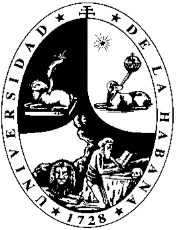
Facultad de Matemática y Ciencias de la Computación

Universidad de La Habana.



Nombre de la tesis

Trabajo de Diploma presentado en opción al título de

Licenciado en Ciencia de la Computación

Autor: Elias Bestard Lorigados

e.bestard@estudiantes.matcom.uh.cu

Tutor: Pedro Quinteras Rojas

La Habana, 2019

Tabla de Contenidos

[Introducción 3](#_Toc8916101)

[Objetivo general 3](#_Toc8916102)

[Objetivos específicos 3](#_Toc8916103)

[Estado del Arte: 4](#_Toc8916104)

[Procesar y extraer información de documentos de textos: 4](#_Toc8916105)

[Graficar Datos: 5](#_Toc8916106)

[Herramientas de visualización y procesamiento de información: 6](#_Toc8916107)

[Diseño de la aplicación: 7](#_Toc8916108)

[Módulo de Interacción con el usuario: 7](#_Toc8916109)

[Módulo Identificador de Datos: 8](#_Toc8916110)

[Módulo Formatos Conocidos (KF): 8](#_Toc8916111)

[Módulo Graficar (Generación de gráficos): 9](#_Toc8916112)

[Retroalimentación: 10](#_Toc8916113)

[Detalles de Implementación: 11](#_Toc8916114)

[Módulo de Interacción con el usuario (CLI): 11](#_Toc8916115)

[Módulo *Parsers:* 12](#_Toc8916116)

[Módulo Gráficos: 13](#_Toc8916117)

[Requerimientos: 13](#_Toc8916118)

# Introducción

## Objetivo general

Desarrollar un *software* que permita graficar información a partir de documentos de los cuales se pueda desconocer cómo está estructurado.

## Objetivos específicos

1. Estudiar herramientas actuales que permitan el desarrollo del *software*.
2. Proponer diseño modular que permita extraer información de cadenas de textos y graficarla.
3. Proponer un mecanismo de aprendizaje sencillo que permita seleccionar de forma inteligente gráficos partiendo de un conjunto de datos.
4. Implementar un prototipo del modelo establecido con una interfaz de usuario amigable, (CLI).
5. Validar el prototipo presentado.

# Estado del Arte:

Dada la gran información de que se dispone tanto en el terreno de las ciencias, el comercio o la vida social es que se hace evidente la necesidad de contar con métodos o procederes que permitan procesar datos y extraer de los mismos información útil.

Para visualizar la información contenida en documentos de texto mediante gráficas existen diversas etapas que son de vital importancia: procesar la información, extraer los datos de interés y crear gráficos. En cada etapa existen diversas ramas de la investigación dedicadas a su estudio. En el caso del presente documento, para el desarrollo de la aplicación objeto de esta tesis, se realizó un estudio de estas ramas.

## Procesar y extraer información de documentos de textos:

Si se parte de un documento, del cual se desconoce su estructura y el contenido en sí, se hace complejo poder extraer datos que le sean de utilidad al usuario. En este punto es donde se busca reconocer estructuras en los datos, acomodarlos para partir de algo conocido y extraer información que resulte valiosa. Actualmente existen ramas de la inteligencia artificial y de recuperación de información que estudian como extraer datos y procesarlos, transformándolos a estructuras comprensibles posteriormente, como es el caso de la minería de textos (por sus términos en ingles *text mining*) y minería de datos (por sus términos en ingles *data mining*).

La minería de datos es el proceso de obtener conocimiento nuevo a partir de grandes cantidades de datos (Han, Kamber, & Pei, 2012). Es un campo interdisciplinario, al que contribuyen muchas áreas, como son: la estadística, el aprendizaje de máquinas, recuperación de la información, reconocimiento de patrones y bioinformática. El objetivo de la minería de datos es el descubrimiento de patrones, perfiles, anomalías y tendencias a través del análisis de los datos. Las principales técnicas en este campo incluyen clasificación y predicción, agrupación o *clustering* (por su término en inglés), detección de anomalías, reglas de asociación, redes neuronales, arboles de decisión, etc.

Entre las áreas que abarca la minería de datos se encuentra la minería de textos, en la que los datos son documentos de texto. La minería de textos se caracteriza por poseer la información muy poco estructurada y confusa, mientras que la minería de datos trabaja con bases de datos estructuradas. Dentro de la minería de datos, la minería de textos es considerada como una de las áreas de mayor potencial.

Podemos definir la minería de textos como el proceso de extracción de patrones y conocimientos de interés y no triviales a partir de textos no estructurados. Es, por tanto, un campo multidisciplinar, que incluye conocimientos de recuperación de la información, análisis de textos, extracción de la información, *clustering*, categorización, aprendizaje de máquinas, etc. (Doctor Bracho, 2018)

## Graficar Datos:

El primer paso para graficar es buscar la mejor manera de mostrar la información. Los gráficos ayudan mucho a comprender los datos numéricos de un texto. La rama del periodismo de datos se centra en estudiar como mostrar la información y como contar una historia con datos.

El periodismo de datos utiliza las herramientas estadísticas y de visualización de datos con el objetivo de contar las viejas historias de otra forma, de un modo más claro para el público, y descubrir otras nuevas, ocultas hasta entonces. (Flores Vivar & Salinas Aguilar, 2012)

En la actualidad el periodismo de datos ha alcanzado gran éxito a nivel internacional. Para ello, partimos del hecho que las tecnologías de la información han cambiado por completo el concepto que teníamos del mundo. Se han creado miles de herramientas nuevas que han diversificado los usos de las que ya conocíamos. En esta vorágine de novedades, se comenzó a “mezclar” distintas herramientas para nuevos usos. La ciencia informática y la telemática han dado pie al surgimiento de un entorno profesional que se explota cada vez más en los medios. (Flores Vivar & Salinas Aguilar, 2013)

Existen diversas herramientas tecnológicas disponibles que limpian, analizan y cruzan los datos. De form tal que el publico pueda visualizarlos y entenderlos con facilidad. Algunos de los programas más utilizados por los periodistas de datos son: Data Wrangler, una aplicación de la Universidad de Standford que permite explotar hojas de cálculo; Google Fusion Tables, que convierte bases de datos en mapas y geolocaliza direcciones en ellos; Google Refine, que gracias a sus filtros depura la información, homogeneizando las bases de datos utilizadas; Statwing, recomendado para el análisis de datos estadísticos y cruces de variables; Data Wrapper, especialmente indicado para la creación de gráficos, y Piktochart, que también permite crear infografías. (Chaparro Domínguez, 2014)

Para Giannina Segnini, directora de la Maestría de las Ciencias de Periodismo de Datos de la Universidad de Columbia, Nueva York, en el periodismo de datos deben contemplarse cinco pasos básicos (Ramírez, 2019):

1. Obtención de los datos.
2. Limpieza de los datos.
3. Análisis.
4. Verificación de la información.
5. Visualización.

## Herramientas de visualización y procesamiento de información:

Las herramientas para graficar datos son bastantes utilizadas y exigidas hoy en día, existen diversas aplicaciones que permiten visualizar datos mediante gráficas, mapas, tablas, etc. Estas herramientas tienen fines diversos, algunas son simplemente librerías de lenguajes de programación, herramientas online, en su mayoría estos procesan la información de una base de datos o un archivo que conocen la estructura, dígase archivo CSV o archivos Excel. Entre los ejemplos más utilizados de aplicaciones tenemos: Tableau, un *software* que se especializa en técnicas de visualización, enfocado en inteligencia empresarial, analiza y procesa volúmenes de datos, se enfoca en datos bien estructurados, establece un formato, bases de datos relacionales y hojas de cálculo (Tableau, 2019); SPSS desarrollado por IBM y ofrece un avanzado análisis estadístico, extensible con *open-source,* varios algoritmos de *matching-learning,* análisis de textos e integración con *BIG-DATA* (IBM, 2019) muy usado en las ciencias sociales y exactas, este puede ser usado con múltiples tipos de bases de datos, bases de datos relacionales y textos, en caso de recibir texto el usuario especifica de qué forma se analiza, especificando la estructura (IBM).

Herramientas más simples permiten generar gráficos de igual manera sin hacer minería de datos como Infigr.am que es una herramienta para crear infografías, reportes, gráficos, etc., donde se puede importar los datos de archivos CSV o XLS; Google Charts, una aplicación de Google para realizar estadística web con buen monto de gráficos; Microsoft Excel que es una herramienta bien conocida permite crear gráficos, pero como las anteriores no hace pre-procesamiento de los datos, espera los datos en formatos establecidos y solo trabaja con datos estructurados o datos que se insertan de forma manual.

En resumen, podemos apreciar varias herramientas y métodos que se usan en la actualidad para cada parte de este proceso. Vemos que para procesar y extraer información de documentos la minería de datos en general está abordando estos aspectos, apoyándose principalmente en técnicas de inteligencia artificial. De igual forma para graficar el periodismo de datos utiliza varias herramientas y ha creado la necesidad del surgimiento de nuevas aplicaciones que facilitan la creación de gráficos e infografías fusionando varias de estas para mostrar información y los pasos que deben contemplarse para mostrar buenos resultados en gráficos.

# Diseño de la aplicación:

La aplicación consta de cuatro módulos principales: el módulo de interacción con el usuario, el identificador de datos, una capa intermedia de formatos conocidos (KF) y el módulo que genera los gráficos y un módulo extra para retroalimentar el proceso. En la Figura 1 se puede apreciar el flujo general de la aplicación:



Figura . Diagrama de flujo general de la aplicación.

A continuación, se detallan cada uno de los módulos de la aplicación.

## Módulo de Interacción con el usuario:

Este módulo se encarga de procesar los parámetros de entrada del programa y establecer la configuración que tendrá el mismo durante todo el proceso. La Figura 2 describe el funcionamiento detallado de este módulo.

Este módulo recibe los parámetros del usuario, y establece la configuración del programa, la cual se mantiene hasta el fin del proceso. Con la configuración establecida pasa a mostrar ayudas, generar juegos de datos, o carga fichero y analizarlo. Solo se realiza una opción en caso de solicitar varias, las mismas se realizarán de acuerdo al orden descrito. En caso de mostrar ayuda, el proceso termina en este módulo luego mostrarla. Si se acciona generar juegos de datos se pasa a graficar o a generar datos en los módulos correspondientes (Gráficos o Identificador de Datos), dependiendo la configuración, al generarlos termina el proceso. En caso que se cargue el fichero, se pasa al Identificador de Datos continuando el flujo del programa hasta su fin.



Figura . Flujo del módulo Interacción con el Usuario.

## Módulo Identificador de Datos:

El módulo trata de reconocer la cadena creando formatos para graficarlos posteriormente. La Figura 3 muestra detalladamente el flujo de este módulo.

El módulo recibe los datos como cadena de texto, donde primero hace un pre-procesamiento de estos acomodándolos para extraer información. Luego se pasa a analizar la cadena completa. De ser reconocida se construyen los formatos conocidos (KF) pertinentes. En caso contrario, se hace un procesamiento más profundo, al analizar línea a línea la cadena, y creamos los KF apropiados. Al analizar todas las líneas, y tener un conjunto de KF pasamos a compactarlos, tratando de unir los KF similares en líneas adyacentes.



Figura . Flujo del Módulo Identificador de datos. Donde X representa un XOR, KF: formato conocido

## Módulo Formatos Conocidos (KF):

Los Formatos Conocidos constituyen una capa intermedia. Este módulo está compuesto solamente por objetos que definimos como formatos conocidos, independientes de otro módulo en sí, pero a la vez constituyen los pilares de la aplicación. Todos los *parsers* saben que tipos “KF” generan, y cada grafico sabe que tipos de “KF” aceptan para poder graficar(Figura 4).



Figura . Módulo Formatos conocidos.

Los Formatos conocidos son estructuras que sirven como capa traductora de un módulo al otro, pues se necesita transformar la información extraída a un objeto que se sepa graficar. De esta forma se definen un conjunto de formatos conocidos de acuerdo a los tipos de datos diferentes que se necesiten para poder graficar información, mientras que los *parsers* son capaces de crear varios formatos conocidos acorde a la información que procesa como se ve en la Figura 5 y un formato conocido puede generar uno o más gráficos, Figura 6.



Figura . Relación entre un ParserA y los Formatos conocidos.



Figura . Relación entre un Formato Conocido (KFA) y los Gráficos.

## Módulo Graficar (Generación de gráficos):

Este último módulo está compuesto por un conjunto de gráficos definidos. Estos parten de un formato conocido y el objetivo del módulo es crear todos los gráficos posibles como vemos a continuación en la Figura 77:



Figura . Flujo del Módulo Graficar.

Primero se buscan todos los gráficos compatibles con el formato conocido, luego en dependencia de la configuración del programa, se seleccionan los gráficos que más se asemejan a los datos o se prosigue con el conjunto de gráficos inicial, se genera el código de los gráficos para luego construir el archivo de salida y concluir el programa.

La selección de Inteligente de Gráficos está diseñada con ideas de un Sistema basado en reglas. En la Figura 8 se explica su diseño.



Figura . Descripción del paso llamado Evaluar Reglas del Módulo Graficar.

Como vemos en el flujo de este sub-módulo, primero se evalúan las reglas en los gráficos seleccionados, contando con la retroalimentación o *feedback* de los gráficos que han sido útiles para el usuario. De igual forma se evalúan las reglas para ver qué tipo de mensaje se quiere transmitir contando con el *feedback* de estos, así se selecciona el mensaje a transmitir y se termina de escoger los k-mejores gráficos para mostrar.

## Retroalimentación:

Este módulo es una pequeña aplicación que muestra un archivo generado en el proceso principal y permite al usuario seleccionar los gráficos que fueron de utilidad guardando esta información en una base de datos, para utilizarla en próximas ejecuciones.

# Detalles de Implementación:

La aplicación es implementada en el lenguaje de programación *Python.*

Para explicar el flujo del programa visto en la Figura 1, vamos a mostrar cómo fue implementado, viendo su estructura y distribución.

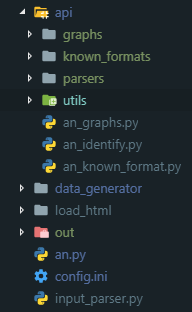


Figura . Distribución de la aplicación.

La raíz de la aplicación consta con un archivo “an.py”, que es el archivo que se debe correr para iniciar el flujo del programa, un archivo “input\_parser.py” y un archivo “config.ini” que se explicaran más adelante. La carpeta “api” contiene los módulos principales del programa, “graphs”, “known\_formats” y “parsers”, estos son gestionados por “an\_graphs.py”, “an\_identify.py”, “an\_known\_format.py”, respectivamente. Contiene una carpeta “utils” la cual contiene archivos útiles para la implementación de la app y una base de datos de la cual nos apoyamos más adelante para poder retroalimentar la aplicación. Las carpetas “data\_generator” y “out” contienen los archivos con juegos de datos que se generan y los archivos resultantes al terminar el proceso. Por ultimo tenemos la carpeta “load\_html” que es la encargada de la retroalimentación. La Figura 9 muestra la distribución de la aplicación.

## Módulo de Interacción con el usuario (CLI):

El archivo “input\_parser.py” es el encargado del módulo de interacción con el usuario, el que esta implementado con una interfaz de líneas de comando (CLI por sus siglas en inglés). Se apoya de la librería *argparser*[[1]](#footnote-1) y así nuestra interfaz recibe un monto de parámetros y ofrece una ayuda al usuario. Un ejemplo de interacción con esta interfaz se ve a continuación en la Figura 10.



Figura . Interacción con el CLI, recibiendo parámetro “parser” con valor p\_numbers\_list.

Acorde a los parámetros recibidos la aplicación establece la configuración y pasa a mostrar ayuda, generar juegos de datos o analizar un archivo, acorde al pedido del usuario. Para establecer la configuración, nos apoyamos en una clase “Config()” la que inicialmente contiene los parámetros por defecto de la aplicación y almacenará la configuración de la app durante todo el flujo de la misma. Para establecer los parámetros de esta clase primero se busca la existencia de un fichero “ *config.ini* ”, en caso de existir, leemos sus parámetros y actualizamos la clase, luego tomamos los parámetros de la entrada estándar y sobrescribimos los anteriores, estableciendo claramente una prioridad en el proceso de ajuste de la configuración: primera prioridad son los parámetros de la entrada estándar, seguido del archivo config.ini, y en un final los parámetros definidos por defecto en “*Config()*”.

El archivo “config.ini”, es un archivo compuesto por secciones (establecidas entre corchetes) y valores (los cuales se definen parámetros y su valor), lo cual hace que sea fácil la extracción de los valores, se ve un ejemplo de este file en la Figura 11, este último no tiene necesidad de existir, en caso de no existir se establece la configuración acorde los parámetros de la entrada estándar. No todos los valores tienen que estar presentes en el archivo.

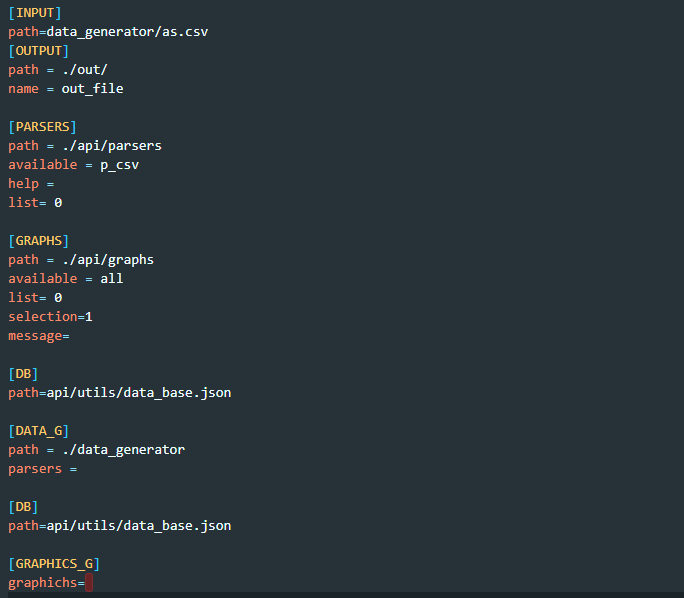


Figura . Ejemplo de archivo “config.ini” donde se establece la sección PARSERS y tiene parámetros “path”, “available”, “help”, “list”, con sus respectivos valores.

El flujo de configuración se puede ejemplificar si se asume que la entrada es la mostrada en Figura 10 y el archivo “*config.ini*” es el mostrado en Figura 11, por la entrada estándar el usuario nos pide que se analicen los datos solamente con el *parser p\_numbers\_list,* mientras que en el fichero está habilitado el *parser* p\_csv, por lo antes expuesto la configuración final tendrá como *parser* habilitado *p\_numbers\_list.*

La clase “*Config()”* tiene como singularidad que implementa el patrón *singleton*, así solo se puede crear una instancia de la misma y proporcionando un acceso global a esta.

## Módulo *Parsers:*

El archivo “an\_identify.py”

El archivo “An\_Known\_ Format.py” solo es una capa para hacer visible los KF, importando los formatos conocidos definidos en el directorio “known\_formats” que en estos momentos son muy simples.

## Módulo Gráficos:

El archivo “An\_Graph”

Pingüino

Los resultados se guardan en la carpeta output

## Requerimientos:

1. python 3
2. configparser
3. json
4. python highcharts
5. argparse
6. electron

1. Es un módulo de parseo por comando de línea recomendado por la librería estándar de Python. (Van Rossum, 2018) [↑](#footnote-ref-1)