Funciones de Similitud sobre Cadenas de Texto: Una Comparación Basada en la Naturaleza de los Datos

Iván Amón
Universidad Pontificia Bolivariana
ivan.amon@upb.edu.co

Claudia Jiménez
Universidad Nacional de Colombia
csjimene@unal.edu.co

Resumen

La detección de duplicados hace referencia al conflicto que se presenta en los datos cuando una misma entidad del mundo real aparece representada dos o más veces a través de una o varias bases de datos, en registros o tuplas con igual estructura pero sin un identificador único y presentan diferencias en sus valores. Múltiples *funciones de similitud* han sido desarrolladas para detectar cuáles cadenas son similares mas no idénticas, es decir, cuáles se refieren a una misma entidad. En el presente artículo se compara, mediante una métrica de evaluación llamada *discernibilidad*, la eficacia de nueve de estas *funciones de similitud* sobre cadenas de texto (Levenshtein, Brecha Afín, Smith-Waterman, Jaro, Jaro-Winkler, *Bi-grams*, *Tri-grams*, Monge-Elkan y SoftTF-IDF) usando para ello seis situaciones problemáticas (introducción de errores ortográficos, uso de abreviaturas, palabras faltantes, introducción de prefijos/sufijos sin valor semántico, reordenamiento de palabras y eliminación/adición de espacios en blanco). Los resultados muestran que algunas funciones de similitud tienen a fallar en ciertas situaciones problemáticas y que ninguna es superior al resto en todas ellas.

Abstract

The duplicate detection refers to the conflict that occurs in the data when an entity from the real world is represented with different values, two or more times in one or more databases of tuples with the same structure but without a unique identifier. Multiple similarity functions have been developed to detect which data are similar but not identical, i.e. which of them refer to the same entity. This paper compares, through an evaluation metric called discernability, the effectiveness of nine of these functions on data similarity (Levenshtein, Affine Gap, Smith-Waterman, Jaro, Jaro-Winkler, Bi-grams, tri-grams, and Monge-Elkan SoftTF-IDF) by using six common problems (introduction of misspellings, use of abbreviations, missing words, introduction of prefixes/suffixes with no semantic value reordering of words, and elimination/addition of blank spaces). The results show that some similarity functions fail in certain problems and none of them is globally better than the others.

Palabras Clave

Calidad de datos, Limpieza de datos, Detección de Duplicados, Funciones de Similitud.

Key Words

Data Quality, Data Cleansing, Record Linkage, Data Deduplication, Similarity Functions.

Funciones de Similitud sobre Cadenas de Texto: Una Comparación Basada en la Naturaleza de los Datos

1. Introducción

En la actualidad, las bases de datos juegan un papel fundamental en la planeación y la toma de decisiones en todo tipo de instituciones. Sin embargo, es común que los datos almacenados presenten errores o conflictos que pueden conducir a decisiones erróneas, con graves efectos como pérdida de tiempo, dinero y credibilidad. Uno de tales conflictos se da cuando una misma entidad del mundo real aparece representada dos o más veces (duplicada) a través de una o varias bases de datos, en tuplas con igual estructura, que no comparten un identificador único (por ejemplo el número de cédula de ciudadanía) y presentan diferencias textuales en sus valores. Así, en cierta base de datos una persona puede aparecer representada como

Nombre	**e-mail**
Jorge Eduardo Rodríguez López*	jorge.rodriguez@gmail.com

Mientras que en otra, debido, por ejemplo, a errores de digitación, la misma persona puede aparecer como

Nombre_persona	**email_persona**
Jorje Eduardo Rodrigues Lópes	jorge.rodrigues@jmail.com

El proceso que detecta este conflicto se conoce con múltiples nombres: *record linkage* o *record matching* entre la comunidad estadística; *database hardening* en el mundo de la Inteligencia Artificial; *merge-purge*, *data deduplication* o *instance identification* en el mundo de la bases de datos; otros nombres como *coreference resolution* y *duplicate record detection* también son usados con frecuencia. Aquí se utilizará el término genérico *detección de duplicados*.

El proceso de *record linkage* fue primero planteado por Dunn (1946). Algunos fundamentos probabilísticos fueron luego desarrollados por Newcombe *et al.* (1959), (Newcombe y Kennedy, 1962), siendo formalizados por Fellegi y Sunter (1969) como una regla de decisión probabilista. Algunas mejoras de este modelo han sido propuestas por Winkler (1989, 1990, 1993, 2000). En su forma más simple, dicho proceso es como sigue: dado un conjunto R de registros 1) Se define un umbral real $\theta \in [0,1]$. 2) Se compara cada registro de R con el resto 3) Si la *similitud* entre una pareja de registros es mayor o igual que θ, se asumen duplicados; es decir, se consideran representaciones de una misma entidad real.

Entonces es necesaria alguna *función de similitud* que, dados dos registros con la misma estructura, devuelva un número real en el intervalo $[0,1]$: igual a uno si ambos registros son idénticos y menor entre más diferentes sean. Tal valor depende de la similitud entre cada pareja de atributos respectivos. Luego, es necesaria otra función de similitud al nivel de atributo (no de

* En algunos países de América Latina, es común que los nombres de las personas consten de dos nombres y dos apellidos.

registro), siendo frecuente en este contexto tratar los atributos, sin importar su tipo de datos, como cadenas de texto (*strings*). Dichas funciones han sido tema de investigación por años. Actualmente existen diversas propuestas, las cuales pueden ser clasificadas en dos categorías: basadas en caracteres y basadas en palabras completas o *tokens* (Elmagarmid *et. al.*, 2007). Las primeras consideran cada cadena como una secuencia ininterrumpida de caracteres; las segundas como un conjunto de subcadenas delimitadas por caracteres especiales como espacios en blanco, comas y puntos; esto es, consideran una cadena como un conjunto de *tokens* y calculan la similitud entre cada pareja de *tokens* mediante alguna función basada en caracteres.

El valor de un atributo puede aparecer representado de muchas formas diferentes. Por ejemplo, el nombre "Jorge Eduardo Rodríguez López" también puede aparecer como "Rodríguez López Jorge Eduardo", "Jorge E Rodríguez López" o "Jorge Eduardo Rodríguez L."; con errores de escritura, como "Jorje Eduardo Rodrigues Lópes"; con información adicional, como "PhD Jorge Eduardo Rodríguez López", entre otras. Las causas pueden ser muchas: restricciones de formato, de longitud y/o en el conjunto de caracteres permitidos, errores humanos al capturar los datos, errores que surgen integrando bases de datos diferentes o haciendo migración entre sistemas, modelos de datos mal diseñados, entre otras causas.

Por todo lo anterior, Elmagarmid *et al.* (2007) concluyen que la gran cantidad de formas en que el valor de un mismo atributo puede aparecer representado convierten la elección de la función de similitud en todo un problema que requiere aún mayor investigación. Se han realizado algunos estudios comparativos: Christen (2006) compara la eficacia de algunas funciones de similitud sobre nombres personales. Yancey (2006) lo hace sobre nombres personales extraídos del censo realizado en 2000 en Estados Unidos. Cohen *et al.* (2003) utilizan diversos conjuntos de datos: nombres personales, de animales, de empresas, de videojuegos, de parques, entre otros. Moreau *et al.* (2008) se centran en nombres de ubicaciones, empresas y personas. Sin embargo, ninguno de estos trabajos compara la eficacia de las técnicas basándose en distintas situaciones problemáticas como las planteadas en el presente trabajo, esto es, con base en las variaciones textuales presentes en las representaciones de una misma entidad.

Para este estudio comparativo se utilizó una metodología basada en casos, con conjuntos de datos, especialmente diseñados para cada una de las seis situaciones problemáticas estudiadas (introducción de errores ortográficos, uso de abreviaturas, palabras faltantes, introducción de prefijos/sufijos sin valor semántico, reordenamiento de palabras y eliminación/adición de espacios en blanco). Las nueve funciones de similitud motivo de comparación en este trabajo (Levenshtein, Brecha Afín, Smith-Waterman, Jaro, Jaro-Winkler, *Bi-grams*, *Tri-grams*, Monge-Elkan y SoftTF-IDF) fueron comparadas mediante una métrica de evaluación llamada *discernibilidad*, buscando determinar la eficacia de las funciones ante cada situación problemática.

El resto del presente artículo está organizado como sigue: la sección 2 describe las funciones de similitud sobre cadenas de texto motivo de comparación en este artículo. La sección 3 describe la técnica de evaluación utilizada. La sección 4 describe la metodología utilizada para la comparación. La sección 5 muestra los resultados obtenidos y en la sección 6 se presentan las conclusiones y trabajos futuros.

2. Funciones de Similitud sobre cadenas de texto

La notación para el presente artículo es la siguiente: Σ denota algún conjunto finito ordenado de caracteres y Σ^* el conjunto de cadenas formadas por la concatenación de cero o más caracteres de Σ. A y B denotan dos cadenas de texto de longitud n y m definidas sobre Σ^*, donde $n \geq m$. a_i representa algún carácter de A para $1 \leq i \leq n$, y b_j es análogo respecto a B.

Cuando se habla de la distancia entre dos cadenas, no se está hablando exactamente de la similitud entre ellas ya que una magnitud puede calcularse a partir de la otra. Una distancia real $d \in [0,1]$, donde 0 indica que ambas cadenas son idénticas y 1 que no tienen ni un solo carácter en común, equivale a una similitud $s = 1 - d$, lo que quiere decir que a mayor distancia menor es la similitud y viceversa.

2.1 Funciones de Similitud Basadas en caracteres

Estas funciones de similitud consideran cada cadena de caracteres como una secuencia ininterrumpida de caracteres. En este trabajo se consideraron cinco de las más conocidas.

2.1.1 Distancia de edición

La distancia de edición entre dos cadenas de texto A y B se basa en el conjunto mínimo de operaciones de edición necesarias para transformar A en B (o viceversa). Las operaciones de edición permitidas son eliminación, inserción y substitución de un carácter.

En el modelo original, cada operación de edición tiene costo unitario, siendo referido como distancia de Levenshtein (Levenshtein, 1966). Needleman y Wunsch (1970) lo modificaron para permitir operaciones de edición con distinto costo, permitiendo modelar errores ortográficos y tipográficos comunes. Por ejemplo, en el idioma español es frecuente encontrar la letra “n” en lugar de la “m” (Ramírez y López, 2006), entonces, tiene sentido asignar un costo de sustitución menor a este par de caracteres que a otros dos sin relación alguna.

Los modelos anteriores al ser absolutos y no relativos tienen una desventaja: la distancia entre dos cadenas carece de algún tipo de normalización. Por ejemplo, tres errores son más significativos entre dos cadenas de texto de longitud cuatro que entre dos cadenas de longitud 20. Por esto, se han propuesto varias técnicas de normalización. Las más simples normalizan dividiendo por la longitud de la cadena más larga (Christen, 2006) o por la suma de la longitud de ambas cadenas (Weigel y Fein, 1994). Más recientemente, Yujiang y Bo (2007) desarrollaron la primera técnica de normalización que satisface la desigualdad triangular[1].

La técnica de normalización propuesta por Marzal y Vidal puede ser ejecutada con un orden computacional de $O(n^2 m)$, siendo reducida a $O(nm)$ en (Vidal *et. al.*, 1995) mediante programación fraccionaria.

2.1.2 Distancia de brecha afín

Como se muestra más adelante, la distancia de edición y otras funciones de similitud tienden a

[1] En el sentido matemático estricto, toda medida de distancia (métrica) debe satisfacer la desigualdad triangular: la distancia directa para ir del punto *x* al *z* nunca es mayor que aquella para ir primero del punto *x* al *y* y después del *y* al *z*. Sin embargo, esta propiedad carece de importancia para procesos de detección de duplicados, y por lo tanto no será discutida.

fallar identificando cadenas equivalentes que han sido demasiado truncadas, ya sea mediante el uso de abreviaturas o la omisión de *tokens* ("Jorge Eduardo Rodríguez López" vs "Jorge E Rodríguez"). La distancia de brecha afín ofrece una solución al penalizar la inserción/eliminación de k caracteres consecutivos (brecha) con bajo costo, mediante una función afín $\rho(k) = g + h \cdot (k - 1)$, donde g es el costo de iniciar una brecha, h el costo de extenderla un carácter, y $h \ll g$ (Gotoh, 1982). Bilenko y Mooney (2003) describen un modelo para entrenar automáticamente esta función de similitud a partir de un conjunto de datos.

La distancia de brecha afín no normalizada puede ser calculada con un orden computacional de $O(nm)$ mediante el algoritmo de programación dinámica propuesto por Gotoh (1982).

2.1.3 Similitud Smith-Waterman

La similitud Smith-Waterman entre dos cadenas A y B es la máxima similitud entre una pareja (A', B'), sobre todas las posibles, tal que A' es subcadena de A y B' es subcadena de B. Tal problema se conoce como *alineamiento local*. El modelo original de Smith y Waterman define las mismas operaciones de la distancia de edición, y además permite omitir cualquier número de caracteres al principio o final de ambas cadenas (Smith y Waterman, 1981). Esto lo hace adecuado para identificar cadenas equivalentes con prefijos/sufijos que, al no tener valor semántico, pueden ser descartados. Por ejemplo, "PhD Jorge Eduardo Rodríguez López" y "Jorge Eduardo Rodríguez López, Universidad Nacional de Colombia" tendrían una similitud cercana a uno.

Es posible normalizar la similitud de Smith-Waterman con base en la longitud de la cadena de mayor longitud, la de menor longitud o la longitud media de ambas cadenas (Christen, 2006), que corresponden a los coeficientes de Jaccar, Overlap y Dice respectivamente. La similitud Smith-Waterman puede ser calculada en $O(nm)$ mediante el algoritmo de Smith-Waterman (1981). Baeza-Yates y Gonnet (1992) presentan un algoritmo que toma $O(n)$ para verificar si la similitud Smith-Waterman entre dos cadenas es menor que cierta constante k.

2.1.4 Similitud de Jaro

Jaro (1976) desarrolló una función de similitud que define la trasposición de dos caracteres como la única operación de edición permitida. Los caracteres no necesitan ser adyacentes, sino que pueden estar alejados cierta distancia d que depende de la longitud de ambas cadenas.

Winkler (1990) propone una variante que asigna puntajes de similitud mayores a cadenas que comparten algún prefijo, basándose en un estudio realizado por Pollock y Zamora (1984). Cohen *et al.* (2003) proponen un modelo basado en distribuciones Gaussianas. Yancey (2006) compara la eficacia de la similitud de Jaro y algunas de sus variantes. La similitud de Jaro puede ser calculada con un orden de complejidad de $O(n)$.

2.1.5 Similitud de q-grams

Un *q-gram*, también llamado *n-gram*, es una subcadena de longitud q (Yancey, 2006). El principio tras esta función de similitud es que, cuando dos cadenas son muy similares, tienen muchos *q-grams* en común.

Es común usar *uni-grams* ($q = 1$), *bi-grams* o *di-grams* ($q = 2$) y *tri-grams* ($q = 3$). Es posible agregar $q - 1$ ocurrencias de un carácter especial (no definido en el alfabeto Σ original) al

principio y final de ambas cadenas. Esto llevará a un puntaje de similitud mayor entre cadenas que compartan algún prefijo y/o sufijo, aunque presenten diferencias hacia el medio (Yancey, 2006). Un modelo alternativo se conoce bajo el nombre de *k-skip-q-grams*: *q-grams* que omiten *k* caracteres adyacentes. Por ejemplo, la cadena "Peter" contiene los *bi-grams* "Pe", "et", "te", "er" y los *1-skip-2-grams* "Pt", "ee", "tr".

Mediante el uso de funciones hash adecuadas, la similitud de *q-grams* puede ser calculada en $O(n)$ (Ukkonen, 1997), (Cohen, 1997). Ukkonen (1997) presenta un algoritmo alternativo basado en autómatas de sufijos que también toma $O(n)$.

2.2 Funciones de Similitud Basadas en tokens

Estas funciones de similitud consideran cada cadena como un conjunto de subcadenas separadas por caracteres especiales, como por ejemplo espacios en blanco, puntos y comas, esto es, como un conjunto de *tokens*, y calculan la similitud entre cada pareja de *tokens* mediante alguna función de similitud basada en caracteres. En esta sección se cubren dos de las funciones basadas en *tokens* más comunes: Monge-Elkan y coseno TF-IDF (Elmagarmid *et. al.*, 2007).

2.2.1 Similitud de Monge-Elkan

Dadas dos cadenas A y B, sean $\alpha_1, \alpha_2 \dots \alpha_K$ y $\beta_1, \beta_2 \dots \beta_L$ sus *tokens* respectivamente. Para cada *token* α_i existe algún β_j de maxima similitud. Entonces la similitud de Monge-Elkan entre A y B es la similitud máxima promedio entre una pareja (α_i, β_j) (Monge y Elkan, 1996).

Gelbukh *et al.* presentan un modelo basado en la media aritmética generalizada, en lugar del promedio, el cual supera al modelo original sobre varios conjuntos de datos. El algoritmo para la similitud de Monge-Elkan tiene un orden computacional $O(nm)$ (Gelbukh *et. al.*, 2009).

2.2.2 Similitud coseno TF-IDF

Dadas dos cadenas A y B, sean $\alpha_1, \alpha_2 \dots \alpha_K$ y $\beta_1, \beta_2 \dots \beta_L$ sus *tokens* respectivamente, que pueden verse como dos vectores V_A y V_B de K y L componentes. Cohen (1998) propone una función que mide la similitud entre A y B como el coseno del ángulo que forman sus respectivos vectores.

La similitud coseno TF-IDF no es eficaz bajo la presencia de variaciones a nivel de caracteres, como errores ortográficos o variaciones en el orden de los *tokens*. Por ejemplo, las cadenas "Jorge Rodríguez López" y "López Jorge Eduardo" tendrían similitud cero. Cohen *et al.* (2003) proponen una variante llamada SoftTF-IDF para solucionar este problema, que tiene en cuenta parejas de *tokens* (α_i, β_j) cuya similitud es mayor que cierto umbral (mediante alguna función de similitud basada en caracteres).

3 Evaluación de funciones de similitud sobre cadenas de texto

Para evaluar y comparar funciones de similitud sobre cadenas de texto, es usada comúnmente la clásica *mean average precisión* (MAP) (Heuser *et. al.*, 2007). MAP se obtiene a partir de un tipo especial de consultas conocidas como *top-k queries*, en las cuales se retornan las *k* instancias más similares a la cadena buscada. Entonces, para una función de similitud particular, MAP mide la capacidad que tiene para mover las cadenas relevantes a la buscada dentro de los primeros *k* resultados. Sin embargo, en el proceso de detección de duplicados se utilizan *consultas de rango*, en las cuales se retornan todas las cadenas cuya similitud con la buscada es mayor que cierto

umbral *t* previamente definido. Una técnica de evaluación acertada debe tener en cuenta (Heuser *et. al.*, 2007):

1. Si la función de similitud consigue separar correctamente cadenas relevantes de irrelevantes. El grado de separación entre cadenas relevantes e irrelevantes. Una buena función de similitud no sólo debe distinguir entre unos y otros, sino ponerlos dentro de una distancia razonable de forma que creen dos conjuntos distintos claramente definidos.
2. La variación del umbral *t* seleccionado, pues la distribución de valores de similitud puede variar de un conjunto de datos a otro.

MAP sólo tiene en cuenta el primer aspecto. Por esta razón, en el presente trabajo se utiliza una métrica de evaluación propuesta por Da Silva *et al.* (2007), llamada *función de discernibilidad* especialmente diseñada para consultas de rango.

3.1 Función de discernibilidad

La discernibilidad de una función de similitud se calcula como (da Silva *et. al.*, 2007):

$$\frac{c_1}{c_1 + c_2}\left(t_{max}^{óptimo} - t_{min}^{óptimo}\right) + \frac{c_2}{c_1 + c_2}\left(\frac{F_{max}}{2|Q|}\right) \quad (1)$$

La componente $[t_{min}^{óptimo}, t_{max}^{óptimo}]$ determina el intervalo de umbrales que mejor separa cadenas relevantes de irrelevantes, pues actúa como indicador de la distancia entre éstas y puede ser calculado por cualquiera de los dos algoritmos propuestos por da Silva. El término $F_{max}/2|Q|$ indica el porcentaje de separaciones correctas logrado. Los coeficientes c_1 y c_2 permiten controlar la importancia de cada uno de los dos aspectos anteriores. Independientemente de los valores dados a c_1 y c_2, los valores de discernibilidad siempre caerán en el intervalo [-1,1]: -1 en el peor caso y 1 en el caso de la función de similitud ideal, que logra separar correctamente todas las cadenas relevantes de las irrelevantes a la distancia máxima posible.

La discernibilidad es una métrica que ha sido utilizada en trabajos comparativos como es el caso de la herramienta *SimEval* (Heuser *et. al.*, 2007) para determinar cuán eficaces son las funciones de similitud identificando las entradas que realmente corresponden a un mismo objeto. Esta métrica intrínsecamente incorpora los cuatro elementos tradicionales de una tabla de contingencia o matriz de confusión de 2*2: aciertos, desaciertos, falsos positivos y falsos negativos.

4 Metodología utilizada para el estudio comparativo

La metodología utilizada en la comparación de las funciones de similitud, se enfocó principalmente en determinar la eficacia de los métodos para detectar duplicados más que en la eficiencia computacional. Aunque la eficiencia computacional es un aspecto siempre importante, máxime con los altos volúmenes de datos de las bases de datos actuales, se consideró que la comparación debía realizarse con base en la calidad de la detección realizada por las diferentes funciones más que en la rapidez de la ejecución de los procesos. Esto, tomando en consideración que una vez almacenados los datos, muy probablemente sea posible esperar un poco más para identificar los problemas en los datos con tal de realizar una buena tarea de limpieza. En consecuencia, se definió como métrica la discernibilidad, como medida de bondad de la calidad

lograda por cada técnica.

Para el estudio comparativo, se identificaron diferentes variaciones textuales frecuentes, que pueden originar el conflicto de la duplicación en cadenas de texto. Las seis situaciones identificadas se presentan en la Tabla 1 y la idea es establecer si algunas de las funciones de similitud son más eficaces para detectar duplicados en presencia de ellas.

Situación Problemática	**Ejemplo**
Errores ortográficos y tipográficos (ERR)	"Jorge Eduardo Rodríguez López" vs. "Jorje Eduadro Rodrigues Lopes"
Abreviaturas: truncamiento de uno o más *tokens* (ABR)	"Jorge Eduardo Rodríguez López" vs. "Jorge E Rodríguez L"
Tokens faltantes: eliminación de uno o más tokens (TFL)	"Jorge Eduardo Rodríguez López" vs. "Jorge Rodríguez"
Prefijos/sufijos sin valor semántico: presencia de subcadenas al principio y/o al final (PSF)	"Jorge Eduardo Rodríguez López" vs. "PhD Jorge Eduardo Rodríguez López, U Nal"
Tokens en desorden (TDR)	"Jorge Eduardo Rodríguez López" vs. "Rodríguez López Jorge Eduardo"
Espacios en blanco: eliminación o adición de espacios en blanco (ESP)	"Jorge Eduardo Rodríguez López" vs. "JorgeEduardo Rodríguez López"

Tabla 1: Situaciones Problemática para comparación de funciones de similitud

En 2007, Elmagarmid *et. al.* denunciaban la falta de un conjunto de datos estandarizado a gran escala que sirviera de punto de referencia para realizar estudios comparativos [Elmagarmid *et. al.*, 2007]. Al no encontrar evidencia de que esa necesidad esté satisfecha al día de hoy, se construyeron conjuntos de datos propios especialmente diseñados para correr las nueve funciones de similitud en consideración, bajo las seis situaciones problemáticas. Así, mediante la herramienta Web FakeNameGenerator[2] se generaron aleatoriamente conjuntos de datos compuestos por registros con atributos como nombre, apellidos, dirección, ocupación y correo electrónico. A partir de estos registros, otros fueron derivados de acuerdo con la variación textual o situación problemática a probar.

Para la situación problemática ERR (Errores ortográficos y Tipográficos), se generaron nuevas cadenas a las cuales se les introdujeron errores agregando, eliminando y cambiando letras (por ejemplo g por j, v por b, c por s, entre otras). Para ABR (Abreviaturas), se generaron nuevas cadenas a las cuales se les recortaron los segundos nombres, segundos apellidos o las ocupaciones, dejando sólo la inicial o primera letra. Para TFL (Tokens Faltantes), se generaron

[2] http://www.fakenamegenerator.com/

nuevas cadenas a las cuales se les omitieron una o varias palabras. Para PSF (prefijos/sufijos), se generaron nuevas cadenas a las cuales se les antepuso o pospuso al nombre completo de la persona, textos en forma aleatoria como "Doctor", "Magíster", "Estudiante", "Universidad Nacional de Colombia", entre otros. Para TDR (Tokens en desorden), se cambió el orden de las palabras al nombre completo colocando primero los dos apellidos y luego los nombres y a los oficios que constaban de más de una palabra también se les cambió el orden de las mismas. Para ESP (Espacios en Blanco), se agregaron o eliminaron espacios en blanco entre las palabras.

Con la intención de poder realizar múltiples corridas y obtener así resultados con mayor validez estadística, no se generó un único conjunto de datos de prueba, sino diez de ellos para cada una de las seis situaciones problemáticas, para un total de 60 archivos de prueba. El volumen de datos de cada conjunto se estableció en 400 cadenas representando 200 entidades diferentes (dos cadenas por entidad). Esta cantidad de registros se consideró suficiente para determinar la eficacia de cada función. Debe tenerse presente que típicamente los algoritmos para detección de duplicados ejecutan comparaciones tipo *"nested-loop"*. Esto es, comparan cada uno de los registros de una tabla con todos los demás y la comparación simple de dos cadenas requiere adicionalmente múltiples operaciones, haciendo que el costo sea prohibitivamente alto para conjuntos de datos inclusive de tamaño moderado. Aunque existen técnicas para reducir el número de comparaciones requeridas [Elmagarmid *et. al.*, 2007], éstas se escapaban del alcance de este trabajo.

Para los diez conjuntos de datos correspondientes a cada situación problemática, se corrieron los algoritmos respectivos de cada una de las nueve funciones de similitud calculando la métrica de la discernibilidad. Para el cálculo de la discernibilidad, se definió $c_1 = 3$ y $c_2 = 7$ en (1), de forma que el intervalo óptimo contribuyera en 30% al valor de discernibilidad y $F_{max}/2|Q|$ en 70%, ya que, para fines prácticos, es más importante que la función de similitud separe correctamente cadenas relevantes de irrelevantes, independientemente de la distancia a la que ubique unas de otras. Debe aclararse que los cálculos fueron verificados confrontándolos con los arrojados por SimMetrics, librería de código abierto de métricas de distancia o similitud[3].

Para poder llegar a conclusiones válidas, se quiso realizar un análisis de varianza (ANOVA) con el fin de determinar si las diferencias encontradas con las distintas técnicas eran estadísticamente significativas, según el índice de discernibilidad, pero luego de realizar la prueba de Kolmogorov-Smirnov y pruebas gráficas de ajuste a la distribución, se determinó que el supuesto de normalidad no se cumplía. Por ello, se aplicó la prueba no paramétrica de Kruskal-Wallis, la cual no requiere que se cumplan los supuestos de normalidad y homoscedasticidad [Álvarez, 2007]. Esta prueba permitió verificar la hipótesis nula de igualdad de las medianas del grado de discernibilidad para las nueve funciones de similitud.

Luego de realizada la prueba de Kruskal-Wallis, con el fin de determinar cuáles eran las medianas significativamente diferentes entre sí, se realizaron Gráficos de Caja y Bigotes. El análisis de dichos gráficos, especialmente del punto central de cada caja, permitió identificar a las funciones de similitud de mejores resultados en cada caso.

[3] http://www.dcs.shef.ac.uk/~sam/simmetrics.html

5 Resultados

La discernibilidad se calculó para las nueve funciones de similitud bajo estudio, para cada uno de los diez conjuntos de datos con cada situación problemática. Se presentan a modo de ejemplo, los resultados obtenidos para la situación problemática ABR (Ver tabla 2).

	FUNCIÓN DE SIMILITUD								
Conjunto de datos	**Levenshtein**	**Affine Gap**	**Smith Waterman**	**Jaro**	**Jaro Winkler**	**2-grams**	**3-grams**	**Monge Elkan**	**Soft TF-IDF**
1	0,5737	0,7180	0,6995	0,4258	0,3579	0,6070	0,6632	0,6802	0,7060
2	0,5965	0,7090	0,7025	0,4719	0,4246	0,6193	0,6662	0,6737	0,6965
3	0,5772	0,7210	0,6972	0,4860	0,3509	0,6088	0,6561	0,6737	0,6982
4	0,5930	0,7090	0,6995	0,6995	0,3807	0,6263	0,6579	0,6702	0,7000
5	0,6053	0,7150	0,7030	0,4439	0,3544	0,6175	0,6509	0,6725	0,6999
6	0,5825	0,7180	0,6930	0,4807	0,4035	0,6035	0,6609	0,6719	0,6990
7	0,5860	0,7180	0,6930	0,4211	0,3451	0,6065	0,6544	0,6754	0,6995
8	0,5877	0,7180	0,6925	0,4509	0,4263	0,6123	0,6439	0,6702	0,6982
9	0,5965	0,7240	0,7060	0,4667	0,4088	0,6333	0,6719	0,6807	0,7030
10	0,5965	0,7180	0,7060	0,4351	0,3193	0,6140	0,6609	0,6837	0,6965
Promedio	**0,5895**	**0,7168**	**0,6992**	**0,4781**	**0,3771**	**0,6149**	**0,6586**	**0,6752**	**0,6997**

Tabla 2: Discernibilidad situación problemática: Abreviaturas

Para todas las situaciones problemáticas, las funciones de similitud arrojaron resultados variables de la discernibilidad. Para poder llegar a conclusiones válidas, se quiso realizar un análisis de varianza (ANOVA), pero el supuesto de normalidad no se cumple como lo muestra el gráfico de probabilidad normal de los residuos presentado en la figura 1, en el que puede verse el pobre ajuste a la distribución normal. Por ello, se aplicó la prueba no paramétrica de Kruskal-Wallis. La prueba realizada mediante el paquete estadístico StatGraphics Plus versión 5, arrojó como resultado un valor *p* de 0,0. Puesto que el valor *p* de la prueba es inferior a 0,05, se puede afirmar que hay diferencias estadísticamente significativas entre las medianas, con un nivel de confianza del 95,0%. Para determinar cuáles son las medianas significativamente diferentes entre sí, se realizó el Gráfico de Caja y Bigotes presentado en la figura 2.

El análisis del gráfico, permite identificar a la función de similitud Brecha Afín como la de mejores resultados, ante la situación problemática Abreviaturas. Un análisis similar para las otras situaciones problemáticas, permitió identificar a las funciones de mejores resultados en cada caso. La tabla 3 presenta las posiciones obtenidas según su eficacia para cada situación.

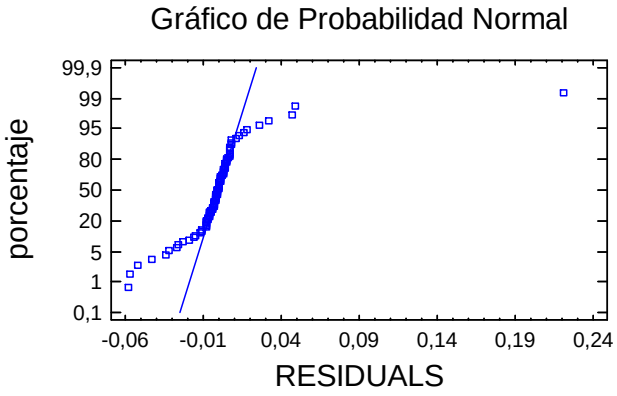

Figura 1: Gráfico de Probabilidad Normal de los residuos para la situación problemática ABR.

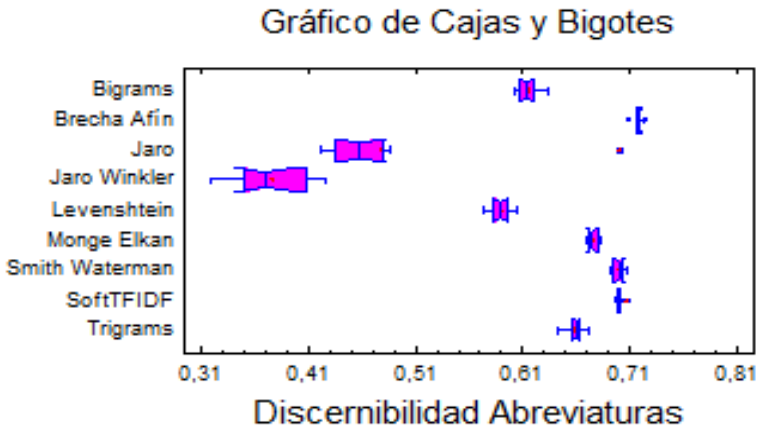

Figura 2: Gráfico de Cajas y Bigotes para la situación problemática ABR.

Situación Problemática	**Posición**								
	1	2	3	4	5	6	7	8	9
Errores ortográficos	LE	SW	2G	3G	BA	ME	JA	SW	JW
Abreviaturas	BA	ST	SW	ME	3G	2G	LE	JA	JW
Tokens Faltantes	SW	3G	BA	LE	2G	SW	ME	JA	JW
Prefijos / Sufijos	ST	BA	3G	SW	2G	ME	LE	JA	JW
Tokens en desorden	3G	2G	ST	ME	JA	BA	JW	SW	LE
Espacios en blanco	SW	3G	LE	2G	BA	SW	ME	JA	JW

BA: Brecha Afín ST: Soft TF-IDF SW: Smith-Waterman LE: Levenshtein
2G: Bi-grams 3G: Tri-grams ME: Monge-Elkan JA: Jaro JW: Jaro Winkler

Tabla 3: Funciones de similitud más eficaces para cada situación problemática.

6 Conclusiones

Muchas son las funciones de similitud sobre cadenas existentes. Como se mostró, algunas tienden a fallar bajo la presencia de ciertas variaciones textuales, acá llamadas situaciones problemáticas. En este trabajo, mediante la construcción de conjuntos de datos especialmente diseñados para cada situación, se determinó las funciones más eficaces en cada caso.

Como trabajo futuro, se plantea realizar una guía metodológica de naturaleza gráfica que apoye la selección de las técnicas más adecuadas para casos particulares de la vida real.

Referencias

Álvarez R. (2007). "Estadística aplicada a las ciencias de la salud". Ed. Díaz de Santos. España. 1030p.

Baeza-Yates, R., y Gonnet, G.H. (1992) "A new approach to Text Searching. *Communications of the ACM*. 35 (10), pp. 74-82.

Bilenko, M. y Mooney, R.J. (2003) "Learning to Combine Trained Distance Metrics for Duplicate Detection in Databases", *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 39-48.

Christen, P. (2006) "A Comparison of Personal Name Matching: Techniques and Practical Issues", *Sixth IEEE International Conference on Data Mining*, pp. 290-294.

Cohen, W.W., Ravikumarand, P. Fienberg, S.E. (2003) "A Comparison of String Distance Metrics for Name-Matching Tasks", *International Joint Conference on Artificial Intelligence*, pp. 73-78.

Cohen, D. (1997) "Recursive Hashing Functions for n-Grams", *ACM Transactions on Information Systems*. 15(3), pp. 291-320.

Cohen, W.W. (1998) "Integration of Heterogeneous Databases without Common Domains Using Queries Based on Textual Similarity", *Proceedings of the SIGMOD International Conference Management of Data SIGMOD'98*, pp. 201-212.

da Silva, R., Stasiu, R., Orengo, V.M. y Heuser, C.A. (2007) "Measuring Quality of Similarity Functions in Approximate Data Matching", *Journal of Informetrics*. 1(1), pp. 35-46.

Dunn, H.L. (1946) "Record Linkage", *Americal Journal of Public Health*, 36(12), pp. 1412-1416.

Elmagarmid, A.K., Ipeirotis, P.G. y Verykios, V.S. (2007) "Duplicate Record Detection: A Survey", *IEEE Transactions on Knowledge and Data Engineering*, 19 (1), pp. 1-40.

Fellegi, I.P y Sunter, A.B. (1969) "A Theory for Record Linkage", *Journal of the American Statistical Association*, 64(328), pp. 1183-1210.

Gelbukh, A., Jiménez, S., Becerra, C. y González, F. (2009) "Generalized Mongue-Elkan Method for Approximate Text String Comparison", *Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing*, pp. 559-570.

Gotoh, O. (1982) "An Improved Algorithm for Matching Biological Sequences", *Journal of Molecular Biology*, 162 (3), pp. 705-708.

Heuser, C.A., Krieser, F.N. y Orengo, V.M. (2007) "SimEval - A Tool for Evaluating the Quality of Similarity Functions", *Tutorials, posters, panels and industrial contributions at the 26th International Conference on Conceptual Modeling*, pp. 71-76.

Jaro, M.A. (1976) *Unimatch: A Record Linkage System User's Manual, technical report*, Washington, D.C.: US Bureau of the Census.

Levenshtein, V.I. (1966) "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals". *Soviet Physics Doklady*, 10(8), 707-710.

Masek, W.J. (1980) "A Faster Algorithm for Computing String Edit Distances", *Journal of Computer and System Sciences*, 20, pp. 18-31.

Monge, A.E. y Elkan, C.P. (1996) "The Field Matching Problem: Algorithms and Applications", *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 267-270.

Moreau, E., Yvon, F. y Cappé, O. (2008) "Robust Similarity Measures for Named Entities Matching", *Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 593-600.

Needleman, S.B. y Wunsh, C.D. (1970) "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins", *J Mol Biol,* 48 (3), pp. 443–53.

Newcombe, H y Kennedy, J. (1962) "Record Linkage: Making Maximum Use of the Discriminating Power of Identifying Information", *Communications of the ACM*, 5(11), pp. 563- 566.

Newcombe, H., Kennedy, J., Axford, S. y James, A. (1959) "Automatic Linkage of Vital Records", *Science*, 130(3381), pp. 954-959.

Pollock, J.J. y Zamora, A. (1984) "Automatic Spelling Correction in Scientific and Scholarly Text", *Communications of the ACM*, 27(4), pp. 358-368.

Ramírez, F. y López, E. (2006) "Spelling Error Patterns in Spanish for Word Processing Applications", *Proceedings of Fifth international conference on Language Resources and Evaluation, LREC 2006*.

Smith, T.F. y Waterman, M.S. (1981) "Identification of Common Molecular Subsequences", *Journal of Molecular Biology*, 147(1), pp. 195-197.

Ukkonen, E. "Aproximate String-Matching With Q-grams and Maximal Matches", Theoretical Computer Science, vol. 92, no. 1, pp. 191-211, 1992.

Vidal, E., Marzal, A. y Aibar, P. (1995) "Fast Computation of Normalized Edit Distances", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9), pp. 899-902.

Wagner, R.A. y Fischer, M.J. (1974) "The String-to-String Correction Problem", *Journal of the ACM*, 21(1), pp. 168-173.

Weigel, A. y Fein, F. (1994) "Normalizing the Weighted Edit Distance", *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, 2(2), pp. 399-402.

Winkler, W.E. (1989) "Frequency-Based Matching in the Fellegi-Sunter Model of Record Linkage", *Proceedings of the Section on Survey Research Methods*, pp. 778-783.

Winkler, W.E. (1990) "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage", *Proceedings of the Section on Survey Research Methods*, pp. 354-359.

Winkler, W.E. (2000) "Improved Decision Rules in the Felligi-Sunter Model of Record Linkage", *Proceedings of the Section on Survey Research Methods*, pp. 274-279.

Winkler, W.E. (2000) "Using the EM Algorithm for Weight Computation in the Fellegi-Sunter Model of Record Linkage", *Proceedings of the Section on Survey Research Methods*, pp. 667-671.

Yancey, W.E. (2006) "Evaluating String Comparator Performance for Record Linkage", (Statistics #2005-05). Washington, DC: U.S. Census Bureau, Statistical Research Division.

Yin, R. (1994) "Case Study Research: Design and Methods". Sage Publications, Thousands Oaks, CA.

Yujian, L. y Bo, L. (2007) "A Normalized Levenshtein Distance Metric", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6), pp. 1091-1095.