Universidad Nacional Autónoma de Mexico.

Instituto de Ingeniería.

Grupo de Ingeniería Lingüística.

Manual Técnico del Sistema de Categorización de Definiciones.

Involucrados:

Ignacio Arroyo Fernández.

Ramón Pantoja Velasco.
masterraymus@gmail.com

México D.F. a 31 de Noviembre de 2015.

Índice. **Pag.**

Introducción.

Descripción del problema.

Se requiere un sistema de apoyo al sistema Anotador de definiciones, el cual, dadas una lista de definiciones sobre un término en específico y las acepciones del mismo término obtenidas de Wikipedia, deberá generar una categorización asignando cada definición con una acepción, se busca obtener un archivo de salida con los datos de la relación antes mencionada y un ranking de los mismos dado por su porcentaje de similitud. Esto deberá llevarse a cabo analizando unigramas, bigramas y trigramas. Es necesario un grupo no informativo para las definiciones que no cumplan un cierto porcentaje de similitud con ninguna acepción.

Propuesta de Solución.

La propuesta de solución es: por medio de procesar las definiciones y acepciones con TFIDF y calculando la similitud coseno entre definiciones y acepciones, en unigramas, bigramas y trigramas, y por otro lado, analizando definiciones y acepciones sin TFIDF, solo generando vectores por vocabulario obtenido de las definiciones y calculando la similitud coseno igualmente en unigramas, bigramas y trigramas, se elegirá una de las 6 opciones posibles y se generará un diccionario guardado en un documento en formato de .txt donde se muestran las acepciones, y las definiciones asignadas a ellos, el valor de similitud y se acomodan en un ranking de mayor similitud a menor similitud.

Objetivos.

- Apoyar a los anotadores de definiciones con sugerencias para asignar una definición a cierto grupo.
- Tener una alternativa de Categorización.
- Verificar el correcto trabajo de los anotadores.

Tecnologías usadas

- Distribución Fedora 22 (Linux).- Se desarrolló completamente en un ambiente Linux, cualquier distribución de Linux es útil solo tendrá pequeñas diferencias en la instalación de las herramientas que han sido utilizadas para crear este sistema. Esto se decidió ya que es más sencillo instalar las herramientas y bibliotecas de programación requeridas en un sistema Linux.
- Python 2.7 instalado de base en Fedora y en la mayoría de distribuciones de kernel Linux, se puede verificar esto con el comando "$ python –V" desde línea de comandos de Linux y en caso de no estar instalado se instala (en Fedora 22) con el comando "sudo apt-get install python2.7".
- Se utilizaron las bibliotecas de scikit learn el cual se encuentra un manual de instalación en la dirección: http://scikit-learn.org/stable/install.html y en el caso de

fedora solo fue necesario ejecutar el comando “sudo dnf install python-scikit-learn”. Esto nos sirve para poder utilizar las clases TfidfVectorizer para análisis con TFIDF y para usar la clase CountVectorizer y analizar sin TFIDF.

- Se utilizaron así mismo las bibliotecas de scipy con el instructivo que define la página oficial de scipy, la cual es: http://www.scipy.org/install.html, y es con el comando “sudo yum install numpy scipy python-matplotlib ipython python-pandas sympy python-nose”. Esto sirve para utilizar la función de similitud coseno.

Términos utilizados.

- Acepción.- Sentido en que se puede tomar una palabra o expresión y que, una vez aceptado y reconocido por el uso, se expresa en los diccionarios a través de la definición.
- Definición.- Una definición es una proposición mediante la cual trata de exponer de manera unívoca y con precisión la comprensión de un concepto o término o dicción o –si consta de dos o más palabras– de una expresión o locución.
- Término.- Palabra de una lengua, especialmente la que designa una noción en un ámbito de especialidad determinado.
- Unigrama.- Analisis hecho con una sola palabra.
- Bigrama.- Un bigrama o digrama es un grupo de dos letras, dos sílabas, o dos palabras. Los bigramas son utilizados comúnmente como base para el simple análisis estadístico de texto. En este caso de usan grupos de palabras.
- Trigrama.- Un trigrama es un grupo de tres letras, tres sílabas, o tres palabras. Los trigramas son utilizados comúnmente como base para el simple análisis estadístico de texto. En este caso de usan grupos de palabras.
- Similitud coseno.- La similitud coseno es una medida de la similitud existente entre dos vectores en un espacio que posee un producto interior con el que se evalúa el valor del coseno del ángulo comprendido entre ellos. Esta función trigonométrica proporciona un valor igual a 1 si el ángulo comprendido es cero, es decir si ambos vectores apuntan a un mismo lugar. Cualquier ángulo existente entre los vectores, el coseno arrojaría un valor inferior a uno.
- TFIDF.- frecuencia de término – frecuencia inversa de documento (o sea, la frecuencia de ocurrencia del término en la colección de documentos), es una medida numérica que expresa cuán relevante es una palabra para un documento en una colección. Esta medida se utiliza a menudo como un factor de ponderación en la recuperación de información y la minería de texto. El valor tf-idf aumenta proporcionalmente al número de veces que una palabra aparece en el documento, pero es compensada por la frecuencia de la palabra en la colección de documentos, lo que permite manejar el hecho de que algunas palabras son generalmente más comunes que otras.

Descripción de los casos de uso.

Caso de uso:	1.1 Generar asignación

Actores:	Sistema anotador
Propósito:	Generar diccionario con las asignaciones de definiciones a una definición y acepción de Wikipedia.
Resumen:	El sistema anotador, podrá generar un diccionario que servirá de guía a la hora de anotar, asignando definiciones a acepciones obtenidas de Wikipedia.
Precondiciones:	Se deben tener 3 archivos .csv, uno de definiciones, otro de acepciones y el último de definiciones de Wikipedia.
Tipo:	Primario
Descripción:	El sistema anotador, obtendrá un archivo de apoyo para el etiquetado.

Flujo Normal		**Flujos de Excepción**
Acciones del Actor	**Respuesta del Sistema**	
1.1- Llama al programa mandado la opción 1 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia, la dirección y nombre de salida de archivo y el porcentaje de mínima relación.	2.1-Procesa los archivos, genera la relación usando TFIDF para unigramas y genera el diccionario y lo guarda en la dirección asignada.	
Flujos Alternos		
2.1.- Llama al programa mandado la opción 2 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia, la dirección y nombre de salida de archivo y el porcentaje de mínima relación.	2.2.-Procesa los archivos, genera la relación usando TFIDF para bigramas y genera el diccionario y lo guarda en la dirección asignada.	
3.1.- Llama al programa mandado la opción 3 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia,	3.2.-Procesa los archivos, genera la relación usando TFIDF para trigramas y genera el diccionario y lo	

la dirección y nombre de salida de archivo y el porcentaje de mínima relación.	guarda en la dirección asignada.	
4.1.- Llama al programa mandado la opción 4 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia, lal dirección y nombre de salida de archivo y el porcentaje de mínima relación.	4.2.-Procesa los archivos, genera la relación sin usar TFIDF para unigramas y genera el diccionario y lo guarda en la dirección asignada.	
5.1.- Llama al programa mandado la opción 5 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia, la dirección y nombre de salida de archivo y el porcentaje de mínima relación.	5.2.-Procesa los archivos, genera la relación sin usar TFIDF para bigramas y genera el diccionario y lo guarda en la dirección asignada.	
6.1.- Llama al programa mandado la opción 6 entre los parámetros, los archivos de definiciones, acepciones y definiciones de Wikipedia, lal dirección y nombre de salida de archivo y el porcentaje de mínima relación.	6.2.-Procesa los archivos, genera la relación sin usar TFIDF para trigramas y genera el diccionario y lo guarda en la dirección asignada.	
Poscondiciones:	Haber generado un diccionario con la asignación definición->acepción Wikipedia	

Para el caso de pruebas, para ver cómo funciona el programa antes de incluirlo en el sistema se tiene otro caso de uso que maneja los mismos flujos pero ahora con un actor humano que es el tester, por lo tanto es importante recordar y tener presente que el siguiente caso de uso solo tiene como cambio el actor y la manera en que se desempeña en los nuevos flujos.

Caso de uso:	1.2 Generar asignación.

Actores:	Tester.
Propósito:	Generar diccionario con las asignaciones de definiciones a una definición y acepción de Wikipedia.
Resumen:	El tester, podrá generar un archivo que servirá de guía a la hora de anotar, asignando definiciones a acepciones obtenidas de Wikipedia.
Precondiciones:	Se deben tener 3 archivos .csv, uno de definiciones, otro de acepciones y el último de definiciones de Wikipedia.
Tipo:	Primario
Descripción:	El tester, obtendrá un archivo de apoyo para el etiquetado.

Flujo Normal		**Flujos de Excepción**
Acciones del Actor	**Respuesta del Sistema**	
1.1.- Inicia el programa Categorizador.py	1.2.- Despliega las opciones para hacerlo con o sin TFIDF y para unigrama, bigrama o trigrama.	
1.3- Selecciona la opción 1.	2.4-Procesa los archivos, genera la relación usando TFIDF para unigramas y genera el diccionario y lo guarda en la dirección asignada.	
Flujos Alternos		
2.1.- Inicia el programa Categorizador.py	2.2.- Despliega las opciones para hacerlo con o sin TFIDF y para unigrama, bigrama o trigrama.	
2.3- Selecciona la opción 2.	2.4-Procesa los archivos, genera la relación usando TFIDF para bigramas y genera el diccionario y lo guarda en la dirección asignada.	
3.1.- Inicia el programa Categorizador.py	3.2.- Despliega las opciones para hacerlo con o sin	

	TFIDF y para unigrama, bigrama o trigrama.	
3.3- Selecciona la opción 3.	3.4-Procesa los archivos, genera la relación usando TFIDF para trigramas y genera el diccionario y lo guarda en la dirección asignada.	
4.1.- Inicia el programa Categorizador.py	4.2.- Despliega las opciones para hacerlo con o sin TFIDF y para unigrama, bigrama o trigrama.	
4.3- Selecciona la opción 4.	4.4-Procesa los archivos, genera la relación sin usar TFIDF para unigramas y genera el diccionario y lo guarda en la dirección asignada.	
5.1.- Inicia el programa Categorizador.py	5.2.- Despliega las opciones para hacerlo con o sin TFIDF y para unigrama, bigrama o trigrama.	
5.3- Selecciona la opción 5.	5.4-Procesa los archivos, genera la relación sin usar TFIDF para bigramas y genera el diccionario y lo guarda en la dirección asignada.	
6.1.- Inicia el programa Categorizador.py	6.2.- Despliega las opciones para hacerlo con o sin TFIDF y para unigrama, bigrama o trigrama.	
6.3- Selecciona la opción 6.	6.4-Procesa los archivos, genera la relación sin usar TFIDF para trigramas y genera el diccionario y lo guarda en la dirección asignada.	

Poscondiciones:	Haber generado un diccionario con la asignación definición->acepción Wikipedia

Desarrollo del sistema.

Descripción de módulos.

Módulo: main().

Este módulo es el único que se modificará obligatoriamente al unirlo al sistema, ya que recibirá parámetros, los cuales son: dirección de los archivos .csv , porcentaje mínimo de relación expresado en decimales ente 0 y 1, el número de opción de las 6 disponibles y la dirección completa donde se guardará el archivo final.

Por ahora no recibe los parámetros, tiene por default las direcciones de los archivos que se utilizarán y tiene ifs anidados que deben permanecer para seleccionar la opción, ya sea recibiéndola como parámetro o preguntándola con el menú al usuario.

Las variables que este método maneja son:

listaDefiniciones .- Almacena la lista obtenida del .csv de Definiciones.

listaWiki .- Almacena la lista obtenida de las definiciones que se obtuvieron de Wikipedia.

listaAcepciones .- almacena la lista de los nombres de Acepciones obtenidas de Wikipedia.

listaSimilitud .- almacena el valor de similitud entre 0 y 1 de la asignación de relación entre Definiciones y Definiciones de Wikipedia.

listaWiki2.- almacena el ID que relaciona la definición de Wikipedia con la Acepción de lista Acepciones.

listaAcepciones2.- almacena el ID de las acepciones.

archivoDefiniciones.- Almacena la dirección del archivo de Definiciones que se utilizará para realizar el proceso. Se convertirá en parámetro del main() en la versión que se unirá al sistema.

archivoWiki.- Almacena la dirección del archivo de Definiciones de Wikipedia que se utilizará para realizar el proceso. Se convertirá en parámetro del main() en la versión que se unirá al sistema.

archivoAcepciones.- Almacena la dirección del archivo de Acepciones de Wikipedia que se utilizará para realizar el proceso. Se convertirá en parámetro del main() en la versión que se unirá al sistema.

opc.- Almacena la opción para realizar el proceso de diferentes maneras con o sin TFIDF y para bigramas, trigramas o unigramas. Será un parámetro recibido en el main() al momento de unirlo al sistema.

minimaSimilitud.- Almacena el valor mínimo de similitud que puede tener una definición para ser asignada a una acepción y definición de Wikipedia. Será un parámetro de entrada para el main() cuando se una al sistema.

vectorizador.- Es un objeto de TfidfVectorizer o CountVectorizer según sea requerido el caso, para las opciones con TFIDF y sin TFIDF. Nos servirá para poder utilizar el método fit_transform() para vectorizar la lista de Definiciones.

vectorizadorWiki.- Es un objeto de TfidfVectorizer o CountVectorizer según sea requerido el caso, para las opciones con TFIDF y sin TFIDF. Nos servirá para utilizar el método fit_transform() para vectorizar la lista de Definiciones de Wikipedia.

vectorDefiniciones.- Almacena el vector creado por fit_transform() de la lista de definiciones.

vectorWiki.- Almacena el vector creado por fit_transform() de la lista de definiciones de Wikipedia.

listaRelacion.- Almacena la relación de asignación de definiciones->definiciones de Wikipedia.

A continuación se muestra el código fuente QUE SUSTITUITRÁ al main para usuarios cuando este módulo necesite unirse a otro programa.

```
def main(archivoDefiniciones,archivoWiki,archivoAcepciones,opc,minimaSimilitud):
    """

        Aquí se definen las listas que serán las principales en el programa.

        listaefiniciones almacena las definiciones que se obtienen de Describe.

        listaWiki almacena las definiciones que provee wikipedia

        listaAcepciones almacena las ascepciones de wikipedia

        listaSimilitud almacena el porcentaje de relacion que se establece entre los vectores de wikipedia y de definiciones de Describe

        listaWiki2 almacena el id que asocia la definicion de wikipedia con la ascepcion

        listaAcepciones2 almacena el id de cada ascepcion


    """
    listaDefiniciones = []

    listaWiki = []

    listaAcepciones = []

    listaSimilitud = []

    listaWiki2 = []
```

```
listaAcepciones2 = []
"""
```

Se asignan los valores a las listas por medio de funciones que leen el archivo .csv que reciben de parametro.

```
"""
listaDefiniciones = crearListaDefiniciones(archivoDefiniciones)
listaWiki = crearListaDefiniciones(archivoWiki)
listaWiki2 = crearListaWiki2(archivoWiki)
listaAcepciones = crearListaAcepciones(archivoAcepciones)
listaAcepciones2 = crearListaAcepciones2(archivoAcepciones)
"""
```

El siguiente while junto con los print y las opciones solamente son para probar el funcionamiento del programa, deberan de eliminarse o deshabilitarse

al momento de unirlo con el sistema para el que fue creado.

```
"""
while(opc != 0):
    opc = int(input("Digite:"))
    """
```

Los siguientes if anidados son para seleccionar las direfentes acciones:

opc = 1.- Realiza la clasificación con TFIDF para unigramas.

opc = 2.- Realiza la clasificación con TFIDF para bigramas.

opc = 3.- Realiza la clasificación con TFIDF para trigramas.

opc = 4.- Realiza la clasificación sin TFIDF para unigramas.

opc = 5.- Realiza la clasificación sin TFIDF para bigramas.

opc = 6.- Realiza la clasificacion sin TFIDF para trigramas.

opc = 0.- Es sólo en casi de usarse de manera manual, sirve para terminar el programa.

else.- En caso de ser de forma manual para asegurarnos de que el usuario introduzca una opción valida.

```
        """

            #Las siguientes opciones son muy similares y solo cambia considerando lo que mencioné arriba, así que solo comentaré la primera.

    if(opc == 1):

            #Se crea un objeto de la clase TfidfVectorizer con los parametros importantes de analyzer en palabras e idioma inglés.

        vectorizador = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english')

            #Se crea una matriz con los vectores al transformar la listaDefiniciones con TFIDF y se asigna a vectorDefiniciones, así mismo se crea un vocabulario que se guarda en el objeto vectorizador.

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()

            #Se crea un objeto de la clase TfidfVectorizer con los parametros importantes de analyzer en palabras e idioma inglés, y como vocabulario el que se creó para vectorizador.

        vectorizadorWiki = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english'
,vocabulary=vectorizador.vocabulary_)

            #Se crea una matriz con los vectores al transformar la listaWiki con TFIDF y se asigna a vectorWiki.

        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

            #Se crea la lista con la relación entre las definiciones y acepciones.

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

            #Se crea una lista con el porcentaje de similitud de los vectores.

        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

            #Genera el reporte final de relacion de acpeciones y definiciones.

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki, listaWiki2, listaDefiniciones, "ResultadoTFIDFUnigrama.txt")

    elif(opc == 2):

            # Lo mismo que en el caso anterior pero con el nuevo parametro ngram_range = (2,2) para hacer el analisis con bigramas

        vectorizador = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english',ngram_range = (2,2))

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()
```

```
                # Lo mismo que en el caso anterior pero con el nuevo parametro ngram_range = (2,2) para hacer el analisis con bigramas

        vectorizadorWiki = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english', vocabulary=vectorizador.vocabulary_,ngram_range = (2,2))

        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki, listaWiki2, listaDefiniciones,  "ResultadoTFIDFBigrama.txt")

    elif(opc == 3):

                # Lo mismo que en el caso 1 pero con el nuevo parametro ngram_range = (3,3) para hacer el analisis con trigramas

        vectorizador = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english', ngram_range = (3,3))

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()

                # Lo mismo que en el caso 1 pero con el nuevo parametro ngram_range = (3,3) para hacer el analisis con trigramas

        vectorizadorWiki = TfidfVectorizer(sublinear_tf=True,max_df=0.5,analyzer='word',stop_words='english'
,vocabulary=vectorizador.vocabulary_, ngram_range = (3,3))

        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki, listaWiki2, listaDefiniciones,  "ResultadoTFIDFTrigrama.txt")

    elif(opc == 4):

                #Lo mismo que la opcion 1 pero ahora se usa CountVectorizer sustituyendo a TfidfVectorizer, se hará el analisis sin TFIDF

        vectorizador = CountVectorizer(max_df=0.5,analyzer='word',stop_words='english')

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()

        vectorizadorWiki = CountVectorizer(max_df=0.5,analyzer='word',stop_words='english',vocabulary=vectorizador.vocabulary_)
```

```
        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki,
listaWiki2, listaDefiniciones,  "ResultadoUnigrama.txt")

    elif(opc == 5):

                #Lo mismo que en la opcion 4 pero con el parametro ngram_range =
(2,2) para hacer el analisis sin TFIDF y para bigramas

        vectorizador                                                    =
CountVectorizer(max_df=0.5,analyzer='word',stop_words='english',ngram_range=(2,2))

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()

        #Lo mismo que en la opcion 4 pero con el parametro ngram_range = (2,2) para
hacer el analisis sin TFIDF y para bigramas

        vectorizadorWiki                                                =
CountVectorizer(max_df=0.5,analyzer='word',stop_words='english',vocabulary=vectorizad
or.vocabulary_,ngram_range=(2,2))

        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki,
listaWiki2, listaDefiniciones,  "ResultadoBigrama.txt")

    elif(opc == 6):

                #Lo mismo que en la opcion 4 pero con el parametro ngram_range =
(3,3s) para hacer el analisis sin TFIDF y para trigramas

        vectorizador                                                    =
CountVectorizer(max_df=0.5,analyzer='word',stop_words='english',ngram_range=(3,3))

        vectorDefiniciones = vectorizador.fit_transform(listaDefiniciones).toarray()

                #Lo mismo que en la opcion 4 pero con el parametro ngram_range =
(3,3) para hacer el analisis sin TFIDF y para trigramas

        vectorizadorWiki                                                =
CountVectorizer(max_df=0.5,analyzer='word',stop_words='english',vocabulary=vectorizad
or.vocabulary_,ngram_range=(3,3))

        vectorWiki = vectorizadorWiki.fit_transform(listaWiki).toarray()

        listaRelacion = generarRelacion(vectorDefiniciones, vectorWiki,minimaSimilitud)
```

```
        listaSimilitud = generarSimilitud(vectorDefiniciones, vectorWiki)

        generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2, listaWiki,
listaWiki2, listaDefiniciones,  "ResultadoTrigrama.txt")

    elif(opc == 0):

        print "Termina el programa"

    else:

        print "Error, elija una opcion valida"
```

Método: crearListaDefiniciones(archivoCsv)

Recibe la ruta del archivo .csv, archivoCsv

Este método lee el archivo .csv que recibe como parámetro, y almacena en una lista las definiciones, retornará la lista creada. Solamente guarda el dato que esté en el campo llamado 'definicion' del archivo. No requiere ser modificado a menos que se requiera obtener información de un archivo que no sea .csv o que se requiera guardar en la lista otro campo diferente.

Variables:

- listaDefiniciones.- Es la lista donde se guardarán las definiciones obtenidas del .csv.
- lecturaDefiniciones.-Es el objeto de csv que nos servirá para ir leyendo renglón por renglón el archivoCsv.

Método: generarRelacion(vectorDefiniciones, vectorWiki, minimaSimilitud)

Recibe el vectorDefinciones, vectorWiki y miminaSimilitud.

Este método está encargado de generar la relación definición->definiciónWiki por medio de la implementación del método cosine_similarity(), el cual recibe como parámetro dos vectores los cuales va a calcular la similitud coseno. Únicamente genera la relación y la guarda en una lista, la cual es retornada al finalizar la función. En caso de no cumplir con el mínimo señalado en el parámetro de minimaSimilitud, se asigna el valor de 0 para manejarlo como no informátivo.

Variables:

- listaRelacion.- Almacena la lista de relación, donde la posición significa la posición en la listaDefinicion de la definición y el elemento de esa posición significa la posición de la listaWiki con el cual se relaciona.
- i.- contador que ayuda a avanzar de definición en definición.
- j.- contador que ayuda a avanzar de definición de Wikipedia en definición de Wikipedia.
- valorMasAlto.- Almacena el valor más alto, ya que el que tenga valor más alto es el que se relacionará.
- Result.- va guardando el resultado actual de la comparación de vectorDefinicion con vectorWiki por medio de la implementación de cosine_similarity().

Método: generarSimilitud(vectorDefiniciones, vectorWiki)

Este método realiza una operación similar al método generarRelacion() pero este solo recibe de parámetros el vectorDefiniciones y vectorWiki, y tiene como objetivo generar una lista guardando el valor máximo calculado por cosine_similarity() ya que esto lo ocuparemos en el método generarCsv().

Variables:

- listaSimilitud.- Almacena la lista de valor entre 0 y 1 de similitud entre vectores relacionados, donde la posición significa la posición en la listaDefinicion de la definición y el elemento de esa posición significa el valor de similitud obtenido de cosine_similarity().
- i.- contador que ayuda a avanzar de definición en definición.
- j.- contador que ayuda a avanzar de definición de Wikipedia en definición de Wikipedia.
- valorMasAlto.- Almacena el valor más alto, ya que el que tenga valor más alto es el que se relacionará.
- Result.- va guardando el resultado actual de la comparación de vectorDefinicion con vectorWiki por medio de la implementación de cosine_similarity().

Método: crearListaAcepciones(archivoCsv)

Recibe el archivo .csv que contiene las acepciones de Wikipedia, tiene como objetivo obtener los nombres de las acepciones, para poder utilizarlo en generarCsv(). Regresa una lista con los nombres de las acepciones.

Variables:

- listaAcecpciones.- Es la lista de nombres de acepciones donde se guardarán estos mismos, está lista es la que se regresa al terminar el programa.
- lectura.- Es el objeto de csv que nos servirá para ir leyendo renglón por renglón el archivoCsv.

Método: crearListaAcepciones2(archivoCsv)

Recibe el archivo .csv que contiene las acepciones de Wikipedia, tiene como objetivo obtener los ID de las acepciones, para poder utilizarlo en generarCsv(). Regresa una lista con los ID de las acepciones.

Variables:

- listaAcecpciones.- Es la lista de ID de acepciones donde se guardarán estos mismos, está lista es la que se regresa al terminar el programa.
- lectura.- Es el objeto de csv que nos servirá para ir leyendo renglón por renglón el archivoCsv.

Método: crearListaWiki2(archivoCsv)

Recibe el archivo .csv que contiene las definiciones de Wikipedia, tiene como objetivo obtener los ID de las acepciones ligadas con las definiciones de Wikipedia del archivo, para

poder utilizarlo en generarCsv(). Regresa una lista con los ID de las acepciones de cada definición de Wikipedia.

Variables:

- listaWiki2.- Es la lista de ID de acepciones de cada definición de Wikipedia donde se guardarán estos mismos, está lista es la que se regresa al terminar el programa.
- lectura.- Es el objeto de csv que nos servirá para ir leyendo renglón por renglón el archivoCsv.

Método generarDic(listaRelacion, listaSimilitud, listaAcepciones, listaAcepciones2,listaWiki, listaWiki2, listaDefiniciones, nombreArchivo)

Recibe como parámetros listaRelacion que es la lista donde se guarda la relación entre definición y definiciones de Wikipedia, con las listaDefiniciones y listaWiki, listaSimilitud que es la lista con los valores resultado de aplicar la similitud coseno, listaAcepciones que contiene el nombre de las acepciones, listaAcepciones2 que contiene el id de las acepciones, listaWiki que contiene las definiciones obtenidas de Wikipedia, listaWiki2 que contiene los ID que identifican a que acepción pertenece la definición de Wikipedia, listaDefiniciones que contiene las definiciones y por último el nombre del archivo que contiene la ruta y el nombre donde se desea que se guarde el archivo.

Variables:

- c.- objeto de writer para poder escribir en el archivo.
- cont.- Sirve para ir contando las vueltas dentro del for de acepciones, se utiliza porque varias listas ocupan el mismo orden y así con esta variable se pueden ir recorriendo las demás listas sin generar más fors, ya que esto alentaría el programa.
- contWiki.- Sirve para ir contando las vueltas que genera el for de listaWiki2, se utiliza porque la listaWiki ocupa el mismo orden y tamaño y así con esta variable se pueden ir recorriendo la lista sin generar más fors, ya que esto alentaría el programa.
- cont2.- Sirve para ir contando las vueltas dentro del for de listaRelacion, se utiliza porque listaDefiniciones y listaSimilitud ocupan el mismo orden y tienen el mismo tamaño y así con esta variable se pueden ir recorriendo las demás listas sin generar más fors, ya que esto alentaría el programa.
- matriz.- Es una lista de listas que sirve para ir guardando los valores de listaSimilitud y las definiciones, claro en una lista dentro de esta lista se guardan las definiciones y los valores de similitud que se corresponden entre sí. Esto es para poder acomodar de mayor a menor y así generar el Ranking.
- Lista.- Lista que guarda un valor de similitud con su definición correspondiente. Sirve para guardar temporalmente estos dos valores y posteriormente agregarlos a la matriz.
- contMatriz.- Sirve para ir contando las vueltas dentro del for de matriz, se utiliza porque se requiere acomodar el ranking así que es para numerar los mismos.
- contNoInformativo.- Sirve para manejar dentro del for de listaDefiniciones una manera de que se pueda visualizar contenido de la listaRelacion sin necesidad de loops que no son requeridos.

Requerimientos de los archivos.

Los archivos deben de cumplir ciertas reglas para que pueda ser manejado adecuadamente por el programa, aquí listaré lo que cada archivo de entrada debe contener.

Archivo de Definiciones:

Debe ser un archivo en formato .csv de dos campos, el primero no contiene nada y lleva una etiqueta ‘nada’ y el segundo campo lleva las definiciones y una etiqueta llamada ‘definicion’. Estas etiquetas van hasta arriba dentro del documento separados por comas y sin espacios. Los valores de definiciones van entre comillas y después de una coma, pondré un ejemplo gráfico para que sea más claro. Revisar Ilustración 1.

Es el principal archivo que recibe el módulo ya que es la información que se requiere procesar.

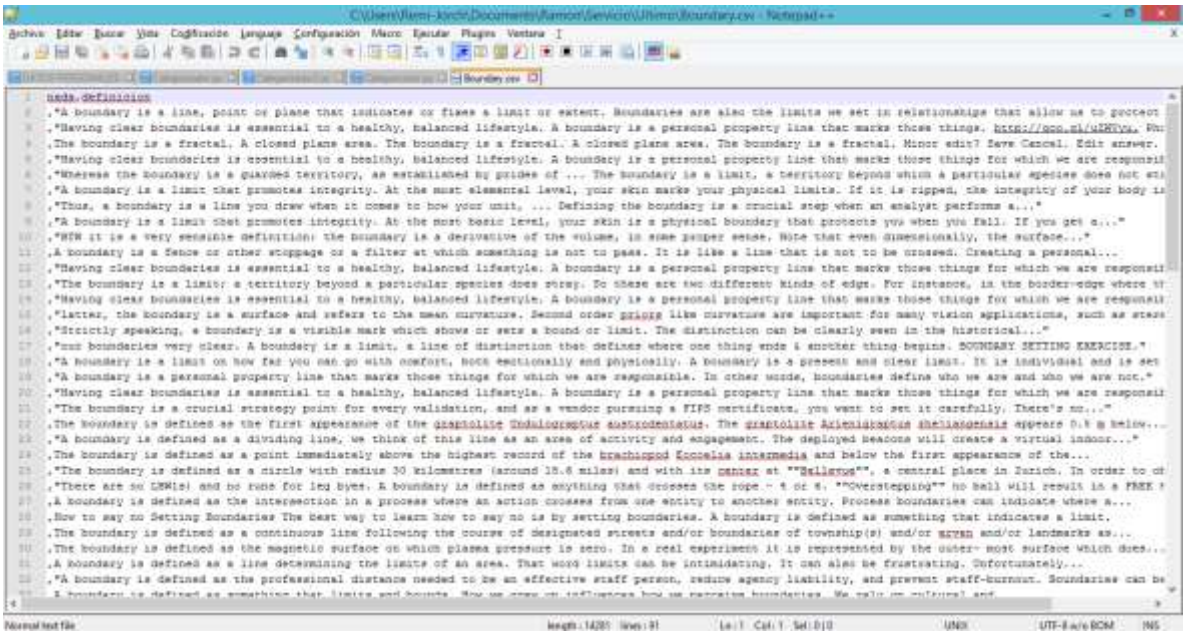

Ilustración 1Ejemplo de Archivo de Definiciones.

Archivo de Artículos de Wikipedia.

Este archivo debe ser un .csv de dos campos, el primero contiene el ID de la acepción a la que pertenece esa definición y lleva una etiqueta ‘id’ y el segundo campo lleva las definiciones y una etiqueta llamada ‘definicion’. Estas etiquetas van hasta arriba dentro del documento separados por comas y sin espacios. Los valores de Id van uno por línea, seguidos de una coma ‘,’ los valores de definiciones van uno por línea entre comillas y después de la coma. Es importante que las definiciones tengan en promedio 40 palabras ya que sino se vuelve más difícil calcular la similitud coseno, pondré un ejemplo gráfico para que sea más claro. Revisar Ilustración 2.

Es importante ya que son los artículos con los que serán comparadas las definiciones.

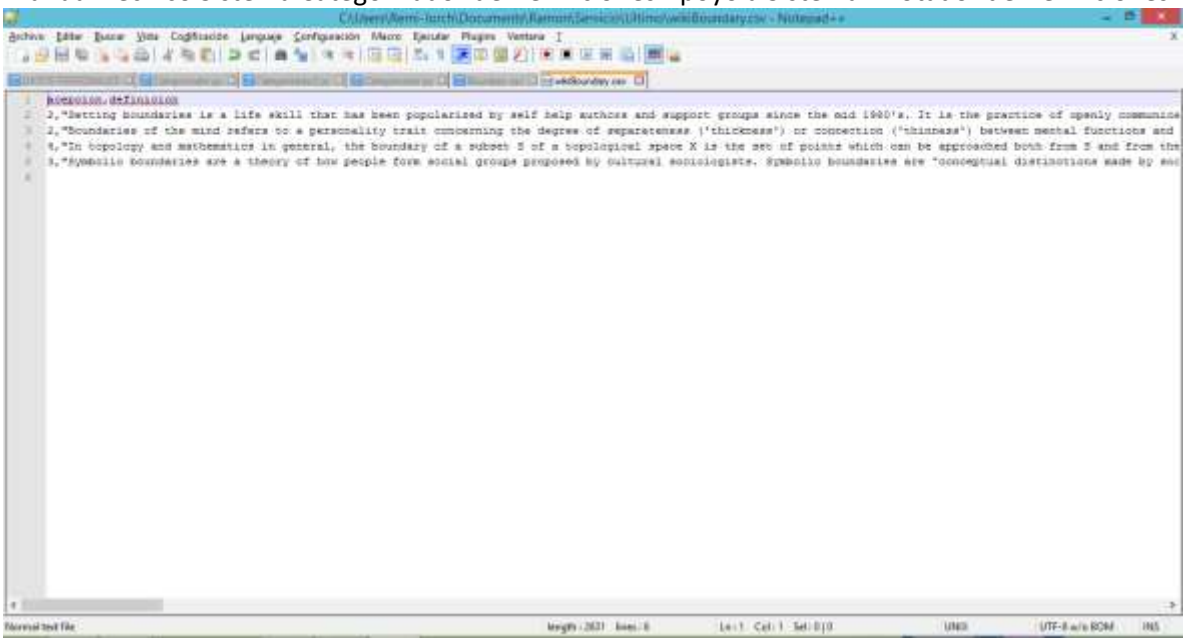

Ilustración 2Ejemplo de Archivo de Artículos.

Archivo de Acepciones.

Este archivo deberá ser un .csv donde se almacenará el nombre de la acepción y la definición, esto debe de ser relacionado con las definiciones que se obtuvieron de Wikipedia y las definiciones a procesar, esto significa que deben ser referentes al mismo término. El .csv consiste de dos campos, el primero consiste en el ID de la acepción en la cual recomiendo que sea secuencial, en el segundo se encuentra el nombre de la acepción entre comillas. Estas etiquetas van hasta arriba dentro del documento separados por comas y sin espacios. Los valores de Id van uno por línea, seguidos de una coma ‘,’ los valores de nombres de acepciones van entre comillas y después de la coma. Revisar Ilustración 3.

Es importante ya que es el archivo que contiene los artículos con los que serán comparadas las definiciones.

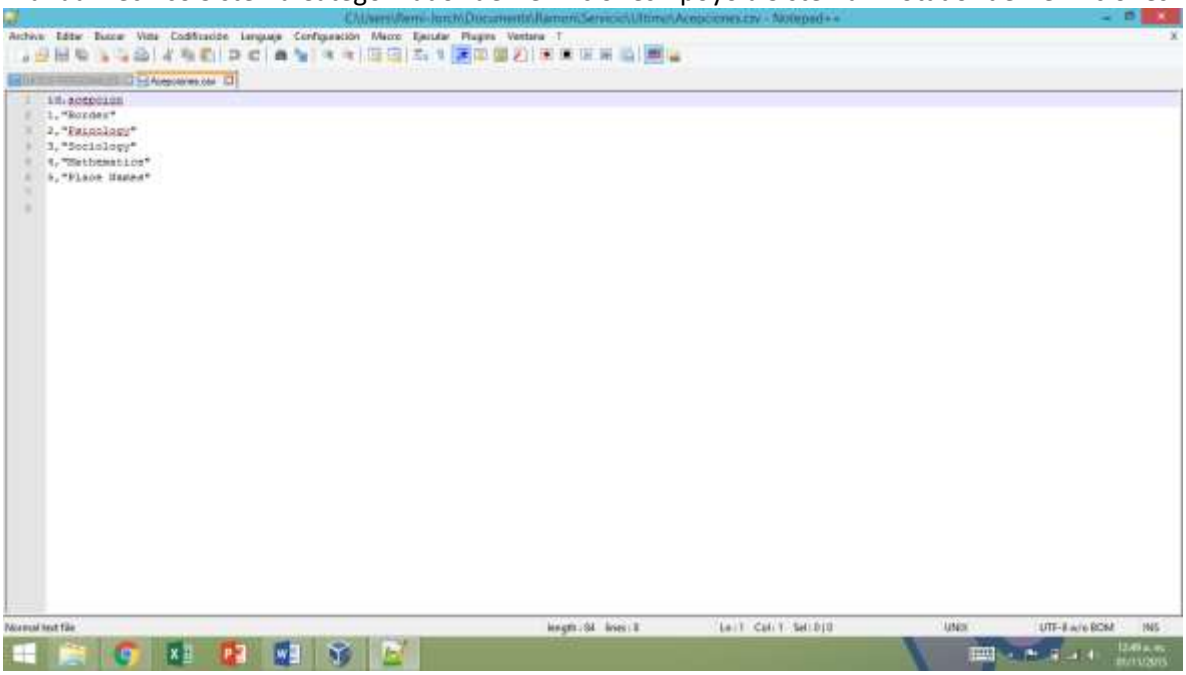

Ilustración 3Ejemplo de archivo de Acepciones

Archivo de Salida.

Lleva el nombre que indica si se utilizó TFIDF o no y de que forma fue el análisis, si por unigrama, bigrama o trigrama, es un archivo en formato .txt que contiene el diccionario de la asociación entre definiciones y acepciones, este se almacena en la misma carpeta donde se encuentra el archivo fuente "Categorizador.py" El nombre de salida ejemplo si utilizamos TFIDF para unigramas es: "TFIDFUnigrama.txt". En la ilustración 4 se puede observar un ejemplo de este archivo de salida.

La organización del diccionario es:

{ascpcion:{<nombreDeAcepcion1>:{Articulo1:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_3:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo2:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo3:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, <nombreDeAcepcion2>:{Articulo1:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_3:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo2:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo3:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, <nombreDeAcepcionN>:{Articulo1:"Artículo de

Manual Técnico Sistema Categorizador de Definiciones Apoyo a Sistema Anotador de Definiciones Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_3:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo2:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, {Articulo3:"Artículo de Wikipedia",Definiciones:{def_1:(<valorDeSimilitud1>,"Definición1"), def_2:(<valorDeSimilitud2>,"Definición2"), def_n:(<valorDeSimilitudN>,"DefiniciónN")}}, {NoInformativos:{definiciones:["Definición1","Definición2","DefiniciónN"]}}}

Donde:

Acepciones = diccionario de acepciones principal, donde se guardan todo el analisis.

nombreDeAcepcionN = diccionario creado sobre el enésimo nombre de las acepciones para un término, los artículos que perteneces a este y las definiciones que se asignaron a este.

ArticuloN = Enésimo artículo obtenido de wikipedia.

Definiciones = diccionario con las definiciones y su grado de similitud, asignadas a un id de definiciones.

def_N = ID de definiciones.

valorDeSimilitudN = valor de similitud de la enésima definición.

DefinicionN = Enésima Definición.

NoInformativos = Almacena la lista de las definiciones no informativas.

Ilustración 4Ejemplo de archivo de salida (Diccionario)