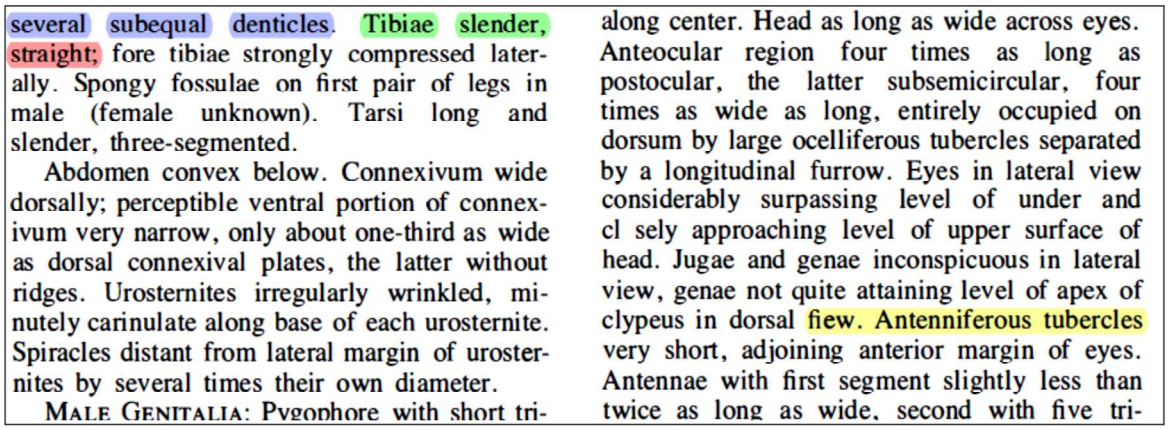
Existen varias librerías para Python que son utilizadas para extraer texto desde archivos .PDF. Se realizaron pruebas con algunas de ellas, obteniendo diferentes tipos de resultados. Ya que, el contenido no se encuentra estructurado por tratarse de imágenes escaneadas, los resultados varían dependiendo de la librería probada.

Durante las pruebas de cada una de las diferentes librerías, se obtuvieron resultados cercanos a los esperados, pero fueron descartadas por presentar alteraciones en el texto. Se detallan, a continuación, las librerías descartadas y la librería elegida para la extracción del texto de la fuente bibliográfica seleccionada en 2.1.

3.1.4.1. PdfMiner

Esta librería, desarrollada por Yusuke Shinyama, es una herramienta para extraer texto desde documentos PDF, con características particulares que ayudan a localizar un determinado fragmento de texto o convertir los archivos PDF en otros formatos, por ejemplo, HTML.

Esta librería realiza la extracción del texto aparentemente de manera correcta. Al realizar el análisis comparativo del texto extraído con esta librería y el texto original en la fuente bibliográfica, se encontró que las palabras extraídas se encuentran léxicamente correctas, pero en el orden incorrecto, por lo que fue descartada. En la **Figura 8** se presenta un ejemplo del texto extraído con pdfminer, comparado con el texto de la fuente bibliográfica.



several subequal denticles. Tibiae slender, straight; fore tibiae strongly compressed laterally. Spongy fossulae on first pair of legs in male (female unknown). Tarsi long and slender, three-segmented.

Abdomen convex below. Connexivum wide dorsally; perceptible ventral portion of connexivum very narrow, only about one-third as wide as dorsal connexival plates, the latter without ridges. Urosternites irregularly wrinkled, minutely carinulate along base of each urosternite. Spiracles distant from lateral margin of urosternites by several times their own diameter.

MALE GENITALIA: Pygophore with short tri-

along center. Head as long as wide across eyes. Anteocular region four times as long as postocular, the latter subsemicircular, four times as wide as long, entirely occupied on dorsum by large ocelliferous tubercles separated by a longitudinal furrow. Eyes in lateral view considerably surpassing level of under and cl sely approaching level of upper surface of head. Jugae and genae inconspicuous in lateral view, genae not quite attaining level of apex of clypeus in dorsal fiew. Antenniferous tubercles very short, adjoining anterior margin of eyes. Antennae with first segment slightly less than twice as long as wide, second with five tri-

Texto original

flew. Antenniferous tubercles Tibiae slender, fore tibiae strongly several subequal denticles. straight; ally. Spongy fossulae on first pair of legs in male (female unknown). Tarsi long and slender, three- segmented. compressed later- Abdomen convex below. Connexivum wide perceptible ventral portion dorsally; of connex- ivum very narrow, only about one-third as wide as dorsal connexival ridges. Urosternites irregularly nutely carinulate a Spiracles distant nites by several times their own diameter. from lateral wrinkled, long base of each urosternite. margin of uroster- the latter without plates, mi- MALE GENITALIA: Pygophore with short tri- Articulatory Basal plate struts very short, apparently not examined in apically. angular platelike process apparatus long and slender, detail. fused into an upper platelike nated structure, structures. Dorsal but faint; vesica and lateral endosoma not observed. and 1 + 1 lower subcircular

Texto extraído con pdfminer

Figura 8. Ejemplo de un fragmento de texto, extraído con pdfminer, comparado con el texto original de la fuente bibliográfica.

3.1.4.2. PyPDF2

Esta librería, desarrollada por Phaseit Inc, es una herramienta para extraer texto desde documentos PDF con características particulares que ayudan dividir un documento, unir diferentes documentos o también encriptar o desencriptar documentos.

Esta librería realiza la extracción del texto, de tal manera que este se encuentra léxicamente correcto, en el orden adecuado y respeta la división de columnas que presenta cada página. Sin embargo, esta librería fue descartada, ya que el texto se presenta de manera continua, es decir, no cuenta con el espaciado entre palabras. Tampoco se distingue la separación entre párrafos ni la división de páginas. En la **Figura 9** se presenta un ejemplo del análisis comparativo entre el texto extraído con PyPdf2 y el texto de la fuente bibliográfica.

ABSTRACT

The reduviid subfamily Triatominae contains numerous species which are actual or potential vectors of Chagas' disease, caused by *Trypanosoma cruzi*. All Triatominae require bloodmeals for their complete development. Most species feed on small terrestrial or arboreal mammals, especially didelphids, edentates, and rodents; others are associated with bats, and some are found with birds. Several triatomines have colonized human habitations, where they transmit Chagas' disease to man and his domestic animals. The epidemiologically most important species are the wide-ranging *Triatoma infestans*, *T. dimidiata*, and *Rhodnius prolixus*. The following are also active vectors of Chagas' disease, although more localized: *Triatoma barberi*, *brasiliensis*, *car-*

level. A newly discovered character of first instar nymphs is the absence or presence of a group of elongate sensory hairs on the metatarsus. Nymphs of all instars of *Microtriatoma* and *Parabelminus* have three pairs of spongy fossulae as do the adults, a condition unique in the subfamily.

The subfamily Triatominae is restricted to the Nearctic, Neotropical, and Oriental regions, entering the Australian region marginally. *Triatoma rubrofasciata* is tropicopolitan. There are 13 genera in the Western Hemisphere, with the center of diversity in South America. The Oriental region contains one endemic genus and an apparently monophyletic species group of *Triatoma*, a genus occupying the entire range of the subfamily.

Texto original

```
ABSTRACTThereduviidsubfamilyTriatominaecontainsnu-
merousspecieswhichareactualorpotentialvectorsofChagas'disease,causedbyTrypa
nosomacruzi.AllTriatominaerequirebloodmealsfortheircom-
pletedevelopment.Mostspeciesfeedonsmallter-
restrialorarborealmammals,especiallydidelphids,edentates,androdents;othersa
reassociatedwithbats,andsomearefoundwithbirds.Severaltri-
atomineshavecolonizedhumanhabitations,wheretheytransmitChagas'diseasetomana
ndhisdomes-ticanimals.Theepidemiologicallymostimportantspeciesarethewide-
rangingTriatomainfestans,T.dimidiata,andRhodniusprolixus.Thefollowingareals
oactivevectorsofChagas'disease,althoughmorelocalized:Triatomabarberi,brasil
iensis,car-rioni,guasayana,maculata,patagonica,pallidipen-
nis,phyllosoma,pseudomaculata,rubrofasciata,sordida,andrubida,Panstrongylus
chinai,megistus,andrufotuberculatus,Rhodniusecuadoriensis,andpallescens.Asu
rveyoftheexternalstructuresoftheTri-
atominaeandtheirterminologyispresentedtofacili-
```

Texto extraído con pyPDF2

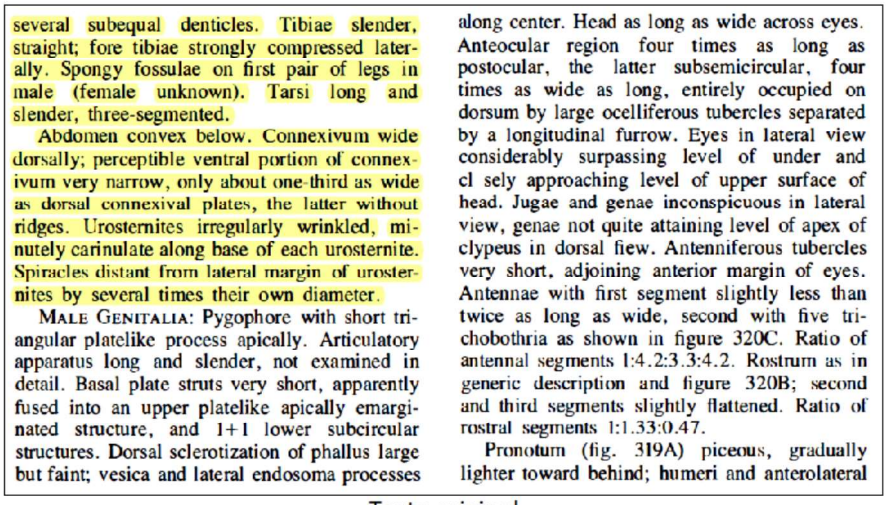
Figura 9. Ejemplo de un fragmento de texto, extraído con pypdf2, comparado con el texto original de la fuente bibliográfica.

3.1.4.3. FITZ (PyMuPdf)

Esta librería llamada PyMuPdf y formalmente conocida como FITZ, desarrollada por Ruikai Lui, es una herramienta para extraer texto desde documentos PDF con características particulares que ofrecen un gran rendimiento y su alta calidad de renderización.

Esta librería presenta resultados más satisfactorios. Ya que, extrae el texto de tal forma que, se respeta la separación en columnas de cada página, se respeta la separación entre párrafos, es posible distinguir el texto en cada página y las modificaciones de texto son mínimas, pero estas se dan por errores de lectura por el formato en el que se encuentra la fuente bibliográfica. Esta fue la librería utilizada para extraer el texto.

En la **Figura 10** se presenta un ejemplo del análisis comparativo entre el texto extraído con FITZ y el texto de la fuente bibliográfica.



several subequal denticles. Tibiae slender, straight; fore tibiae strongly compressed laterally. Spongy fossulae on first pair of legs in male (female unknown). Tarsi long and slender, three-segmented.

Abdomen convex below. Connexivum wide dorsally; perceptible ventral portion of connexivum very narrow, only about one-third as wide as dorsal connexival plates, the latter without ridges. Urosternites irregularly wrinkled, minutely carinulate along base of each urosternite. Spiracles distant from lateral margin of urosternites by several times their own diameter.

MALE GENITALIA: Pygophore with short triangular platelike process apically. Articulatory apparatus long and slender, not examined in detail. Basal plate struts very short, apparently fused into an upper platelike apically emarginated structure, and 1+1 lower subcircular structures. Dorsal sclerotization of phallus large but faint; vesica and lateral endosoma processes

along center. Head as long as wide across eyes. Anteocular region four times as long as postocular, the latter subsemicircular, four times as wide as long, entirely occupied on dorsum by large ocelliferous tubercles separated by a longitudinal furrow. Eyes in lateral view considerably surpassing level of under and cl sely approaching level of upper surface of head. Jugae and genae inconspicuous in lateral view, genae not quite attaining level of apex of clypeus in dorsal fiew. Antenniferous tubercles very short, adjoining anterior margin of eyes. Antennae with first segment slightly less than twice as long as wide, second with five trichobothria as shown in figure 320C. Ratio of antennal segments 1:4.2:3.3:4.2. Rostrum as in generic description and figure 320B; second and third segments slightly flattened. Ratio of rostral segments 1:1.33:0.47.

Pronotum (fig. 319A) piceous, gradually lighter toward behind; humeri and anterolateral

Texto original

```
several subequal denticles. Tibiae slender,
straight; fore tibiae strongly compressed later-
ally. Spongy fossulae on first pair of legs in
male (female unknown).
Tarsi long
and
slender, three-segmented.
Abdomen convex below. Connexivum wide
dorsally; perceptible ventral portion of connex-
ivum very narrow, only about one-third as wide
as dorsal connexival plates, the latter without
ridges. Urosternites irregularly wrinkled, mi-
nutely carinulate along base of each urosternite.
Spiracles distant from lateral margin of uroster-
nites by several times their own diameter.
```

Texto extraído con fitz

Figura 10. Ejemplo de un fragmento de texto, extraído con FITZ, comparado con el texto original de la fuente bibliográfica.

3.1.5. Extracción de las descripciones de los insectos

Con el archivo de texto que contiene el índice de los insectos clasificados, la librería definida, el objeto "insect" definido y la eliminación del texto innecesario, en la fuente bibliográfica, se inicia el proceso de extracción de la información de cada uno de los insectos. A partir de la elección de la librería, se extrae el contenido del archivo de texto que contiene el índice y se almacena en una estructura de datos tipo lista. La estructura de datos es ordenada, con respecto a su página inicial, para que la información se encuentre de forma ascendente. En la **Tabla 4** se presenta un extracto del contenido de la estructura de datos con los insectos extraídos del archivo de texto y ordenados según su página inicial.

Tabla 4. Extracto del contenido de la estructura de datos con los insectos extraídos del archivo de texto y ordenados según su página inicial

Insectos
180: Triatomini Jeannel
181: Triatoma Laporte
199: Triatoma Amicitiae
201: Triatoma Arthurneivai
203: Triatoma Barberi
204: Triatoma Bouvieri
206: Triatoma Brasiliensis

Para obtener la descripción de cada insecto, se itera la estructura de datos. Es importante mencionar, que cada objeto de la estructura de datos es procesado separando la página inicial, nombre de la familia y tipo de especie de cada insecto, y se almacena esta información en un objeto tipo insecto. Para obtener la descripción de la morfología de cada insecto es necesario definir el rango de páginas en las que esta se encuentra definida.

Para esto, se toma la página del objeto del insecto actual para definir la página inicial y, como la estructura de datos se encuentra ordenada de forma ascendente, según su página, se crea un objeto tipo insecto, temporal, que almacenará la información del objeto siguiente en la estructura de datos y se extrae su página, para definir la página en la que finaliza la descripción del objeto del insecto actual.

Con el rango de páginas definido, en cada objeto insecto, se extrae todo el texto contenido entre estas páginas, en la fuente bibliográfica, y es almacenado en la descripción del objeto del insecto actual, para que esta información sea procesada y limpiada, de tal forma que se obtenga solamente aquella que sea relevante para el estudio.

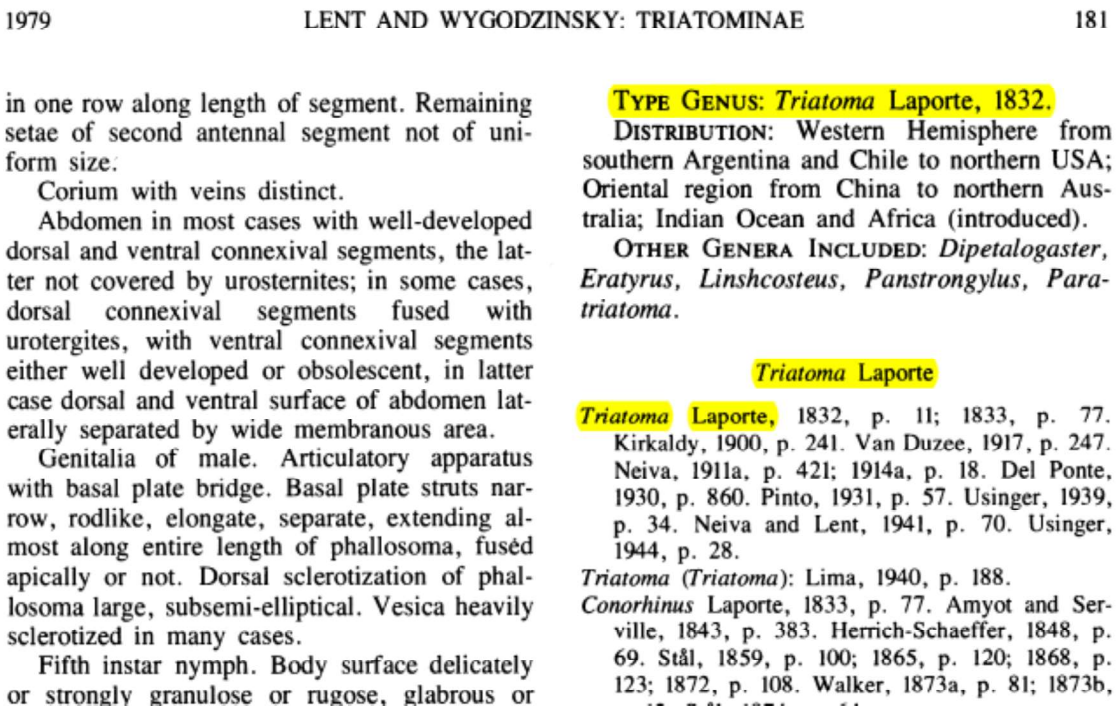
Para obtener el texto limpio, es decir, que contiene solamente la descripción de la morfología de cada insecto, es necesario procesar la descripción almacenada en cada objeto insecto. Por esta razón, el texto cruza por seis etapas: corrección de errores de lectura, eliminación de frases repetidas que causan conflicto en la delimitación del texto, delimitación del texto relevante, eliminación de referencias a figuras dentro del texto, separación entre párrafos y corrección de errores correspondientes a números decimales y puntuaciones.

Con respecto a la corrección de errores de lectura, se realiza un análisis comparativo de todo el texto original contenido en la fuente bibliográfica con el texto extraído, con el fin de hallar alteraciones. Entre las alteraciones halladas, se encontraron: confusiones en la lectura de ciertas letras, por ejemplo: las letras E y F, espacios añadidos entre letras de una palabra, números suplantados por signos de puntuación, entre otros. Los errores hallados fueron corregidos reemplazando la palabra errónea por la palabra correcta. En la **Tabla 5** se muestran algunos ejemplos de las correcciones realizadas a palabras alteradas en el proceso de extracción del texto de la fuente bibliográfica.

Tabla 5. Ejemplos de las correcciones realizadas a palabras alteradas en el proceso de extracción del texto de la fuente bibliográfica

Texto Extraído	Texto Corregido
MARTINFZ	MARTINEZ
CARCA VALLO	CARCAVALLO
Be/minus	Belminus
ST AL	STAL
mazzatti	Mazzotti
jlavida	Flavida
TRIA TO MINI	TRIATOMINI
stgments	Segments
seg ments	segments

Para la delimitación del texto relevante, se utiliza el nombre del insecto como punto de partida y finaliza en la especificación del tipo de insecto, esta es identificada por la palabra "*TYPES*" y es definida al finalizar la descripción morfológica de cada insecto en la fuente bibliográfica. Existen conflictos al tomar como punto de partida el nombre del insecto, ya que, en una página se pueden encontrar referencias que incluyan el nombre del insecto, dentro de la descripción de otro insecto. En la **Figura 11**, se presenta un fragmento de texto, extraído de la fuente bibliográfica, en el que se identifica el conflicto por causa de la referencia del insecto *Triatoma Laporte* en la descripción del tipo de género de otro insecto, localizada en la misma página en la que se describe al insecto *Triatoma Laporte*.



1979 LENT AND WYGODZINSKY: TRIATOMINAE 181

in one row along length of segment. Remaining setae of second antennal segment not of uniform size.

Corium with veins distinct.

Abdomen in most cases with well-developed dorsal and ventral connexival segments, the latter not covered by urosternites; in some cases, dorsal connexival segments fused with urotergites, with ventral connexival segments either well developed or obsolescent, in latter case dorsal and ventral surface of abdomen laterally separated by wide membranous area.

Genitalia of male. Articulatory apparatus with basal plate bridge. Basal plate struts narrow, rodlike, elongate, separate, extending almost along entire length of phallosoma, fused apically or not. Dorsal sclerotization of phallosoma large, subsemi-elliptical. Vesica heavily sclerotized in many cases.

Fifth instar nymph. Body surface delicately or strongly granulose or rugose, glabrous or

TYPE GENUS: *Triatoma* Laporte, 1832.

DISTRIBUTION: Western Hemisphere from southern Argentina and Chile to northern USA; Oriental region from China to northern Australia; Indian Ocean and Africa (introduced).

OTHER GENERA INCLUDED: *Dipetalogaster, Eratyrus, Linshcosteus, Panstrongylus, Paratriatoma*.

Triatoma Laporte

Triatoma Laporte, 1832, p. 11; 1833, p. 77. Kirkaldy, 1900, p. 241. Van Duzee, 1917, p. 247. Neiva, 1911a, p. 421; 1914a, p. 18. Del Ponte, 1930, p. 860. Pinto, 1931, p. 57. Usinger, 1939, p. 34. Neiva and Lent, 1941, p. 70. Usinger, 1944, p. 28.

Triatoma (*Triatoma*): Lima, 1940, p. 188.

Conorhinus Laporte, 1833, p. 77. Amyot and Serville, 1843, p. 383. Herrich-Schaeffer, 1848, p. 69. Stål, 1859, p. 100; 1865, p. 120; 1868, p. 123; 1872, p. 108. Walker, 1873a, p. 81; 1873b,

Figura 11. Ejemplo de un fragmento de texto que causa inconvenientes en la delimitación de texto del insecto *Triatoma Laporte*.

Se analizan todos los casos de las descripciones de insectos que presenten este conflicto y, previo al proceso de delimitación del texto, se eliminan las frases que causan los inconvenientes, dentro de la descripción de los objetos de los insectos especificados en la **Tabla 6**, lo que permite que el texto sea delimitado correctamente.

Tabla 6. Especificación de insectos y las frases que causan inconvenientes en la delimitación del texto de su descripción morfológica.

Nombre del Insecto	Frase que causa inconveniente en la delimitación del texto

Triatoma laporte	Triatoma Laporte, 1832.
Paratriatoma hirsuta	Paratriatoma hirsute Barber. DISTRIBUTION:
Alberprosenia goyovargasi	Alberprosenia goyovargasi Martinez and Carcavallo, 1977
Triatoma rubrofasciata	Triatoma rubrofasciata, which is superficially similar to rubida,
Linshcosteus costalis	Linshcosteus costalis, OBSERVATIONS:
Dipetalogaster maximus	Dipetalogaster maximus possesse other characters
Rhodnius Stal	Rhodnius Stal, 1859, Other genus included: Psammolestes Bergroth, 1911.
Cavernicola pilosa	Cavernicola pilosa Barber, 1937.
Bolbodera scabrosa	Bolbodera scabrosa Valdes;type species of

Para la eliminación de referencias a figuras, es necesario conocer las diferentes formas de referenciar una figura dentro del texto. En la **Tabla 7**, se muestran los patrones de texto que siguen las referencias a figuras que fueron eliminadas.

Tabla 7. Patrones de texto encontrados con respecto a las referencias a figuras dentro de las descripciones de los insectos.

Tipos de Referencias Internas
figs ([letra - número]; [letra - numero]; …)
fig ([letra - número]; [letra - numero]; …)
as in ([letra - número]; [letra - numero]; …)
see ([letra - número]; [letra - numero]; …)

Una vez eliminadas las referencias a figuras quedarán espacios innecesarios en el texto, así que es necesario distinguir la separación entre párrafos para descartar saltos de línea innecesarios, ya que pueden generar que la información relevante se divida. Se identifican los párrafos y se eliminan los saltos de línea restantes, obteniendo un texto ordenado.

Posteriormente, es necesario identificar los errores relacionados con los números decimales y los signos de puntuación. Se encontraron errores de lectura que no permitían la distinción de los números decimales con la separación con comas entre palabras. Para corregir este error, es necesario analizar detalladamente los errores de lectura correspondientes a los números decimales.

Durante el análisis de los números decimales, estos son algunos de los errores encontrados: espacios entre dígitos, espacios entre el separador del número decimal y sus dígitos, confusiones entre dígitos y letras, por ejemplo: 1 e l. Se utilizan expresiones regulares para corregir estos errores, eliminando los espacios adicionales e intercambiando, en los números decimales, sus separadores por puntos.

Una vez concluida la etapa de procesamiento y limpieza del texto, las descripciones obtenidas de cada insecto se almacenan en diferentes archivos de texto, estos son identificados por el nombre del insecto como muestra en el fragmento de texto de la **Figura 12.** Estos archivos de texto serán útiles para realizar la extracción de características morfológicas de los insectos en la siguiente etapa.

Triatomini Jeannel, 1919, p. 176; Usinger, 1944, p. 36. Triatominae Pinto, 1926c, p. 485. Small- to large-sized Triatominae (9,5-42,0 long). Body integument from smooth torugose, with or without small setiferous granules. Body from glabrous to heavily hirsute. Antenniferous tubercles without apical spinelike projection. Antenna inserted from very close to anterior margin of eye to center of anterior half of anteocular region of head. Genae from falling short of to slightly surpassing level of apex of clypeus, the latter widened at base, narrow apically. Ocelli situated posterolaterally behind eyes, in most cases on conspicuous elevations, behind interocular sulcus, the latter not fully developed. Head without setiferous callosities behind eyes. Second antennal segment with 4-10 trichobothria arranged in one row along length of segment. Remaining setae of second antennal segment not of uniform size. Corium with veins distinct. Abdomen in most cases with well-developed dorsal and ventral connexival segments, the latter not covered by urosternites; in some cases, dorsal connexival segments fused with urotergites, with ventral connexival segments either well developed or obsolescent, in latter case dorsal and ventral surface of abdomen laterally separated by wide membranous area. Genitalia of male. Articulatory apparatus with basal plate bridge. Basal plate struts narrow, rodlike, elongate, separate, extending almost along entire length of phallosoma, fused apically or not. Dorsal sclerotization of phallosoma large, subsemi-elliptical. Vesica heavily sclerotized in many cases. Fifth instar nymph:Body surface delicately or strongly granulose or rugose, glabrous or with short setae, if setae long, then spinulose; head not or very rarely strongly convex dorsally; genae not or only slightly projecting beyond level of apex of clypeus; eyes situated laterally at or behind middle of head; antenniferous tubercles without prominent apicolateral process; fourth antennal segment not or only slightly longer than any of the preceding; fourth or third and fourth antennal segments delicately annulate; stridulatory sulcus present in most genera (except Linshcosteus, genus with abbreviated rostrum); fore and mid femora slender or incrassate, with or without denticles; femora without trichobothria; fore tarsi much shorter than half the length of tibiae; abdomen without or with series of large tubercles dorsally along midline. First instar nymph:Head, thorax and legs uniformly dark or with light-colored annuli, never mottled; setae simple or spinulose; fourth antennal segment not longer than first, second and third combined; stridulatory sulcus present, rarely absent; mesonotum longer at midline than at sides; metanotal plates large, distance between plates varied; femora without trichobothria; hind tarsi apically with or without numerous long, delicate sensory hairs; setae of urotergites arranged in two transversal rows, very rarely three irregular rows. Eggs laid singly.

Figura 12. Fragmento del archivo de texto correspondiente al insecto *Triatomini Jeannel*.

También se genera un archivo de texto que almacenará todas las descripciones de los insectos, donde cada descripción ocupa una línea dentro del archivo. En la **Figura 13** se muestra un extracto de las primeras líneas del archivo de texto, donde el nombre del insecto es separado con su descripción con el símbolo @. Este archivo será útil para obtener la descripción de una característica específica de un insecto determinado.

Triatomini Jeannel@Triatomini Jeannel, 1919, p. 176; Usinger, 1944, p. 36. Triatominae Pinto, 1926c, p. 485. Small- to large-sized Triatominae (9,5–42,0 long). Body integument from smooth torugose, with or without small setiferous granules. Body from glabrous to heavily hirsute. Antenniferous tubercles without apical spinelike projection. Antenna inserted from very close to anterior margin of eye to center of anterior half of anteocular region of head. Genae from falling short of to slightly surpassing level of apex of clypeus, the latter widened at base, narrow apically. Ocelli situated posterolaterally behind eyes, in most cases on conspicuous elevations, behind interocular sulcus, the latter not fully developed. Head without setiferous callosities behind eyes. Second antennal segment with 4-10 trichobothria arranged in one row along length of segment. Remaining setae of second antennal segment not of uniform size. Corium with veins distinct. Abdomen in most cases with well-developed dorsal and ventral connexival segments, the latter not covered by urosternites; in some cases, dorsal connexival segments fused with urotergites, with ventral connexival segments either well developed or obsolescent, in latter case dorsal and ventral surface of abdomen laterally separated by wide membranous area. Genitalia of male. Articulatory apparatus with basal plate bridge. Basal plate struts narrow, rodlike, elongate, separate, extending almost along entire length of phallosoma, fused apically or not. Dorsal sclerotization of phallosoma large, subsemi-elliptical. Vesica heavily sclerotized in many cases. Fifth instar nymph:Body surface delicately or strongly granulose or rugose, glabrous or with short setae, if setae long, then spinulose; head not or very rarely strongly convex dorsally; genae not or only slightly projecting beyond level of apex of clypeus; eyes situated laterally at or behind middle of head; antenniferous tubercles without prominent apicolateral process; fourth antennal segment not or only slightly longer than any of the preceding; fourth or third and fourth antennal segments delicately annulate; stridulatory sulcus present in most genera (except Linshcosteus, genus with abbreviated rostrum); fore and mid femora slender or incrassate, with or without denticles; femora without trichobothria; fore tarsi much shorter than half the length of tibiae; abdomen without or with series of large tubercles dorsally along midline. First instar nymph:Head, thorax and legs uniformly dark or with light-colored annuli, never mottled; setae simple or spinulose; fourth antennal segment not longer than first, second and third combined; stridulatory sulcus present, rarely absent; mesonotum longer at midline than at sides; metanotal plates large, distance between plates varied; femora without trichobothria; hind tarsi apically with or without numerous long, delicate sensory hairs; setae of urotergites arranged in two transversal rows, very rarely three irregular rows. Eggs laid singly.

Figura 13. Extracto de la primera línea del archivo de texto que contiene todas las descripciones de los insectos.

3.1.6. Descripción de insectos

Producto del preprocesamiento, se obtienen cada una de las descripciones de los insectos del texto tratado, en la **Figura 14** tenemos un ejemplo de una descripción obtenida en esta fase.

Triatoma amicitiae Lent Figures 34-36 Triatoma amicitiae Lent, 195ld, p. 427, figs. 1-3; Monteith, 1974, p. 91.

Female (male not known). Length 15 mm; width of pronotum 4 mm, of abdomen 7 mm Overall color uniformly orange-brown, with only extreme apex of corium and entire mem brane, dark. Integument granulose on head and thorax. Setae very short and sparse. Head (figs. 34, 35, 36A, B) strongly convex dorsally between eyes, slightly less than twice as long as wide across eyes (1:0.55) and as long as pronotum. Anteocular region twice as long as postocular (1:0.45); postocular with sides slightly rounded, somewhat converging posteriorly. Clypeus widened on posterior half, elevated in lateral view. Genae slightly tapering distally, their apex rounded, extending to level of apex of clypeus. Eyes unusually small, in lateral view not quite attaining level of lower surface and remote from level of upper surface of head. Ratio width of eye to synthlipsis 1:4.2. Ocelli small. Antenniferous tubercles situated behind middle of anteocular region. First and second antennal segments dark brown; first with apex falling short of level of apex of clypeus, second segment only with very short setae. Second segment four times as long as first. Rostrum (fig368) with short setae on all segments, slightly longer on apex of second and on third segment. First rostral segment attaining level of base of antenniferous tubercles, second falling slightly short of level of posterior border of head. Ratio of rostral segments 1:1.4:0.5.

Anterior lobe of pronotum (figs. 34, 35, 36A) without discal or lateral tubercles. Posterior lobe irregularly rugose-granulose. Submedian carinae evanescent before hind margin of pronotum; humeri rounded. Anterolateral angles with short, triangular, anterolaterally directed projections. Scutellum with central depression limited by distinct carinae. Posterior process almost as long as main body of scutellum, stout, subconical, its apex slightly upward turned. Hemelytra (figs. 34, 35) attaining posterior margin of seventh urotergite. Corium yellowish, lighter colored than rest of body, only its apex darkened. Membrane dark. Legs stout, with fore femur about four times as long as wide, and without denticles. Tibiae without spongy fossulae (probably present in male). Venter slightly flattened longitudinally along middle, its microsculpture consisting of mean dering or labyrinthine wrinkles; transverse striation not developed. Dorsal connexival segments (fig34) with conspicuous rugosities along outer margin. Spiracles adjacent to connexival suture. Connexivum unicolorous, not spotted. TYPE: British Museum (Natural History). DISTRIBUTION: Sri Lanka [Ceylon]. This species was described (Lent, 195ld) from a specimen supposedly collected in Australia (Wiraurla) but it was shown by Monteith (1974) that the type and only specimen was actually collected in southern Sri Lanka (Wirawala). BIOLOGY: Unknown. OBSERVATION: This is the smallest of all Old World species of Triatoma. Only a single specimen is known.

Figura 14. Descripción obtenida del preprocesamiento de la especie Triatoma Amicitae

3.2. Extracción de características morfológicas

Para la extracción semiautomática de las características morfológicas de los insectos triatominos, es necesario determinar los posibles N-gramas que puedan ser determinados como características morfológicas. Para esto, se utiliza el lenguaje de programación R que cuenta con las librerías RWeka y Text Mining que serán útiles para obtener los N-gramas.

3.2.1. Lectura de información

Para la extracción de la matriz de N-gramas, es importante definir cuál será la información que se va a procesar. Para esto, se extrae el contenido de los archivos de texto obtenidos en la etapa de extracción de descripciones y se almacena en el corpus, que será útil para la definición y obtención de los N-gramas.

3.2.2. Obtención de la matriz de N-Gramas

Para definir los N-gramas, se crea una función que recibirá como parámetro un texto y utilizará la función NGramTokenizer, proporcionada por la librería RWeka, para asociar el texto dependiendo de los parámetros de control. Para determinar los parámetros de control, se utiliza la función Weka_control y en esta se define el número máximo y el mínimo de palabras en los N-gramas. El número de palabras en un N-Grama, se realizó a partir de un análisis manual de las características morfológicas de los insectos triatominos determinando que pueden tener desde uno hasta cinco palabras.

Para obtener los N-gramas, es necesario extraer el texto sin modificaciones, por lo que, se debe procesar, deshabilitando la conversión automática de todas las letras a minúsculas. Ya que, los números tampoco son útiles en la extracción de los N-gramas, son descartados, así como los signos de puntuación y los espacios en blanco extra. Una vez realizado el procesamiento del texto, se aplica la función que obtendrá los N-gramas y se almacena en una matriz frecuencia de términos por documento, para ser exportada en un archivo CSV. En la **Tabla 8** se muestra un fragmento del archivo obtenido, en el que cada N-grama es acompañado con columnas que representan la frecuencia en cada archivo.

Tabla 8. Fragmento del archivo .csv que contiene la matriz de términos por documento obtenida en R.

N-Grama	Triatomini Jeannel.txt	Triatoma Laporte.txt	Triatoma circummaculata.txt	Rhodnius neglectus.txt
with	13	9	11	8
without	9	6	1	0
not	8	2	2	0
and	6	10	18	13
antennal	5	1	3	1
segment	5	4	7	1
setae	5	1	2	0
the	5	0	0	1
antennal segment	4	1	2	0
behind	4	2	1	0
cases	4	0	1	1
fourth	4	1	1	1

3.2.3. Obtención de características morfológicas

A partir del archivo CSV con los N-gramas y su frecuencia por documento, se determinará cuántos y cuáles de estos son considerados como características morfológicas. Se utiliza la biblioteca Pandas, en Python, para manipular los datos correspondientes a la frecuencia de cada N-grama de forma estadística, es decir, que sea posible contabilizar el total de ocurrencias de cada N-grama en todas las descripciones de la morfología de cada insecto. Y así, poder seleccionar las características, considerando la presencia de los N-gramas en la mayoría de las descripciones de los insectos.

3.2.4. Preprocesamiento de datos.

Para obtener el listado de posibles características, es importante procesar la información porque la matriz obtenida contiene N-gramas que referencian a las mismas palabras, pero contienen espacios en blanco extras. En la **Tabla 9**, se muestra un ejemplo en el cual el primer n-grama "*Triatoma Laporte"* muestra una frecuencia de solamente una aparición en el documento correspondiente al insecto *Triatoma Laporte,* mientras que en el segundo n-grama *"Triatoma Laporte"* se muestra una frecuencia de dos apariciones en el mismo

documento, esto se debe a que el segundo n-grama contiene un espacio al final y son considerados n-gramas distintos.

Tabla 9. Ejemplo de n-gramas con las mismas palabras, pero con distintas cantidades de espacios extra.

Nombre del Insecto	Triatomini Jeannel.txt	Triatomini Laporte.txt	Triatoma circummaculata.txt	Rhodnius neglectus.txt
Triatoma Laporte	0	1	0	0
Triatoma Laporte	0	2	0	0

Para esto, es necesario eliminar los espacios extra al inicio y final de los N-gramas. Se almacena el conjunto de datos, de la matriz de N-gramas obtenida, en un dataframe. Este es ordenado según la columna que especifica su nombre y a continuación se eliminan los espacio iniciales y finales, teniendo como resultado un dataframe ordenado, pero con valores repetidos.

Considerando que, los valores leídos con espacios extras pueden ser tomados en cuenta durante la extracción de algunas descripciones, y en otras no, se hace una intersección entre coincidencias de N-gramas, para obtener el conjunto total de ocurrencias en un solo N-Grama. Con esto, se obtiene una matriz en la cual cada N-Grama es único y es acompañado con su frecuencia única, en cada documento que contiene la descripción morfológica de cada insecto.

A partir de esta matriz, cada N-Grama sumará todos los valores correspondientes a su frecuencia en las descripciones de los insectos, teniendo como resultado total un listado de N-gramas acompañado de su número total de ocurrencias en todas las descripciones de los insectos. Como muestra en la **Tabla 10,** los N-gramas están acompañados del número total de ocurrencias en los documentos de las descripciones, por ejemplo, el N-Grama “Abdomen” se repite en 69 documentos y el N-Grama “Antennae” se repite en 30 documentos.

Tabla 10. Fragmento del listado de N-gramas acompañados de su frecuencia total.

N-Grama	Frecuencia
Abdomen	69
Antennae	30
Antenniferous	93
Antenniferous tubercles	92
Antenniferous tubercles situated	47
Anteocular	110
Anteocular region	109
Anterior	79
Anterior lobe	79
Anterolateral	96
Anterolateral angles	45
Anterolateral projections	44
Body	36
Clypeus	84

3.2.5. Selección de N-gramas.

Para la selección de los N-gramas que corresponden a las características morfológicas de los insectos, es necesario realizar un análisis manual de la fuente bibliográfica. Una vez realizado el análisis manual de la morfología de los insectos, se considera que, para ser una característica morfológica, es necesario que esta esté presente en al menos el 80% de descripciones.

Por lo tanto, en el estudio realizado de 132 insectos, cada N-Grama debe cumplir con una frecuencia total de al menos 25. Si no se cumple esta condición, se descarta el N-Grama como característica morfológica. En la **Tabla 11** se muestran los 48 N-gramas seleccionados que corresponden a una característica morfológica de los insectos. Se exportan los resultados en un nuevo archivo csv para realizar la extracción de las descripciones de las características morfológicas de cada insecto.

Tabla 11. Conjunto de N-gramas considerados como características morfológicas.

N-grama	Frecuencia
Abdomen	69
Antennae	30
Antenniferous tubercles	92
Anteocular region	109
Anterior lobe	79
Anterolateral angles	45
Anterolateral projections	44
Body	36
Clypeus	84
Connexivum	81
Corium	71
Eyes	106
Femora	30
First antennal segment	74
First rostral segment	45
Fore	77
Genitalia	30
Genae	93
General color	29
Head	129
Hemelytra	119
Humeral angles	67
Integument	29
Jugae	54
Legs	121
Length	110
Membrane	43
Neck	73
Ocelli	36
Overall color	84

Pilosity	33
Posterior lobe	84
Posterior process	49
Pronotum	113
Ratio of antennal segments	90
Ratio of rostral segments	96
Ratio width of eye to synthlipsis	79
Rostrum	115
Scutellum	113
Second antennal segment	27
Setae	50
Spiracles	89
Spongy fossulae	36
Submedian carinae	54
Tibiae	28
Urosternites	37
Venter	84

3.2.6. Extracción de descripciones de características por insecto

Partiendo del archivo csv que contiene el listado de características morfológicas seleccionadas, se obtienen las descripciones de las características de cada insecto. También, es necesario utilizar el archivo texto que contiene todas las descripciones de los insectos, obtenido en la etapa de extracción de descripciones (2.1).

3.2.7. Obtención de descripciones de características

Para la obtención de las descripciones de las características por insecto, es necesario almacenar en una estructura de datos, tipo lista, el archivo de texto que contiene las características morfológicas seleccionadas y en otra estructura de datos, tipo lista, el archivo de texto que contiene todas las descripciones de los insectos.

Se itera la estructura de datos que contiene las descripciones de los insectos y a partir de cada objeto se itera la estructura de datos que contiene las características morfológicas para extraer las características que estén contenidas en esta descripción al mismo tiempo que son almacenadas en un objeto tipo json y este es exportado a un archivo json. En la **Figura 15,** se muestra el resultado de unos de los objetos contenidos en el archivo json en

el que cada insecto contiene sus características y en ellas están detalladas sus especificaciones.

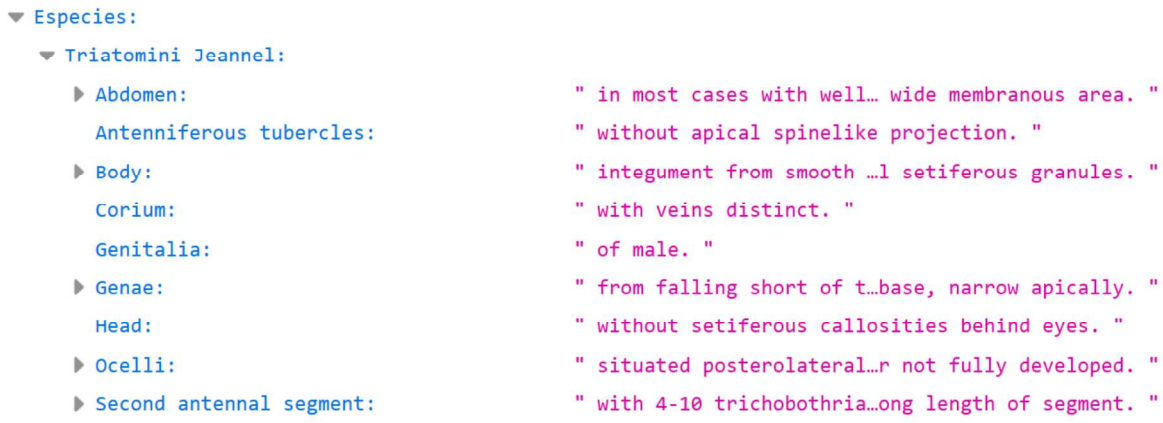


Figura 15. Ejemplo del objeto json “Especies” y la especie “Triatomini Jeannel” que contiene sus características y descripciones.

3.2.8. Obtención de archivos a partir de características

Se realiza una iteración del objeto json obtenido para crear archivos a partir de las características de cada insecto y se almacena su descripción. Durante la iteración del objeto, se creará un archivo a partir de cada característica encontrada y se almacena su descripción acompañada del nombre del insecto. En caso de que el archivo exista, se agregará el contenido de su descripción y el nombre del insecto en una nueva línea. En la **Figura 16** se muestra un fragmento de texto de las tres primeras líneas contenidas en el archivo generado a partir de los insectos que posean especificada la característica “Abdomen”.

Triatomini Jeannel@ in most cases with well-developed dorsal and ventral connexival segments, the latter not covered by urosternites; in some cases, dorsal connexival segments fused with urotergites, with ventral connexival segments either well developed or obsolescent, in latter case dorsal and ventral surface of abdomen laterally separated by wide membranous area.

Figura 16. Contenido del archivo obtenido a partir de la característica Abdomen.

3.2.9. Descripciones de características morfológicas

Como resultado de la fase de extracción de características, se obtienen las descripciones de todas las características morfológicas descritas en la fase de análisis en R. En la **Tabla 12** tenemos un ejemplo de 5 textos correspondientes a la misma característica morfológica (eyes) en 5 especies distintas, obtenidos en dicha fase.

Tabla 12. Ejemplo de 5 descripciones de la misma característica morfológica en diferentes especies

Especie	Descripción de la característica morfológica “eyes”
Triatoma amicitiae	small, lateral view quite attaining level lower surface remote level upper surface head.
Triatoma arthurneivai	lateral view surpassing level surface approaching level upper surface head.
Triatoma barberi	lateral view attaining level dorsal close level surface head.
Triatoma bouvieri	lateral view attaining level surface remote upper surface head.
Triatoma brasiliensis	lateral view approaching attaining level surface remote level upper surface head.

3.3. Codificación

En esta etapa se procesaron los archivos obtenidos de cada característica usando técnicas de inteligencia artificial, para comparar cada cadena contra todas las demás dentro de una misma característica para poder cuantificar su similitud y poder clasificarla dentro de clústeres.

3.3.1. Creación de la matriz de distancias

Para poder cuantificar las relaciones que existen entre cada cadena se utilizó el concepto de matriz de distancias para así poder representar la similitud de cada cadena en referencia a las otras cadenas dentro de una misma característica.

Cada distancia será calculada con una variación adaptada a nuestro estudio de la llamada distancia de Leveshtein. Esta nos dice que la “distancia” de palabras es el número mínimo de operaciones de inserción, eliminación y transformación requeridas para que una palabra sea idéntica a otra. En el presente trabajo este concepto fue adaptado a frases completas, tomando en cuenta operaciones sobre palabras completas en lugar de caracteres.

Este concepto de distancias de Leveshtein aplicado a frases, se lo adaptó con el uso de la teoría de conjuntos, para lo cual, definimos la “distancia” como el número de elementos resultantes de la resta entre la unión de una frase con otra y la intersección de dichas frases (diferencia simétrica); es decir, tomaremos en cuenta el conjunto de palabras que solo se repiten en una frase, no en ambas, ya que las palabras comunes en ambas frases no requieren de ninguna transformación.

En la **Figura 17** la parte coloreada representa esta diferencia simétrica, el número de términos dentro de este conjunto representa la distancia entre la frase F y la frase B.

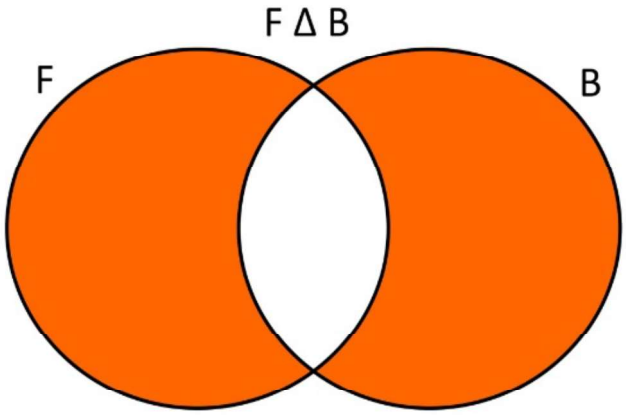


Figura 17. Diferencia simétrica de conjuntos.

Todas estas operaciones se las realiza enfrentando cada frase contra todas las demás dentro de su respectiva característica, obteniendo así una matriz simétrica, la cual es nuestra matriz de distancias.

Con la matriz de distancias generada, se procede a la ubicación de los puntos dentro de un espacio bidimensional, para ello utilizamos el Análisis de Componentes Principales o PCA (según sus siglas en inglés), el cual es una función dentro del paquete de inteligencia artificial SKLEARN, esta función utiliza la matriz de distancias y la escala a la dimensión requerida, en este caso 2, para dar un punto de referencia en el cual posicionar los puntos que representan a cada una de las frases.

En la **Figura 18** tenemos un ejemplo de la representación bidimensional interpretada por PCA de la matriz de distancias generada al analizar las distancias entre cada cadena de cada especie en la característica morfológica Corium.

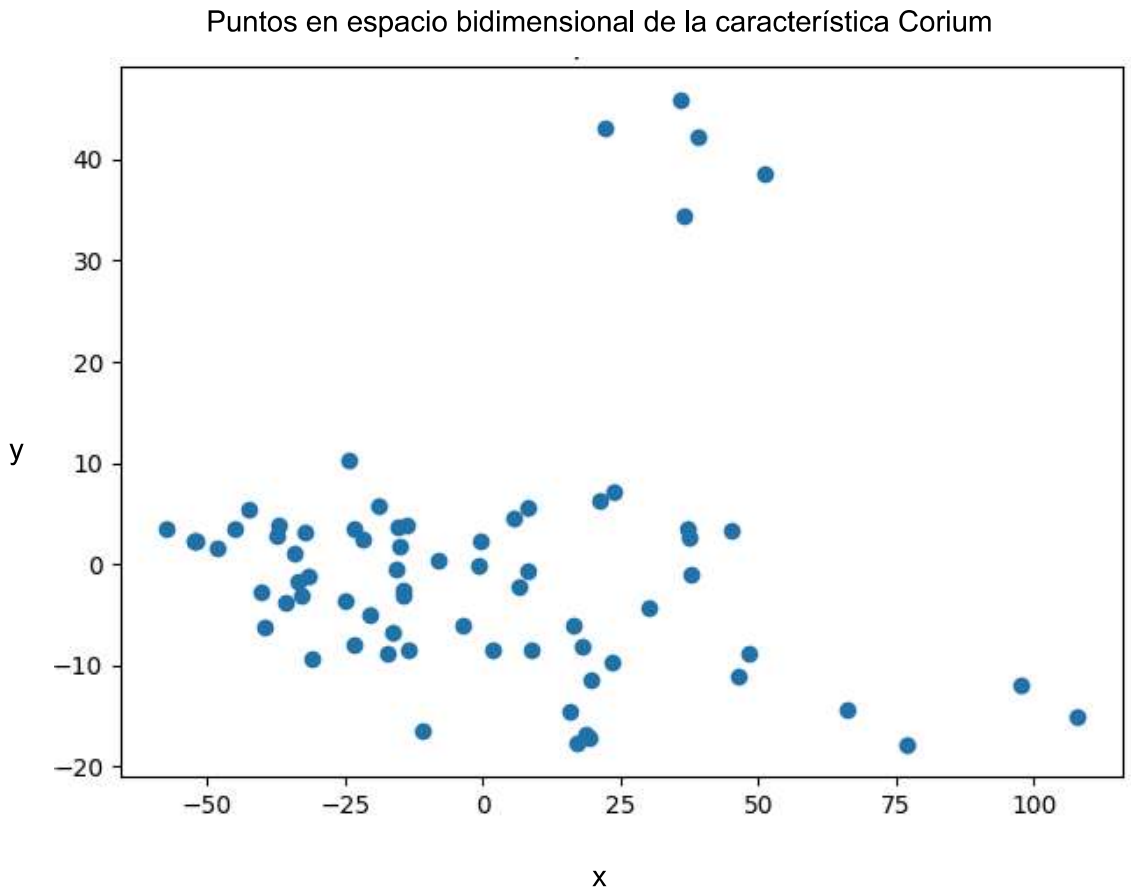


Figura 18. Representación bidimensional de la matriz de distancias en la característica morfológica Corium

3.3.2. Determinación del número óptimo de clústeres

Con la representación bidimensional de las distancias de cada frase, el siguiente paso fue la clusterización de dicho espacio bidimensional, para ello utilizaremos un algoritmo de inteligencia artificial para creación de clústeres, denominado K-Means [27], el cual va a calcular las distorsiones generadas dependiendo del número de clústeres, es decir, a diferentes números de clústeres dentro de un espacio bidimensional, diferentes distorsiones generadas, esta función (Número de Clústeres vs Distorsiones) se la conoce como la función “Codo”, la cual, como bien indica su nombre, cuando se la dibuja presenta la forma de un codo flexionado.

En la **Figura 19**, se muestra el gráfico generado de “Distorsión vs Número de Clústeres” en la característica morfológica Corium.

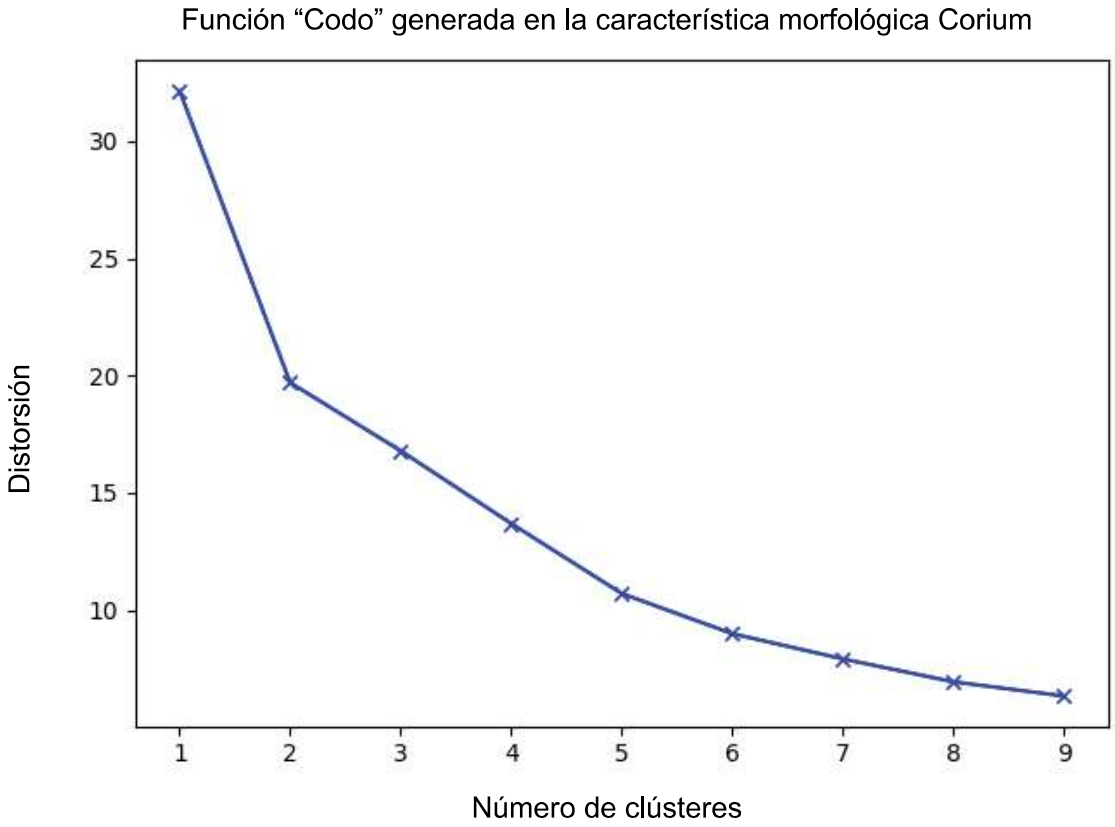


Figura 19. Función codo generada por la característica morfológica corium

El objetivo de la creación de esta función es hallar el punto donde la distorsión, relativa al punto anterior y siguiente, es la máxima, y este punto será el número óptimo de clústeres, para ello utilizamos un concepto del cálculo integral llamado la segunda derivada, la cual, según su concepto, nos sirve para hallar los puntos de máxima y mínima curvatura dentro de una función, en este caso la función codo.

Pero, debido a que nuestra función está definida por puntos, es decir, no es una función continua, debemos utilizar una aproximación a la segunda derivada de manera discreta, esta función de aproximación es la denominada, “Diferencia Central”, función la cual toma como entrada las distorsiones anteriores y siguientes de un punto para aproximar a la segunda derivada y así, con estos datos obtener el punto de inflexión del codo y así hallar el punto con mayor distorsión, el que será para nuestro caso de estudio el número óptimo de clústeres.

En la **Figura 20** se muestra la Ecuación que calcula la diferencia central de la segunda derivada.

$$\ddot{x}_t = \frac{x_{t+1} - 2x_t + x_{t-1}}{(\Delta t)^2}$$

Figura 20. Función de diferencia central de la segunda derivada

Debemos obtener el punto con mayor distorsión relativa debido a que los puntos con poca distorsión o con mucha distorsión pierden sentido, en un extremo, al máximo número de clústeres, es decir, tantos clústeres como puntos en nuestro espacio bidimensional, cada punto (que representa a una frase) tendría su propio clúster, y caso contrario, con el número mínimo de clústeres, es decir, un único clúster, todos los puntos en el espacio bidimensional pertenecerían al mismo clúster.

3.3.3. Creación de Clústeres

Una vez obtenido el número óptimo de clústeres en cada característica, procedemos a la construcción de los clústeres, los cuales servirán de referencia para su codificación, para ello utilizamos la función KMeans, la cual utiliza el número real de clústeres (el valor obtenido en el punto anterior y sumado dos debido a que debemos tomar en cuenta los indices) para obtener un modelo, el cual se compone de unos elementos llamados centroides, que no son más que la representación del centro de cada clúster en el espacio bidimensional, a estos centroides podemos hallar atados una lista de términos ordenados según su significancia dentro del clúster, es decir, términos ordenados conforme a su cercanía al centroide, y dicho de otra manera, una lista de palabras ordenadas según la importancia para definir cada clúster.

Es importante aclarar que cada uno de los clústeres no son excluyentes en cuanto a términos, es decir, que cada término se encuentra en las listas de todos los centroides pero en diferente posición, por ejemplo, supongamos que dentro de una característica X, se obtuvieron 2 clústeres, y el término Z puede encontrarse en la posición 1 del primer clúster y al mismo tiempo en la posición 3 del segundo clúster, esto quiere decir que el término Z tiene una significancia mayor para el primer clúster que para el segundo. En la **Figura 21** podemos observar visualmente este concepto de significancia.

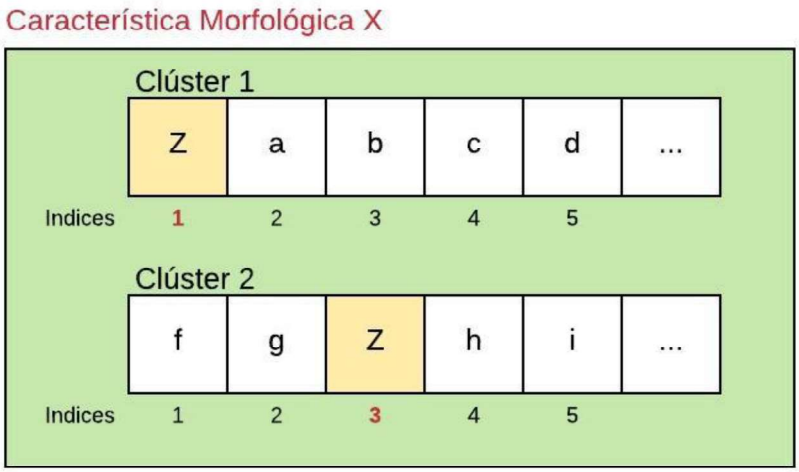


Figura 21. Representación gráfica de significancia en clústers

Por motivos de optimización, se optó por obtener solamente los diez términos con mayor significancia de cada centroide para luego almacenarlos dentro de otra lista, así obtenemos una lista que contiene cada una de las listas de los centroides con solo diez términos.

Además, creamos una nueva lista que nos servirá más adelante en la contrucción del árbol filogenético para dar una categoría a cada clúster al momento de la codificación.

3.3.4. Matrices de distancias

Dentro de la fase de clústerizacion, para poder ubicar las descripciones en un espacio bidimensional, construimos matrices de distancias, en la **Tabla 13** tenemos un ejemplo de las 10 primeras filas y columnas de la matriz de distancias para la característica morfológica Abdomen.

Tabla 13. Extracto de la matriz de distancias de la característica morfológica Abdomen

Característica Morfológica Abdomen	Jeannel	Bouvieri	Brasiliensis	Carrioni	Cavernicola	Dimidiata	Dispar	Flavida	Gerstaeckeri	Guazu
Triatomini Jeannel	0	21	23	23	21	27	23	23	23	24
Triatoma Bouvieri	21	0	12	12	10	16	12	12	14	13

Triatoma Brasiliensis	23	12	0	6	6	12	8	4	8	7
Triatoma Carrioni	23	12	6	0	4	10	6	4	6	9
Triatoma Cavernicola	21	10	6	4	0	10	2	2	8	5
Triatoma Dimidiata	27	16	12	10	10	0	12	12	12	13
Triatoma Dispar	23	12	8	6	2	12	0	4	8	7
Triatoma Flavida	23	12	4	4	2	12	4	0	8	5
Triatoma Gerstaeckeri	23	14	8	6	8	12	8	8	0	11
Triatoma Guazu	24	13	7	9	5	13	7	5	11	0

3.3.5. Asignación de caracteres a clústeres

Una vez obtenidos los clústeres, debemos asignar cada uno de los puntos (que son las frases) a cada uno de los clústeres para así asignarles un código dependiendo del número de clústeres; para ello utilizamos nuevamente la teoría de conjuntos y hacemos una intersección entre una frase con cada lista de términos y, utilizando los índices de la lista de términos (que representan su posición e importancia dentro del clúster), definimos mediante una suma el clúster al que pertenece cada frase.

Con el procedimiento para codificar cada descripción de cada característica en cada insecto, construiremos una matriz que represente en sus filas a cada insecto y en sus columnas cada característica morfológica definida.

3.3.6. Matriz de datos codificada

Como resultado final de la fase de clusterización tenemos la matriz codificada, la cual representa a que clúster pertenece cada característica morfológica de cada insecto. En la **Tabla 14** se muestra un ejemplo de las diez primeras columnas (características morfológicas) con las diez primeras filas (especies de insectos).

Tabla 14. Extracto de la matriz codificada

	Abdomen	Antennae	Antenniferous Tubercles	Anteocular Region	Anterior Lobe	Anterolateral Angles	Anterolateral Projections	Body	Connexivum	Corium
Triatomini Jeannel	1	?	?	?	?	?	?	0	?	0
Triatoma Laporte	1	?	1	?	2	?	?	1	?	?
Triatoma Amicitiae	?	?	1	1	2	0	?	?	?	1
Triatoma Arthurneivai	?	?	1	2	2	?	0	?	0	1
Triatoma Barberi	?	?	1	1	2	2	?	?	0	?
Triatoma Bouvieri	0	?	?	1	?	2	?	?	0	1
Triatoma Brasiliensis	0	?	1	2	1	?	?	?	?	1
Triatoma Breyeri	?	?	1	2	2	?	?	?	0	?
Triatoma Carrioni	0	?	1	2	1	0	1	?	0	1
Triatoma Cavernicola	2	?	1	2	?	0	?	0	?	1

3.4. Modelado e Implementación

En esta etapa se procedió a la definición de herramientas y algoritmos útiles para la construcción y evaluación del árbol filogenético.

La matriz codificada obtenida se insertó dentro de un archivo, el cual fue subido a la plataforma Morphobank [28] para generar los archivos y comandos necesarios para poder procesar los datos y finalmente visualizar el árbol filogenético.

3.4.1. Construcción del árbol filogenético

En esta etapa se procesó la matriz codificada dentro de un servicio WEB provisto por Morphobank, para posteriormente procesar los archivos generados con PAUP para obtener el árbol fologenético.

3.4.2. Creación del archivo .tnt

Debido a que el servicio para generar árboles de Morphobank utiliza los archivos con extensión .tnt (Tree analysis using new technologies) para procesarlos, se creó un archivo el cual debe representar las características morfológicas, los nombres de los insectos, así como la codificación de cada carácter según el formato de archivo deducido de un archivo descargado de la plataforma[3].

3.4.3. Procesamiento del archivo en Morphobank

Una vez obtenido el archivo .tnt, se procedió a crear una cuenta en la plataforma Morphobank, luego se creó un proyecto como se presenta en la **Figura 22**, posteriormente se subió el archivo .tnt al sitio y, si no hay errores, en el apartado "Matrices" se deberá mostrar lo que se acabó de subir como se muestra en la **Figura 23.**

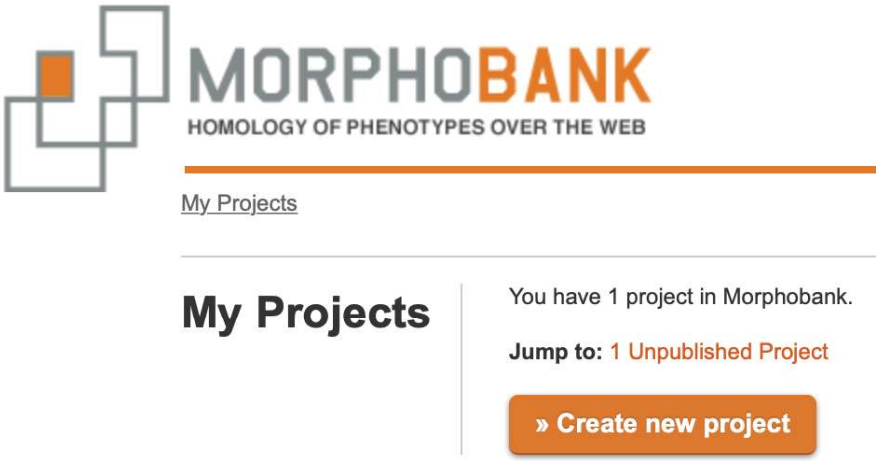


Figura 22. Interfaz WEB de Morphobank luego de crear un usuario

[3] https://morphobank.org/index.php/Projects/Matrices/project_id/3151

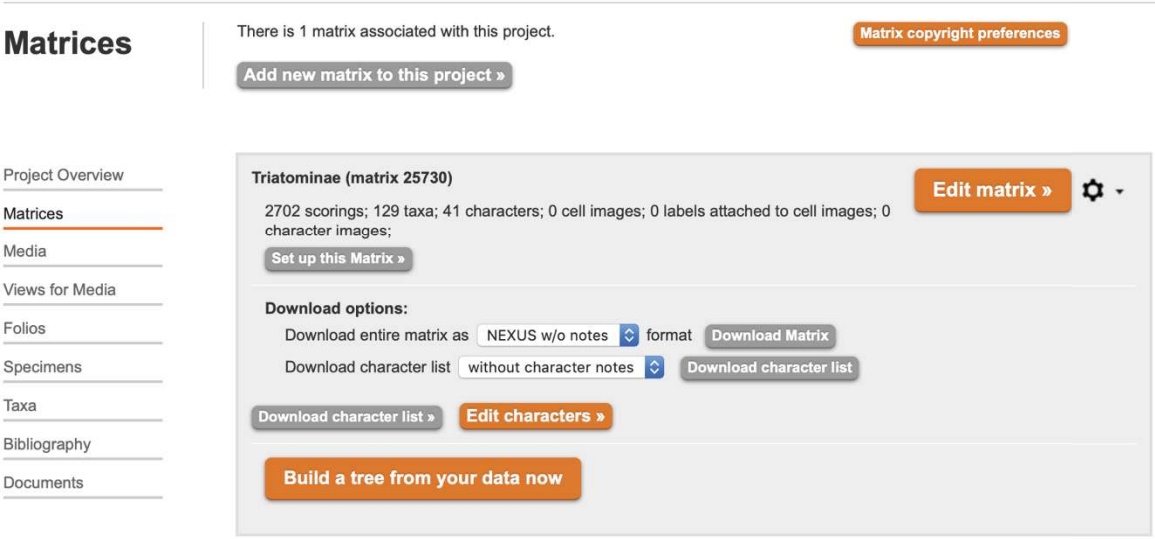


Figura 23. Interfaz WEB del apartado Matrices

Ahora Podemos ver la matriz y su representación visual dentro de Morphobank, en la **Figura 24** se muestra un extracto de las primeras diez filas y las primeras 5 columnas de la matriz.

	[1] Abdomen	[2] Antennae	[3] Antenniferous tubercles	[4] Anteocular region	[5] Anterior lobe
[1] *TRIATOMINI jeannel*	1	?	?	?	?
[2] *Triatoma laporte*	1	?	1	?	2
[3] *Triatoma amicitiae*	?	?	1	1	2
[4] *Triatoma arthurneivai*	?	?	1	2	2
[5] *Triatoma barberi*	?	?	1	1	2
[6] *Triatoma bouvieri*	0	?	?	1	?
[7] *Triatoma brasiliensis*	0	?	1	2	1
[8] *Triatoma breyeri*	?	?	1	2	2
[9] *Triatoma carrioni*	0	?	1	2	1
[10] *Triatoma cavernicola*	2	?	1	2	?

Figura 24. Extracto de la representación de la matriz en Morphobank

Posterior a la carga de la matriz dentro de Morphobank, utilizamos la opción “Build a tree from your data now”, dentro de este apartado podemos modificar el número de iteraciones

y el número de caracteres permutables para la creación del árbol, todo esto utilizando el algoritmo de la máxima parsimonia de Ratchet. En la **Figura 25** se muestran los apartados modificados para el análisis, se utilizaron 10 iteraciones y 20% de permutaciones por motivos de optimización.

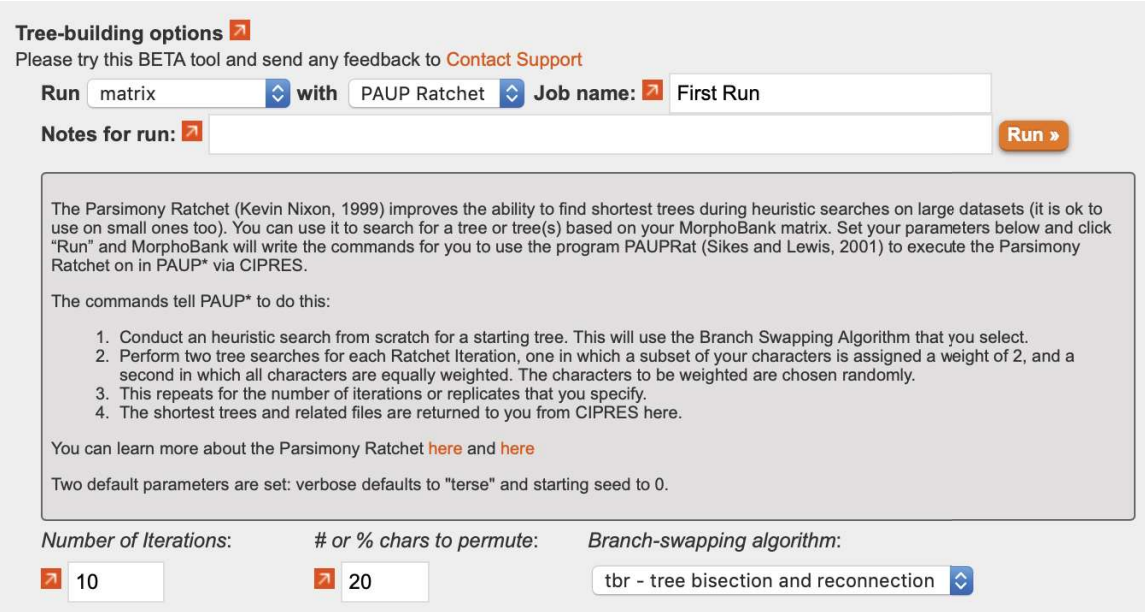


Figura 25. Interfaz de Morphobank para la creación del árbol

Después de ejecutar la herramienta Morphobank generará diversos archivos disponibles para la descarga dentro del mismo proyecto de Morphobank, en la **Tabla 15** se muestran los archivos generados por Morphobank y la descripción de su contenido.

Tabla 15. Archivos generados por Morphobank

Archivo	Descripción
paup_commands.txt	Contiene los comandos a ser ejecutados en PAUP
done.txt	Contiene información de la marca temporal en la cual comenzó la creación del árbol
infile.nex	Contiene la información generada a partir de la matriz codificada
ratchet.nex	Contiene la ejecución del algoritmo de Ratchet sobre el archivo “infile.nex”
setup.nex	Contiene distintos parámetros para la modificación de PAUP
start.txt	Contiene información de la marca temporal en la cual terminó la creación del árbol
term.txt	Contiene información de los recursos utilizados en la creación del árbol
STDERR	Contiene código necesario para manejo de errores
stderr.txt	Contiene código necesario para manejo de errores
STDOUT	Contiene información sobre la ejecución
stdout.txt	Contiene información sobre la ejecución

3.4.4. Generación del árbol

Con los archivos de Morphobank procesados, se procede a interpretarlos utilizando PAUP*, programa el cual utiliza los comandos descritos en uno de los archivos por Morphobank para generar nuestro árbol. En la **Figura 26** se muestra la interfaz de PAUP*, en la cual debemos ingresar los comandos descritos en la **Figura 27.**

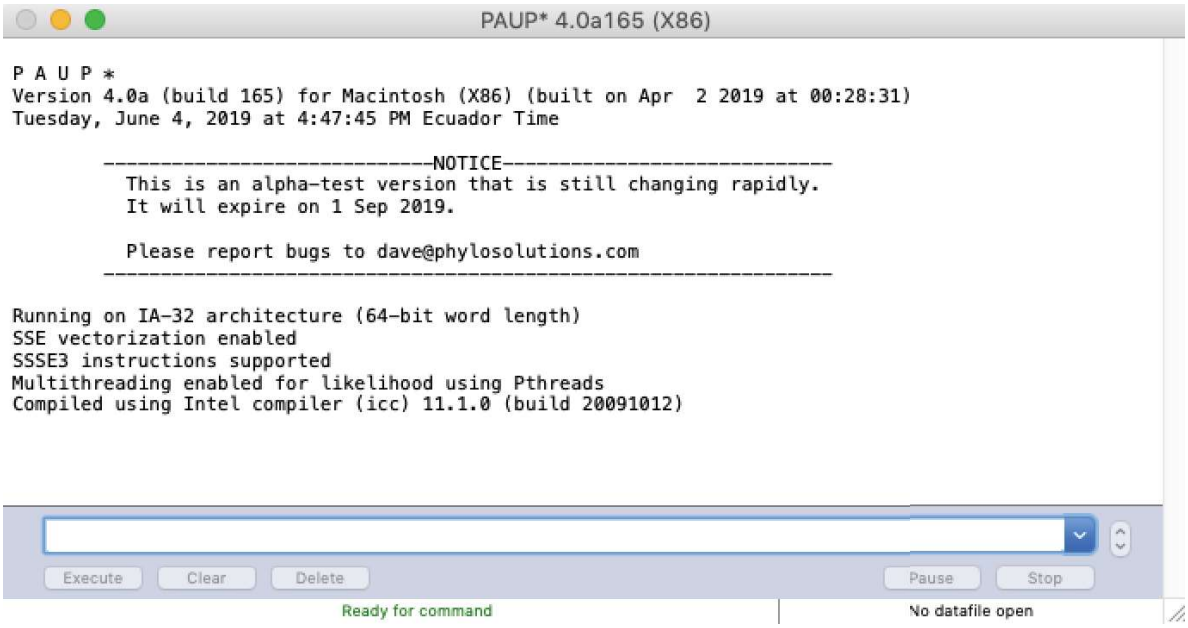


Figura 26. Interfaz de PAUP*

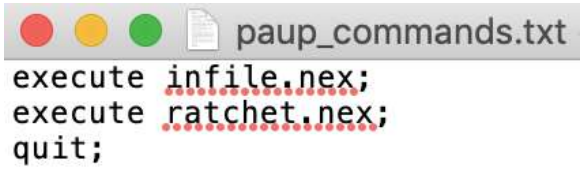


Figura 27. Comandos para ejecutar en PAUP*

Una vez ejecutados los comandos de PAUP*, se generarán nuevos archivos. En la **Tabla 16** se describen los archivos con los creados recientemente y su contenido.

Tabla 16. Archivos generados por Morphobank y PAUP*

Archivo	Descripción
paup_commands.txt	Contiene los comandos a ser ejecutados en PAUP
done.txt	Contiene información de la marca temporal en la cual comenzó la creación del árbol
infile.nex	Contiene la información generada a partir de la matriz codificada
ratchet.nex	Contiene la ejecución del algoritmo de Ratchet sobre el archivo "infile.nex"
setup.nex	Contiene distintos parámetros para la modificación de PAUP
start.txt	Contiene información de la marca temporal en la cual terminó la creación del árbol
term.txt	Contiene información de los recursos utilizados en la creación del árbol
STDERR	Contiene código necesario para manejo de errores
stderr.txt	Contiene código necesario para manejo de errores
STDOUT	Contiene información sobre la ejecución
stdout.txt	Contiene información sobre la ejecución
mtree.nex	Contiene la descripción de la matriz codificada, así como los nombres de las filas y las columnas (especies y características morfológicas)
pauprat.best.tre	Contiene el árbol filogenético

Finalmente, ejecutamos el programa FigTree y dentro del mismo, abrimos el archivo pauprat.best.tree para visualizar el árbol filogenético. La **Figura 28** muestra la interfaz gráfica de FigTree y la **Figura 29** muestra un extracto del árbol presentado en FigTree.

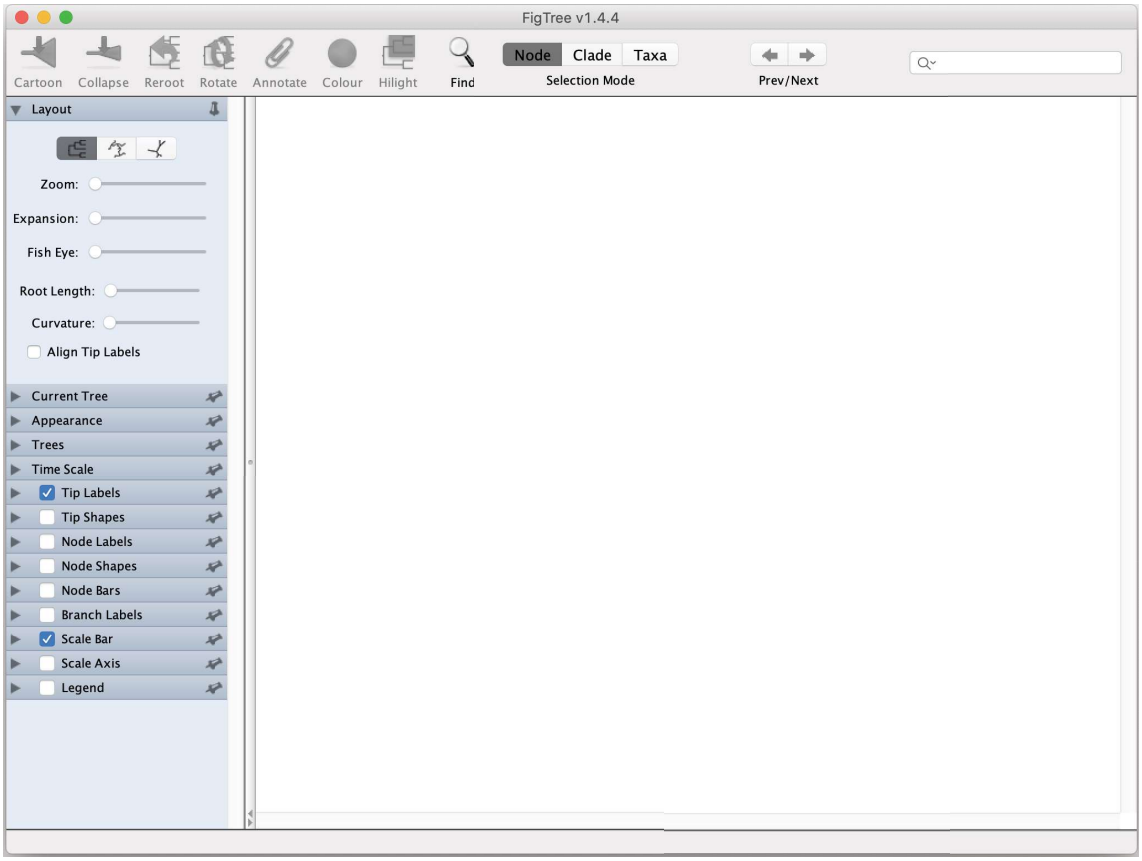


Figura 28. Interfaz gráfica de FigTree

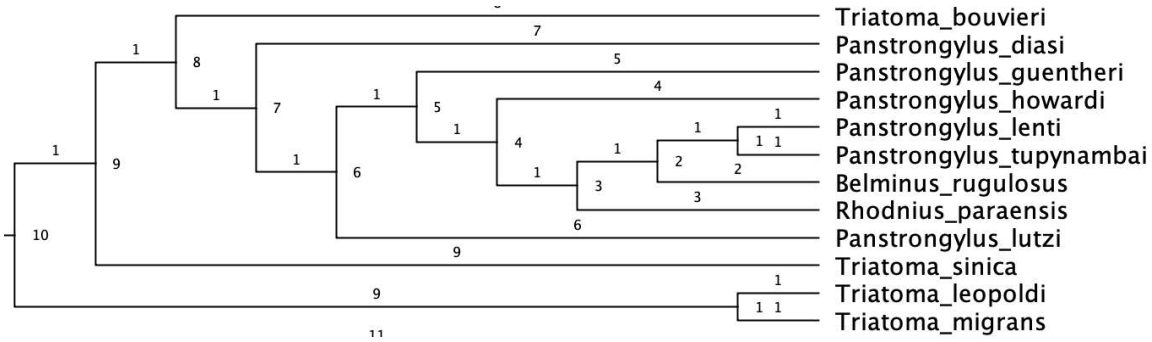


Figura 29. Extracto del árbol filogenético

3.5. Evaluación

Los resultados obtenidos se evaluaron inicialmente con un experto del dominio que recomendó la comparación en el futuro con datos procesados manualmente, proyectos que se encuentran actualmente en ejecución, aunque indicó que los resultados obtenidos en este proyecto tienen coherencia y serán de gran utilidad para la comunidad científica.

3.6. Discusión

El enfoque semiautomático combinado de extracción de información y agrupamiento fue efectivo para reducir el tiempo de extracción y codificación manual de las características.

El modelo de minería propuesto en CRISP-DM fue particularmente útil en el desarrollo de este proyecto debido a que los datos, evidentemente, necesitan un tratamiento luego de ser extraídos de forma textual de un libro que carece de estructura en su redacción.

En fases preliminares del proyecto se analizó la posibilidad de utilizar paralelismo computacional para realizar las operaciones que fueran prudentes en el proyecto, pero esto se descartó, principalmente, a los criterios heurísticos tomados para la extracción de los datos, por ejemplo, el uso de solamente el 20% de características morfológicas, debido a que el resto no se encontraban en todas las especies de insectos y no supondrían una diferencia al construir el árbol filogenético.

En la fase de preprocesamiento, el texto se procesó para eliminar encabezados e imágenes. En la segunda y tercera fase, la extracción y codificación características se desarrollaron utilizando agrupación de cadenas con buenos resultados. El libro de muestra es una referencia en biología y es la base del análisis morfológico actual, pero el texto se redactó sin criterios técnicos, es decir en base a las observaciones subjetivas de los autores, por ejemplo, podemos ver como se describen ciertas características como “negras” y en otros casos como “oscuras”, aquí se evidencia la subjetividad, la cual no podemos cuantificar de manera técnica, esta es la principal razón del uso de clusterización.

En adición, debemos recalcar que la utilización de conceptos básicos de teoría de conjuntos y cálculo matemático fueron vitales en el desarrollo y resolución de problemas del proyecto.

4. CONCLUSIONES Y TRABAJOS FUTUROS

Mediante todo lo expuesto en este trabajo se puede concluir que es posible la construcción de árboles filogenéticos utilizando minería de texto, en este caso, mediante la metodología CRISP-DM [15] junto con un análisis de frecuencias de N-gramas en R y el uso de herramientas de software externo ([24] y [28]).

Este proyecto utilizó la metodología CRISP-DM principalmente en la fase de preprocesamiento, debido a que se requería como paso fundamental la preparación de datos para su posterior lectura y minería porque el texto no se encuentra formateado y su redacción no presenta una estructura definida en cuanto a las descripciones de las especies, y CRISP-DM brinda una guía en cuanto a preparación y transformación de los datos para adaptarse a las necesidades del problema.

Además, en este proyecto se utilizó un enfoque constituido con una aproximación de la distancia de Leveshtein, pero adaptada a frases completas, lo que brindó resultados satisfactorios, se puede afirmar que el uso de esta medida es posible y muy usable en el ámbito de la minería de texto, específicamente en la clusterización para definir la matriz de distancias y posteriormente analizar en planos 2D.

La clusterización como modelo de minería de texto debe ser utilizado a la par o en conformidad con una metodología establecida para que sus resultados sean interpretados correctamente.

4.1. Trabajos futuros

A partir del proyecto realizado, se propone implementar un sistema web en el que se puedan visualizar los datos extraídos. Esto ayudaría a que la comunidad de biólogos interesados en la filogenia de los insectos triatominos pueda acceder a la información morfológica de una forma más simple y, así, ahorrar tiempo y costes en la búsqueda y extracción de la información.

También, se propone desarrollar, a partir de la información extraída, un entorno digitalizado para mostrar de forma didáctica los insectos estudiados. Esto ayudaría a que se comprendan de forma visual las características de los insectos triatominos y puede utilizarse como una herramienta educativa.

Al igual que, se propone realizar un sistema que logre generalizar la extracción de información de fuentes bibliográficas, a partir de las técnicas utilizadas en este proyecto. Es posible aplicar técnicas avanzadas que permitan automatizar la extracción y

clasificación de información sin importar el formato en el que se encuentre la fuente bibliográfica.

Además, los mismos datos, extraídos en este proyecto, podrían compararse con los datos de ADN, e incluso combinarse, para crear una filogenia de triatominae más confiable y completa.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] M. A. Mora y J. E. Araya, «Semi-automatic Extraction of Plants Morphological Characters from Taxonomic Descriptions Written in Spanish,» *Biodiversity data journal,* nº 6, 26 June 2018.

[2] W. Wheeler, Systematics: a course of lectures, John Wiley & Sons, 2012.

[3] H. Lent, P. Wygodzinsky y others, «Revision of the Triatominae (Hemiptera, Reduviidae), and their significance as vectors of Chagas' disease,» *Bulletin of the American Museum of Natural History,* vol. CLXIII, nº 3, pp. 123-520, 1979.

[4] J. D. Stanaway y G. Roth, «The burden of Chagas disease: estimates and challenges,» *Global Heart,* vol. X, nº 3, pp. 139-144, September 2015.

[5] B. Y. Lee, K. M. Bacon, M. E. Bottazzi y P. J. Hotez, «Global economic burden of Chagas disease: a computational simulation model,» *The Lancet infectious diseases,* vol. XIII, nº 4, pp. 342-348, April 2013.

[6] S. A. Justi, C. A. Russo, J. R. dos Santos Mallet, M. T. Obara y C. Galvão, «Molecular phylogeny of Triatomini (Hemiptera: Reduviidae: Triatominae),» *Parasites & vectors,* vol. VII, nº 1, pp. 7-149, 31 March 2014.

[7] S. A. Justi, C. Galvão y C. G. Schrago, «Geological changes of the Americas and their influence on the diversification of the Neotropical kissing bugs (Hemiptera: Reduviidae: Triatominae),» *PLoS neglected tropical diseases,* vol. X, nº 4, 8 April 2016.

[8] J. R. Hobbs y E. Riloff, «Information Extraction,» *Handbook of natural language processing,* vol. II, 2010.

[9] V. L. Trainer, S. S. Bates, N. Lundholm, A. E. Thessen, W. P. Cochlan, N. G. Adams y C. G. Trick, «Pseudo-nitzschia physiological ecology, phylogeny, toxicity, monitoring and impacts on ecosystem health,» *Harmful Algae,* vol. XIV, pp. 271-300, February 2012.

[10] J. P. Balhoff, W. M. Dahdul, C. R. Kothari, H. Lapp, J. G. Lundberg, P. Mabee, P. E. Midford, M. Westerfield y T. J. Vision, «Phenex: ontological annotation of phenotypic diversity,» vol. V, nº 5, 5 May 2010.

[11] A. R. Deans, M. J. Yoder y J. P. Balhoff, «Time to change how we describe biodiversity,» *Trends in ecology & evolution,* vol. XXVII, nº 2, pp. 78-84, February 2012.

[12] M. Zhou, G. Geng y S. Huang, «Ontology development for insect morphology and taxonomy system,» *2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops,* pp. 324-330, 8 January 2007.

[13] T. H. Nguyen, «Deep Learning for Information Extraction,» January 2018.

[14] P. Terdchanakul, H. Hata, P. Phannachitta y K. Matsumoto, «Bug or not? bug report classification using n-gram idf,» *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME),* pp. 534-538, 7 November 2017.

[15] C. Shearer, «The CRISP-DM model: the new blueprint for data mining,» *Journal of data warehousing,* vol. V, nº 4, pp. 13-22, 2000.

[16] M. Brown, «What IT Needs To Know About The Data Mining Process,» *Published by Forbes,* vol. XXIX, 2015.

[17] G. Harper y S. D. Pickett, «Methods for mining HTS data,» *Drug Discovery Today,* vol. XI, nº 15-16, pp. 694-699, August 2006.

[18] R Development Core Team, «R: What is R?,» 2006. [En línea]. Available: https://www.r-project.org/about.html. [Último acceso: 17 06 2019].

[19] S. Masumi , H. Takahiro y N. Shojiro , «N-gram IDF: A Global Term Weighting Scheme Based on,» *WWW '15 Proceedings of the 24th International Conference on World Wide Web,* vol. 24, pp. 960-970, 2017.

[20] D. Swofford, PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods), Sunderland, Massachusetts: Sinauer Associates, 2002.

[21] D. Maddison y W. Maddison, MacClade 4: Analysis of phylogeny and character evolution, Sunderland, Massachusetts: Sinnauer Associates, 2000.

[22] D. Maddison, D. Swofford y W. Maddison, «Nexus: An Extensible File Format for Systematic Information,» *Systematic Biology,* vol. XLVI, nº 4, pp. 590-621, 1 December 1997.

[23] B. Hall, Phylogenetic trees made easy, Sunderland, Massachusetts: Sinauer Associates, 2011.

[24] A. Rambaut, «Figtree,» 25 November 2018. [En línea]. Available: http://tree.bio.ed.ac.uk/software/figtree/. [Último acceso: June 2019].

[25] D. MacKay, «An example inference task: Clustering,» de *Information theory, inference and learning algorithms*, vol. XX, Cambridge, Cambridge University Press, 2003, pp. 284-289.

[26] J. A. Hartigan y M. A. Wong, «Algorithm AS 136: A k-means clustering algorithm,» *Journal of the Royal Statistical Society. Series C (Applied Statistics),* vol. XXVIII, nº 1, pp. 100-108, 1979.

[27] A. K. Jain, «Data clustering: 50 years beyond K-means,» *Pattern recognition letters,* vol. XXXI, nº 8, pp. 651-666, 1 June 2010.

[28] M. O'Leary y S. Kaufman, «MorphoBank 3.0: Web application for morphological phylogenetics and taxonomy,» *Available at website http://www. morphobank. org,* 2012.

[29] A. Stamatakis, «RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies,» *Bioinformatics,* vol. XXX, nº 9, pp. 1312-1313, 1 May 2014.

6. ANEXOS

ANEXO I

```
 # Sistema semiautomático para la extracción de información desde la fuente bibliográfica

import fitz
import re

# Representación de un insecto triatomino para la extracción de información
class Insect:
    page = ''
    name = ''
    last_name = ''

    def __init__(self, page, name, last_name):
        self.page = page
        self.name = name
        self.last_name = last_name

class Information_extract:
    # Lectura del índice de insectos creado para la extracción de información de la fuente bibliográfica
    def extract_index():
        file = open('insects_index.txt', 'r')
        index_content = file.read()
        split_index = index_content.split('\n')
        split_index = sorted(split_index)
        return split_index

    # Extracción de contenido de la fuente bibliográfica en un rango de páginas específicas
    def get_Content(initial_page, final_page):
        specific_content = ''
        pdf = fitz.open("LibroTriatominae.pdf")
        for i in range(initial_page - 120 - 1, final_page - 120):
            page = pdf.loadPage(i)
            specific_content += page.getText("text")
        return specific_content

    # Corrección de errores de lectura en los nombres de los insectos
    def edit_wrong_names(insect_description):
        insect_description = insect_description.replace('-\n', '')
```

```
        insect_description = insect_description.replace('  ', ' ')
        insect_description = insect_description.replace('\'', '')
        insect_description = insect_description.replace('\"', '')
        insect_description = insect_description.replace('MARTINFZ', 'MARTINEZ')
        insect_description = insect_description.replace('CARCA VALLO',
'CARCAVALLO')
        insect_description = insect_description.replace('Be/minus', 'Belminus')
        insect_description = insect_description.replace('ST AL', 'STAL')
        insect_description = insect_description.replace('mazzattii', 'mazzottii')
        insect_description = insect_description.replace('jlavida', 'flavida')
        insect_description = insect_description.replace('eratyrusif ormis',
'eratyrusiformis')
        insect_description = insect_description.replace('TRIA TO MINI',
'TRIATOMINI')
        insect_description = insect_description.replace('stgments', 'segments')
        insect_description = insect_description.replace('seg ments', 'segments')
        insect_description = insect_description.replace('tu bercles',
'tubercles')
        insect_description = insect_description.replace('denti cles.',
'denticles.')
        insect_description = insect_description.replace('trapezoi dal',
'trapezoidal')
        insect_description = insect_description.replace('un der', 'under')
        insect_description = insect_description.replace('an tenniferous',
'antenniferous')
        insect_description = insect_description.replace('sub median',
'submedian')
        insect_description = insect_description.replace('St!l', 'Stal')
        insect_description = insect_description.replace('MATERIAL EXAMINED',
'TYPE')
        insect_description = insect_description.replace('!', 'l')
        insect_description = insect_description.replace('fig. ', 'fig.')
        insect_description = insect_description.replace('antenna)', 'antennal')
        insect_description = insect_description.replace('antenna]', 'antennal')
        insect_description = insect_description.replace('Urostemites',
'Urosternites')
        insect_description = insect_description.replace('Antennifrous',
'Antenniferous')
        insect_description = insect_description.replace('Cori um', 'Corium')
        return insect_description

    # Metodo que extrae la información relevante de cada insecto
    def extract_content_insect():
```

```
        index_insects = Information_extract.extract_index()
        for index_item in range(len(index_insects)):
            split_index_insects = index_insects[index_item].split(':')
            split_insect_description_insects = split_index_insects[1].split(',')
            actual_insect = Insect(split_index_insects[0],
split_insect_description_insects[0],
                                   split_insect_description_insects[1])
            if index_insects[index_item] != index_insects[-1]:
                split_next_index_insects = index_insects[index_item +
1].split(':')
                next_insect = Insect(split_next_index_insects[0], '', '')
            else:
                next_insect = Insect('464', '', '')
            insect_description =
Information_extract.get_Content(int(actual_insect.page), int(next_insect.page))
            insect_description =
Information_extract.edit_wrong_names(insect_description)
            insect_description =
Information_extract.delete_repeated_phrases(actual_insect.last_name,

insect_description)
            insect_description =
Information_extract.delimit_content(insect_description, actual_insect.page,

actual_insect.name, actual_insect.last_name)
            insect_description =
Information_extract.delete_fig_references(insect_description)
            insect_description =
Information_extract.separe_paraghraps(insect_description)
            insect_description =
Information_extract.fix_decimals(insect_description)

            # GUARDAR EN ARCHIVOS DIFERENTES PARA LA EXTRACCION DE
CARACTERISTICAS
            path = 'descriptions_separated_files\\' + str(index_item) + '.txt'
            Write_Files.save_insects_different_txtfiles(path, insect_description)
            # PASA DE PYTHON A R --> Archivo R-DataSet.R

            # GUARDAR EN UN SOLO ARCHIVO TODAS LAS DESCRIPCIONES
            path = 'descriptions_files\\insect_descriptions.txt'
            Write_Files.save_insect_descriptions_in_txtfile(path,
insect_description, actual_insect.name,
```

```
actual_insect.last_name)
        # PASA A LA SEPARACION DE ARCHIVOS DE CARACTERISTICAS

    # Elimina conflictos con frases repetidas en el texto
    def delete_repeated_phrases(last_name, insect_description):
        if (last_name == 'Laporte'):
            insect_description = insect_description.replace('Triatoma Laporte,
1832. ', '')
        if (last_name == 'hirsuta Barber'):
            insect_description = insect_description.replace('Paratriatoma hirsuta
Barber. DISTRIBUTION:', '')
        if (last_name == 'goyovargasi'):
            insect_description = insect_description.replace('Alberprosenia
goyovargasi Martinez and Carcavallo, 1977',
                                                            '')
        if (last_name == 'rubrofasciata'):
            insect_description = insect_description.replace(
                'Triatoma rubrofasciata, which is superficially similar to
rubida, but the former species is conspicuously granulose on the head and
pronotum, and rubida is not.',
                '')
        if (last_name == 'costalis'):
            insect_description = insect_description.replace('Linshcosteus
costalis. OBSERVATIONS: ', '')
        if (last_name == 'maximus'):
            insect_description = insect_description.replace('Dipetalogaster
maximus possesses other characters', '')
        if (last_name == 'Stal'):
            insect_description = insect_description.replace(
                'Rhodnius Stal, 1859. Other genus included: Psammolestes
Bergroth, 1911. ', '')
        if (last_name == 'pilosa'):
            insect_description = insect_description.replace('Cavernicola pilosa
Barber, 1937. ', '')
        if (last_name == 'scabrosa'):
            insect_description = insect_description.replace('Bolbodera scabrosa
Valdes; type species of', '')
        if (last_name == 'Valdes'):
            insect_description = insect_description.replace('BIOLOGY:', 'TYPE:')
        return insect_description

    # Delimita el contenido para obtener solamente el párrafo de la descripción
de las características de cada insecto
```

```
    def delimit_content(insect_description, page, name, last_name):
        try:
            insect_description = insect_description[insect_description.index(name
+ " " + last_name):]
        except ValueError:
            print('Insect not found' + name + ' ' + last_name,
insect_description)
        try:
            insect_description =
insect_description[:insect_description.index('TYPE')]
            return insect_description
        except ValueError:
            print('Delimit not found')

    # Elimina referencia a figuras dentro del texto
    def delete_fig_references(insect_description):
        insect_description = str(insect_description.encode('utf-8'))
        insect_description = insect_description[1:].replace('\\',
'/').replace('/xc2/xad ', '').replace('/xc2/xad',

'').replace(
            '/xc2/xb7', '').replace('/xef/xbf/xbd', '').replace('/xc2/xb1',
'').replace('/', '\\').replace('\'',

'').replace(
            '\\n', '\n').strip(' ')
        insect_description = re.sub(r'(\(figs[\.\w\s\d,;\-]*\))', '',
insect_description)
        insect_description = re.sub(r'(\(fig[\.\w\s\d,;\-]*\))', '',
insect_description)
        insect_description = re.sub(r'(\(as in [\.\w\s\d,;]*\))', '',
insect_description)
        insect_description = re.sub(r'(\(see [\.\w\s\d,;]*\))', '',
insect_description)
        return insect_description

    # Define la separación entre párrafos en el texto
    def separe_paraghraps(insect_description):
        insect_description = insect_description.replace('  ', ' ')
        insect_description = insect_description.replace('\n', '@')
        insect_description = re.sub(r'(\.@)', '.\n', insect_description)
        insect_description = re.sub(r'(\.\s@)', '.\n', insect_description)
        insect_description = insect_description.replace('@', '')
```

```
        insect_description = insect_description.replace('fig. ', 'fig')
        insect_description = re.sub(r'(mm\.)', '', insect_description)
        insect_description = insect_description.replace('T.', 'T')
        insect_description = " ".join(insect_description.split())
        return insect_description

    # Elimina conflictos con números decimales en el texto
    def fix_decimals(insect_description):
        insect_description = re.sub(r':\s', ':', insect_description)
        decimals = []
        decimals = re.findall(r'\d\.\s\d', insect_description)
        for k in range(len(decimals)):
            insect_description = insect_description.replace(decimals[k],
decimals[k].replace('. ', '.'))
        decimals = re.findall(r'\d\.\d', insect_description)
        for k in range(len(decimals)):
            insect_description = insect_description.replace(decimals[k],
decimals[k].replace('.', ','))
        return insect_description

# Clase utilizada para el manejo de archivos
class Write_Files:
    def save_insect_descriptions_in_txtfile(file_path, text, insect_name,
insect_last_name):
        file = open(file_path, 'a', encoding='utf-8')
        file.write(insect_name + ' ' + insect_last_name + '@' + text + '\n')
        file.close()

    def save_insects_different_txtfiles(file_path, text):
        file = open(file_path, 'w', encoding='utf-8')
        file.write(text)
        file.close()

# Clase principal
def main():
    # Primera fase
    Information_extract.extract_content_insect()


if __name__ == "__main__":
    main()
```

ANEXO II

```
#Este script se utiliza para la creación de un vector de n-gramas que servirán
para la selección de características
#morfológicas de los insectos vectores de la enfermedad del Chagas


#Librerías utilizadas para:
#1. Text mining
library(tm)
#2. Creación de N-gramas
library(RWeka)

#Lectura de los archivos necesarios para crear los n-gramas
path = "C:/Users/bryan/Documents/GitHub/Filogenia-
Triatominae/descriptions_separated_files/"
dir = DirSource(paste(path,"/",sep=""), encoding = "UTF-8")
corpus = VCorpus(dir)

#Creación de los n-gramas con sus parámetros necesarios
BigramTokenizer <- function(x) NGramTokenizer(x, Weka_control(min = 1, max = 5))
tdm.bigram = TermDocumentMatrix(corpus,control = list(tolower =
FALSE,removeNumbers= T, removePunctuation = T ,stripWhitespace = T, tokenize =
BigramTokenizer))

#Exportar los n-gramas con su frecuencia en cada documento
write.table(as.matrix(tdm.bigram), file =
"C:/Users/bryan/Documents/GitHub/Filogenia-Triatominae/filesData.csv",sep = ",")
```

ANEXO III

```
# Sistema semiautomático para la extracción de las características morfológicas
de los insectos vectores
# de la enfermedad del Chagas

import pandas as pd
import numpy as np
import csv
import re
import json


# Clase creada para el manejo de archivos
class ReadWrite_Files:

    def write_in_csv(path, df):
        df.to_csv(path, index=False, header=False)

    def read_csv(path):
        characteristicsArray = []
        with open(path, newline='') as File:
            reader = csv.reader(File)
            for row in reader:
                characteristicsArray.append(row)
        return characteristicsArray


class Get_Characteristics:
    # Obtención de un vector de características evaluadas por su frecuencia
    def get_csv_characteristics():
        # Lectura del archivo de n-gramas
        CSV_PATH = "C:/Users/bryan/Documents/GitHub/Filogenia-
Triatominae/filesData.csv"
        df_complete = pd.read_csv(CSV_PATH)

        # Definición de un Dataset que contiene la información de los n-gramas
ordenados por nombre
        df_ordered = df_complete.sort_values(by=['name'], ascending=True)

        # Modificación del DataSet para eliminar caracteres no deseados
        df_ordered['name'] = df_ordered['name'].str.lstrip()
        df_ordered['name'] = df_ordered['name'].str.rstrip()
        df_ordered['name'] = df_ordered['name'].str.replace('  ', ' ')
```

```
        df_ordered = df_ordered.sort_values(by=['name'], ascending=True)

        # Agrupar n-gramas repetidos mediante una intersección.
        df_cut = df_ordered.groupby([df_ordered.name]).agg(
            {'0.txt': 'sum', '1.txt': 'sum', '2.txt': 'sum', '3.txt': 'sum',
'4.txt': 'sum', '5.txt': 'sum',
             '6.txt': 'sum', '7.txt': 'sum', '8.txt': 'sum', '9.txt': 'sum',
'10.txt': 'sum', '11.txt': 'sum',
             '12.txt': 'sum', '13.txt': 'sum', '14.txt': 'sum', '15.txt': 'sum',
'16.txt': 'sum', '17.txt': 'sum',
             '18.txt': 'sum', '19.txt': 'sum', '20.txt': 'sum', '21.txt': 'sum',
'22.txt': 'sum', '23.txt': 'sum',
             '24.txt': 'sum', '25.txt': 'sum', '26.txt': 'sum', '27.txt': 'sum',
'28.txt': 'sum', '29.txt': 'sum',
             '30.txt': 'sum', '31.txt': 'sum', '32.txt': 'sum', '33.txt': 'sum',
'34.txt': 'sum', '35.txt': 'sum',
             '36.txt': 'sum', '37.txt': 'sum', '38.txt': 'sum', '39.txt': 'sum',
'40.txt': 'sum', '41.txt': 'sum',
             '42.txt': 'sum', '43.txt': 'sum', '44.txt': 'sum', '45.txt': 'sum',
'46.txt': 'sum', '47.txt': 'sum',
             '48.txt': 'sum', '49.txt': 'sum', '50.txt': 'sum', '51.txt': 'sum',
'52.txt': 'sum', '53.txt': 'sum',
             '54.txt': 'sum', '55.txt': 'sum', '56.txt': 'sum', '57.txt': 'sum',
'58.txt': 'sum', '59.txt': 'sum',
             '60.txt': 'sum', '61.txt': 'sum', '62.txt': 'sum', '63.txt': 'sum',
'64.txt': 'sum', '65.txt': 'sum',
             '66.txt': 'sum', '67.txt': 'sum', '68.txt': 'sum', '69.txt': 'sum',
'70.txt': 'sum', '71.txt': 'sum',
             '72.txt': 'sum', '73.txt': 'sum', '74.txt': 'sum', '75.txt': 'sum',
'76.txt': 'sum', '77.txt': 'sum',
             '78.txt': 'sum', '79.txt': 'sum', '80.txt': 'sum', '81.txt': 'sum',
'82.txt': 'sum', '83.txt': 'sum',
             '84.txt': 'sum', '85.txt': 'sum', '86.txt': 'sum', '87.txt': 'sum',
'88.txt': 'sum', '89.txt': 'sum',
             '90.txt': 'sum', '91.txt': 'sum', '92.txt': 'sum', '93.txt': 'sum',
'94.txt': 'sum', '95.txt': 'sum',
             '96.txt': 'sum', '97.txt': 'sum', '98.txt': 'sum', '99.txt': 'sum',
'100.txt': 'sum', '101.txt': 'sum',
             '102.txt': 'sum', '103.txt': 'sum', '104.txt': 'sum', '105.txt':
'sum', '106.txt': 'sum', '107.txt': 'sum',
             '108.txt': 'sum', '109.txt': 'sum', '110.txt': 'sum', '111.txt':
'sum', '112.txt': 'sum', '113.txt': 'sum',
             '114.txt': 'sum', '115.txt': 'sum', '116.txt': 'sum', '117.txt':
```

```
'sum', '118.txt': 'sum', '119.txt': 'sum',
            '120.txt': 'sum', '121.txt': 'sum', '122.txt': 'sum', '123.txt':
'sum', '124.txt': 'sum', '125.txt': 'sum',
            '126.txt': 'sum', '127.txt': 'sum', '128.txt': 'sum', '129.txt':
'sum',
            '130.txt': 'sum'}).reset_index().reindex(columns=df_ordered.columns)

        # Eliminar n-gramas que empiecen con minúsculas o espacios
        df_cut['name'].replace(regex=True, inplace=True, to_replace=r'^[a-
z][\d\w\W\D\s]+', value=r'nan')
        df_cut['name'].replace(regex=True, inplace=True, to_replace=r'^\s',
value=r'nan')
        df_cut['name'].replace(regex=True, inplace=True, to_replace=r'',
value=r'nan')
        df_cut = df_cut[df_cut.name != 'nan']

        # Definir ocurrencias únicas de cada n-grama por archivo
        df_sum = df_cut
        df_aux = df_sum._get_numeric_data()
        df_aux[df_aux > 1] = 1
        df_sum['sum'] = df_sum[:][df_sum.columns].sum(1)
        df_result = pd.DataFrame({'name': df_sum['name'], 'sum': df_sum['sum']})

        # Condicionar resultados mediante la Regla de Paretto
        df_condition = df_result[df_result['sum'] > 25]
        df_condition = df_condition[df_condition['sum'] < 132]

        # Exportar resultados a un archivo .csv
        ReadWrite_Files.write_in_csv("Clasified_characteristics2.csv",
df_condition)


# Clase principal
def main():
    # Segunda fase
    Get_Characteristics.get_csv_characteristics()


if __name__ == "__main__":
    main()
```

ANEXO IV

```
# Sistema semiautomático para la extracción de las descripciones de las
características morfológicas de los insectos vectores
# de la enfermedad del Chagas.

import csv
import re
import json


# Clase creada para el manejo de archivos
class ReadWrite_Files:
    def write_in_json(path, json_file):
        with open(path, 'w') as outfile:
            archivo = json.dump(json_file, outfile)

    def read_csv(path):
        characteristicsArray = []
        with open(path, newline='') as File:
            reader = csv.reader(File)
            for row in reader:
                characteristicsArray.append(row)
        return characteristicsArray


class Get_Descriptions:

    # Se obtienen las características seleccionadas bajo ciertos criterios
    def get_characteristics():
        characteristicsArray =
ReadWrite_Files.read_csv('Clasified_characteristics.csv')
        return characteristicsArray

    # Se obtienen las descripciones de cada característica de cada insecto
    def get_descriptions():
        descriptionsArray = []
        descriptionsFile = open('descriptions_files/insect_descriptions.txt',
'r')
        descriptionsText = descriptionsFile.read()
        splitDescriptions = descriptionsText.split('\n')
        for i in range(len(splitDescriptions)):
            descriptionsArray.append(splitDescriptions[i].split('@'))
        return descriptionsArray
```

```
    # Se almacena la información de cada insecto en un archivo .json
    def get_json_characteristics():
        characteristicsArray = Get_Descriptions.get_characteristics()
        descriptionsArray = Get_Descriptions.get_descriptions()
        insects = {}
        characteristics = {}
        textoAux = ''
        for i in range(131):
            description = descriptionsArray[i][1]
            for j in range(len(characteristicsArray)):
                coincidences = re.findall(r'(' + characteristicsArray[j][0]
                                          + r'\s[\sa-zA-Z,;\(\)\-0-
9\[\]:\+\?]*\.?\s)', description)
                if len(coincidences) != 0:
                    characteristics[characteristicsArray[j][0]] =
coincidences[0].replace(characteristicsArray[j][0],

'')
                    insects['' + descriptionsArray[i][0]] = characteristics
            characteristics = {}
        input = {"Especies": insects}
        Get_Descriptions.get_characteristics_files(characteristicsArray, insects)
        path = 'json_insects_data.json'
        ReadWrite_Files.write_in_json(path, input)
        return input

    # Se agrupan las características dependiendo de su tipo, de cada insecto en
un archivo de texto
    def get_characteristics_files(characteristicsArray, insects):
        for i in range(len(characteristicsArray)):
            auxFile = open('characteristics_files/' + characteristicsArray[i][0]
+ '.txt', 'w')
            for j in insects:
                try:
                    auxFile.write(
                        j + '@' +
insects[j][characteristicsArray[i][0]].replace(characteristicsArray[i][0] + ' ',

'') + '\n')
                except KeyError:
                    print('', end=" ")
```

```python
# Clase principal
def main():
    # Tercera fase
    Get_Descriptions.get_json_characteristics()


if __name__ == "__main__":
    main()
```

ANEXO V

```
#Script que codifica mediante clusterización los archivos de características

import sys
import numpy as np
import matplotlib.pyplot as plt
import re
from sklearn.decomposition import PCA
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from scipy.spatial.distance import cdist
from os import listdir
from os.path import isfile, join

sys.modules[__name__].__dict__.clear()

#Obtenemos los nombres de archivos y los ordenamos
onlyfiles = [f for f in listdir('characteristics_files') if
isfile(join('characteristics_files', f))]
onlyfiles.sort()
characteristics = []
insectNames = []
charToFile = []

#Recorremos los archivos
for path in onlyfiles:
    lineToFile = []
    partName = path.replace('.txt','')
    lineToFile.append(partName)
    valuesOfCharacteristic = []

    #Abre los archivos y los limpia
    file = open('characteristics_files/' + path, 'r').read()
    file = re.sub(r'[:;,.]', '', file)
    file = re.sub(path,'',file)
    file = re.sub(r'[\w ]*@','',file)
    if file != '':
        content = file.split('\n')
        distanceMatrix = []
        #Creación de la matriz de distancias
        for line in content:
            if line != '':
                lineWords = line.split()
```

```
        lon = []
        for lineAux in content:
            if lineAux != '':
                #Aplicación de la diferencia simétrica
                unionValue = len(set(lineWords)
                                 .union(lineAux.split()))
                intersectionValue = len(set(lineWords)
                        .intersection(lineAux.split()))
                distance = unionValue - intersectionValue
                lon.append(distance)
        distanceMatrix.append(lon)
#Descomenta esto para obtener la matriz de distancias
'''print('\n'.join(['.join(['{:4}'.format(item) for item in row])
                for row in distanceMatrix]))'''
#Crea la representación bidimensional de la matriz de distancias
pca = PCA(n_components=2)
X3d = pca.fit_transform(distanceMatrix)
xpoints = []
ypoints = []
for i in range(0, len(X3d)):
    xpoints.append(float(X3d[i][0]))
    ypoints.append(float(X3d[i][1]))
x1 = np.array(xpoints)
x2 = np.array(ypoints)
#Descomenta esto para ver los puntos en un espacio bidimensional
'''plt.plot()
plt.title('2 Dimension points in ' + path)
plt.scatter(x1,x2)
plt.show()'''
#Construyendo el 'codo'
plt.plot()
X = np.array(list(zip(x1, x2))).reshape(len(x1), 2)
colors = ['b', 'g', 'r']
markers = ['o', 'v', 's']
# kmeans para determinar el número de clústeres
derivadas = []
distortions = []
K = range(1, 10)
for k in K:
    kmeanModel = KMeans(n_clusters=k).fit(X)
    kmeanModel.fit(X)
    distortions.append(sum(np.min(
        cdist(X, kmeanModel.cluster_centers_,
```

```
                    'euclidean'), axis=1)) / X.shape[0])
        #usamos la diferencia central para obtener el valor real de k
        for t in range(1, len(distortions) - 1):
            derivadas.append(distortions[t + 1]
                             + distortions[t - 1]
                             - (2 * distortions[t]))
        #Descomenta esto para dibujar el 'codo'
        '''
        plt.plot(K, distortions, 'bx-')
        plt.xlabel('Number of clusters')
        plt.ylabel('Distortion')
        plt.title('The Elbow Method showing the optimal k in ' + path)
        plt.show()'''
        #Creación de Clústeres
        distanceMatrix = []
        sumas = []
        distance = 0
        vectorizer = TfidfVectorizer(stop_words='english')
        X = vectorizer.fit_transform(file.split('\n'))
        #Añadimos 2 a k debido a los índices
        true_k = derivadas.index(max(derivadas)) + 2
        #Descomenta y usa init1 en la igualdad de init para usar puntos de
partida
        '''init1 = np.array([[0.0, 0.0], [100.0, 100.0]], np.int32)'''
        model = KMeans(n_clusters=true_k,init='k-means++',max_iter=600,n_init=10)
        model.fit(X)
        order_centroids = model.cluster_centers_.argsort()[:,
                          ::-1]
        terms = vectorizer.get_feature_names()
        #Se crea un array bidimensional para almacenar el tod de términos de cada
clúster
        allTerms = []
        for i in range(true_k):
            termsCluster = []
            for ind in order_centroids[i, :10]:
                termsCluster.append(terms[ind])
            lineAux = ''
            lineAux = lineAux + termsCluster[0].capitalize() + '_'
            for t in range(1,3):
                lineAux = lineAux  + termsCluster[t] + '_'
            lineAux = lineAux + termsCluster[4]
            lineToFile.append(lineAux)
            allTerms.append(termsCluster)
```

```
        #Asignación de cada frase a su respectivo clúster usando índices
        for line in content:
            values = []
            intersections = []
            for cluster in allTerms:
                intersection = set(
                    line.split()).intersection(cluster)
                if len(intersection) > 0:
                    intersections.append(intersection)
                else:
                    intersections.append(0)
                values.append(len(intersection))
            maxValue = max(values)
            #Si la línea está vacía asignamos un signo de interrogación
            if maxValue == 0:
                valuesOfCharacteristic.append('?')
            else:
                repeatedIndex = []
                for elem in range(0,len(values)):
                    if values[elem] == maxValue:
                        repeatedIndex.append(elem)
                if len(repeatedIndex)> 1:
                    aditions = []
                    for auxIndex in repeatedIndex:
                        sumAux = 0
                        for word in intersections[auxIndex]:
                            sumAux += allTerms[auxIndex].index(word)
                        aditions.append(sumAux)

                    #Asignamos al clúster con mayor afinidad

valuesOfCharacteristic.append(repeatedIndex[aditions.index(min(aditions))])
                else:
                    valuesOfCharacteristic.append(values.index(maxValue))
        characteristics.append(valuesOfCharacteristic)
        charToFile.append(lineToFile)

finalMatrix = []

#Se prepara los datos para escribir el archivo tnt
for m in range(0,len(characteristics[0])):
    auxVector = []
    for n in range(0,len(characteristics)):
```

```
        auxVector.append(characteristics[n][m])
    finalMatrix.append(auxVector)
print('\n'.join(['',join(['{:4}'.format(item) for item in row])
                        for row in finalMatrix]))
fileInsects = open('sortedInsects.txt','r')
insectNames = fileInsects.read().split('\n')

#Escribimos el encabezado
finalFile = open('tntFile.tnt','w')
finalFile.write('xread\n')
finalFile.write(str(len(finalMatrix[0])))
finalFile.write(' ')
finalFile.write(str(len(insectNames)-1))
finalFile.write('\n\n&[num]\n')
i = j = 0

#Escribimos los insectos y sus valores codificados
for i in range(0,len(insectNames)-1):
    finalFile.write(insectNames[i].replace(' ', '_'))
    for k in range(1,49-len(insectNames[i])): finalFile.write(' ')
    for j in range(0,len(finalMatrix[0])):
        finalFile.write(str(finalMatrix[i][j]))
    finalFile.write('\n')
finalFile.write(';\n\ncnames\n')
i = j = 0
for i in range(len(charToFile)):
    finalFile.write('{ '+str(i))
    finalFile.write(' ')
    finalFile.write(charToFile[i][0].replace(' ', '_')+'_')
    for j in range(1,len(charToFile[i])):
        finalFile.write(' ')
        finalFile.write(charToFile[i][j].replace(' ','_'))
    finalFile.write(';\n')
finalFile.write(';')
finalFile.close()
fileInsects.close()
```