borrosa, criterios heurísticos que aprovechan la semántica inherente a algunas etiquetas HTML, así como a la posición del término dentro del texto. La idea fundamental es tratar de reproducir la manera en que una persona lee las partes que considera más representativas de una página web para obtener una visión general de su contenido y así poder concluir si esa página puede o no interesarle.

Existen varias diferencias entre nuestro enfoque y el presentado en WEBSOM. En primer lugar nuestra representación está orientada a páginas web. Además, asumimos que todo el proceso de representación de documentos será no supervisado; algo que no siempre se asume en los modelos que utilizan SOMs para el agrupamiento de documentos.

Este artículo se estructura como sigue: en el apartado 2 se resumirá de forma básica el proceso de creación de un SOM; en el apartado 3 se revisan algunos modelos de representación de documentos aplicados al SOM; en el 4 se describe el modelo propuesto, basado en lógica borrosa; en el 5 se explica la experimentación llevada a cabo para, posteriormente, analizar los resultados en el apartado 6. Finalmente se exponen las conclusiones en el apartado 7.

2. Mapas autoorganizativos

Los mapas autoorganizativos de Kohonen son estructuras neuronales que utilizan aprendizaje competitivo para tratar de generar una relación espacial-topológica entre los vectores que caracterizan sus neuronas, a partir de un entrenamiento y en función de los vectores de entrada.

En este tipo de aprendizaje las neuronas compiten entre si, activándose sólo una de ellas ante la presencia de un patrón o estímulo de entrada. El objetivo es, a partir de un proceso iterativo de comparación con la colección de datos de entrada, agrupar estos datos en base a su similitud. Para ello se presentan al mapa vectores de entrada de igual dimensión que la de sus vectores característicos.

Para la creación de un SOM lo primero que ha de hacerse es inicializar la red, definiendo el número de neuronas y su topología, e inicializando el vector de pesos de cada neurona, algo que puede realizarse simplemente de forma aleatoria.

La neurona ganadora establecerá el conjunto de neuronas cuyos vectores deben modificarse. Las neuronas están conectadas con sus vecinas mediante una relación de vecindario que impone la propia estructura del SOM. El tamaño del vecindario disminuirá a lo largo del entrenamiento; esa es la clave de la autoorganización. La función de actualización de pesos del mapa tiene la forma:

$$m_i(t+1) = m_i(t) + h_{ci}(t)\,[x(t) - m_i(t)] \quad (1)$$

donde t es el instante de tiempo discreto correspondiente a una iteración del algoritmo, $x(t)$ es el vector de entrada en la iteración t y h_{ci} es la región de influencia que el vector de entrada tiene sobre el SOM, también llamado núcleo de vecindad. Esta función es la que define la "rigidez" de la "red élástica" del SOM en el espacio de los datos((Kohonen et al., 1996)).

La función que define el núcleo de vecindad h_{ci} puele ser de tipo gaussiano (2), como en nuestro caso, y se expresa como:

$$h_{ci} = \alpha(t) \cdot exp\left(-\frac{||r_c - r_i||^2}{2\sigma^2(t)}\right) \quad (2)$$

lo que hará que la modificación de valores disminuya con la distancia en el vecindario, siendo $\sigma(t)$ la función que define este radio de vecindario, que se va reduciendo después de cada iteración t. En los mapas bidimensionales las neuronas pueden ordenarse en un retículo rectangular o hexagonal, con lo que cada neurona tendrá 6 u 8 vecinos respectivamente. En nuestro caso se utilizó un mapa rectangular.

La modificación de pesos depende también de la distancia entre una neurona n_i y la ganadora n_c (siendo r_c y r_i las posiciones de las neuronas en el grid) y tiende a cero según aumenta ésta. La tasa de aprendizaje $\alpha(t)$ es una función monótona decreciente respecto al tiempo t en el rango [0,1] ($0 < \alpha(t) < 1$). En otras palabras, tiende a cero a medida que se van procesando los vectores del conjunto de entrenamiento.

De este modo, una vez entrenado el mapa, después de un número fijo de iteraciones o cuando se cumpla la condición de parada establecida, ya se pueden introducir vectores clasificados manualmente que permiten etiquetar las zonas del mapa correspondientes.

Por tanto, y a modo de resumen, los tres pasos fundamentales en la creación de un SOM son: inicialización, entrenamiento y calibración. Una vez etiquetado, un SOM

puede usarse como un clasificador que asigna a cada vector de entrada la categoría con la que se haya etiquetado la neurona que se active en cada caso.

3. Representación de documentos

En la literatura pueden encontrarse diversas propuestas para la representación de documentos en sistemas basados en SOMs, destinados al clustering, la clasificación o la visualización de grandes colecciones. Algunas de estas propuestas serán analizadas a lo largo de este apartado. Además, se describirá la representación que proponemos y que queremos evaluar.

3.1. Trabajos relacionados

En el sistema WEBSOM, la representación de los documentos se realiza dentro del modelo de espacio vectorial (Salton, Wong, y Yang, 1975). De este modo, la entrada es un conjunto de vectores de representación de documentos donde cada dimensión representa el peso de un término en el contenido del mismo. Este peso se puede calcular, bien de forma sencilla en base al número de ocurrencias del término en el documento, por ejemplo usando la frecuencia inversa de documento, o bien, si existe información sobre las categorías de los documentos, utilizando la entropía de Shannon sobre el conjunto de clases de documentos, para lo que se utiliza la información de clasificación. Además, las distintas aproximaciones al problema se han basado en documentos textuales, mientras la que aquí se presenta está orientada especificamente a páginas web en formato HTML, aunque sería fácilmente aplicable a documentos XML con vocabularios con semántica relacionada con la documentación electrónica, como es el caso de docbook.

En (Bakus, Hussin, y Kamel, 2002) la representación utilizada se basa en sintagmas en lugar de palabras para formar los vectores de representación, utilizando dichos sintagmas como unidades de entrada para las funciones de pesado tradicionales: Binaria, TF y TF-IDF. Por otro lado, el modelo ConSOM (Liu, Wang, y Wu, 2008) usa dos vectores en lugar de uno para representar tanto los documentos de entrada, como las neuronas del mapa, con el objetivo de combinar el espacio vectorial con lo que denominan *espacio conceptual*. Esto supone una modificación en el SOM, por lo que no sólo afecta a la representación, sino que va más allá al proponer un nuevo modelo, lo que se aleja de nuestra propuesta, que ataca el problema desde el punto de vista de la representación de los documentos y no pretende modificar el algoritmo utilizado para agruparlos.

3.2. *Fuzzy Combination of Criteria* (FCC)

La lógica borrosa se basa principalmente en la aplicación de heurísticas con el objeto de resolver la ambigüedad inherente a procesos de razonamiento cualitativo, permitiendo establecer cierta relación entre los factores observados. Profundizando un poco más, podemos decir que mediante la lógica borrosa se tratan de modelar relaciones entre variables que, en nuestro caso, se definirán a partir de las frecuencias de aparición de los términos en determinados elementos HTML. Esto la convierte en un entorno adecuado para capturar el conocimiento experto humano.

La pieza básica sobre la que se construye todo sistema borroso es la llamada variable lingüística, cuyo valor puede venir dado por palabras del lenguaje natural y se define por medio de conjuntos borrosos (Zadeh, 1965), cuyos límites son imprecisos. Con estos conjuntos se permite describir el grado de pertenencia de un objeto a una determinada clase y se definen a partir de conocimiento experto.

La arquitectura básica de un sistema de inferencia borroso se compone de tres etapas de procesamiento: borrosificación de entradas, aplicación de las reglas de inferencia que constituyen la base de conocimiento del sistema, y desborrosificación, que permite obtener el valor final. La base de conocimiento se define mediante un conjunto de reglas IF-THEN que describirán, a partir del conocimiento experto, el comportamiento que debería tener el sistema con la máxima precisión posible; es decir, reflejan, junto con la propia definición de las variables lingüisticas y los conjuntos borrosos, el conocimiento heurístico que se tiene sobre el problema. La finalidad de estas reglas es la combinación de uno o varios conjuntos borrosos de entrada, llamados antecedentes, asociándolos a un conjunto borroso de salida, llamado consecuente. Una vez obtenidos los consecuentes de cada regla, y tras una etapa de agregación, se obtiene un conjunto agregado final, que será la entrada para la etapa de desborrosificación, donde

se hace corresponder el conjunto borroso de salida con un punto concreto, llamado salida nítida o "crisp".

En nuestra propuesta para la representación de documentos asumimos que no usaremos ningún tipo de información de clasificación previa que pudiera existir. Esta información sólo se utilizará para la evaluación de los resultados, ya que como veremos más adelante, para llevar a cabo la comparación entre las distintas representaciones, fijaremos el tamaño del mapa en función del número de *clústers* que queremos obtener y que se corresponderá con el número de clases a las que pertenecen los documentos de entrada.

Las variables lingüísticas que usaremos como entrada del sistema serán la frecuencia del término en el documento, en el título (contenido en el elemento *title*), en los enfatizados (contenidos en los elementos *em*, *h1*, *b*, etc.) y la posición global del término dentro de la página. Las frecuencias son normalizadas con el mayor valor encontrado para cada criterio, con el objetivo de independizar las reglas del tamaño del documento y del tamaño de los textos presentes en cada criterio. La posición global se calcula mediante un sistema borroso auxiliar, que tomando como entrada las posiciones en las que aparece el término dentro del documento, devuelve la posición global por medio de dos conjuntos borrosos: estándar y preferente. Las figuras 1 y 2 muestran los conjuntos borrosos empleados.

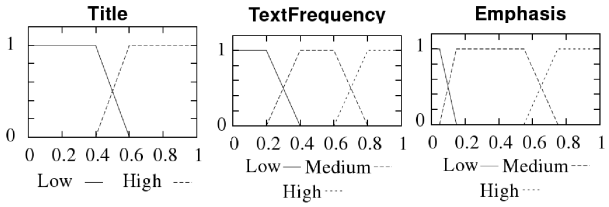

Figura 1: Reglas basadas en frecuencia de aparición

La salida del sistema borroso es una única variable lingüística denominada relevancia, cuyos valores pueden ser: no relevante, poco relevante, medianamente relevante, bastante relevante y muy relevante. Los conjuntos borrosos definidos para esta variable pueden verse en la figura 3. Las reglas utilizadas se han basado en los siguientes aspectos:

- Una página web puede no tener palabras enfatizadas.

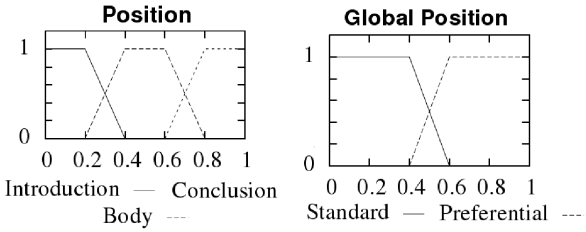

Figura 2: Sistema borroso auxiliar para el cálculo del valor global de la posición

- Una palabra que aparece en el título puede que no siempre sea relevante (el título podría haber sido generado, por ejemplo, por un editor de HTML), o bien podría tener una componente retórica.
- Generalmente, la posición es un criterio que da más peso en páginas largas que en cortas.
- Una palabra con alta frecuencia de aparición en una página podría tener un significado muy general, y por lo tanto, no discriminante.

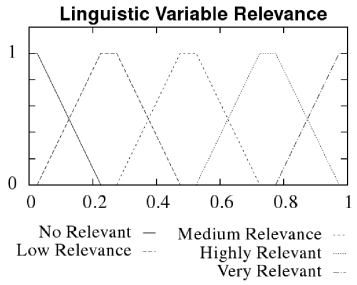

Figura 3: Conjuntos borrosos para definir la relevancia del término

Los conjuntos completos de reglas tanto del sistema borroso auxiliar como del global se muestran en los cuadros 1 y 2

	posición relativa		**posición global**
IF	introducción	THEN	preferente
IF	cuerpo	THEN	estándar
IF	conclusión	THEN	preferente

Cuadro 1: Conjunto de reglas del sistema borroso auxiliar

Por último, el motor de inferencia está basado en el algoritmo de centro de masas, que evalúa la salida de cada regla en función del grado de verdad de cada antecedente.Una explicación detallada del sistema borroso puede encontrarse en (Fresno, 2006).

	Título		**Frecuencia**		**Enfatizado**		**Posición**		**Relevancia**
IF	Alto	AND	Alta	AND	Alto			THEN	Muy Alta
IF	Alto	AND	Media	AND	Alto			THEN	Muy Alta
IF	Alto	AND	Media	AND	Medio			THEN	Alta
IF	Alto	AND	Alta	AND	Medio			THEN	Muy Alta
IF	Alto	AND	Baja	AND	Bajo	AND	Preferente	THEN	Media
IF	Alto	AND	Baja	AND	Bajo	AND	Estándar	THEN	Poca
IF	Bajo	AND	Baja	AND	Bajo			THEN	Nada
IF	Bajo	AND	Alta	AND	Alto	AND	Preferente	THEN	Muy Alta
IF	Bajo	AND	Alta	AND	Alto	AND	Estándar	THEN	Alta
IF	Alto	AND	Baja	AND	Medio	AND	Preferente	THEN	Alta
IF	Alto	AND	Baja	AND	Medio	AND	Estándar	THEN	Media
IF	Alto	AND	Baja	AND	Alto	AND	Preferente	THEN	Muy Alta
IF	Alto	AND	Baja	AND	Alto	AND	Estándar	THEN	Alta
IF	Alto	AND	Alta	AND	Bajo	AND	Preferente	THEN	Muy Alta
IF	Alto	AND	Alta	AND	Bajo	AND	Estándar	THEN	Alta
IF	Bajo	AND	Baja	AND	Medio	AND	Preferente	THEN	Media
IF	Bajo	AND	Baja	AND	Medio	AND	Estándar	THEN	Poca
IF	Bajo	AND	Baja	AND	Alto	AND	Preferente	THEN	Alta
IF	Bajo	AND	Baja	AND	Alto	AND	Estándar	THEN	Media
IF	Bajo	AND	Media	AND	Bajo	AND	Preferente	THEN	Poca
IF	Bajo	AND	Media	AND	Bajo	AND	Estándar	THEN	Nada
IF	Bajo	AND	Media	AND	Medio	AND	Preferente	THEN	Media
IF	Bajo	AND	Media	AND	Medio	AND	Estándar	THEN	Poca
IF	Bajo	AND	Media	AND	Alto	AND	Preferente	THEN	Muy Alta
IF	Bajo	AND	Media	AND	Alto	AND	Estándar	THEN	Alta
IF	Bajo	AND	Alta	AND	Bajo	AND	Preferente	THEN	Media
IF	Bajo	AND	Alta	AND	Bajo	AND	Estándar	THEN	Poca
IF	Bajo	AND	Alta	AND	Medio	AND	Preferente	THEN	Alta
IF	Bajo	AND	Alta	AND	Medio	AND	Estándar	THEN	Media
IF	Alto	AND	Media	AND	Bajo	AND	Preferente	THEN	Media
IF	Alto	AND	Media	AND	Bajo	AND	Estándar	THEN	Poca

Cuadro 2: Conjunto de reglas del sistema borroso global

4. Experimentación

Los pasos seguidos para realizar la experimentación se describen a continuación.

4.1. Colección

Para la experimentación se ha utilizado la colección Banksearch (Sinka y Corne, 2005), en concreto un subconjunto de 10 clases etiquetadas como: Commercial banks, Building societies, Insurance agencies, Java, C/C++, Visual Basic, Astronomy, Biology, Soccer y Motor sport. Cada una de estas clases consta de 1000 páginas web en formato HTML, haciendo un total de 10000 documentos. De estos, algunos fueron descartados por problemas con el parser HTML utilizado, ya que algunos documentos no estaban bien formados o, incluso, la página estaba incompleta por problemas en su descarga durante la creación de la colección. Finalmente, tras los descartes, 9897 documentos fueron usados en los experimentos.

Antes de extraer la información necesaria para la entrada del sistema borroso, se eliminaron un conjunto de términos de una lista de *stopwords* compuesta por 621 términos en inglés, se convirtieron las entidades particulares del lenguaje HTML, se eliminaron los signos de puntuación y se utilizó el algoritmo de Porter para hacer *stemming* de las palabras.

4.2. Detalles del SOM

El tamaño del SOM utilizado en la experimentación es 5x2, haciendo un total de 10 neuronas, con el objetivo de que exista una única neurona por cada clase. Este hecho supone un pequeño grado de supervisión, pero ésta no se aplica a la fase de representación, sino al proceso posterior de *clustering*, ya que fijamos el k. Los vectores de entrada fueron reducidos a varios tamaños entre 100 y 5000 con la intención de evaluar el comportamiento de las diferentes representaciones

en un rango de dimensiones. De este modo podremos ver si el comportamiento empeora al representar con un número reducido de rasgos y, además, averiguar con que dimensiones podemos encontrar un equilibrio entre la calidad de la representación y el coste computacional. Este aspecto adquiere gran importancia en tareas como el *clustering* que generalmente conllevan un alto coste computacional.

Durante el entrenamiento del mapa, la tasa de aprendizaje inicial se fijo en 0,1, el vecindario inicial en 5 y el número de iteraciones en 50000. Todos estos datos se eligieron después de la realización de diversas pruebas, por ser aquellos con los que se obtuvieron unos resultados de *clustering* y calidad del mapa más satisfactorios. El resto de información acerca del mapa coincide con la utilizada en la implementación SOMlib (Dittenbach, Merkl, y Rauber, 2000), distribuído como proyecto de software libre[1]. Dicha librería, escrita en lenguaje Java, ha sido la utilizada para la creación del SOM.

4.3. Reducción del número de rasgos

Para la comparación usamos tres funciones de pesado de términos diferentes: TF, Bin-IDF y TF-IDF. Cada vector debe contener una entrada por cada término del vocabulario, es decir, por cada término que aparezca en la colección, lo que da lugar a vectores con gran número de dimensiones. Esto supone un problema en lo que a rendimiento se refiere. Para paliarlo, se utilizan distintos tipos de reducciones que permiten utilizar un número menor de dimensiones sin perder la información esencial.

En todos los casos se probaron tanto la reducción por frecuencia de documentos, como la proyección aleatoria (Kaski, 1998) con cinco unos distribuídos aleatoriamente en cada columna de la matriz de proyección. Esta reducción tiene la ventaja de reducir el coste computacional. En este último caso se ha añadido al preproceso descrito en el apartado 4.1, la eliminación de los términos que aparecían en la colección con una frecuencia global de menos de 50, tal como se indica en Kohonen et al. (2000).

En nuestro caso, la relevancia de un término no depende únicamente de la frecuencia de aparición del término en un documento o en la colección, por lo que no tiene sentido reducir usando únicamente la frecuencia de documentos. Dado que la representación pondera cada término y le asigna un valor que indica su relevancia, eliminar los términos menos relevantes consistiría sólo en eliminar aquellos con las puntuaciones de pesado más bajas, o bien quedarnos con los que tengan las puntuaciones más altas. Por otra parte, queremos que, de alguna forma, cada documento se vea reflejado en el vocabulario final y valorar positivamente que un término aparezca bien puntuado en diferentes documentos.

Por todo lo anterior, la reducción que se presenta en este estudio, llamada MFT_n (*More Frequent Terms*) consiste en la extracción de los términos más puntuados por niveles, es decir, por cada documento se hace un *ranking* de sus términos más representativos, es decir, los que tienen mayor peso dentro del documento, y se van tomando secuencialmente los términos que aparecen en primera posición, después en segunda, etc. hasta que se cubren las dimensiones requeridas. A medida que se extraen los términos de un nivel, se ordenan en una lista global por frecuencia, esto es, se colocan primero aquellos que han aparecido en un mayor número de documentos. Entre aquellos que resultan empatados tras la primera ordenación, se utiliza la relevancia para determinar su posición. Al final de cada nivel se comprueba si se tienen suficientes términos para el tamaño de vocabulario solicitado y si es así, se toman, ordenadamente, los términos necesarios de la lista global.

Además de esta reducción, se han realizado experimentos con otras basadas sólo en el valor de la relevancia, tomándolo por niveles o de forma global, combinando otros métodos como la reducción por frecuencia de documentos o la proyección aleatoria. No obstante los mejores resultados fueron obtenidos utilizando la reducción MFT_n y así, por claridad y brevedad, los resultados obtenidos con el resto de reducciones han quedado fuera de este artículo.

Finalmente, para validar la función de pesado FCC, hemos aplicado también la reducción MFT_n a TF, Bin-IDF y TF-IDF, con el objetivo de verificar que la mejora no venga dada únicamente por la reducción.

[1] http://www.ifs.tuwien.ac.at/ andi/somlib/

4.4. Métodos de evaluación

Para evaluar el *clustering*, una vez entrenado el SOM, se mapea toda la colección sobre él, de forma que cada documento quedará asociado a la neurona del mapa a la que más se asemeje. Después se etiqueta cada neurona eligiendo para ello la clase predominante en función de los vectores que activaron dicha neurona, es decir, se utiliza como etiqueta la clase a la que pertenecen el mayor porcentaje de documentos mapeados en la neurona. Todos los documentos que hayan activado esa neurona durante el proceso de mapeo y no pertenezcan a la clase que etiqueta dicha neurona, son contados como errores.

Utilizaremos dos medidas para evaluar los resultados. La primera es la tasa de aciertos (*accuracy*), es decir, el porcentaje de documentos que activan una neurona etiquetada con su misma clase. Esta medida y la forma de llevarla a cabo ha sido basada en Kohonen et al. (2000): "*[...] each document was mapped onto one of the grid points of each map, and all documents that represented a minority class at any grid point were counted as classification errors.*"

El segundo método elegido es la medida F, véase la fórmula 3, siendo i la clase y j el *cluster*. El *recall* y la precisión vienen dados por las fórmulas 4 y 5.

$$F(i,j) = \frac{2 \cdot Recall(i,j) \cdot Precision(i,j)}{Recall(i,j) + Precision(i,j)} \tag{3}$$

$$Recall(i,j) = \frac{n_{ij}}{n_j} \tag{4}$$

$$Precision(i,j) = \frac{n_{ij}}{n_i} \tag{5}$$

Siendo n_{ij} es el número de documentos etiquetados con la clase i en el *cluster* j, n_i el número de documentos etiquetados con la clase i, n_j el número de documentos en el *cluster* j y n el número total de documentos. Para todos los *clusters*, la medida F se calcula según la fórmula 6. Un mayor valor de esta medida indica una mayor calidad del *clustering*.

$$F = \sum_i \frac{n_i}{n} \cdot \max_j \{F(i,j)\} \tag{6}$$

5. Análisis de resultados

En las figuras 4 y 5 se muestran los resultados para la tasa de aciertos y la medida F obtenidos en los diferentes casos. Cabe destacar que cada uno de los resultados presentados en ellas corresponde a la media de cinco ejecuciones diferentes con los mismos parámetros. El motivo para ello es la inicialización aleatoria del mapa, que provocará que cada ejecución del proceso concluya con resultados diferentes, y aunque por la convergencia del mapa serán bastante similares, se han querido evitar los valores demasiado buenos o demasiado malos.

Se puede apreciar cómo FCC supera a las funciones tradicionales que, a medida que aumenta el número de rasgos, se aproximan a los resultados de nuestra propuesta a la vez que sus resultados se estabilizan. Además, si se selecciona un número excesivo de rasgos (a partir de 1000 aproximadamente en las figuras 4 y 5), se introducirán sucesivamente términos poco relevantes, pudiendo introducir ruido y afectando a los resultados.

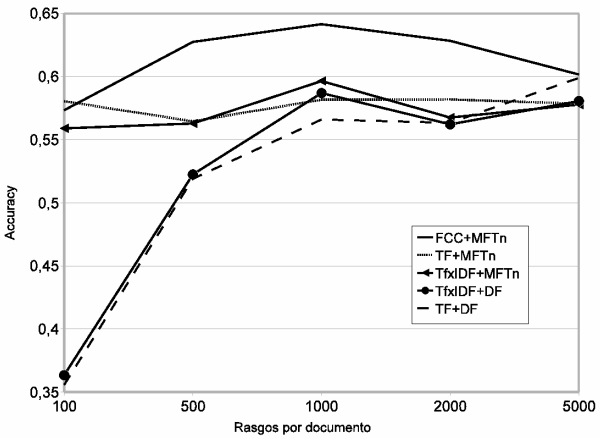

Figura 4: Tasa de aciertos para diferentes dimensiones de los vectores de documentos

En ambos casos con un número de rasgos pequeño, en concreto por debajo de 2000, la representación propuesta obtiene los mejores resultados tanto en tasa de aciertos como en calidad del *clustering*, o al menos resultados tan buenos como cuando se utilizan 2000 o más rasgos por documento. Asimismo, las funciones basadas en la frecuencia (TF y TF-IDF) se muestran mucho más estables con la reducción MFT_n, es decir, que con dimensiones reducidas sus resultados no disminuyen drásticamente, situándose al nivel de FCC con el mínimo número de rasgos elegido,

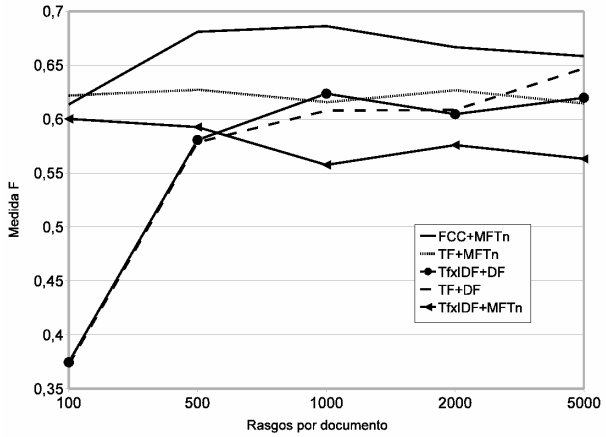

Figura 5: Medida F para diferentes dimensiones de los vectores de documentos

aunque posteriormente su mejora es menor que la obtenida por FCC. En resumen, la representación propuesta funciona mejor que las clásicas y con pocos rasgos está más cerca de sus propios máximos.

6. Conclusiones

A lo largo del presente trabajo se ha descrito un método de representación basado en lógica borrosa, de forma que se trata de recoger parte de la semántica implícita en el lenguaje HTML, con el objetivo de realizar *clustering* de documentos basado en mapas autoorganizativos. Los experimentos realizados han demostrado que la representación propuesta mejora el agrupamiento por medio de SOM respecto a las representaciones clásicas basadas únicamente en la frecuencia de los términos.

Cabe destacar que la representación basada en lógica borrosa mejora no sólo los valores máximos obtenidos por las representaciones clásicas, sino que con el mínimo número de rasgos probado, esto es 100 rasgos para representar cada documento, está prácticamente al nivel de los máximos de las clásicas. Esto permite la obtención de los mismos resultados con un vocabulario menor, lo que reduce notablemente el tamaño de los datos de entrada del SOM y de los vectores de pesos de sus neuronas, teniendo como principal efecto una reducción importante en el tiempo de computo necesario.

Bibliografía

Bakus, J., M.F. Hussin, y M. Kamel. 2002. A som-based document clustering using phrases. En *ICONIP '02*.

Dittenbach, Michael, Dieter Merkl, y Andreas Rauber. 2000. The growing hierarchical self-organizing map. *IJCNN*.

Fresno, Victor. 2006. *Representacion autocontenida de documentos HTML: una propuesta basada en combinaciones heuristicas de criterios*. Ph.D. tesis.

Kaski, S. 1998. Dimensionality reduction by random mapping: fast similarity computation for clustering. En *Neural Networks Proceedings, 1998*.

Kohonen, T. 1990. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

Kohonen, T., J. Hynninen, J. Kangas, y J. Laaksonen. 1996. Som pak: The self-organizing map program package.

Kohonen, T., S. Kaski, K. Lagus, J. Salojarvi, J. Honkela, V. Paatero, y A. Saarela. 2000. Self organization of a massive document collection. *Neural Networks, IEEE Transactions on*.

Lagus, Krista. 1998. Generalizability of the websom method to document collections of various types.

Liu, Yuanchao, Xiaolong Wang, y Chong Wu. 2008. Consom: A conceptional self-organizing map model for text clustering. *Neurocomput.*

Perelomov, Ivan, Arnulfo P. Azcarraga, Jonathan Tan, y Tat Seng Chua. 2002. Using structured self-organizing maps in news integration websites.

Russell, Ben, Hujun Yin, y Nigel M. Allinson. 2002. Document clustering using the 1 + 1 dimensional self-organising map. En *IDEAL '02*.

Salton, G., A. Wong, y C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*.

Sinka, Mark P. y David W. Corne. 2005. The banksearch web document dataset: investigating unsupervised clustering and category similarity. *J. Netw. Comput. Appl.*

Vesanto, J. y E. Alhoniemi. 2000. Clustering of the self-organizing map. *IEEE-NN*, 11(3):586, May.

Zadeh, L. A. 1965. Fuzzy sets. *Information and control*.