LSI-02-6

FUNCIONES DE COMPARACIÓN DE CARACTERES PARA APNM: LA DISTANCIA *DEA*

Rafael Camps Paré
Departamento LSI
Universitat Politècnica de Catalunya
Barcelona

1.- Generalidades
    1.1.- Introducción
    1.2.- Funciones cuantitativas y booleanas. Funciones de distancia
    1.3.- Distancias métricas

2.- Las distancias de edición
    2.1.- Distancia de edición y tipos de operaciones
    2.2.- Costes y distancia generalizada
    2.3.- Determinación de los costes DEA
        2.3.1.- Introducción
        2.3.2.- Desventajas simétricas
        2.3.3.- Posiciones de las operaciones
    2.4.- Umbrales y longitudes

3.- Otras distancias
    3.1.- Distancias basadas en *n-grama*s
    3.2.- Alberga
    3.3.- Jaro (US Census Bureau)
    3.4.- Zobel: EDITEX

4.- Evaluación y comparación de las funciones de distancia
    4.1.- Metodología y corpus
    4.2.- Sobreidentificación, subidentificación y umbrales: Gráfico SUSO
    4.3.- Gráfico P/R
    4.4.- Eficacias E y J
    4.5.- Conclusiones

5.- Salida y factor β: Pruebas de búsqueda con DEA

1.- Generalidades

1.1- Introducción

En el campo del APNM (Approximate Personal Name Matching) se utilizan tradicionalmente criterios de semejanza basados en codificaciones fonéticas y, con menos frecuencia, funciones de comparación de caracteres. En este trabajo vamos a tratar sobre funciones de comparación que explotan las coincidencias o diferencias de caracteres entre las dos cadenas a comparar.

Vemos el antropónimo como una cadena de caracteres. Se determina una medida de distancia o "des-semejanza", $\delta(x, y)$ comparando de alguna manera los caracteres de una cadena con los de la otra. La distancia más conocida es la distancia de edición, *edit-distance*. Hay muchas variantes de la distancia de edición, siendo la más popular la que se calcula como el número mínimo de operaciones de inserción `I` , supresión `D` y substitución `S` , necesarias para que las dos cadenas sean iguales. A esta distancia la llamaremos (como en [Nav01]) *distancia de edición simple* o *distancia simple*.

La medida de semejanza propuesta en [Fau64] nos servirá como un primer ejemplo para introducir el tema de las funciones de comparación de caracteres. Su medida de semejanza consiste en la suma de tres factores: posicional, ordinal y material.

- 2 coincidencias por posición (la *G* y la *C*) — Factor posicional
- 3 consonantes en el mismo orden (*G,R,C*) y 2 vocales también en el mismo orden (*I,A*) — Factor ordinal
- 5 símbolos que están en las dos cadenas (*A,C,G,I,R*) — Factor material

Comparando *GARCIA* con *GRICA* encontramos:

De ahí se obtiene un valor de la semejanza igual a 2+(3+2)+5 = 12

Imaginemos ahora otra medida de semejanza consistente en el número de bigramas que coinciden en las dos cadenas. En nuestra pareja *GARCIA* / *GRICA* la semejanza valdría cero, ya que *GARCIA* consta de *GA, AR, RC, CI, IA,* y en cambio *GRICA* consta de otros bigramas diferentes; *GE, RI, IC, CA.*

La medida más extendida de "des-semejanza", o distancia, es la distancia simple. En nuestro ejemplo será:

$$\delta(GARCIA\ ,\ GRICA) = 3$$

ya que se puede transformar un antropónimo en el otro, por medio de la inserción (o supresión) de una *A* y dos substituciones; la de *I* y la de *C*.

Todas las funciones de ese tipo, basadas en comparaciones de caracteres, tienen en cuenta esencialmente los aspectos físicos, por lo que parecen más gráficas que fonéticas. Hay una notable correlación entre las deformaciones físicas y las fonéticas. De la misma manera que las codificaciones fonéticas resuelven algunos errores gráficos, las funciones de comparación de caracteres también resuelven algunos problemas fonéticos, que suelen aparecer como metaplasmos, sinalefas, etc. Una función que tenga en cuenta las operaciones `I, D` y `S`, podrá aceptar la desaparición del sonido de una *S* final, la transformación fonética del sonido de una *B* en el de una *P*, la transformación de *DE LA HOZ* en *DELOZ*, etc.

En este trabajo proponemos una función de distancia, a la que llamamos *DEA.* Es una distancia de edición, para la que hemos diseñado un umbral variable en función de la longitud del antropónimo buscado, y unos costos variables de acuerdo con un modelo probabilístico que tiene en cuenta:

- el tipo de operación (`I, D` o `S`)
- la posición en donde se aplica
- las letras involucradas

Generalmente los métodos de comparación someten a *transformaciones* previas las cadenas de caracteres a comparar. Como mínimo se procura ponerlas en un formato estandarizado o *normalizado* que, en principio, dependerá de la aplicación, de los procedimientos de entrada de datos y del método de comparación utilizado. Aquí, en nuestras pruebas, sometemos los antropónimos a la normalización descrita en LSI-02-1-R.

1.2.- Funciones cuantitativas y booleanas. Funciones de distancia

Las funciones de semejanza basadas simplemente en una codificación fonética son funciones *booleanas* porque se limitan a indicar si la pareja de cadenas son o no semejantes segun que los códigos sean o no idénticos.

Las funciones basadas en la comparación de caracteres suelen devolver un valor real no negativo que es una medida de la semejanza o de la distancia (des-semejanza). Son *cuantitativas*.

Semejanza (x , y) o bien $\delta(x , y)$

Pero toda función cuantitativa es facilmente transformable en booleana, filtrándola con un umbral *k*.

SonSemejantes (x , y , k)

Entre las funciones de comparación de caracteres, la función booleana más utilizada es la siguiente versión mínima de la distancia simple:

- Dos cadenas son semejantes si para pasar de una a otra es suficiente una sola operación del tipo `I, D` o `S`

o dicho de otra forma, son semejantes si la distancia simple es igual a la unidad.

Las funciones cuantitativas son más flexibles que las booleanas y permiten ordenar los resultados de una búsqueda por orden creciente de distancia, lo cual suele ser de gran utilidad. Por ejemplo, permiten la elección automática del más semejante:

ElMasSemejante (x , C)

Esta función devuelve el antropónimo de *C*, que más se asemeja a un antropónimo dado *x* , o sea:

ElMasSemejante (x , C) = $\text{argmin}_{y \in C}\ \delta(x , y)$

Hasta ahora hemos hablado de funciones de distancia de manera bastante informal. Para podernos expresar mas formalmente, usaremos la siguiente notación:

Notaremos como Σ al alfabeto, o sea el conjunto de caracteres que aceptamos en los antropónimos

La longitud del alfabeto será $|\Sigma| = \alpha$

Σ^* será el conjunto de todos los antropónimos (o supuestos antropónimos) posibles, incluyendo el antropónimo cadena vacía ε

Una función de distancia, $\delta(.,.)$, se definirá como la aplicación de un conjunto de parejas de antropónimos sobre el conjunto de los números reales no negativos.

$$\delta : \Sigma^* \times \Sigma^* \rightarrow \square_+$$

La distancia entre dos antropónimos x e y , la notaremos como

$$\delta(x, y)$$

1.3.- Distancias métricas

Las funciones de comparación de caracteres, dan generalmente mejores resultados, son más eficaces, que las de codificación. Pero en BD con grandes volúmenes se utiliza mucho la codificación porque permite usar técnicas de búsqueda eficientes a base de índices, mientras que las funciones de distancia se suelen utilizar, en la práctica, con barridos totales lo cual las hace demasiado caras. Los algoritmos de búsqueda de cierta eficiencia para el caso de funciones de distancia, suelen exigir que el espacio de búsqueda sea métrico (ver LSI-02-59) o sea que la distancia sea métrica. Pero la funciones eficaces para los problemas de búsqueda aproximada de palabras (del lenguaje natural o de antropónimos) no siempre son distancias métricas. Una función de distancia es una métrica en Σ^*, si se cumplen las siguientes propiedades :

reflexiva: $\delta(x,x) = 0$

$0 \leq \delta(x,y)$ si $x \neq y$

simétrica: $\delta(x,y) = \delta(y,x)$

desigualdad triangular: $\delta(x,y) + \delta(y,z) \geq \delta(x,y)$

$\forall x,y,z \in \Sigma^*$

La distancia de edición puede ser métrica o no serlo, depende de los costes de las operaciones. Con costes unitarios, la distancia de edición es métrica.

2.- Las distancias de edición

2.1- Distancia de edición y tipos de operaciones

Levenstein propuso en 1966 [Lev66] una distancia entre cadenas de caracteres medida como número mínimo de operaciones `I`, `D` y `S`, necesarias para transformar una cadena en otra. Esta es la distancia de edición más utilizada; la distancia simple.

En 1964 Damerau [Dam64] propuso un método de comparación por el que dos cadenas se consideran semejantes si una de ellas se puede obtener de la otra con una sola operación `I,D,S` o `T` (transposición de caracteres en posiciones adyacentes). En 1974 Wagner y Fischer [Wag74] llamaron a esos cuatro tipos de operaciones, *operaciones de edición*.

La distancia simple considera solamente las tres operaciones de edición clásicas, `I,D,S`. Pero algunos autores proponen, para aplicaciones específicas, el uso de solo la substitución [Sal89] [Pev95] [Ami00]). Otros proponen solo las operaciones `I` y `D` y aun otros solo la `I` (en este caso se la conoce como *episode distance*). Estas distancias con un juego más limitado de operaciones, no solo se justifican por la aplicación concreta (ciertos casos de *pattern-matching* o de biología molecular por ejemplo) sino también porque los algoritmos de cálculo pueden ser mucho más eficientes.

También se han propuesto otros tipos de operaciones adicionales a los cuatro tradicionales (`IDST`). Citemos algunos ejemplos:

Transposición no contígua
Transposición de digramas contíguos o no contíguos
Transposición seguida de substitución [Lee97] [Oom97a]
Rotación [Mor82]
Substitución de dos caracteres , Fusiones y Particiones [Sen96]
Squashing y Expansión [Oom95]
Inserción/supresión de subcadenas [Leu97]

Se suele exigir que en una secuencia de operaciones no se pueda aplicar más de una operación de edición a la misma posición. En caso contrario la distancia no siempre sería calculable [Nav01] [Lop99]. Otra limitación similar [Gali90] consiste en que las operaciones se apliquen en el siguiente orden: 1º las `D` , 2º las `S` , 3º las `I`.

2.2- Costes y distancia generalizada

En la distancia simple cada operación tiene un coste, un peso, igual a 1. Wagner [Wag74] definió una distancia de edición en la que las tres operaciones clásicas (`I, D, S`) pueden tener costes variables en función de los caracteres. Así por ejemplo, a la substitución de la letra *M* por *N* se le podrá asignar un coste inferior a la substitución de *M* por *R.* En [Cam81] utilizábamos costes en función de las frecuencias de las operaciones `IDS` para los distintos caracteres.

En 1975 Lowrance [Low75] extendió la distancia de Wagner a cuatro operaciones, `IDST`, pero con la limitación de que el coste de cada tipo de operación sea constante y que se cumpla que: el coste de la `I` más el de la `D` sea menor o igual al doble del coste de la `T`.

Posteriormente se han propuesto para distintos tipos de aplicaciones distancias de edición con costes más sofisticados. Por ejemplo, Leung [Leu97] propone, en aplicaciones que manejan cadenas de proteinas, una distancia en la que la `I` y la `D` se aplican a subcadenas (no limitadas a un solo caracter) y cuyo coste es sensible al contexto ya que depende del caracter anterior, del posterior y de la longitud de la subcadena. Las `S` son de un solo caracter y su coste es independiente del contexto.

Cuando los costes de las operaciones no están limitados, se dice que se trata de una *distancia de edición generalizada*. Por ejemplo [Kur96] trata solo con las operaciones `I, D` y `S` pero con costes arbitrarios.

El valor de una función de distancia de edición, $\delta(x,y)$, se puede definir como el coste mínimo de todas las secuencias posibles de operaciones de edición que transforman *x* en *y* (este coste es infinito si no existe tal secuencia). El coste de una secuencia es la suma de los costes de las operaciones elementales que la forman. Los costes de las operaciones elementales son números reales no negativos (en el caso de la distancia simple son = 1) y que notaremos como δ_c

Si vemos los caracteres individuales como cadenas de un solo caracter, la distancia entre dos de esos caracteres *x* e *y* será $\delta(x,y)$. Entonces:

$\forall x,y \in \Sigma$

el coste de la `I` (x) será $\delta_c(\varepsilon,x) = \delta(\varepsilon,x)$

el coste de la `D` (x) será $\delta_c(x,\varepsilon) = \delta(x,\varepsilon)$

el coste de la `S` (x,y) será $\delta_c(x,y) = \delta(x,y)$ para $x \neq y$

Podemos considerar que las operaciones elementales `I`, `D` y `S`, no son mas que casos particulares de un único tipo de operación de edición, $E(x,y)$, $\forall x,y \in \Sigma \cup \{\varepsilon\}$, y cuyo coste es $\delta_c(x,y)$.

La inserción de un caracter en una cadena es equivalente funcionalmente a la supresión de ese caracter en la otra cadena. Y la substitución del caracter *x* por *y*, es equivalente a la de *y* por *x*. O sea, se cumplirá que:

$\forall x,y \in \Sigma$

$\delta_c(x,\varepsilon) = \delta_c(\varepsilon,x)$

$\delta_c(x,y) = \delta_c(x,y)$

2.3.- Determinación de los costes DEA

2.3.1.- Introducción

En la distancia simple, la más utilizada, toda operación $E(x,y)$ tiene un coste = 1.

$$\delta_c(x,y) = 1 \qquad \forall x,y \in \Sigma \cup \{\varepsilon\}$$

La mayoría de distancias definidas con otros juegos de operaciones (más limitados o más amplios) utilizan también costes unitarios. ¿Porqué se suelen utilizar costes unitarios? Hay dos razones básicas: a) la dificultad en encontrar los costes adecuados y b) las técnicas realmente eficientes que se conocen para la implementación del cálculo de la distancia suponen costes unitarios.

¿Como podemos determinar cuales son los costes que hemos de asignar a las operaciones elementales para la distancia de edición generalizada? Es obvio que los costes deben depender de alguna manera de las características de las deformaciones (gráficas, fonéticas, etc) que la aplicación tenga que considerar aceptables. En nuestro caso, dado que se pueden dar tanto errores fonéticos como gráficos, queremos que los costes nos aporten también características fonéticas a los criterios de comparación de caracteres, como también hacen [Ver88] y [Zob96]. Pero como no tenemos un conocimiento total sobre el mundo que queremos modelar (el mundo de las deformaciones, errores y variantes) tendremos que incorporar al modelo algún tipo de estimación probabilística basada en datos experimentales (ver LSI-02-61-R). A los costes que proponemos aquí les llamamos costes DEA.

La diversidad de propuestas existentes es un indício de la dificultad del problema [Kas84] [Jou93] [Dag95] [Wei95] [Bun95] [Oom97b]. Algunos investigadores tratan de resolver el problema aplicando técnicas de optimización para la determinación automática de los costes óptimos (o casi óptimos). Aplican técnicas de aprendizaje automático, a partir de corpus de entrenamiento, en algunos casos basadas en redes neuronales [Ris98] .

Se han propuesto algoritmos eficientes para políticas de costes muy restringidas. Por ejemplo en [Ars99] se supone solamente tres costes, uno para cada tipo de operación `I,D,S`. Algunos autores [Bun95] [Oom96] fijan a 1 el coste de las `I` y las `D`, y buscan un coste óptimo (pero constante) para las operaciones `S`. A esas distancias se las suele llamar *distancias paramétricas*.

2.3.2.- Desventajas simétricas

En los trabajos que proponen distancias con costes, se supone que el coste de una operación debe depender de su probabilidad. No hay unanimidad en como plantear el detalle de esta relación. La mayoría de los trabajos sobre búsquedas aproximadas, se limitan a proponer que el coste debe ser inversamente proporcional a la probabilidad (frecuencia en tanto por uno) de cada operación elemental en el corpus de errores tomado como referencia. Por ejemplo la operación elemental `S` (*V,U*) debería tener un coste menor que `S` (*V,Q*) ya que esta última es una confusión mucho menos probable que aquella. Frecuentemente se propone que en lugar de utilizar la probabilidad se utilice su logaritmo [Oom97b] [Jou93] [Mar93] [All87].

Pero esos enfoques no tienen en cuenta la probabilidad "prior". Por ejemplo, tienen en cuenta la probabilidad de que una substitución lo sea del caracter *x* por el *y*, pero no consideran la probabilidad de aparición de *x* en el conjunto de las cadenas. El hecho de que en el corpus de errores aparezcan muchas substituciones de *A* por *E*, pero muy pocas substituciones de *D* por *T*, habrá que ponderarlo con el hecho de que la *A* es una letra muchísimo más frecuente en los antropónimos que la *D*.

Parece más razonable utilizar algún concepto al estilo de las probabilidades condicionales, como por ejemplo se hace en [Chu91] . En esta linea podríamos utilizar unos costes inversamente proporcionales a las tendencias, T, calculadas en LSI-02-61-R, pero hemos preferido desarrollar un enfoque basado en la idea de desventaja. Llamaremos *desventaja* de una operación de edición, a la relación entre la probabilidad de que se dé en un corpus de parejas al azar y la probabilidad de que se dé en un corpus de parejas con errores reales. Como corpus parejas al azar utilizaremos nuestro fichero *CONTROL*, y como corpus de parejas con error usaremos nuestro fichero *TEST*. Observemos que esta idea se corresponde con la idea del *poder de discriminación* usado especialmente en aplicaciones de clasificación automática, en donde interesa maximizar el ratio entre las diferencias interclase y las diferencias intraclase.

$$D_{op} = \frac{\Pr(\text{op en CONTROL})}{\Pr(\text{op en TEST})}$$

Veamos ahora como podemos determinar las dos probabilidades que intervienen en el cálculo de la desventaja para el caso de las substituciones. Para el numerador necesitamos la probabilidad de que dada una pareja de *CONTROL*, escogida al azar, se encuentre en ella una substitución de x por y (o al revés).

Para estimar este valor, empezaremos llamando $P(x)$ a la probabilidad de encontrar un símbolo *x* al escoger un caracter, al azar, entre todos los caracteres que forman el

conjunto de antropónimos de nuestro universo de trabajo. También la podríamos llamar probabilidad *prior*. En nuestro caso el conjunto de antropónimos escogido es el de los antropónimos que figuran en las parejas del fichero *CONTROL* (ver LSI-02-61-R) y la probabilidad $P(x)$ se calculará así:

$$P(x) = \frac{\text{número total de } x}{\text{número total de letras}}$$

Escojamos dos antropónimos al azar a_1 y a_2 y de cada uno de ellos escojamos al azar un caracter. Llamaremos $P(xy)$ a la probabilidad de que el caracter de a_1 sea x y el de a_2 sea y. Entonces:

$$P(xy) = K_1 P(x) P(y) \qquad 0 \le K_1 \le 1$$

La constante K_1 es un factor de corrección para no incluir los casos en que $x = y$.

$$K_1 = 1 - P(x=y) \text{ siendo}$$
$$P(x=y) = \sum (P(x) P(y)) \qquad \forall x,y \mid x=y$$

Por lo tanto $K_1 = 1 - \sum (P(x))^2 \qquad \forall x$

Observemos que $P(xy) = P(yx)$

Veamos ahora como calcular el denominador de la desventaja: probabilidad de que una pareja de TEST, escogida al azar, tenga en ella una substitución de x por y (o al revés).

Las probabilidades de las operaciones de substitución las estimaremos a partir de las frecuencias mostradas en LSI-02-61-R. Para algunas parejas de caracteres no existe en los corpus ningun caso de substitución, lo cual puede plantear problemas ya que la probabilidad no debería ser cero. Para esas parejas hemos supuesto una frecuencia absoluta igual a 0,5.

El fichero *TEST* está formado por parejas con error, o sea parejas de antropónimos, a_1/a_2, tales que uno es una variante, anomalía o deformación del otro. El orden de los antropónimos dentro la pareja es indiferente. No hay una cadena correcta y otra incorrecta. Por ello se deberá cumplir que:

$$\delta(a_1,a_2) = \delta(a_2,a_1)$$

La pareja a_1, a_2 representará tanto la transformación $a_1 \rightarrow a_2$ como la transformación $a_2 \rightarrow a_1$.

La substitución de una x por una y puede ser vista también como una substitución de una y por una x. En LSI-02-61-R introducimos la idea de *substitución simétrica* SS (x,y) para incluir ambas substituciones.

Definiremos ahora la probabilidad $P_{ss}(xy)$ de esta substitución simétrica, determinada a partir de las parejas del fichero *TEST*, como :

$$P_{ss}(xy) = \frac{\text{número de subs } (x \text{ por } y) \text{ o } (y \text{ por } x)}{\text{número total de parejas}}$$

Obsérvese que $\frac{P_{ss}(xy)}{2}$ es la media de las probabilidades de `S`(*x,y*) y `S`(*y,x*).

Ahora ya podemos expresar la desventaja simétrica $D_{ss}(xy)$ de una operación `SS` (*x,y*), como:

$$D_{ss}(xy) = \frac{2P(xy)}{P_{ss}(xy)}$$

Para los costes DEA, hemos aproximado el valor de $P_{ss}(xy)$ utilizando solamente las parejas con distancia simple igual a uno.

Para el caso de las inserciones y supresiones, el numerador de la desventaja será la probabilidad de que en una pareja de *CONTROL,* escogida al azar, exista una inserción o supresión de *x.* Podemos expresar esta probabilidad como

$$K_2\, P(x)$$

Para dar un valor a K_2 hemos adoptado el siguiente enfoque: Si las longitudes de las cadenas de la pareja son l1 y l2, siendo l1 $\geq$ l2, entonces habrá l1-l2 inserciones (o supresiones). Por lo tanto K_2 será el valor medio de l1-l2. (En nuestro corpus este valor es 2 aproximadamente).

Para el denominador, al igual que en el caso de las substituciones, consideraremos la operación simétrica `ID`(*x*) que incluye tanto la inserción como la supresión de *x*. Como no hay una cadena correcta y otra incorrecta, deberá ser:

$$\delta_c(x, \varepsilon) = \delta_c(\varepsilon, x)$$

o sea que la supresión de un símbolo cuesta lo mismo que su inserción.

La probabilidad $P_{id}(x)$ de la operación simétrica `ID`(*x*), determinada a partir de las parejas de *TEST*, será:

$$P_{id}(x) = \frac{\text{número de ins o sup de } x}{\text{número total de parejas}}$$

Por lo tanto, la desventaja simétrica $D_{id}(x)$ de una operación `ID`(*x*) la podremos expresar como:

$$D_{id}(x) = K_2 \frac{2P(x)}{P_{id}(x)}$$

Para los costes DEA, hemos aproximado el valor de $P_{id}(x)$ utilizando solamente las parejas con distancia simple igual a uno.

Es facil comprobar que las desventajas, tal como proponemos calcularlas, son muy similares a las inversas de las tendencias al error, T , presentadas en LSI-02-61-R

No podremos utilizar directamente las desventajas $D_{ss}(xy)$ y las $D_{id}(x)$ obtenidas con las formulas anteriores, como costes de las operaciones de edición elementales, si

queremos que la distancia sea métrica. Hemos definido las desventajas D_{ss} *(xy)* y D_{id} *(x)* de manera que ya cumplen la propiedad simétrica. Pero también se deberá cumplir que δ_c *(x,x)* = 0 , y en el caso de $x \neq y$ deberá ser $0 \leq \delta_c$ *(x,y)* . Todo ello no plantea ningún problema. Pero también se deberá cumplir la desigualdad triangular, la cual nos impone fuertes limitaciones en los valores de los costes. Se tendrá que cumplir que

$$\delta_c (x,y) + \delta_c (y,z) \geq \delta_c (x,z)$$

o sea, el coste de una operación no ha de ser mayor que el de una secuencia equivalente. Por lo tanto nos exige que:

- el coste de substituir *x* por *y* no exceda el de
  substituir *x* por *z,* más el de *z* por *y*
  ni el de suprimir *x,* más el de insertar *y*
- el coste de suprimir *x* no exceda el de substituir *x* por *y,* más el de suprimir *y*
- el coste de insertar *y* no exceda el de insertar *x,* más el de substituir *x* por *y*

Es obvio que la distancia de edición con un coste fijo y único para todas las operaciones (aunque incluyéramos la transposición, T) cumple con esas desigualdades. Si adoptamos como costes de las operaciones las desventajas $D_{ss}(xy)$ y $D_{id}(x)$, nos encontramos que sus valores varian en un intervalo enorme. Entre el máximo y el mínimo obtenidos en nuestros corpus, hay un ratio del orden de 10^3, y obviamente, resultará una distancia que no cumple la desigualdad triangular. Pero necesitamos que la cumpla por dos razones:

a) las técnicas de búsqueda eficientes pueden necesitar la triangularidad [Lop99]

b) la naturaleza del problema de los errores en antropónimos, no acepta que un error sea substituido por dos en la misma posición, aunque el coste conjunto de los dos sea menor que el de uno solo. Así por ejemplo, el confundir una *D* por una *F*, no debería ser reemplazado por la supresión de una *D* y la inserción de una F, ni tampoco por la substitución de una *D* por una *T* seguida de una substitución de una *T* por una *F*, sean cuales sean los costes de esas operaciones.

Una forma facil de hacer cumplir la desigualdad triangular sería haciendo que el coste elemental mas alto no supere al doble del más pequeño. Para ello adoptamos, arbitrariamente, como límites de los costes elementales, 0,6 y 1,2. Para acotar los costes a este intervalo, convertimos las desventajas D en costes haciendo:

$$Coste = 0{,}6 * \left(2 - \frac{D - Dmax}{Dmax - Dmin} \right)$$

Tras hacer esta conversión, hemos disminuido los valores pequeños y aumentado los grandes, hasta el punto en que ya no se cumpliría la ley triangular. Al final han resultado unos costes que oscilan entre 0,41 y 1,2.

Ejemplos de costes elementales DEA:

δ_c *(R,C) =* δ_c *(C,R) = 1*
δ_c *(C,U) =* δ_c *(U,C) =1*
δ_c *(R,* ε*) =* δ_c *(*ε*,R) =1,08*
δ_c *(U,* ε*) =* δ_c *(*ε*,U) =0,83*

Teniendo en cuenta estos costes, el valor de δ*(ACUSO,ARCSO)* será 1,91 ya que este es el coste de la secuencia `D`*(R),* `I`*(U)* y cualquier otra secuencia válida tiene un coste mayor, por ejemplo; `S`*(R,C) ,* `S`*(C,U)*

Aunque en la distancia propuesta, DEA, solo utilizamos las operaciones `I`, `D` y `S`, hemos hecho pruebas incluyendo las operaciones de transposición contigua, `T`. Los costes de la transposición se han calculado basándose en la desventaja D_{tt}*(xy)* calculándola como la relación entre las probabilidades *prior* de que se den operaciones `T` en parejas al azar (*CONTROL*) y que se den en parejas con error (*TEST*). En la siguiente expresión, la abreviatura "bigr" quiere decir "bigramas".:

$$D_{tt}(xy) = \frac{2\,(\text{num total bigr})^2}{\text{num total de T}} * \frac{(\text{num bigr } xy) * (\text{num bigr } yx)}{\text{num de T}xy \text{ o T}yx}$$

Para que al añadir la operación `T` se cumpla la ley triangular, tendremos que añadir la desigualdad de que el coste de `T`*(x,y)* no exceda del doble de `S`*(x,y).*

Los resultados, en cuanto eficacia, obtenidos en las pruebas al añadir la operación `T` apenas mejoran los obtenidos sin ella. Como la inclusión de la `T` complica los algoritmos de cálculo y búsqueda, perdiendo eficiencia, hemos decidido quedarnos con solo las tres operaciones tradicionales, `IDS`.

2.3.3.- Posiciones de las operaciones

Hasta aquí hemos tenido en cuenta dos fuentes de conocimiento; las frecuencias de los distintos tipos de errores (operaciones de edición) encontradas en *TEST* y las frecuencias *prior* de las letras. Para recoger más información del contexto en el que se efectuan las operaciones de edición, decidimos tener en cuenta también la posición en que ocurren, ya que la distribución de las letras y de las operaciones no es uniforme según las posiciones. Para mejorar la eficacia proponemos pues unos costes que son función no solo del tipo de operación y de los caracteres afectados sino también de su posición.

En una primera versión distinguimos entre la 1ª posición, la 2ª, la última y el resto de posiciones. Pero comprobamos que el distinguir la 2ª posición apenas mejoraba la eficacia y en cambio complicaba la implementación. Por lo tanto hemos utilizado tres matrices de confusión y tres conjuntos de frecuencias *prior* (1ª, última y el resto). A las operaciones en las posiciones que no son ni la 1ª ni la última, les asignamos el mismo coste que usábamos en el caso de no distinguir posiciones.

Los dos antropónimos a comparar pueden tener longitudes distintas, por lo que conviene aclarar qué entendemos por substitución en última posición; será aquella en la cual cada una de las dos letras es la última de su cadena.

Para construir el nuevo modelo probabilístico (los seis conjuntos de probabilidades) hemos necesitado las frecuencias de las 1053 distintas operaciones elementales, simétricas, por posición y las 78 frecuencias *prior* de las letras segun las tres posiciones:

$1053 = 3*(26+((26^2-26)/2))$

$78 = 3*26$

Nótese que ahora todos los valores que entran en juego en el cálculo del coste, pueden variar en cada posición. Por ejemplo:

- la constante k_1, prevista para no contar en $P(xy)$ las parejas en las que $x=y$, será distinta en cada posición.

- el denominador de $P(x)$ ya no será el número de letras sino el número de parejas

Veamos dos ejemplos de diferencias entre los costes según la posición:

`ID`*(G)*	→	1ª pos = 1,3	últ.pos = 1,1	resto = 1,2
`SS`*(J,G)*	→	1ª pos = 1	últ.pos = 0,6	resto = 0,52

Obsérvese que ahora, para la obtención de los costes de las 1053 operaciones de edición simétricas hemos tenido que aplicar también las limitaciones impuestas por la desigualdad triangular pero teniendo en cuenta las posiciones.

En el resultado final obtenido, los valores oscilan entre 0,41 y 1,3 aunque la mayoría de los 1053 costes se situa entre 0,6 y 1,2.

En la sección 4 evaluamos la eficacia de la distancia de edición DEA, resultante de aplicar esos costes detallados por posición. Pero también hemos evaluado la eficacia en el caso de que en el cálculo de las desventajas se usen expresiones logarítmicas. Hemos probado dos variantes. La primera pretende que el ratio no sea entre probabilidades sino entre cantidad de información [Lin98]:

$$Dss_{xy} = \frac{Log2 + Log(Prior_{xy})}{Log(Pss_{xy})}$$

$$Did_{x} = \frac{Log(K_2) + Log2 + Log(Prior_{x})}{Log(Pid_{x})}$$

En la segunda variante hemos hecho que las probabilidades se conviertan en entropía [Zav97], así:

$$entropía = -prob * log_2\, prob$$

Pero los resultados obtenidos con las desventajas calculadas con ambas variantes han sido sensiblemente inferiores a los obtenidos sin expresiones logarítmicas.

2.4.- Umbrales y longitudes

Teniendo una función de distancia $\delta(x,y)$, podemos construir funciones como *SonSemejantes (x,y,k)* o el procedimiento que aquí en este trabajo necesitamos

$$LosSemejantes\ (x, C_1, C_2, k)$$

que hemos definido así: Dado un conjunto C_1 de antropónimos y un antropónimo x, obtener un conjunto $C_2 \subseteq C_1$ con aquellos antropónimos que son semejantes a x. El conjunto C_2 puede quedar vacío. Podremos dar el valor de un parámetro k para regular la dureza del criterio de determinación de la semejanza.

¿Que valor dar al parámetro *k*? Es obvio que dos errores en un antropónimo de 15 caracteres es más tolerable que en un antropónimo de tan solo 4 caracteres. El número de errores (el número de operaciones de edición) que se cometen no es independiente de la longitud de los antropónimos. El número de errores crece al crecer la longitud, especialmente a partir de la longitud = 10.

Algunos autores (especialmente en el campo del *pattern recognition*) [Ars99] [Vid95] [Mar93] proponen el uso de distancias *normalizadas*. Para cada posible secuencia de operaciones de edición que transforme una cadena *x* en otra *y* determinan el coste medio por operación (o sea el coste total de la secuencia dividido por su número de operaciones). El coste medio más pequeño es la distancia normalizada. Hacemos notar que esta distancia no puede calcularse dividiendo la distancia de edición por la longitud de su secuencia.

Hemos decidido utilizar un umbral que dependerá de la longitud. Los dos antropónimos pueden tener longitudes diferentes. Hemos hecho pruebas con la longitud del primer antropónimo de la pareja, con la del segundo y con la del mas corto. Como era de esperar, los resultados obtenidos con la longitud del primero y la del segundo son casi iguales. Pero ambos son ligeramente mejores que con la longitud del mas corto. Decidimos por lo tanto utilizar la longitud del patrón de búsqueda. La longitud considerada es la que queda después de someter el antropónimo a las transformaciones previas.

El tema de los prefijos, sufijos y abreviaturas (que pueden dar diferencias de longitud considerables) se trata aparte de la función de distancia (ver LSI-02- VIII). Así por ejemplo: si buscamos *FUENTES*, el antropónimo *DELASFUENTES* debería será considerado semejante aunque la función de distancia calculase una distancia superior al umbral que corresponde a la longitud de *FUENTES*.

Decidimos que el umbral para cada longitud tenga que ser mayor que la distancia media en *TEST* para esa longitud. Agrupamos las parejas de *TEST* por la longitud de su primer antropónimo y para cada longitud calculamos la media de la distancia. Hicimos lo mismo para las parejas de *CONTROL*. Parece razonable que el umbral para cada longitud, deba estar entre esos dos valores. Tras varios tanteos de "prueba y error", determinamos siete juegos de valores de los umbrales, para poder así escoger entre siete grados de dureza (*A,B,C,D,E,F,G*) en el criterio de semejanza. Por lo tanto, el parámetro *k* del procedimiento *LosSemejantes* permitirá indicar cual de los siete juegos de valores se desea utilizar.

Long	A	B	C	D	E	F	G
2	1,71	1,79	1,9	1,9	1,9	2,4	2,6
3	1,73	1,83	1,9	2,5	2,8	3,3	3,4
4	1,87	1,95	2,1	2,7	2,9	3,4	3,6
5	1,89	2	2,16	2,78	3	3,5	3,8
6	1,98	2,2	2,6	2,8	3,1	3,6	4
7	2,25	2,4	2,9	2,9	3,3	3,8	4,1
8	2,45	2,5	3	3,2	3,5	4	4,3
9	2,5	2,5	3,1	3,2	3,6	4,1	4,8
10	2,6	2,6	3,5	3,5	3,7	4,2	5,1
11	3,5	3,8	4	5	5	5,5	5,9
12	4	4,5	4,5	5	5	5,5	6
13	4,5	4,5	4,5	5	6	6,5	6,5
≥14	5	5	5	5	6	6,5	7

En la sección 4 veremos la eficacia de la distancia DEA, en la que se utilizan los umbrales variables junto con los costes por posición y la compararemos con la de otras distancias.

3.- Otras distancias

A través de los años se han propuesto muchísimas formas de evaluación de la semejanza entre dos palabras basadas en la comparación de caracteres. Hasta ahora hemos hablado del enfoque más habitual, la distancia de edición. Otro enfoque popular es el de las distancias basadas en *n-grama*s, que estudiaremos a continuación. También comentaremos las ideas básicas que hay detrás de las distancias propuestas por Alberga [Alb67] Jaro [Jar89] y Zobel [Zob96] y en la sección 4 compararemos la eficacia de todas esas funciones.

3.1.- Distancias basadas en *n-grama*s

El término *n-grama* se usa en lingüística tanto en el sentido de un conjunto de *N* letras como en el de un conjunto de *N* palabras. Son muy utilizados los *n-grama*s de letras, especialmente en la detección y corrección de errores ortográficos.

Es muy frecuente medir la semejanza entre dos palabras, mediante una distancia definida como la diferencia entre el número total de *n-grama*s entre las dos palabras y dos veces el número de *n-grama*s comunes a ambas [Pet00]. Esta distancia es una aproximación de la distancia definida por Ukkonen [Ukk92] el cual estudió también la relación entre su distancia y la distancia simple. En LSI-02-59-R se ve como la coincidencia de *n-grama*s se puede utilizar para mejorar la eficiencia (el tiempo de búsqueda) de la distancia de edición DEA.

Los valores de *n* más utilizados en la práctica son;

n = 2 y entonces hablamos de *bigramas* (o digramas)
n = 3 y entonces hablamos de *trigramas*

Los *n-grama*s pueden ser solapados o no y con delimitadores de palabra o no. Aquí adoptamos *n-grama*s no solapados y con delimitadores (para incorporar el conocimiento del principio y final de palabra). Así la palabra *BAALO* constará de seis bigramas (los símbolos *[* y *]* representan los delimitadores de principio y fin de palabra respectivamente): *[B BA AA AL LO O]*

Si L_x es la longitud de la palabra x, el número de *n-grama*s será igual a $L_x + N$ - 1. Por lo tanto el número de bigramas será $B_x = L_x + 1$. En la distancia que utilizamos aquí y que se describe mas abajo, solo contamos los bigramas diferentes. O sea que en *BAABA* solo contamos cinco bigramas.

En la mayoría de pruebas expuestas en la literatura para el lenguaje natural, dan mejores resultados los bigramas que los trigramas. Vease por ejemplo [Kuk92] [Pfe95] [Zob95] [Vil96] [Pet00]. Cabe suponer que esto ocurre porque el trigrama es excesivamente largo para la longitud habitual de las palabras. En nuestro caso hemos hecho pruebas con distintas formas de cálculo de la distancia basadas en la coincidencia de bigramas y trigramas entre los dos antropónimos a comparar. La función de distancia que mejores resultados nos ha dado tiene la expresión siguiente:

$$\delta(x,y) = \frac{B_x + B_y - 2B_{xy}}{2B_{xy}}$$

en donde B_x es el número de bigramas diferentes que aparecen en la palabra x , B_y el de los que aparecen en la palabra y, B_{xy} es el número de bigramas diferentes comunes a ambas palabras. A B_{xy} le damos el valor 0,5 en el caso de que no haya nigún bigrama común. En la sección 4 comparamos la eficacia de esta distancia con los otros métodos presentados en este trabajo.

3.2.- Alberga

Uno de los trabajos más interesantes (y muy ignorado) sobre búsqueda aproximada de cadenas de caracteres, es el que en 1976 publicó J.Alberga. En él hizo un estudio de 65 métodos de comparación, casi todos cuantitativos. Damos a continuación las ideas básicas del método que a él le dió mejores resultados, al que denominamos método ALB25.

El algoritmo consiste en un conjunto de operaciones a realizar sobre una matriz de coindidencia. En una matriz de coincidencia el elemento (*i, j*) vale uno o cero según que el carácter de la posición *i* de la 1ª cadena coincida o no con el caracter de la posición *j* de la 2ª. El algoritmo consta de tres fases*: Ponderación, Selección* y *Cálculo de la distancia.* Esta división en tres fases de operaciones sobre una matriz de coincidencia no es privativa de ALB25, sino que es el esquema general que Alberga utilizó para describir métodos de comparación. Son muchos los algoritmos que pueden adaptarse a tal esquema. Veamos el detalle de las tres fases para ALB25:

- En la fase de *Ponderación* se reemplazan los valores (*i, j*) = 1 de la matriz por

$$1 - \left| \frac{i-1}{m-1} - \frac{j-1}{n-1} \right|$$

siendo *m* la longitud de la 1ª cadena y *n* la de la 2ª.

- En la fase de *Selección* se transforma la matriz de modo que no haya columnas ni filas con más de un elemento no nulo. Para ello se empieza seleccionando el mayor elemento de la 1ª fila y se suprimen el resto de elementos correspondientes a su fila y columna. A continuación se selecciona el mayor de los elementos que quedan en la fila 2ª y se suprimen el resto de elementos de su fila y columna, y así sucesivamente. Hacemos notar que esta fase es asimétrica.

- En la última fase se efectua el *Cálculo de la distancia* de la siguiente forma: Para cada conjunto de elementos no nulos diagonalmente consecutivos se hace una suma acumulativa. Así, en un conjunto de *n* elementos diagonalmente consecutivos, el 1º (el de menores *i j*) se suma *n* veces, el segundo *n*-1 veces, etc. Si llamamos *S* a la suma de esas sumas y *L* la longitud de la mayor de las dos cadenas, la distancia se define como

$$\delta(x,y) = 1 - \frac{2S}{L^2 + L}$$

De la forma que se ha concebido este algoritmo se consigue que:

a) A las coincidencias de dos caracteres se les dé menor importancia cuanto mas se alejen de posición (Fase de Ponderación).

b) Para cada caracter de una cadena se tenga en cuenta su coincidencia con otro como máximo de la otra (Fase de Selección).

c) Los caracteres coincidentes, formando grupos, cuenten tanto más cuanto más al principio del grupo estén, primándose al mismo tiempo la longitud de los grupos. (Fase de Cálculo de la distancia).

Como la distancia es asimétrica (por la fase de Selección) Alberga proponía que se tome como primera cadena la "correcta". Pero en nuestro caso no hay cadena crrecta por lo que en nuestras pruebas (sección 4) calculamos las dos distancias y escogemos la más pequeña.

Para disponer de una función booleana se puede fijar un umbral, que la experiencia aconseja situar entre 0,750 y 0,820.

3.3.- Jaro (US Census Bureau)

En los procesos de detección de coincidencia de personas en la gestión del censo norteamericano (*US Census Bureau*) se utiliza una distancia basada en comparaciones de caracteres, pensada especificamente para antropónimos, a la que llamaremos JARO [Win95]. En ella se tienen en cuenta :

- Las longitudes de los antropónimos

- Las transposiciones de caracteres

- las coincidencias de caracteres, siempre que sus posiciones no estén separadas más de la mitad de la longitud de la cadena más corta

- Los caracteres similares. Considera similares todas las vocales entre si y los pares de caracteres siguientes: BV B8 CG CK CQ EF EY Eespacio GJ IJ IL IY I1 KX MN O(letra)0(cero) PR QC QO(letra) Q0(cero) SX SZ S5 Sespacio UV UW VW Yespacio Z2

En las pruebas expuestas en [Win95] este método da unos resultados ligeramente mejores que la distancia simple, pero en la sección 4 veremos que en nuestros corpus su eficacia es inferior.

3.4- Zobel: EDITEX

En el trabajo de Zobel y Dart [Zob96] se propone una función de comparación, llamada EDITEX, que prueban con antropónimos obteniendo buenos resultados. En nuestras pruebas (sección 4) da una eficacia muy similar a la de la distancia simple. Se trata de una variación de la distancia de edición con costes. Solo aplican tres costes diferentes: coincidencia, semejanza, no semejanza. En el caso de inserción o supresión consideran también la posible semejanza o coincidencia de la letra anterior. Se da un tratamiento especial a la H, la W y a la secuencia de dos caracteres iguales.

El criterio de semejanza se basa en los grupos de la codificación fonética PHONIX [Gad90] aunque difieren en algun detalle. Consideran semejantes todas las letras que están en un mismo grupo:

AEIOUY BP CQK DT LR MN GJ FPV SXZ CSZ

Obsérvese que las letras P, C, S y Z aparecen en más de un grupo, de manera que tanto la X como la C son consideradas semejantes a la S y la Z, pero no semejantes entre si.

4- Evaluación y comparación de las funciones de distancia

4.1.- Metodología y corpus

En esta sección evaluaremos y compararemos algunas de las funciones de distancia explicadas anteriormente.

Para este análisis empírico utilizaremos el método presentado en LSI-02-60-R, que consiste básicamente en el análisis del poder de discriminación de las distintas funciones, entre un conjunto de parejas de antropónimos con error (*CONTROL*) y otro de parejas sin error (*TEST*).

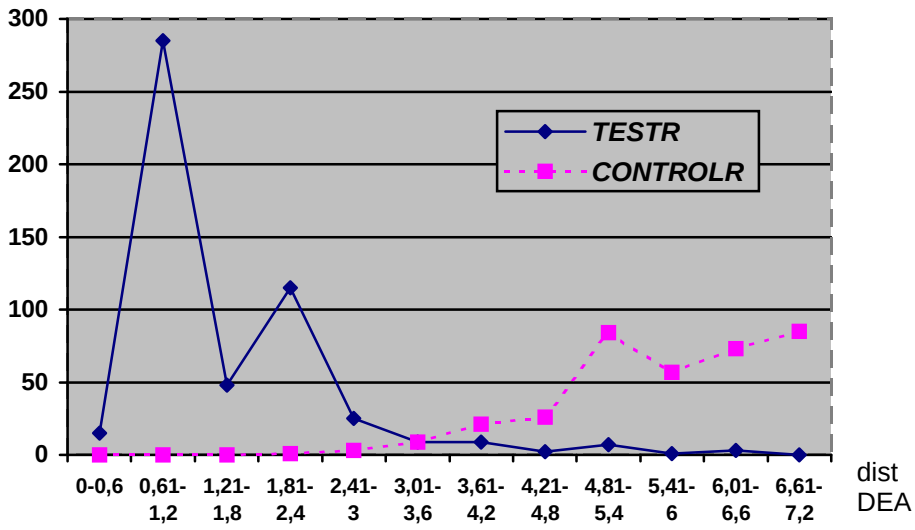


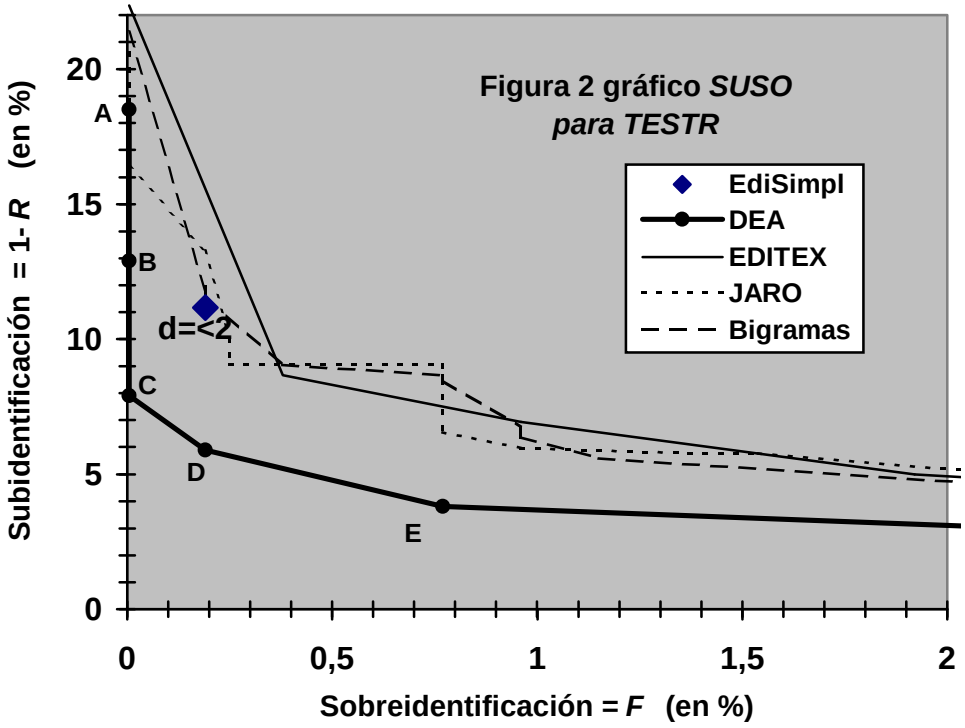
Figura 1: Distribución de la distancia DEA

Intuitivamente, el objetivo del método es la determinación de cual de las funciones de distancia es la que mas separa las dos curvas (una de cada fichero) de distribución de las frecuencias segun la distancia . En la figura 1 hemos representado, a título de ejemplo, las dos curvas para la distancia de la función DEA (los ficheros son *TESTR* y *CONTROLR* conteniendo cada uno de ellos 519 parejas). Las métricas que utilizaremos son básicamente la subidentificación (1-R) y la sobreidentificación (F), mediante el gráfico *SUSO*, pero también estudiaremos la precisión, P y las dos métricas de eficacia, E y J (LSI-02-60-R).

El corpus utilizado es el presentado en ISI-02-61-R y que consta de diferentes parejas de ficheros: *TESTU / CONTROL, TESTR / CONTROLR, TESTC / CONTROLC.* Allí comentamos sus características y particularidades. Como la pareja de ficheros *TESTU / CONTROL* la hemos utilizado para determinar los parámetros (costes y umbrales) de la distancia DEA, los resultados obtenidos con esa pareja podrían resultar sesgados. Por ello nos fijaremos basicamente en los resultados obtenidos con la pareja *TESTR / CONTROLR*.

4.2.- Sobreidentificación, subidentificación y umbrales: Gráfico *SUSO*

En las figuras 2 y 3 vemos los gráficos *SUSO* con los resultados obtenidos por las distintas funciones de distancia, con las dos parejas de ficheros *TESTR / CONTROLR* y *TESTU / CONTROL.* Como en la practica dificilmente nos interesarán valores de la subidentificación superiores al 22%, y de la sobreidentificación superiores al 2%, en estas figuras nos hemos limitado a este intervalo ($F \leq$ 2% y $1\text{-}R \leq$ 22%).



Dentro del intervalo de las figuras, la función DEA tiene 5 o 6 umbrales (segun el fichero), que hemos definido en función de las longitudes. aunque se podrían definir muchísimos más pues DEA produce en ese intervalo mas de 100 distancias diferentes. La función de distancia simple, solo produce un punto en la figura 2 y dos en la 3. Dentro de nuestro intervalo, JARO produce unos 60 puntos, BIGRAMAS puede tener una docena y EDITEX produce basicamente 5.

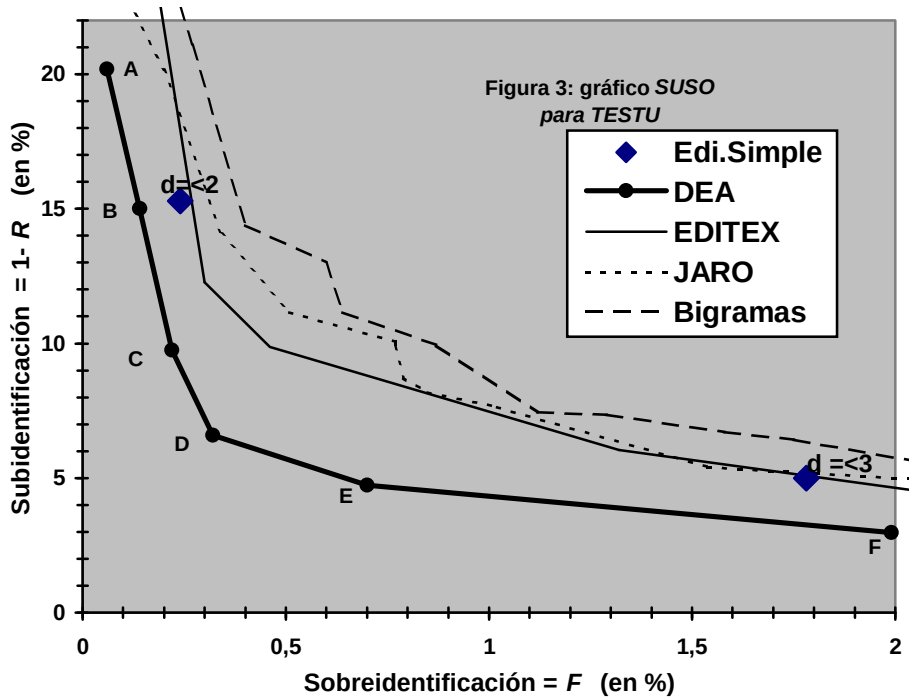
Nuestro objetivo es minimizar a la vez F y $1\text{-}R$. Una visión rápida de las figuras 2 y 3 nos muestra que la función DEA cumple mejor este objetivo que el resto de las funciones, las cuales dan resultados muy similares entre si. Para un mismo nivel de subidentificación, la función DEA da una sobreidentificación 70% a 80% inferior (dentro del intervalo de las figuras). Para un mismo nivel de sobreidentificación, da una subidentificación 40% a 55% inferior. Los resultados de DEA obtenidos con *TESTU* no son esencialmente diferentes a los obtenidos con *TESTR.*

Tabla 1: Evaluación de las funciones de distancia, con un solo apellido

Función		DEA			JARO		BIGRAM	EDITEX	Ed.simple
Umbral		C	D	E	0,14	0,18	1,1	6	2
Subid. 1-*R*		7,9%	5,9%	3,8%	13,29%	6,55%	11,94%	8,67%	11,17%
Sobreid. *F*		0,005%	0,19%	0,77%	0,19%	0,77%	0,19%	0,38%	0,19%
β=1	Precisión *P*	99,995%	99,799%	99,206%	99,781%	99,182%	99,785%	99,58%	99,786%
	Eficacia *E*	96,05%	96,96%	97,72%	93,26%	96,34%	93,94%	95,48%	94,32%
	" *J*	95,89%	96,87%	97,68%	92,79%	96,23%	93,56%	95,28%	93,99%
	Salida	46,05%	47,14%	48,48%	43,45%	47,11%	44,12%	45,85%	44,51%
$\beta=10^3$	*P*	94,85%	33,12%	11,11%	31,33%	10,82%	31,67%	19,38%	31,85%
	E	99,987%	99,804%	99,227%	99,797%	99,224%	99,799%	99,61%	99,799%
	J	93,46%	48,99%	19,91%	46,03%	19,4%	46,59%	31,97%	46,89%
	Salida	0,097%	0,199%	0,86%	0,276%	0.862%	0,277%	0,47%	0,278%

En la tabla 1 figuran algunos resultados numéricos detallados relativos a *TESTR / CONTROLR*.

Dados los pequeños volúmenes de los ficheros de parejas de apellidos formadas al azar *CONTROLR* (519) y *CONTROLC* (1290) , ocurre que algún método no llega a dar ni una sola pareja sobreidentificada. Como una sobreidentificación nula no es realista y



produciría una *P* = 100%, se obtendrian resultados singulares. Por ello en estos casos de no aparición de falsos positivos, daremos a la sobreidentificación el valor 0,005%, equivalente a una pareja sobreidentificada por cada 20000.

En lo que queda de sección vamos a estimar el comportamiento de las funciones de distancia, para una supuesta BD con información de 4 millones de personas. De momento supondremos que solo trabajamos con el primer apellido. Para facilitar el acceso por medio de ese apellido, se dispone de un diccionario conteniendo todos los 100.000 distintos primeros apellidos ("correctos" o no) de esas personas.

Vamos a hacer una búsqueda aproximada de un apellido en ese diccionario e imaginemos que "realmente" existen en él unos 100 apellidos que pueden ser el buscado o sea que una respuesta "correcta" debería mostrar esos 100 apellidos (o el conjunto de personas que tienen esos primeros apellidos... unas 4000 como media). Llamamos factor β a la proporción entre el número de antropónimos que son anomalias o variantes del buscado y el resto. Entonces tendremos β = 100/(100.00-100) $\approx$ 0,001

Imaginemos que no estamos dispuestos a subidentificar (dejar de obtener en la respuesta) mas de unos 5 apellidos (unas 200 personas que podrian ser la buscada). O sea, queremos que la subidentificación sea inferior a 5% lo cual equivale a R > 95%. En cambio, somos más tolerantes en cuanto a la sobreidentificación, estando dispuestos a tolerar hasta unos 1500 falsos positivos, o sea que en la respuesta obtenida no queremos obtener más de 1500 apellidos que no puedan ser el buscado. Esto quiere decir una sobreidentificación $F <$ 1,5%. Mirando la figura 2 vemos que para satisfacer estos dos condicionantes no tenemos mas remedio que utilizar la función DEA y con el umbral E. Este punto tiene una subidentificación igual a 3,8% y una F = 0,77% (tabla1).

Aplicando las fórmulas que se presentan en LSI-02-60--R, encontramos que con este criterio de semejanza, DEA-E, podemos esperar obtener una respuesta total de unos 865 apellidos candidatos, de los cuales tan solo 96 podrían corresponder al apellido buscado. El resto, 769, son falsos positivos. Esto hace que la precisión sea muy baja, P = 11,1%, Y en la respuesta nos faltarán 4 apellidos que quisiéramos haber obtenido. Todo esto entra dentro de los límites que nos habíamos impuesto respecto a la sobreidentificación y la subidentificación.

Si utilizáramos la distancia de edición simple, con el umbral $\delta \leq$ 2, tendríamos una subidentificación del 11,17%, más del doble de la deseada (11 falsos negativos). Pero la sobreidentificación sería muy buena, tan solo el 0,19%, muy inferior al 0,77% de DEA-E. Y utilizando BIGRAMAS con umbral $\leq$ 1,1 obtenemos casi estos mismos resultados. Utilizando JARO con umbral 1,8 se obtiene también una F = 0,77% pero la subdentificación es 6,55%, casi el doble que la de DEA-E.

Si 769 falsos positivos nos parece una sobreidentificación excesiva, F = 0,77%, y queremos reducirla al valor F = 0,19% que es el de BIGRAMAS-1,1, el de JARO-1,4 o de la distancia simple $\delta \leq$ 2, podemos utilizar DEA pero con el umbral D. Ahora obtendremos solamente 190 apellidos no deseados, pero la subidentificación ha pasado de 3,8% a 5,9% lo cual está un poco por encima del límite que nos habíamos impuesto, aunque aun está muy por debajo del valor obtenido con la distancia simple o con BIGRAMAS-1,1 o con JARO-0,14. Para DEA-D (y β = 0,001) la precisión será P = 33,12%

Psicologicamente puede ser muy negativo el obtener respuestas que nos parezcan que no se corresponden con lo solicitado. En cambio la subidentificación puede pasar desapercibida ya que el usuario suele desconocer cuantas respuestas correctas se le han dejado de dar. Por ello en la mayoría de productos comerciales, especialmente en los dirigidos a un público no especializado, en donde se hagan búsquedas por aproximación (por ejemplo en la verificación ortográfica de los procesadores de texto) se procura minimizar la sobreidentificación, aunque aumente la subidentificación, y por ello se utiliza la distancia simple con $\delta \leq$ 1. En ciertas aplicaciones, la sobreidentificación de

DEA-D puede aún resultar psicologicamente inadecuada, pero tenemos tres umbrales mas estrictos, el A, el B y el C. Con ellos se logran precisiones superiores al 90%, pero las subidentificaciones son entoces superiores al 7%.

Normalmente, para buscar una persona no se suele utilizar solamente el primer apellido sino que se dispone de más datos, por ejemplo; la fecha de nacimiento, el otro apellido, el nombre, el sexo, etc., Esto puede reducir mucho la sobreidentificación resultante, pero esos otros datos también pueden tener errores con lo cual aumentará la subidentificación. Supondremos ahora que para la búsqueda utilizamos los dos apellidos. En la tabla 2 aparecen las principales cifras para ese caso, obtenidas con las fórmulas explicadas en LSI-02-60-R.

Tabla 2: Evaluación de las funciones de distancia, con dos apellidos

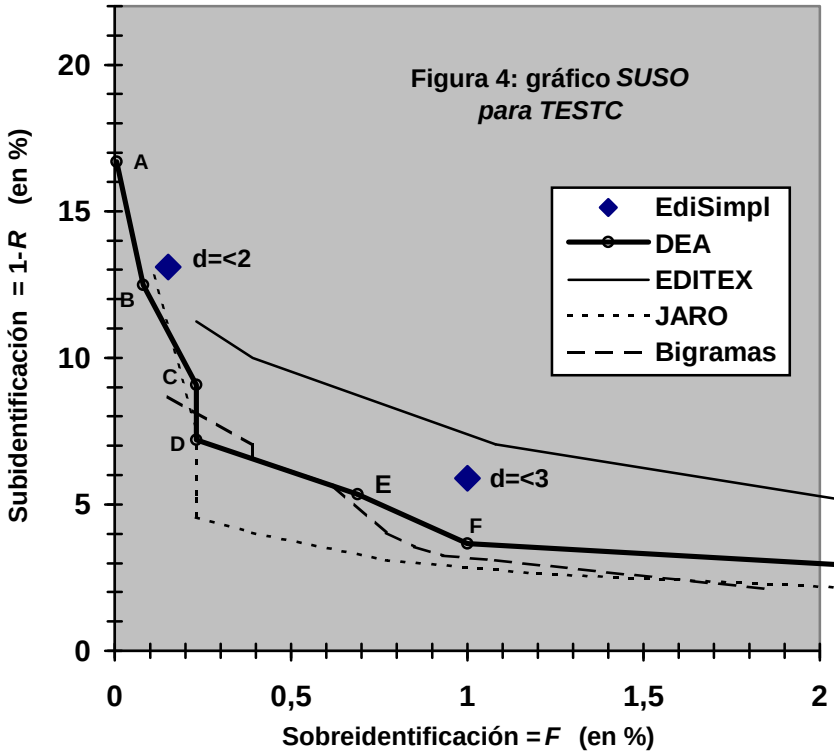
Función		DEA			JARO	BIGRAMA	EDITEX	Edición simple
Umbral		E	F	G	0,32	3	1,2	4
Subid. 1ap 1-R		3,8%	2,7%	1,73%	1,9%	1,56%	2,3%	2,5%
Sobreid. 1ap F		0,77%	2,7%	3,86%	6,55%	5,39%	14,45%	10%
Subid. 2aps 1-R_2		7,46%	5,327%	3,43%	3,76%	3,056%	4,54%	4,94%
$\beta=10^{-3}$	Sobr.2aps F_2	0,007%	0,078%	0,156%	0,441%	0,3%	2,11%	1,017%
	Preci.2aps P_2	1,236%	0,1212%	0,0617%	0,0218%	0,032%	0,0045%	0,009%
	Efica.2aps E_2	99,979%	99,877%	99,779%	99,453%	99,611%	97,676%	98,828%
	" " J_2	2,439%	0,242%	0,123%	0,043%	0,064%	0,009%	0,018%
	Salida ≈ F_2	0,007%	0,078%	0,156%	0,441%	0,3%	2,11%	1,017%

Ahora la subidentificación ha aumentado hasta un valor de aproximadamente el doble que en el caso de un solo apellido. DEA-E tiene una subidentificación (para dos apellidos) de 7,46%. Si nos parece demasiado alta, podremos reducirla utilizando un umbral mas generoso. Con el umbral G la subidentificación, 1-R_2 , queda reducida al 3,43%. Por otra parte la subidentificación disminuye y esa disminución depende del valor de β (ver LSI-02-60-R). Si suponemos que el valor de β (para un solo apellido) sigue siendo β = 0,001 (o sea $\beta_2 \approx$ 0,000001) la nueva sobreidentificación para DEA-G será F_2 = 0,156%.

Estamos suponiendo β = 0,001 lo cual quiere decir que suponemos $\beta_2 \approx$ 0,000001. Y con este factor tan bajo la precisión cae en picado. Como puede verse en la tabla 2 los valores de P_2 son ahora inferiores al 1%. Para lograr P_2 < 50% con esos factores β es necesario irse a umbrales mas pequeños, A, B o C, que producen subidentificaciones superiores al 10%.

Obsérvese en la tabla 2 que utilizando la función DEA, obtenemos mejores resultados que con el resto de funciones: para subidentificaciones similares, la sobreidentificación es mucho menor.

Para terminar con la evaluación de las distancias, utilizando el gráfico SUSO, hemos representado en la figura 4 los resultados de las pruebas con los ficheros norteamericanos *TESTC/CONTROLC*. Al no estar creado por nosotros no podemos garantizar la "objetividad" del fichero *TESTC*, pero segun la información facilitada por el *Bureau of Census* se trata de un fichero con casos reales (no inventados ni creados automaticamente). De los resultados obtenidos queremos destacar dos puntos importantes:



- La función JARO da unos resultados excelentes (similares a los que da DEA en los ficheros españoles) pero hay que tener en cuenta que el fichero *TESTC* es precisamente el fichero con el trabajó Jaro [Jar89] para ajustar su método.

- DEA da en TESTC unos buenos resultados, a pesar de que sus parámetros están basados solamente en el contexto español.

De todo ello deducimos que si se hiciera un función como la DEA con los parámetros (costes en función de las letras y las posiciones, y umbrales en función de las longitudes) adaptados al contexto norteamericano, probablemente se podrían obtener mejores resultados que con JARO.

4.3.- Gráfico *P/R*

Podemos evaluar las funciones utilizando las métricas R (*recall*) y P (precisión), en lugar de utilizar la subidentificación y la sobreidentificación. Podemos utilizar el gráfico P/R en lugar del gráfico *SUSO*, ya que:

$$R = 1 - \text{subidentificación}$$

$$P = \frac{\beta R}{\beta R + F}$$

Vemos que la P depende también del factor β. Pero cuando en la literatura se hacen evaluaciones de los procesos de búsqueda aproximada, no se suele tener en cuenta este hecho, lo cual es como si $\beta = 1$, valor muy superior al que se suele tener en la realidad. Y en consecuencia las precisones P citadas en la literatura suelen llevar a engaño al ser muchísimo mayores que las "reales".

En la figura 5 aparece el gráfico P/R para la pareja de ficheros *TESTR/ CONTROLR* en el supuesto de que β = 0,001. No hemos representado las lineas de las funciones BIGRAMAS y EDITEX porque tienen un comportamiento casi idéntico al de JARO (ver la tabla 1).

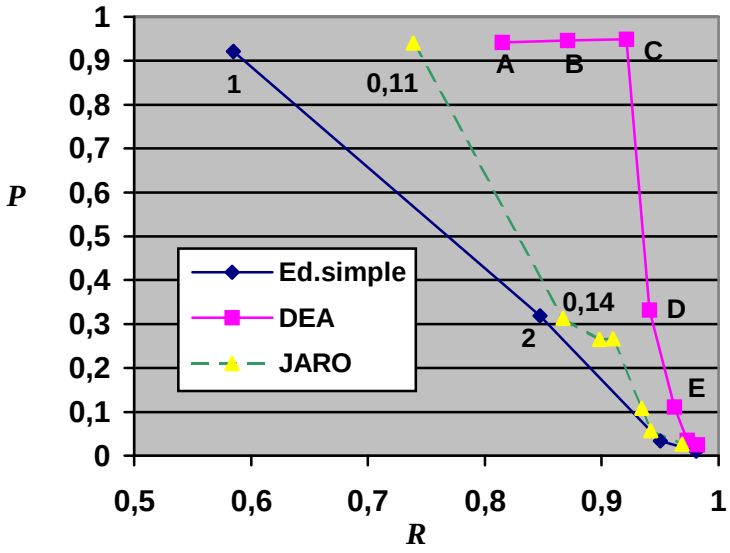


Figura 5: Gráfico P/R para $\beta = 0{,}001$

Vemos que para obtener precisiones superiores al 50% (con $\beta = 0{,}001$) es necesario aceptar valores de R inferiores a 0,95 o sea hay que aceptar subidentificaciones superiores al 5% como ya hemos visto en la sección anterior. La tensión entre P y R se agudiza al usar los dos apellidos, de forma que para obtener valores de P_2 del orden del 50% es necesario aceptar subidentificaciones superiores al 10%. Claro que con factores β mas grandes la precisión aumenta. Por ejemplo, con DEA-E y un factor $\beta =$ 0,01 obtenemos una precisión P = 55% con una subidentificación del 3,8%, pero esos factores β no son muy frecuentes en la práctica.

4.4- Eficacias E y J.

En LSI-02-60-R se explican las propuestas que en el mundo del *Information Retrieval* se han hecho para condensar en una sola métrica la bondad de una función de semejanza; se trata de las métricas de eficacia E y J. Una métrica de eficacia puede ser util para analizar el comportamiento de las funciones, pero no puede substituir al manejo de parejas de métricas como F y 1-R o bien P y R. Nos limitaremos aquí a comentar con ayuda de gráficos, algunos de los valores de E y J obtenidos con los ficheros *TESTR / CONTROLR* (ver tablas 1 y 2).

En la figura 6 representamos la E para la distancia DEA en función del umbral para uno y dos apellidos, y para $\beta = 1$ *y* $\beta = 0{,}001$.

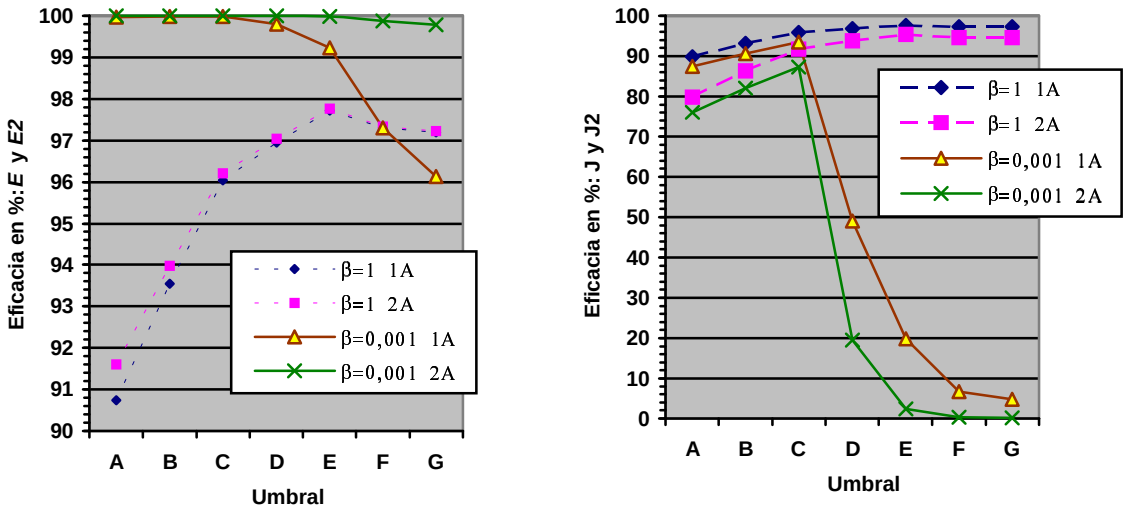
Recordemos que la E es la media entre R y 1-F, ponderada con β. Para valores pequeños de β, resulta $E \approx 1\text{-}F$ y por lo tanto si trabajamos con F muy pequeñas resulta una E muy alta. Como habitualmente trabajamos con $\beta \leq$ 0,001 y con F pequeñas, la eficacia E se suele situar por encima del 95% y al usar los dos apellidos se situa por encima del 99%. Para el caso de un solo apellido, el umbral DEA-F da siempre la misma eficacia E, sea cual sea el valor de β.

El que la eficacia E sea muy alta no impide que la precisión sea muy baja. Por ejemplo; para DEA-E con $\beta = 0{,}001$, tendremos una eficacia E =99,80% pero una precisión baja, P =33,12%. Si utilizáramos los dos apellidos, la eficacia sería aun mayor E_2 =99,97% pero

la precisión sería mucho menor P_2=1,23%. Para tener en cuenta a la vez la precisión y la subidentificación, se utiliza la eficacia J definida como la media armónica entre la precisión P y el *recall* R. También se puede ver como la diferencia simétrica normalizada entre el conjunto real de *parejas-con-error* y el conjunto de parejas identificadas como tales.

En la figura 7 representamos la J para la distancia DEA en función del umbral.

Las métricas de eficacia no son prácticas para hacer comparaciones entre distancias que no tengan umbrales homogéneos. Habrá que substituir los umbrales por una métrica como R o F . Por ejemplo, compararemos los valores de E para un mismo valor de R (o F). En las tablas 1 y 2 vemos que DEA da siempre, para una F o R similares, valores de las eficacias E y J más altos que el resto de las funciones de comparación de caracteres estudiadas aquí.



Figuras 6 y 7 : La eficacia E y la eficacia J, para la distancia DEA

4.5.- Conclusiones

En LSI-02-60-R presentamos métricas utilizando como ejemplo la distancia simple, con un corpus distinto al que utilizamos aquí y allí se ve que esta famosa y tradicional distancia no es muy adecuada para las aplicaciones en las que se necesita una búsqueda aproximada de antropónimos españoles. De los tres únicos umbrales que podrían dar resultados aceptables, $\delta \leq 1$ $\delta \leq 2$ $\delta \leq 3$, el primero da una enorme subidentificación, el segundo subidentifica demasiado pero en determinadas circunstancias podría ser aceptable, y el tercero sobreidentifica enormemente. No existen valores intermedios, lo cual no permite hacer ajustes variando el umbral.

Hemos visto que con los corpus de pruebas, el único umbral aceptable para la distancia simple es el $\delta \leq 2$ si aceptamos una subidentificación que sobrepasa el 11%. El resto de funciones explicadas aquí, excepto DEA, tienen un comportamiento similar entre sí. Tienen mayor densidad de valores que la distancia simple, dentro del intervalo usual, lo que las hace más útiles, pero los valores no son significativamente mejores.

En cambio DEA aporta mejoras significativas. Para un mismo nivel de subidentificación, y dentro del intervalo util, da una sobreidentificación 70% a 80% inferior. Y para un mismo

nivel de sobreidentificación, da una subidentificación 40% a 55% inferior. Para dos apellidos las conclusiones son similares. DEA da siempre, para una F o una R similares, valores mas altos de las eficacias E, J, E_2 y J_2 que el resto de las funciones de comparación de caracteres estudiadas aquí.

Para los valores habituales de β, el umbral DEA-C parece el mas adecuado a la mayoria de situaciones.

5.- Salida y factor β: Pruebas de búsqueda con DEA

En LSI-02-59-R se muestran resultados de pruebas de búsqueda realizadas con nuestro método DEA, usando el fichero *APELLIDOS* (73724 apellidos) y buscando diez apellidos españoles escogidos al azar. El número de apellidos obtenidos en la búsqueda es el siguiente:

Umbral	Apellidos semejantes		
	min	med	max
A	6	24,4	58
B	9	38,8	88
C	**20**	**69,3**	**126**
D	36	122,2	223
E	54	274,9	583
F	170	715,2	1690
G	279	1460,6	2878

Podemos calcular el factor β y los errores cometidos por DEA en estas pruebas. Tomemos por ejemplo el umbral C. Según la tabla 1, este umbral tiene una subidentificación del 7,9% y una sobreidentificación del 0,005%. Por lo tanto (ver LSI-023-60-R) tendremos los siguientes valores medios para la búsqueda de un apellido escogido al azar:

- Apellidos contenidos en *APELLIDOS* y que son "realmente" semejantes al buscado = 71,25

- Apellidos semejantes que no aparecen en la salida al buscar con DEA-C = 5,63

- Apellidos no semejantes que aparecen en la salida al buscar con DEA-C = 3,68

A partir de estos valores se deduce que el factor β medio es 0,0009 , lo cual es coherente con nuestras afirmaciones acerca de los valores de β mas usuales.

Referencias

[Alb67] Alberga C.N.: String similarity and mispellings. *Comm. ACM 10*,5 (May 1967) 302-313.

[All87] Allen M. Automatic correction to misspelled names: a fourth generation language approach. *Comm. ACM 30*,3. (March 1987) 224-228.

[Ami00] Amir,A. et al.: Faster Algorithms for String Matching with K mismatches, *SODA-2000* (2000) 794-803.

[Ars99] Arslan, A.N.& Egecioglu, O.: An Efficient Uniform-Cost Normalized Edit Distance Algorithm. *SSPIPRS-1999* (1999).

[Bun95] Bunke,H. & Csirik,J.: Parametric String Edit Distance and its Application to Pattern Recognition. *IEEE Trans.on Systems, Man, and Cybernetics, 25,*1 (Enero 1995) 202-206.

[Cam81] Camps, R.: *Búsqueda por semejanza ortográfica o fonética*. Tesina. FIB / UPC. (1981).

[Dag95] Dagan, I et al: Contextual word similaity and estimation from sparse data. *Computer Speech and Language 9* (1995) 123-152.

[Dam64] Damerau, F.J.: A technique for computer detection and correction of spelling errors. *Comm. ACM 7*,3 (Mar. 1964) 171-176.

[Fau64] Faulk R.: An inductive approach to language translation. *Comm. ACM 7,*11. (Nov. 1964) 647-653.

[Gad90] Gadd, T.N.: PHONIX: the algorithm. *Program vol 24,* 4 (1990) 363-366.

[Gal89] Galil, Z.. & Park,K.: An improved algorithm for approximate string matching. *SIAM J.Computing 19-6* (1990) 989-999.

[Jar89] Jaro, M.A. : Advances in Record-Linkage methodology as applied to matching the 1985 Census of Tampa, Florida. *Journal of the American Statistical Association* (1989)

[Jou93] Jouvet, D. et al.: Speaker-independent spelling recognition over the telephone. ¿?? (1993) 235-238

[[Kas84] Kashyap, R.L. & Oommen, B.J.: Spelling correction using probabilistic methods. *Patt. Recog. Lett. vol 2*, 3 (Mar. 1984) 147-154.

[Kuk92] Kukich, K.: Techniques for Automatically Correcting Words in Text. *ACM Comp. Surveys, vol 24,* 4 (Dec. 1992) 377-439.

[Kur96] Kurtz,R.: Approximate string searching under weighted edit distance. *Proceed. 3thd SAWSP* (1996).

[Lee97] Lee, J. et al: Efficient algorithms for approximate string matching with swaps. *8th Annual Symposium CPM-97 Proc.* (1997) 28-39

[Leu97] Leung, V.J.: The undecidability of the unrestricted modified edit distance. *Theoretical Computer Science 180(1&2)* (1997) 203-215.

[Lev66] Levenshtein V.I.: Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics- Doklady 10,*8. (Feb. 1966) 707-710.

[Lin98] Lin, D.: An information-theoretic definition of similarity. *Proceed. Intl. Conf. on Machine Learning* (1998).

[Lop99] Lopresti, D. & Wilfong, G.: Cross-Domain Approximate String Matching. *IEEE String Processing and Information Retrieval Symposium* (1999) 120-127.

[Low75] Lowrance, R. & Wagner, R,A.: An Extension of the String-to-String Correction Problem. *Journ. ACM 22,*2 (1975). 177-183.

[Mar93] Marzal, A. & Vidal, E.: Computation of normalized edit distance and applications, *IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 15* (1993) 926-932.

[Mor82] Mor M. Fraenkel A.S.: A hash code method for detecting and correcting spelling errors. *Comm. ACM 25*.12. (Dec. 1982) 935-938.

[Nav01] Navarro, G.: A guided tour to approximate string matching. *ACM Computing Surveys 33,*1 (Marzo 2001) 32-88

[Oom95] Oommen, B.J.: String alignment with substitutions, insertions, deletions, squashing and expansion operations. *Information Sciences, vol. 8*, 3 (1995) 89-107.

[Oom96] Oommen, B.J. & Zhang, K.: The normalized string editing problem revisited. *IEEE Trans. on Pattern Analysis and Mach. Intelligence, vol. 18*, 6 (June 1996) 669-672.

[Oom97a] Oommen, B.J.& Loke, R.K.S.: Pattern recognition of strings with substitutions, insertions, deletions and generalized transpositions. *Pattern Recognition 30*, 5 (1997) 789-800.

[Oom97b] Oommen, B.J.& Loke, R.K.S.: On using parametric string distances and vector quantization in designing syntattic pattern recognition systems. *0-7803-4053-1/97 IEEE* (1997) 511-517.

[Pet00] Petrakis, E.G.M & Tzeras,K.: Similaryty Searching in the CORDIS Text Database *Software Practice and Experience, 13*,(Nov.2000) 1447-1464.

[Pev95]Pevzner,P.A. & Waterman, M.S.: Multiple filtration and approximate pattern matching. *Algorithmica, 13* (1995) 135-154.

[Pfa80] Pfaltz,J.L., Berman,W.J. & Cagley,E.M.: Partial-match retrieval using indexed descriptor files. *Commun.ACM. 23*,9 (Sept. 1980), 522-528.

[Pfe95] Pfeifer, U. et al.: Searching proper names in databases. *Hypertext -Information retrieval-Multimedia. Procee. HIM'95* (Oct. 1995) 259-275.

[Ris98] Ristad, E.S. & Yianilos, P.N.: Learning String-Edit-Distance*. IEEE Trans. on Pattern Analysis and Machine Intelligence, 20,5* (Mayo 1998) 522-532.

[Sal89] Salton, G.: *Automatic Text Processing*. Addison-Wesley. (1989).

[Sen96] Seni, G. et al: Generalizing edit distance to incorporate domain information: handwritten text recognition as a case study. *Patt. Recog. 29*, 3 (1996) 405-414

[Ver88] Veronis, J: Computerized correction of phonografic errors. *Computers in Humanities. 22* (1988) 43-56.

[Vid95] Vidal, E. et al : Fast computation of normalized edit distances. *IEEE Trans. on Pattern Analysis and Mach. Intelligence vol.17,* 9 (Sept. 1995) 899-902.

[Vil96] Villarrubia, L. et al.: Context-dependent units for vocabulary-independent spanish speech recognition. *IEEE ICASSP-96 vol 1* (1996) 451-454.

[Wag74] Wagner, R.A. & Fischer. M.J.: The String-to-String Correction Problem. *Journ. ACM 21*,1 (1974) 168-178.

[Wei95] Weigel, A. et al: Lexical post-processig by heuristic search and automatic determination of the edit costs. *ICDAR-95* (1995) 857-860.

[Win95] Winkler, W.E.: Matching and record-linkage, cap.20 de *Business Survey Methods* (1995) 355-384. John Wiley & Sons.

[Zav97] Zavrel, J. & Daelemans, S.W.: Memory-based learning: Using similarity for smoothing. *8th EACL Conf.* (1997) 436-442.

[Zob96] Zobel, J. & Dart, P.: Phonetic string matching: Lessons from Information Retrieval. *Proceed.19th Annual Intl. ACM SIGIR Conf. on R&D in IR.* (Aug.1996) 166-173.