Universidad Autónoma Metropolitana Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Reporte final de Proyecto Terminal

Matrices de Asociación para un Glosario de Términos Informáticos

Sánchez González Silvia 205205542

Trimestre 13-P

Asesor: Oscar Herrera Alcántara, Profesor Asociado.
Departamento de Sistemas

Contenido

Resumen

El presente documento describe el diseño y la construcción de las matrices de asociación para un glosario de términos informáticos, así mismo prueba que son una herramienta para la comparación entre palabras clave y documentos.

En un intento por dotar a un equipo de cómputo de la capacidad de comparar documentos se requiere proveer conocimiento al, así llamado, Sistema Clasificador de Documentos Texto (SCDT), es por ello que se creó un conjunto de matrices de similitud entre palabras del lenguaje natural, y en particular de términos informáticos, divididas en 4 categorías: Ingeniería de Software, Algoritmos, Mecatrónica y Comunicaciones / Sistemas digitales.

El SCDT puede "saber" qué tan similares son las palabras, de acuerdo a una relación semántica (el significado en un contexto dado). Es fundamental contar con una jerarquía semántica de términos de un glosario de informática y un conjunto de estructuras de datos (matriz) de asociaciones semánticas entre los términos informáticos para ampliar la posibilidad de que las palabras clave y los documentos sean considerados similares de acuerdo a los términos que usan y no solo en base al título.

Por medio de las matrices de asociación se cumple la posibilidad de encontrar que tan similares son las palabras clave y los documentos.

A continuación se muestra el diseño y la construcción de las matrices de asociación y el cumplimiento de cada uno de los objetivos, los cuales son los siguientes:

Objetivos:

Objetivo General:

Diseñar, implementar y probar matrices de asociación para un glosario de términos informáticos.

Objetivos Específicos:

1. Definir una jerarquía semántica de términos de un glosario de informática.
2. Diseñar y construir una matriz de asociación para los términos de cada jerarquía semántica.
3. Diseñar un módulo para aplicar la jerarquía semántica y las matrices de asociación para determinar vecinos similares.
4. Implementar un programa que use vecinos de palabras clave en la clasificación de documentos de texto de acuerdo a su contenido temático.

Desarrollo

Recopilación de las palabras

Se comprueba el cumplimiento del objetivo específico N. 1 a través de la siguiente explicación: Para poder diseñar las matrices de asociación fue necesario definir un número de palabras de términos informáticos, en este caso el número de palabras se delimito en 160, las cuales fueron tomadas de un diccionario de términos informáticos.

```
1.  Matemáticas
2.  Criptografía
3.  Aplicación
4.  Binario
.
.
.
.
160. Microprocesador
```

Figura 1. Simula las 160 palabras recopiladas

Clasificación de palabras

Una vez obtenidas las 160 palabras con términos informáticos, fueron clasificadas en cuatro categorías: Ingeniería de Software (ISW), Algoritmos (ALG), Comunicaciones/Sistemas Digitales (COM/SD), Mecatronica (MEC), como se muestra en la figura 2, la clasificación se ideó con el fin de catalogar las palabras en un área de la informática para no hacer comparaciones innecesarias, ya que existen términos que son considerados dentro del contexto informático pero la relación entre ellos no es relevante, con esta misma ideología en cada una de las categorías antes mencionadas se realizaron subcategorias, es decir que

dentro de las categorías se hicieron grupos de palabras que llevaran una mayor relación, nuevamente con el propósito de no realizar comparaciones innecesarias. Véase la figura 3.

Ingeniería de Software (ISW)	Algoritmos (ALG)	Comunicaciones/Sistemas Digitales (COM/SD)	Mecatronica (MEC)
Abstraccion	3D	Analogico	Actuadores
Almacenamiento	AGP	Arquitectura	Adaptacion
Aplicación	Algoritmo	Banda	Agente
Archivo	Arbol	Binario	Artificial
Atributo	Automata	Bios	Automatizacion
Auditoria	Cifrado	Bluetooth	Biomecanica
Base	Coloracion	Bus	Bionica
Business	Combinacion	Byte	Cibernetica
Calidad	Compilador	Computadora	Circuitos
Clase	Correctitud	Circuito	Cognicion
Dato	Criptografia	Procesador	Conocimiento

Figura 2. Primera Clasificación de las 160 palabras

Bloque 1	Bloque 2	Bloque 3	Bloque 4
Interfaz	Dato	Aplicación	Abstracción
Sistema	Patrón	Pagina	Clase
Software	Base	Portal	Encapsulamiento
Metodología	Atributo	Web	Estructura
Requerimiento	Informática	Buscador	herencia
Calidad	Filtrado	Cookies	Objeto
Gestión	Agrupar	Browser	Modulo
Auditoría	Minería	Link	
Almacenamiento	Business	Hosting	
Vida	Weka		
Integridad			
Desarrollo			
Manejador			
Información			

Figura 3. Ejemplo de subclasificación de la categoría Ingeniería de Software (ISW)

Por lo tanto el objetivo específico N. 2 queda comprobado en base a lo anterior.

Cálculo de Porcentaje de Similitud entre palabras

Una vez establecidas las subclasificaciones de las palabras, se definió un porcentaje de similitud de cada palabra contra el resto de las palabras del bloque en cuestión, el rango del porcentaje de similitud abarca desde el cero, el cual representa que no existe ninguna similitud entre las palabras, hasta el uno, el se trata de la misma palabra, véase la figura 4.

	Dato	Patrón	Base	Atributo	Informática	Filtrado	Agrupar	Minería	Business	Weka
Dato	1	0,2	0,9	0,7	0,4	0,8	0,8	0,8	0,8	0,4
Patrón	0,2	1	0,2	0,09	0,04	0,5	0,5	0,7	0,9	0,02
Base	0,9	0,2	1	0,7	0,8	0,3	0,1	0,7	0,6	0,2
Atributo	0,7	0,09	0,7	1	0,02	0,03	0,4	0,03	0,01	0,01
Informática	0,4	0,04	0,8	0,02	1	0,01	0,03	0,2	0,5	0,4
Filtrado	0,8	0,5	0,3	0,03	0,01	1	0,7	0,4	0,04	0,4
Agrupar	0,8	0,5	0,1	0,4	0,03	0,7	1	0,7	0,1	0,7
Minería	0,8	0,7	0,7	0,03	0,2	0,4	0,7	1	0,2	0,9
Business	0,9	0,9	0,6	0,01	0,5	0,04	0,1	0,2	1	0,6
Weka	0,7	0,02	0,2	0,01	0,4	0,4	0,7	0,9	0,6	1

Figura 4. Ejemplo de porcentaje de similitud entre palabras

Después de calcular los porcentajes de similitud de cada palabra se definieron sus vecinos (V1 y V2), los cuales se obtuvieron por medio de los porcentajes más altos de similitud de cada una de las palabras, por ejemplo en la tabla de la figura 4 la palabra DATO, la cual es comparada con las 10 palabras de la tabla, al observar los porcentajes se encuentra que la cantidad más alta de dicha comparación es de 0.9, mediante las palabras BASE y BUSINESS, por lo tanto esta dos palabras se consideran los dos vecinos (V1 y V2) de la palabra DATO.

	Dato
Dato	1
Patrón	0,2
Base	0,9
Atributo	0,7
Informática	0,4
Filtrado	0,8
Agrupar	0,8
Minería	0,8
Business	0,9
Weka	0,7

Figura 5. Vecinos (V1 y V2)

En este punto queda comprobado el objetivo específico N. 3. Es por ello que se procede a desarrollar un SCDT, el cual pueda demostrar que las matrices de asociación son una herramienta para encontrar documentos similares.

Sistema Clasificador de Documentos de Texto

Para realizar la interfaz y el desarrollo del SCDT se utilizo el lenguaje PHP/HTML y para la construcción de las matrices se utilizó MySQL.

Interfaz del SCDT, esta área del sistema es la que recibe las palabras clave del documento, del cual se desea saber que otros documentos son similares a este.

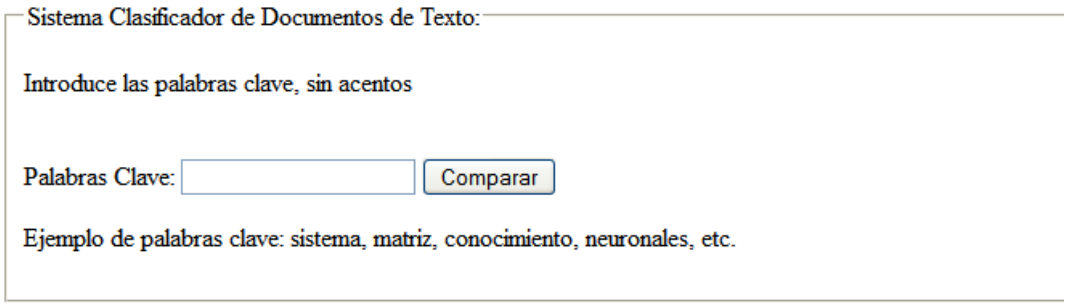

Figura 6. Sistema Clasificador de Documentos de Texto

Una vez que las palabras clave son introducidas, el sistema las depura, por ejemplo el usuario ingresa las siguientes palabras clave:

"Sistema de información"

El sistema recibe las palabras clave y mediante un query a la base de datos compara cada palabra clave con las 160 palabras que se encuentran guardas.

```
//Recibe Título
$titulo = strip_tags($_POST['titulo']);

//Consulta la base de datos para extraer el ID y el NOMBRE de la palabra
$sql="SELECT id, nombre_palabra FROM palabra";
$datos=mysql_query($sql,$db);

// J se usa para saber si existe por lo menos una palabra del título
// dentro de las 160 palabras registradas
$j=0;

//Cuenta el numero de palabras del titulo
$cuenta = count($palabras=explode(" ", $titulo));

//En esta área del código se compara cada palabra del código con las
//160 palabras
for($i=0; $i<$cuenta; $i++)
{
      while ($row=mysql_fetch_array($datos))
      {
            $nombre=$row['nombre_palabra'];
            $resultado = strcasecmp($palabras [ $i ],"$nombre");
            if($resultado == 0)
            {$j++;}
      }mysql_data_seek($datos,0);
}
```

Al terminar la ejecución de esta parte del código se sabe si existe por lo menos una palabra clave dentro de las 160 palabras contenidas en la base de datos y que palabra o palabras son, ya que se extrae el Id y el Nombre de la misma.

Con esta información podemos determinar si el sistema procede a buscar los vecinos de las palabras o si se sale al no encontrar ninguna palabra, con la cual pueda buscar vecinos.

```
//Si J es diferente de cero es que si existe por lo menos una palabra
//del titulo que se encuentre dentro de las 160 palabras registradas
if($j!=0)
{
```

Pensemos que j es diferente de cero, es decir que se cumple la sentencia y por lo tanto podemos buscar vecinos. Entonces se ejecutar el código siguiente, el cual nos ayudará a encontrar los vecinos de las palabras mediante un query y estos serán guardados en un vector junto con la palabra.

```
//La variable K sirve como indice del vector que guarda las palabras
//y los vecinos
$k=0;

//En esta parte del código se busca a los vecinos de las palabras
for($i=0; $i<$cuenta; $i++)
{
      while ($row=mysql_fetch_array($datos))
      {
        $numero=$row['id'];
        $nombre=$row['nombre_palabra'];

        //Aqui se compara las palabras (una de las palabras del titulo se compara con las 160)
        $resultado = strcasecmp($palabras [ $i ],"$nombre");
        //Si el resultado es cero quiere decir que son iguales
        if($resultado == 0)
        {
            //En esta consulta se encuentra los vecinos (V1, V2)
            $consulta = "SELECT id_palabra, id_nombre_v1, id_nombre_v2 FROM diccionario WHERE id_palabra = $numero";
            $vecinos=mysql_query($consulta,$db);
            $num=mysql_fetch_array($vecinos);
            $elemento=$num['id_palabra'];$vecino=$num['id_nombre_v1'];$vecina=$num['id_nombre_v2'];
            //Se guardan los vecinos en un vector
            $vec[$k]= "$elemento";
            $vec[$k=$k+1]= "$vecino";
            $vec[$k=$k+1]= "$vecina";
            $k=$k+1;

        }
       }
      }
      //Una vez que termina de buscar los vecinos de una palabra el contador del
      //arreglo que los contiene vuelve a cero para poder hacer otra comparación
      mysql_data_seek($datos,0);
}
```

Hasta este punto del código los vecinos y las palabras han sido guardados en un vector para ser comparados con los documentos y así encontrar el número de palabras contenidas en cada documento.

```
//En esta área las palabras y sus vecinos serán comparadas con las palabras de los
//documentos y se sacara el numerod de palabras que se encuentren en cada documento
for($r=1; $r<=38; $r++)
{
    $div=0;
    $cuentas=0;
```

```
for($q=0; $q<$k; $q++)
{
    //Aqui se hace la consulta a la base datos definiendo el numero de proyecto y tomando el contenido del vector de los
    //vecinos y palabras
    $pt="SELECT num_palabras, id_proyecto, id_palabra FROM contenido_pts WHERE id_palabra=$vec[$q] AND id_proyecto=$r";
    $proyectos=mysql_query($pt,$db);
    $pts=mysql_fetch_array($proyectos);

        $proyecto=$pts['id_proyecto'];
        $numpalabras=$pts['num_palabras'];
        if($proyecto==$r)
        {
            //Aqui se cuenta cuantas palabras se encuentra en el documento
            $cuentas++;
            $div=$numpalabras;
        }
}
```

En el momento que se termina un recorrido del ciclo FOR se puede saber el número de palabras que están contenidas en cada documento o en su defecto no tener ninguna palabra en común, es por ello que se define el siguiente código:

```
    //Se podria dar el caso de que no hubiera palabras ni vecinos en un documento
    //es por ello que si no existen los valores se guardan en ceros
    if($cuentas!=0 && $div!=0)
    //Pero cuando si existen palabras se hace una operacion para saber el porcentaje de similitud
    {$numpro[$r]=$aux[$r]=$cuentas/$div;}
    else {$numpro[$r]=$aux[$r]=0;}
}
```

Si el documento contiene una o más palabras se procede a realizar una operación, la cual consiste en lo siguiente: en dividir el número de palabras que se encuentran dentro del documento entre el número total de palabras del mismo y posteriormente se guardan en dos vectores para las siguientes ejecuciones. En caso dado que no haya ninguna palabra se guarda un cero en los dos vectores.

En seguida los porcentajes son ordenados de manera descendente, con el siguiente código:

```
//En esta parte los porcentajes de similitud son ordenados de manera descendente
rsort($numpro, SORT_NUMERIC);
foreach($numpro as $result)
{
    for($h=1; $h<=38; $h++)
    {
        if($aux[$h]==$result)
        {
            $duplicados[$h]=$h;
        }
    }
}
```

En la siguiente parte del código mediante un query se buscará el título, el autor y el porcentaje de similitud entre las palabras clave y los documentos con mayor similitud y posteriormente serán mostrados en pantalla.

```
//En esta parte del código se muestran los primeros 10 titulos de los documentos
//con mayor similitud al titulo ingresado
$d=0;
$unicos=array_unique($duplicados);
echo "Las propuestas similares a tu proyecto terminal ".$titulo." son: <br/><br/>";
foreach($unicos as $uno)
{
    if ($d<=9)
    {
        //Se hace una consulta a la tabla proyectos para sacar los titulos y autores de los proyectos similares al titulo
        $revisar = "SELECT id, titulo_pt, autor FROM proyectos WHERE id = $uno";
        $nomproyecto = mysql_query($revisar,$db);
        $nombres=mysql_fetch_array($nomproyecto);
        $tit=$nombres['titulo_pt'];
        $autor=$nombres['autor'];
        print $tit." "." /Autor: ".$autor."<br />";
        $d++;
    }
}
}//Fin del if inicial
```

En la figura 7 se muestra que el sistema es capaz de proyecta el título, el autor y el porcentaje de similitud, así mismo cuenta con un botón de regreso por si se desea realizar otra búsqueda.

Las propuestas similares a tu proyecto terminal matriz sistema neuronales son:

Titulo: Reconocimiento de Huellas Digitales /Autor: Garcia Garcia Elizabeth Adriana /Porcentaje 0.44444444444444
Titulo: Matrices de Asociacion para un Glosario de Terminos Informaticos /Autor: Sanchez Gonzalez Silvia /Porcentaje 0.3125

Regresar

Figura 7. Título, autor y porcentaje de similitud

Recordando la sentencia IF donde el sistema define si existen palabras dentro de la base de datos que coincidan con el titulo introducido, si la sentencia no se cumple simplemente el código se salta todas la sentencias antes explicadas y muestra en pantalla por medio del código siguiente:

```
//Esta sentencia nos muestra que si no se encuentra una palabra dentro del glosario de las
//160 palabras, manda un mensaje de volverlo a intenta con otro titulo
else
{echo "Ninguna palabra del titulo introducido se encuentra, intenta con otro por favor <br/>";}
echo "<br/>";
echo "<input type='button' value='Regresar' onclick='history.go(-1)'>";
```

Ninguna palabra clave introducida se encuentra, intenta con otras por favor

Regresar

Figura 8. Palabra Clave no encontrada

La descripción anterior comprueba el objetivo específico N.4, a través de las imágenes se muestra los resultados y en conjunto la descripción comprueba que se han cumplido tanto el objetivo general como cada objetivo específico.

Conclusión

El objetivo general ha sido cumplido, así mismo también se demostró que los objetivos fueron alcanzados, esto se comprueba mediante el SCDT, el cual es capaz de definir que documentos son similares a las palabras clave por medio de las matrices de asociación, por lo tanto las matrices de asociación son una herramienta para comparación de documentos y palabras calve.

Una posible limitante en la comparación entre palabras clave y documentos, es que el SCDT solo cuenta con 160 palabras y con 2 documentos depurados, por lo tanto habrá palabras dentro del contexto informático que no serán halladas en el SCDT.