UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

INSTITUTO DE INVESTIGACIONES EN MATEMÁTICAS APLICADAS Y EN SISTEMAS

TEORÍA DE LA COMPUTACIÓN

Vectores, entropía, y su aplicación a clasificación

T E S I S

QUE PARA OPTAR POR EL GRADO DE:
MAESTRA EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A:

KARLA ROCÍO VARGAS GODOY

Director de tesis: Dr. José David Flores Peñaloza
Facultad de Ciencias, UNAM

Cotutor: Dr. Edgar Leonel Chávez González
CICESE

Cd. de México Enero 2017

Índice general

I

Introducción

1

Introducción

La clasificación de objetos es una tarea útil en la vida cotidiana. La mayoría de los recursos que usamos se encuentran clasificados de una u otra manera. Por ejemplo, cuando vamos a una biblioteca a buscar un libro, estos están acomodados por categorías para su fácil búsqueda. Usualmente, la asignación de categorías a los libros se lleva a cabo por los empleados de la biblioteca, es decir, no se hace de manera automatizada.

Para la clasificación automática de documentos, existen diversos algoritmos que dado un conjunto de documentos y un conjunto de etiquetas o categorías, clasifican a los documentos, casi como si un humano experto hubiera asignado manualmente una categoría al documento.

Muchos clasificadores de documentos reciben una representación del documento. La representación que es de nuestro interés, es la de *bolsa de palabras*, donde el documento pierde el orden y sólo se consideran todas las palabras distintas y su frecuencia. Es decir, el documento es visto como un multiconjunto de palabras, por lo que la clasificación de un documento podría verse como la clasificación de un multiconjunto. De esta forma, podríamos pensar en una generalización de clasificación de multiconjuntos.

El trabajo que se presenta en los siguientes capítulos, es una heurística de clasificación de multiconjuntos de objetos que pertenecen a algún universo. Para esta clasificación, los objetos se representan como vectores. Estos vectores tienen una propiedad, si se define una relación de similitud entre los objetos entonces la representación vectorial mantiene dicha relación. Es decir, objetos similares aparecen cerca en algún espacio vectorial. A esta representación se le llama *encaje vectorial*.

Los encajes vectoriales son una parte muy importante en la heurística de clasificación propuesta, ya que el éxito de la clasificación de un multiconjunto depende casi completamente de cuánto se preserva la relación de similitud en el *encaje vectorial*.

Al igual que los algoritmos de clasificación de documentos, la heurística de clasificación de multiconjuntos recibe un conjunto de multiconjuntos y un conjunto de categorías. La heurística de clasificación consiste en una votación que realizan algunos descriptores del multiconjunto a las categorías.

Los descriptores son elementos que brindan información acerca del multiconjunto. Un ejemplo concreto de descriptores, son las palabras de un vocabulario que describen a un documento. Los documentos usualmente tienen un vocabulario que pertenece a un idioma en particular. Hay palabras en el documento que aparecen frecuentemente y que no describen al documento. Por ejemplo la palabra “para” es una palabra muy común

en español y si fuera tomada de algún documento, no se podría inferir mucho acerca del documento. En cambio, si se tomara una palabra como "ciencia", se podría saber más acerca del documento.

Uno de los retos importantes para la clasificación que se propone en esta tesis, es la búsqueda de buenos descriptores. Parte del trabajo que se realizó en esta tesis, fue encontrar una medida estadística que ayudara a distinguir entre elementos que describan a un multiconjunto y elementos que no.

La idea detrás de esta medida es que si un elemento aparece frecuentemente en un multiconjunto probablemente ese elemento pueda describir al multiconjunto. Pero si el elemento aparece frecuentemente en todos los multiconjuntos entonces ese elemento no describe al multiconjunto. Es como el ejemplo de las palabras "para" y "ciencia". La palabra "para" aparece frecuentemente en casi cualquier documento en español, pero la palabra "ciencia" no.

Ya que el trabajo que se presenta en esta tesis es una abstracción de la clasificación de documentos, la teoría de clasificación de documentos es importante. En el capítulo 3 se citan algunos algoritmos de clasificación y una medida de relevancia de palabras, la cual también fue generalizada para obtener descriptores de multiconjuntos.

La idea de los encajes vectoriales también fue una abstracción de los encajes de palabras que se muestran en el capítulo 4. Los encajes que se citan, logran preservar la similitud semántica de las palabras, haciendo que palabras semánticamente parecidas tengan vectores cercanos [16].

En el capítulo 5 se muestra la primera propuesta de esta tesis, que consiste en una medida estadística para poder determinar relevancia de elementos en multiconjuntos y poder obtener descriptores de estos. En el capítulo 6 se presentan dos propuestas de heurísticas de clasificación de multiconjuntos.

Finalmente, ya que se cuenta con los encajes de palabras y conjuntos de documentos, algunas pruebas de la obtención de descriptores con la medida propuesta se muestran en el capítulo 7 y algunos experimentos de clasificación de documentos se muestran en el capítulo 8.

2

Antecedentes

En este capítulo se citan algunas definiciones necesarias para la comprensión del trabajo que se presenta en los siguientes capítulos. Los antecedentes están divididos por áreas: matemáticos, lingüísticos y algorítmicos.

2.1. Matemáticos

Definición 1 (Centroide)**.** *[1] Dado un cúmulo*[1] *de vectores V, el centroide es un vector central que representa al cúmulo y se define como el vector promedio de los vectores que se encuentran en su cúmulo.*

$$\vec{\mu}(V) = \frac{1}{|V|} \sum_{\vec{v} \in V} \vec{v}$$

Definición 2 (Similitud de cúmulos)**.** *[1] Dados dos cúmulos de vectores C y D con centroides $\vec{\mu}(C)$ y $\vec{\mu}(D)$ respectivamente, la similitud entre C y D se define como la similitud de sus centroides, es decir el producto punto de $\vec{\mu}(C)$ y $\vec{\mu}(D)$.*

$$\begin{aligned} \text{similitud}(C, D) &= \vec{\mu}(C) \cdot \vec{\mu}(D) \\ &= \left(\frac{1}{|C|} \sum_{\vec{c} \in C} \vec{c} \right) \cdot \left(\frac{1}{|D|} \sum_{\vec{d} \in C} \vec{d} \right) \\ &= \frac{1}{|C||D|} \sum_{c \in C} \sum_{d \in D} \vec{c} \cdot \vec{d} \end{aligned}$$

El producto punto mide la similitud entre dos vectores. Esto lo hace midiendo el coseno del ángulo θ que forman los dos centroides $\vec{\mu}(C)$ y $\vec{\mu}(D)$ y tiene las siguientes propiedades:

1. Si $\theta = 90°$ entonces $\vec{\mu}(C) \cdot \vec{\mu}(D) = 0$
2. Si $\theta < 90°$ entonces $\vec{\mu}(C) \cdot \vec{\mu}(D) > 0$
3. Si $\theta > 90°$ entonces $\vec{\mu}(C) \cdot \vec{\mu}(D) < 0$

[1] Cluster

Entre más cercanos estén los centroides, mayor será el producto punto.

Definición 3 (Similitud coseno)**.** *[6] La similitud coseno entre dos vectores $\vec{v}$ y $\vec{u}$ se define como*

$$\cos(\vec{v}, \vec{u}) = \frac{\vec{v} \cdot \vec{u}}{|\vec{v}||\vec{u}|}$$

Dividiendo entre la longitud de cada vector, se normaliza la longitud de los vectores y de esta forma se logra medir el coseno del ángulo que forman los vectores $\vec{v}$ y $\vec{u}$.

Definición 4 (Variable aleatoria discreta)**.** *[2] Una variable aleatoria discreta X, es una variable que toma un valor x de un conjunto de probabilidades de eventos. La distribución de probabilidad de X es una función f_X que asigna a cada posible valor de X la probabilidad de que ese valor suceda. La distribución de probabilidad de una variable aleatoria discreta cumple con las siguientes propiedades:*

1. $P[X = x] = f_X(x)$
2. $f_X(x) \geq 0$
3. $\sum_i f_X(i) = 1$ *donde i representa todos los posibles valores que X puede tener y* $f_X(i) = P[X = i]$

Definición 5 (Distribución de probabilidad conjunta)**.** *[2] Sean X y Y dos variables aleatorias discretas. La distribución de probabilidad conjunta de las dos variables se define como la distribución de probabilidad de la intersección de las probabilidades de los eventos de X y Y.*

$$\begin{aligned} P(x, y) &= P(X = x \cap Y = y) \\ &= P(Y = y | X = x) P(X = x) \\ &= P(X = x | Y = y) P(Y = y) \end{aligned}$$

Definición 6 (Entropía de la información)**.** *[3] Sea $\Sigma = \{w_1, w_2, ..., w_n\}$ un conjunto de símbolos de alguna fuente de información tales que $P(w_i) = p_i$ y sea X la variable aleatoria con posibles valores $p_1, ..., p_n$. La entropía de la variable aleatoria X se define como*

$$H(X) = -\sum_{i=1}^{n} p_i \log(p_i)$$

La entropía se interpreta como la cantidad de información promedio que contienen los símbolos en un mensaje. Un símbolo con mayor frecuencia aporta menos información y un símbolo con poca frecuencia aporta más información. Cuando los símbolos son equiprobables, la entropía es máxima, por lo que también puede verse como una medida de incertidumbre.

Definición 7 (Multiconjunto)**.** *[4] Un multiconjunto o bolsa, es una colección de elementos en la que cada elemento puede aparecer más de una vez. El número de veces que un elemento e aparece en un multiconjunto, se llama multiplicidad de e. La cardinalidad del multiconjunto es la suma de la multiplicidad de sus elementos.*

Por ejemplo, dado el multiconjunto $M = \{a, b, a, b, c, a\}$, tenemos que la multiplicidad del elemento a es $\text{mul}(a) = 3$ y la cardinalidad de M es $|M| = 6$.

Definición 8 (Softmax)**.** *[5] Softmax es una función que convierte los valores de un vector $\vec{v} = (v_1, v_2, ..., v_k)$ de dimensión k a valores entre $(0, 1)$ y la suma de todas las entradas es lo más 1. La función está dada por:*

$$\sigma(v)_j = \frac{e^{v_j}}{\sum_{i=1}^{k} e^{v_k}} \forall j = 1, ..., k$$

Esta función convierte al vector $\vec{v}$ en una distribución de probabilidad donde un evento aleatorio r puede tomar algún valor de los k posibles valores. El denominador $\sum_{i=1}^{k} e^{v_k}$ se asegura que

$$\sum_{i=1}^{n} \sigma(v)_i = 1$$

Las distribución de probabilidades de r dado como entrada el vector $\vec{v}$ se puede ver como

$$\begin{bmatrix} P(r = 1|\vec{v}) \\ . \\ . \\ . \\ P(r = k|\vec{v}) \end{bmatrix} = \begin{bmatrix} \sigma(v)_1 \\ . \\ . \\ . \\ \sigma(v)_n \end{bmatrix}$$

donde $P(r = c|\vec{v})$ es la probabilidad de que r sea la clase c dado el vector $\vec{v}$ con $c = 1, ..., k$.

2.2. Lingüísticos

Definición 9 (Codificación *1-of-V*)**.** *[6] Dado un conjunto indexado Σ de longitud n y un elemento $w \in \Sigma$ en la i-ésima posición, la codificación* 1-of-V *es la representación del elemento w como un vector de dimensión n con todas las entradas en 0, excepto la i-ésima posición que es 1. A este vector también se le conoce como* one-hot*.*

Por ejemplo si el conjunto es

$$\Sigma = \{array, computer, science, sort\}$$

La codificación *1-of-V* del elemento *computer* es

$$(0, 1, 0, 0)$$

Definición 10 (Contexto)**.** *[6] Sea $T = w_1 w_2 ... w_n$ una sucesión de palabras. Dados un entero positivo k y una palabra w_i, se denomina contexto al conjunto (el orden no importa) o secuencia ordenada de palabras que se encuentran en la ventana de tamaño k que rodea a la palabra w_i.*

$$\text{contexto}_\text{k}(w_i) = \{w_{i-k}, w_{i-k+1}, ..., w_{i+1}, ..., w_{i+k}\}$$

Definición 11 (n-grama)**.** *[7] Sea $S = s_1 s_2 ... s_k$ una cadena. Los n-gramas se definen como subcadenas de S de longitud n. Los 1-gramas se llaman unigramas, los 2-gramas se llaman bigramas.*

Definición 12 (Punto de información mutua)**.** *[6] Sean X y Y dos variables aleatorias discretas. Dados dos eventos x de X y y de Y, PMI se define como*

$$PMI(x, y) = \log \frac{P(x, y)}{P(x)P(y)}$$

con $P(x, y)$ la probabilidad conjunta.

El punto de información mutua (PMI) es una medida de asociación entre un par de variables aleatorias discretas X y Y, que dice que tan seguido ocurren dos eventos x y y (el factor $P(x, y)$) comparado con su distribución individual ($P(x)$ y $P(y)$).

Por ejemplo, consideremos la siguiente tabla de co-ocurrencias de palabras en un texto. *contexto* será la primera palabra a la izquierda de *palabra* en el texto.

palabra / contexto	**algoritmo**	**animal**	**alhambra**
complejidad	3	1	0
elefante	0	4	0

Cuadro 2.1: Co-ocurrencias de palabras en un texto

Y ahora veamos la siguiente tabla con su probabilidad conjunta. La probabilidad conjunta es $P(\text{palabra}, \text{contexto})$.

palabra / contexto	**algoritmo**	**animal**	**alhambra**
complejidad	3/8	1/8	0
elefante	0	4/8	0

Cuadro 2.2: Probabilidades conjuntas

La siguiente tabla muestra las probabilidades individuales de las *palabras* y los *contextos*.

Tenemos entonces que

$$PMI(\text{palabra}, \text{contexto}) = \log_2 \left(\frac{P(\text{palabra}, \text{contexto})}{P(\text{palabra})P(\text{contexto})} \right)$$

Midamos cómo se asocian las palabras *algoritmo* y *animal* con el contexto *complejidad*. $PMI(\text{algoritmo}, \text{complejidad}) = -2$ y $PMI(\text{animal}, \text{complejidad}) = -4{,}3$ de lo que obtenemos que *algoritmo* está más relacionada con *complejidad* que *animal* con *complejidad*.

Palabra	P(palabra)	Contexto	P(contexto)
algoritmo	3/8	complejidad	4/8
animal	5/8	elefante	4/8
alhambra	0		

Cuadro 2.3: Probabilidades individuales

2.3. Algorítmicos

Definición 13 (Redes neuronales)**.** *[8] Una red neuronal artificial (RNA) es un modelo de aprendizaje que consiste en unidades llamadas neuronas que reciben entradas $x_1, x_2, ..., x_n$ a través de interconexiones que son multiplicadas por pesos $w_1, w_2, ..., w_n$, que denotan el peso de las interconexiones. La salida de la red está dada por 3 funciones:*

1. *Una función de propagación que se define como*

$$p = \sum_{i=1}^{n} x_i w_i$$

2. *Una función de activación de las neuronas que modifica a p*

3. *Una función de transferencia, que acota la salida de la función de activación*

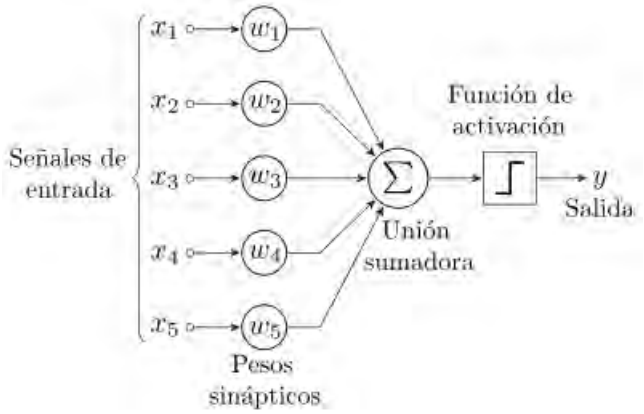

Figura 2.1: Diagrama de un perceptrón el cual es un tipo de RNA

Las RNA, al estar inspiradas en el sistema nervioso humano, son capaces de aprender en una etapa de aprendizaje, la cual consiste en pasarle datos de entrada e indicándole cuál es la salida esperada. En el aprendizaje supervisado se le proporcionan a la red ejemplos de entrada y salida de lo que se quiere calcular y, con base en los ejemplos, se calcula el error.

Existen muchos modelos de RNA en la literatura. El modelo que es de nuestro interés es el de capas en el cual las neuronas están organizadas en 3 capas: entrada, escondida y salida. Usualmente este modelo utiliza el algoritmo de propagación hacía atrás en la etapa de aprendizaje.

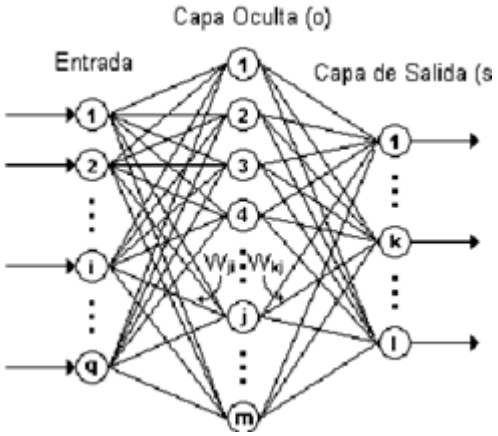

Figura 2.2: RNA multicapa

El algoritmo de propagación hacia atrás (backpropagation) es un algoritmo de aprendizaje supervisado. El objetivo de la propagación hacia atrás, es que la red aprenda y al mismo tiempo minimice el error. El entrenamiento comienza con pesos aleatorios entre las conexiones de las capas y la meta es ajustarlos hasta que el error sea mínimo. Esta es la forma en la que la red aprende.

El funcionamiento general del algoritmo de propagación hacia atrás es el siguiente:

1. Se aplica un patrón a la capa de entrada de la red que se propaga hasta la capa de salida

2. La salida se compara con la entrada deseada y se calcula el error para cada una de las salidas

3. El error se propaga desde la capa de salida hacia la de entrada y se actualizan los pesos

II

Marco teórico

3

Clasificación de multiconjuntos

La clasificación de multiconjuntos es una abstracción de la clasificación de documentos. Los documentos pueden ser vistos como multiconjuntos, donde la multiplicidad de una palabra es la frecuencia con la que aparece el documento. El orden de las palabras en el documento es irrelevante, ya que sólo importa la frecuencia con la que aparecen las palabras en él. A este modelo se le llama *bag of words*.

Nuestro trabajo, presentado en los capítulos 5 y 6, se trata de dicha abstracción, por lo que es importante la teoría de clasificación de documentos que se presenta en las siguientes secciones.

3.1. Clasificación de documentos

La clasificación de documentos es la tarea de asignar a un documento una o más categorías, lo cual se logra a través de un clasificador que recibe al documento o una representación de el y lo etiqueta con alguna categoría. También puede referirse al agrupamiento de un conjunto de documentos con contenido similar. La clasificación de documentos tiene muchas aplicaciones como lo son el filtro de correo no deseado, regresar resultados relevantes adicionales en una consulta, por ejemplo, consultas en la web y recuperación de información [9].

Comúnmente los clasificadores reciben una representación de los documentos. Esta representación se basa en descriptores, que usualmente son palabras que describen al documento. En la siguiente sección se explica más a detalle cómo obtener descriptores de un documento para su clasificación.

3.2. Obtención de descriptores

Para obtener una representación del documento con descriptores, usualmente se lleva a cabo un preproceso del documentoy este se lleva a cabo sin importar qué tipo de clasificador se utilice. Palabras muy frecuentes que no tienen algún significado en el documento se eliminan, por ejemplo, preposiciones, artículos, etc. Algunos algoritmos también eliminan palabras que son el plural de otras, ya que tienen el mismo significado.

Los clasificadores usualmente reciben al documento representado como un vector, en el que cada entrada del vector representa una característica o descriptor del documento.

Los descriptores usualmente son palabras que describen al documento.

La selección de descriptores es un paso importante en el preproceso ya que define la dimensión del vector con la que se representa el documento y la clasificación dependerá únicamente de estos descriptores.

Para la selección de palabras se utilizan medidas como *TF-IDF*[1], *PMI*, entropía, frecuencia normalizada, etc. en la que a las palabras se les da un peso según su relevancia y el orden en el que son seleccionadas no importa. En la siguiente sección se describe el índice de relevancia *TF-IDF*.

3.2.1. Frecuencia de palabras - Frecuencia inversa de los documentos

En la recuperación de información, se busca obtener datos que describan fuentes de información. En particular, las medidas de relevancia en documentos son estadísticas numéricas que indican la relevancia de las palabras en documentos. Para ello, se analizan conjuntos de documentos y con base en frecuencia de palabras, se puede obtener un conjunto de palabras que describen al documento.

Term Frecuency-Inverse Document Frecuency (*TF-IDF*) es una medida que determina el peso o relevancia de una palabra para un documento en un conjunto de documentos [10][6].

La idea de esta medida es que si una palabra aparece frecuentemente en un documento, es importante, pero si esa palabra aparece en muchos documentos, no es una palabra que identifique al documento.

La medida *TF-IDF* se compone de dos funciones. La función *tf* que asigna un peso grande si la palabra aparece frecuentemente en el documento y la función *idf* que compensa el peso dependiendo de si la palabra aparece en muchos documentos o no. De esta forma, las palabras con mayor peso serán las más relevantes para el documento.

Formalmente, dado un conjunto $D = T_1, ..., T_n$ de documentos, el peso *TF-IDF* de la palabra w en el documento T_i está definido como sigue:

$$\text{TF-IDF}(w, T_i, D) = tf(w, T_i) \cdot idf(w, D)$$

donde tf se define como

$$tf(w, T_i) = \frac{f(w,T_i)}{|T_i|}$$

donde $f(w, T_i)$ es el número de veces que la palabra w aparece en T_i y idf es

$$idf(w, D) = log\left(\frac{|D|}{|\{T \in D : w \in T\}|+1}\right)$$

Para obtener las palabras más relevantes para un documento T_i, se calcula el *TF-IDF* para cada $w \in \Sigma_{T_i}$ el vocabulario del documento T_i. Después se ordenan por su índice *TF-IDF* y entonces las palabras con mayor peso son las más relevantes para el documento T_i.

En una de las representaciones vectoriales de los documentos, cada entrada del vector es el peso *TF-IDF* de algún descriptor.

Las ventajas que tiene *TF-IDF* son:

[1]Frecuencia de palabras - Frecuencia inversa de los documentos

- Es fácil de calcular
- Es una medida sencilla

Por otro lado, la desventaja más grande que tiene *TF-IDF* es que no considera ocurrencias de las palabras en otros documentos. La frecuencia normalizada sólo se toma para el documento que se está analizando y aporta un peso grande si la palabra aparece muchas veces en el documento.

Si consideramos un conjunto de documentos del mismo tema, la palabra podría aparecer frecuentemente en los demás documentos pero no con la misma relevancia, por lo que la función *idf* le dará un peso pequeño y hará que la palabra tenga un índice pequeño. Algunos ejemplos de obtención de palabras relevantes se muestran en el capítulo 7.

Por lo tanto, podemos concluir que esta medida funciona bien para determinar la relevancia de las palabras de conjuntos de documentos heterogéneos, es decir, que no sean todos del mismo tema.

3.3. Tipos de clasificación

La clasificación de documentos puede llevarse a cabo de dos formas, de manera supervisada y de manera no supervisada.

Si se quiere etiquetar al documento con una categoría, el clasificador recibe como entrada al documento y realiza la clasificación. A esto se le conoce como clasificación supervisada.

Cuando se quiere agrupar documentos que pertenecen a la misma categoría se utilizan algoritmos de agrupamiento[2].A esto se le conoce como clasificación no supervisada.

En las siguientes secciones se explican ambas clasificaciones.

3.3.1. Clasificación supervisada

La clasificación supervisada está estrechamente relacionada con algoritmos de aprendizaje supervisado. Definamos formalmente cuál es el objetivo del aprendizaje supervisado.

Definición 14 (Aprendizaje Supervisado)**.** *[11] Dado un conjunto de entrenamiento* $T = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ *de n ejemplos de parejas (entrada,salida), donde cada* y_j *fue generada por una función desconocida* $f(x) = y$*, la tarea del aprendizaje supervisado es encontrar una función h que aproxime a la función original f.*

Los valores x y y pueden tomar cualquier valor, no necesariamente numéricos. La función h se llama *hipótesis* y el aprendizaje consiste en buscar en el espacio una hipótesis que aproxime lo mejor posible, incluso para entradas x que no estén en el conjunto de entrenamiento. Cuando la salida y es algún elemento de un conjunto finito de valores, el problema de aprendizaje es llamado clasificación.

Entre los algoritmos de aprendizaje supervisado para clasificación se encuentran las redes neuronales, árboles de decisión, máquinas de vectores de soporte, entre otros [11].

[2]Clustering

Los clasificadores de documentos supervisados aprenden un modelo a partir de un conjunto de entrenamiento T, donde el conjunto tiene la forma $T = \{(D_1, c_1), ..., (D_n, c_m)\}$, con D_i un documento y c_i una categoría, es decir, se toma un conjunto de documentos y cada uno de ellos se etiqueta a mano con sus respectivas categorías. El clasificador es entrenado con el conjunto de entrenamiento T y al recibir como entrada un documento nuevo que no está en T, sin etiquetar, devuelve a qué categoría pertenece.

Sin importar qué algoritmo de aprendizaje se utilice, la calidad del conjunto de entrenamiento que recibe el clasificador debe ser buena y lo suficientemente grande para que cada categoría tenga asignada un buen número de documentos. También es importante que los documentos sean distintos entre ellos para que pueda hacerse una distinción clara entre categorías.

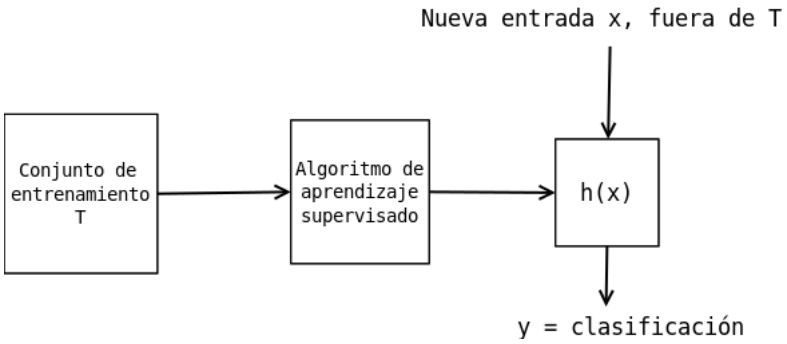

Figura 3.1: Aprendizaje supervisado

3.3.2. Clasificación no supervisada

La clasificación no supervisada o el agrupamiento de documentos, a diferencia de la clasificación supervisada, no recibe un conjunto de entrenamiento ni categorías predefinidas. La asignación de categorías a los documentos se debe llevar a cabo únicamente con el contenido de los documentos [12].

La mayoría de los clasificadores no supervisados, al ser casi todos algoritmos de agrupamiento, reciben al documento representado como vector y lo agrupan con los documentos que tengan vectores similares [12][13].

Para medir la similitud entre vectores, mayormente se utiliza la similitud coseno pero también se utiliza la distancia euclidiana. Otras medidas de similitud de documentos a considerar son la longitud del documento, el número de palabras en común y cuántas veces aparecen en ellos.

Muchos de los algoritmos de agrupamiento se basan en centroides. Entre los algoritmos de agrupamiento basados en centroides se encuentran los siguientes:

- *K-Means* [12]
  Dado un conjunto de n documentos representados en vectores, se seleccionan k documentos para crear k grupos. Cada grupo está representado por el vector del documento, es decir, el vector del documento es el centroide del grupo. El algoritmo lleva a cabo los siguientes dos pasos hasta que cada documento sea asignado a su propio grupo:

1. Asignar el documento d_i al grupo con el centroide más cercano
2. Recalcular los centroides de los grupos recién creados

- *Agrupamiento jerárquico* [12]

  Para los algoritmos de agrupamiento jerárquico, se representan los documentos en vectores. Estos algoritmos producen una jerarquía de grupos llamada dendograma. Existen dos categorías de estos algoritmos, los divisivos y los aglomerativos.

  Para el agrupamiento divisivo, se forma un sólo grupo con todos los vectores de los documentos. En cada paso, el grupo con el mayor diámetro se separa, es decir, el grupo que tiene al par de vectores más distantes. Esto quiere decir que dentro del grupo, el documento menos similar es el que se separa y crea un grupo nuevo. Después, cada vector en el grupo que se acaba de separar, es analizado: si es más parecido al vector del grupo nuevo, se asigna al grupo recién creado.

  Para el agrupamiento aglomerativo, cada documento empieza en un grupo distinto y en cada iteración, se mezclan los grupos más similares hasta llegar a un criterio de paro, el cual puede ser un número fijo de grupos.

4

Encaje de palabras en espacios vectoriales

4.1. Definición

El Procesamiento del Lenguaje Natural (NLP) se refiere a un área de las Ciencias de la Computación que estudia algoritmos que procesan la escritura y el habla del lenguaje natural[14]. La dificultad del procesamiento del lenguaje natural radica en entender el significado de las palabras, analizar la estructura semántica y sintáctica del lenguaje y poder generar el lenguaje mismo. Algunas de las tareas que pueden resolverse son la traducción de un idioma a otro, comprender y representar el contexto de un documento, generar resúmenes de documentos, clasificar palabras, etc.

Para obtener conocimiento de las características del lenguaje natural, se pueden utilizar encajes de palabras[1]. Un encaje de palabras consiste en representar cada palabra como un vector. En un encaje de palabras de *alta calidad* se espera que palabras semánticamente similares aparezcan cerca en el espacio vectorial [16].

Entre los algoritmos de encaje de palabras más populares se encuentran GloVe[2] y word2vec[3] que serán descritos en las siguientes secciones.

4.2. La técnica GloVe

GloVe es un algoritmo de aprendizaje no supervisado basado en conteo de coocurrencias de palabras, es decir, dos palabras son similares semánticamente si aparecen dentro del mismo contexto frecuentemente [15].

Ya que GloVe es un algoritmo de aprendizaje no supervisado, la principal fuente de información para este tipo de algoritmos son las estadísticas de ocurrencias de palabras en el texto. Pero con únicamente estadísticas no es sencillo capturar el significado de las palabras también.

Para resolver el aprendizaje del significado de las palabras, GloVe funciona bajo la idea de que aspectos del significado pueden ser extraídos de la probabilidad de coocu-

[1] Word Embedding

[2] http://nlp.stanford.edu/projects/glove/

[3] https://code.google.com/archive/p/word2vec/

rrencias [15]. Por ejemplo, consideremos dos palabras i y j, lo que se quiere es dar es una métrica con respecto a estas dos palabras para ver cómo se relacionan con otra palabra, digamos k. La relación puede ser examinada al analizar el cociente de la probabilidad de su coocurrencia, es decir a $\frac{P(k|i)}{P(k|j)}$ donde $P(k|i)$ denota la probabilidad de que la palabra k aparezca en el contexto de la palabra i. Los tres casos que pueden pasar son los siguientes:

1. $P(k|i) > P(k|j)$
2. $P(k|j) > P(k|i)$
3. $P(k|i) \simeq P(k|j)$

En el caso 1, el cociente $\frac{P(k|i)}{P(k|j)}$ será relativamente grande, lo cual quiere decir que la palabra k está más relacionada con la palabra i que con la palabra j. En el caso dos, el cociente $\frac{P(k|i)}{P(k|j)}$ será más pequeño, lo cual quiere decir que la palabra k está más relacionada con j que con i. Finalmente, en el tercer caso, cuando las probabilidades sean casi las mismas el cociente será casi 1, lo cual significa que k está relacionada con i y j de la misma manera, es decir, la palabra k puede estar relacionada con ambas y aparecer frecuentemente con i y j, pero también está el caso en el que la palabra no esté relacionada de ninguna forma con i y j.

La siguiente figura muestra un ejemplo de probabilidades obtenidas en conjunto de documentos de entrenamiento.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figura 4.1: Cociente de probabilidades

GloVe es una de las técnicas más exitosas, pero como su autor aceptó, las tareas de analogía que pueden realizarse con los encajes que produce word2vec (que se analizarán en la sección 4.3.1) no eran mejores [15].

4.3. La técnica word2vec

Uno de los esquemas algorítmicos más populares para encajar palabras, debido a su eficiencia y efectividad, es wor2vec. Para encajar las palabras, word2vec utiliza una red neuronal multicapa que aprende supervisadamente relaciones entre palabras y su significado [16].

La siguiente descripción es una simplificación de cómo funciona la red neuronal que utiliza word2vec [18].

El objetivo de esta red en particular es aprender la relación entre bigramas, en el que sólo consideramos una palabra en el contexto. Sea w la palabra que vamos a analizar,

la cual se llama *palabra objetivo* y v la palabra *contexto* Esta relación se aprende de un conjunto de entrenamiento que le será pasado a la red en el que sabemos qué bigramas están relacionados. El objetivo del entrenamiento es maximizar la probabilidad condicional de observar la palabra w dada la palabra v.

Sea Σ el vocabulario que se va a encajar, $n = |\Sigma|$ y d la dimensión del encaje. La capa de entrada tiene n neuronas y la capa escondida d neuronas, por lo que los pesos de las conexiones entre la capa de entrada y la escondida están dados por la matriz W_{in} de dimensión $n \times d$. La capa de salida tiene n neuronas, entonces los pesos de las conexiones entre la capa escondida y la capa de salida están dados por la matriz $\mathrm{W}_{\mathrm{out}}$ de dimensión $d \times n$.

La red recibe como entrada el vector X que corresponde a la codificación $1 - of - V$ de v, la palabra contexto. La capa de entrada produce la matriz $H^\top$ que recibe la capa escondida y está definida como

$$H^\top = X^\top \times \mathrm{W}_{\mathrm{in}}$$

La función de activación de la capa escondida hacia la de salida está dada por

$$\mathrm{activation}_{\mathrm{V}} = H^\top \times \mathrm{W}_{\mathrm{out}}$$

Para encajar palabras similares en vectores similares y alejar vectores de palabras que no estén relacionadas, se debe reflejar la relación entre w y v. El objetivo es hacer que la probabilidad $P(w|v)$ sea grande cuando w y v están juntas en el texto y las probabilidades $P(w_i|v)$ con las demás palabras w_i con $i = 1, ..., n$ deben ser calculadas también. La suma de la salida de las neuronas en la capa de salida debe ser 1 por lo que se utiliza la función *Softmax*, la cual también contrasta las probabilidades, maximizando la probabilidad $P(w|v)$ cuando w y v están juntas en el texto.

Por lo que la salida de la k-ésima neurona está dada por la siguiente expresión

$$o_k = P(w_k|v) = \frac{e^{(\mathrm{activation}_{\mathrm{V}}(k))}}{\sum_{j=1}^{n} e^{(\mathrm{activation}_{\mathrm{V}}(j))}}$$

donde activation(k) es la entrada k-ésima del vector $\mathrm{activation}_{\mathrm{V}}$. Llamemos O al vector de salida que producen las n neuronas. Finalmente, el error es calculado de la siguiente forma

$$\mathrm{error}_{\mathrm{V}} = Y - O$$

donde Y es la codificación $1 - of - V$ de w. Una vez que se calcula el error, los pesos de las matrices W_{in} y $\mathrm{W}_{\mathrm{out}}$ son actualizados usando propagación hacia atrás.

El funcionamiento de la red descrita es la idea del funcionamiento de los algoritmos que propone word2vec, la diferencia es que la relación que aprenden estos algoritmos, es entre *palabras objetivo* y sus *contextos*.

En pocas palabras, la red neuronal que utiliza wor2vec para encajar las palabras en vectores, es entrenada para que dado un par (w, c) de palabra objetivo y contexto aprenda la relación entre ellos. Después de que la red es entrenada, las palabras quedan encajadas en la matriz de pesos entre la capa de entrada y la escondida.

Mientras más grande es el tamaño del contexto, se obtiene mejor calidad en los vectores a cambio de más tiempo de entrenamiento. Pero también tenemos que las palabras más distantes a la palabra objetivo, usualmente están menos relacionadas que las que son más cercanas, por lo cual se les da menos peso haciendo el muestreo de las palabras distantes más pequeño [17].

El entrenamiento de word2vec es sensible a los parámetros que recibe, entre ellos el tamaño del contexto y la dimensión del encaje [16].

El encaje resultante manda a palabras parecidas semánticamente a vectores cercanos en el espacio. La siguiente tabla es un ejemplo de la cercanía de las palabras en el espacio.

Palabra	**Vectores más cercanos**
cow	cows pig bovine
mexico	monterrey peru honduras
science	biology scientist mathematics

Cuadro 4.1: Encajes de palabras y su cercanía

Hay dos observaciones importantes acerca del encaje que produce word2vec. La primera es que experimentalmente se ha observado que la mayoría de sus vectores quedan contenidos en un cono de 110°.

La segunda observación es que sucede que la palabras “american” y “airlines” no son una combinación “natural” de conceptos, pero al ser combinadas obtenemos “american airlines”, que sabemos que es una aerolínea. Para encajar este tipo de frases, la red neuronal se entrena exactamente igual, sólo que ahora la palabra objetivo debe ser un n-grama y sólo se considerarán a los n-gramas que aparezcan con cierta frecuencia en el corpus de entrenamiento [17].

Palabra compuesta	Palabras más cercanas
martial_arts	karate kung_fu kickboxing
national_park	big_bend national_forest national_wildlife_refuge
american_airlines	southwest_airlines to_fly airtran takes_off gulf_air
computer_science	electrical_engineering mechanical_engineering physics

Cuadro 4.2: Encajes de palabras compuestas

Los algoritmos presentados en word2vec producen un encaje de buena calidad, palabras similares tienen vectores similares y además, con una operación algebraica entre vectores, pueden encontrarse relaciones semánticas entre palabras [16]. La siguiente sección explica la estructura semántica que captura el encaje producido por wor2vec.

4.3.1. Estructura semántica capturada por word2vec

Los encajes de palabras en espacios vectoriales sirven para poder capturar características del lenguaje natural. Por ejemplo, pueden aprender similitudes sintácticas y semánticas de palabras.

El encaje que produce wor2vec tiene otra característica, puede encontrar relaciones semánticas. Las relaciones semánticas pueden verse como analogías. Por ejemplo la analogía

hombre es a rey
mujer es a reina

captura el género del adjetivo rey, para el caso de hombre es rey y para el de mujer es reina. Otro ejemplo es la analogía

París es a Francia
Roma es a Italia

que captura que x es la capital de y.

Los encajes de word2vec capturan relaciones semánticas con una simple operación de vectores. Dada una analogía del tipo X es a Y como W es a Z, se encuentra que

$$vector(R) = vector(X) - vector(Y) + vector(W)$$

el vector $vector(R)$ el vector más cercano al vector $vector(Z)$.

Algunos ejemplos de relaciones están en la siguiente tabla:

Analogía	**Palabra esperada**	**Palabras más cercanas en word2vec**	**Relación semántica**
he is to his she is to	her	her Her herself hers my	Género
horse is to horses zebra is to	zebras	zebras giraffes ostriches animals red_ruffed_lemurs	Cuantificación
milk is to cow wool is to	sheep	sheep Angora_goat sheep_wool angora_goats Merino_sheep	Similitud
drive is to car ride is to	bike	bike scooter bikes motocross_bike motorcycle	Acción a objeto
mother is to home teacher is to	school	school teachers classroom elementary Middle_School	Persona a situación

La tabla anterior se obtuvo al correr tareas de similitud con la biblioteca para Python Gensim [4], que implementa wor2vec. Las tareas de similitud se miden con la similitud coseno de los vectores, que mide el coseno del ángulo entre ellos.

4.3.2. word2vec como factorización de una matriz

En "Neural Word Embedding as Implicit Matrix Factorization" [20] se interpreta el encaje de palabras como la factorización de una matriz, particularmente se analiza a word2vec con el modelo Skip-gram con Negative Sampling en la función de salida.

El modelo Skip-gram supone un alfabeto de palabras Σ_W y un alfabeto de *contextos* Σ_C. Ambos alfabetos provienen de $w_1, w_2, ..., w_n$, un corpus de palabras donde n típicamente es del orden de millones de millones.

Cada palabra w está asociada a un vector $\vec{w} \in \mathbb{R}^d$ y cada contexto c está asociado a un vector $\vec{c} \in \mathbb{R}^d$, con d la dimensión del encaje. Podemos ver a los vectores $\vec{w}$ como los renglones de una matriz W de dimensión $|\Sigma_W| \times d$ y a los vectores $\vec{c}$ como los renglones de una matriz C de dimensión $|\Sigma_C| \times d$.

Para la función Negative Sampling, se considera un par (w, c) palabra contexto. Lo que trata de determinar es si ese par proviene del conjunto D (los datos observados). Sea $P(D = 1|w, c)$ la probabilidad de que el par (w, c) proviene de D. La distribución se modela como sigue:

[4] https://radimrehurek.com/gensim/models/word2vec.html

$$P(D = 1|w, c) = \sigma(\vec{w}, \vec{c}) = \frac{1}{1+e^{-\vec{w},\vec{c}}}$$

donde $\vec{w}$ y $\vec{c}$ vectores de dimensión d, son los parámetros del modelo a ser aprendidos.

Skip-gram y Negative Sampling (SGNS) encajan las palabras y los contextos en $\mathbb{R}^d$, que resulta en las matrices W y C, donde W es la matriz que nos interesa ya que ahí están los encajes de las palabras.

Ahora consideremos la matriz $M = W \cdot C^{\intercal}$. De esta forma, SGNS puede ser visto como la factorización de una matriz implícita M de dimensión $|\Sigma_W| \times |\Sigma_C|$. Una entrada de la matriz M corresponde a

$$M_{i,j} = W_i \cdot C_j = \vec{w_i} \cdot \vec{c_j}$$

De lo anterior se puede ver que SGNS factoriza una matriz donde los renglones son $w \in \Sigma_W$ y las columnas son $c \in \Sigma_C$, por lo que cada entrada está dada por $f(w, c)$, una medida de asociación entre la palabra w y el contexto c. En [20] demuestran que esa medida de asociación es

$$M_{i,j} = W_i \cdot C_j = \vec{w_i} \cdot \vec{c_j} = PMI(w_i, c_j) - log\ k = log\left(\frac{\#(w,c)\cdot|D|}{\#(w)\cdot\#(c)}\right) - log\ k$$

con D la colección de palabras y *contextos* observados, $\#(w, c)$ indica el numero de veces que el par (w, c) aparece en D. $\#(w)$ y $\#(c)$ son el número de veces que w y c aparecen en D respectivamente.

Ya que la dimensión del encaje es un parámetro, la elección de la dimensión puede no permitir una perfecta reconstrucción de M pero sí una aproximación. Por esta razón, el problema puede ser visto como una factorización de matrices pesada, donde la función objetivo será buscar una factorización óptima de la matriz M bajo alguna medida que dé más peso a pares (w, c) que aparezcan frecuentemente.

La descripción de la factorización ayuda a entender qué hace word2vec de fondo, pero sigue sin explicar por qué captura relaciones semánticas.

4.4. El algoritmo doc2vec

El algoritmo de encaje de doc2vec es una extensión del algoritmo word2vec, en la que el objetivo es encajar documentos o enunciados en lugar de palabras en el espacio vectorial. El algoritmo doc2vec busca dar una representación más robusta para los documentos que la representación de *bolsa de palabras*, en la que el orden de las palabras se pierde y no hay noción de semántica en los vectores que codifican al documento [9].

Para encajar los enunciados, el algoritmo doc2vec lleva a cabo dos etapas de entrenamiento. En la primera etapa se entrena una red para obtener los vectores de las palabras en una matriz W. Dicha red es la de wor2vec, descrita en la sección 4.3.

Para la segunda etapa de entrenamiento, el enunciado a ser encajado es visto como otra palabra que se representa como un vector en una matriz D. La red recibe como entrada las palabras de *contextos* muestreados aleatoriamente a lo largo del enunciado y la matriz D para ser entrenada para predecir la palabra siguiente dentro del enunciado.

La relación que aprende la red de word2vec es entre parejas (w, c) palabra objetivo y contexto. Al final del entrenamiento, la red es capaz de predecir una palabra objetivo dado

un contexto o viceversa, dependiendo del modelo que se use. Para encajar el enunciado en un vector, la red de doc2vec aprende la relación entre la tripleta (e, c, n) enunciado, contexto y palabra siguiente, es decir, la red es entrenada para poder predecir la siguiente palabra en el enunciado dado un contexto.

El algoritmo doc2vec opera de la siguiente manera:

1. Entrena a una red como la de word2vec para obtener los vectores de las palabras en la matriz W

2. Entrena a otra red con *contextos* tomados a lo largo del enunciado para aprender la relación contexto-palabra siguiente y ajusta los pesos en la matriz D

Ya que doc2vec funciona bajo el mismo esquema que word2vec, la representación que produce es de alta calidad. Enunciados relacionados aparecerán cerca en el espacio vectorial y como la representación del documento es robusta, los algoritmos de agrupamiento logran agrupamientos de alta calidad. Sin embargo, doc2vec solo no es un algoritmo de clasificación, solo una forma de representar enunciados como vectores.

III

Aportaciones

5

Medida de relevancia en conjuntos

En el capítulo 3 se mencionó que algunos clasificadores de documentos recibían como entrada una representación del documento. Algunas veces, el documento se representaba con base en sus descriptores. Para obtener descriptores del documento teníamos la función *TF-IDF* que asigna índices de relevancia a las palabras para poder distinguir entre palabras que describían al documento y palabras que no.

Lo que se propone en las siguiente sección, es una abstracción de la función *TF-IDF* con el mismo propósito, que es asignar índices de relevancia.

5.1. Entropía de frecuencias normalizada (*EFN*)

La Entropía de Frecuencias Normalizada (*EFN*) es una función que asigna un índice de relevancia a elementos de estructuras de las cuales se pueden obtener histogramas como multiconjuntos por ejemplo. *EFN* es una función que se creó en este trabajo de tesis.

Fueron dos las motivaciones para crear una medida de relevancia en conjuntos. La primera fue que *TF-IDF*, descrita en la sección 3.2.1, es una medida de relevancia para palabras en documentos. Para asignarle un índice de relevancia a una palabra en un documento, *TF-IDF* recibe un conjunto de documentos y con base en la frecuencia de la palabra en el documento y en el conjunto de documentos, se calcula el índice de relevancia. La principal desventaja que tiene *TF-IDF* es que la relevancia de una palabra está asociada a si aparece o no en los demás documentos sin medir cuanta importancia tiene en ellos, lo que muchas veces resulta una medida poco precisa.

La segunda motivación es que nos interesa una generalización de una medida de relevancia en objetos que puedan tener histogramas asociados. Esto para poder clasificar con base en descriptores de conjuntos, lo cual se explica en el capítulo 6.

EFN se pensó cómo una generalización de la medida de relevancia en documentos *TF-IDF*. Para poder determinar la relevancia de un elemento, en vez de conjuntos de documentos, se consideran conjuntos de multiconjuntos y la relevancia de los elementos en los multiconjuntos se cuantifica siguiendo mismo principio que *TF-IDF*: si un elemento aparece frecuentemente en un multiconjunto es relevante pero si aparece en todos los multiconjuntos, no lo es.

Al igual que *TF-IDF*, la función *EFN* está compuesta de dos partes: una que mide la relevancia en el conjunto de multiconjuntos que se compensa con la importancia del

elemento en el multiconjunto.

La diferencia fundamental entre *EFN* y *TF-IDF*, es que *EFN* no sólo cuenta las ocurrencias de los elementos en otros multiconjuntos, sino que también mide la relevancia que tienen en ellos. Es decir, la función *idf* cuenta en cuántos documentos del conjunto aparece una palabra, pero sólo cuenta si aparece o no en el documento y no mide con cuánta importancia aparece la palabra en los documentos.

La idea intuitiva es que dado un conjunto de multiconjuntos $\zeta = \{M_1, ..., M_n\}$ definido sobre un universo, el cálculo de la relevancia de un elemento e en el conjunto ζ, se basa en la siguiente idea: si e aparece frecuentemente en todos los multiconjuntos entonces tendrá un histograma homogéneo, lo cual quiere decir que e no aporta mucha información acerca de ζ. En cambio, si e tiene un histograma heterogéneo significa que es relevante en los multiconjuntos en los que aparece. La relevancia de e en el conjunto ζ puede ser medida con la entropía.

El enfoque de la entropía en la literatura, es de una medida que indica cuanta información posee un símbolo en una fuente de información [3]. Otra interpretación de la entropía puede ser vista con histogramas, no necesariamente de símbolos o palabras de alguna fuente de información. En cualquier estructura que tenga elementos en ella, en la cual puedan crearse histogramas a partir de las cardinalidades de los elementos, la entropía puede ser usada para medir la relevancia de sus elementos.

Por otro lado, se necesita también un factor que determine la importancia del elemento e en un sólo multiconjunto M_i. Este es la frecuencia normalizada del elemento en el conjunto ζ. A continuación se da una descripción más formal.

Dado un universo U y $\zeta = \{M_1, ..., M_n\}$ un conjunto de multiconjuntos donde $\forall m \in M_i$ se tiene que $m \in U$, el histograma del elemento e en el conjunto de multiconjuntos ζ se representa con el vector de frecuencias normalizadas, que se calcula de la siguiente forma:

$$\text{nfvector}(e, \zeta) = \left(\frac{\text{mul}(e, M_1) + 1}{|M_1|}, ..., \frac{\text{mul}(e, M_n) + 1}{|M_n|} \right) = (f_1, ..., f_n)$$

donde $\text{mul}(e, M_i)$ denota la multiplicidad del elemento e en el multiconjunto M_i y $|M_i|$ denota la cardinalidad del multiconjunto $|M_i|$.

Para convertir el vector $nfvector$ en un vector de una distribución de probabilidad discreta, se suma cada entrada del vector

$$total = \sum_{i=1}^{n} f_i$$

y la i-ésima entrada del vector es la probabilidad de que el elemento e esté en el multiconjunto M_i

$$\text{dpvector}(\text{nfvector}(e, \zeta)) = \left(\frac{f_1}{total}, ..., \frac{f_n}{total} \right) = (p_1, ..., p_n)$$

Es importante notar que esta distribución es la que determina la importancia de los elementos en cada multiconjunto, es decir, la entrada i-ésima del vector de probabilidades contiene el porcentaje de veces que aparece el elemento e en el multiconjunto M_i del total de veces que aparece en todo el conjunto ζ.

Ya que se definió la distribución de probabilidades, entonces la entropía de un elemento e en el conjunto de multiconjuntos ζ está dada por

$$H(e,\zeta) = -\sum_{i=1}^{n} p_i \log(p_i)$$

Ya que $\log(0) = \infty$, p_i debe ser distinto de 0. Para evitar que $p_i = 0$ se debe hacer que $f_i \neq 0$, lo cual se logra sumando un 1 a la multiplicidad del elemento al calcularse el vector nfvector(e,ζ). Es por eso que

$$f_i = \frac{\text{mul}(e, M_i) + 1}{|M_i|}$$

Las siguientes observaciones deben ser consideradas:

1. Cuando los elementos son equiprobables, la entropía resultará en $\log(n)$
2. Cuando un elemento aparece casi con la misma frecuencia en todos los multiconjuntos, tiene un histograma homogéneo por lo que la distribución de probabilidad es muy parecida a una distribución de elementos equiprobables

De las observaciones anteriores obtenemos que $H(e,\zeta)$ será muy cercana a $\log(n)$ cuando e aparezca homogéneamente en el conjunto de multiconjuntos. Como *EFN* está pensada para la relevancia de elementos en un conjunto de multiconjuntos, entonces se divide a $H(e,\zeta)$ entre el $\log(n)$ y la relevancia del elemento e en el conjunto ζ se define como

$$\text{rel}(e,\zeta) = 1 - \frac{H(e,\zeta)}{\log(n)}$$

de esta forma, el índice de relevancia del elemento e en el conjunto ζ está entre $[0,1]$. Los elementos con índice de relevancia más cercanos a 1, son los elementos más relevantes en el conjunto ζ y los que tienen índice más cercano a 0 son los menos relevantes.

Finalmente, para calcular la relevancia del elemento e en el multiconjunto M_i, se define la función EFN(e, M_i) como

$$\text{EFN}(e, M_i) = \text{rel}(e,\zeta) p_i$$

por lo que el índice de relevancia del elemento e en el multiconjunto M_i se mantiene entre $[0,1]$ y los elementos EFN(e, M_i) más cercanos a 1, son los elementos más relevantes en el multiconjunto M_i.

Ahora analicemos los siguientes casos:

1. El elemento e aparece casi con la misma frecuencia en todos los multiconjuntos. Esto significa que tendrá un histograma casi homogéneo, lo cual resultará en que $\frac{H(e,\zeta)}{\log(n)}$ sea muy cercano a 1. Por lo que rel(e,ζ) será casi 0 y al calcular EFN(e, M_i) para algún multiconjunto M_i, dicho factor se mantiene al multiplicar por p_i.

2. El elemento e aparece de manera heterogénea en todos los multiconjuntos. Esto significa que $\frac{H(e,\zeta)}{\log(n)}$ será más cercano a 0, lo cual hará que $\text{rel}(e,\zeta)$ sea más cercano a 1. Ahora, para determinar si e es relevante para M_i, hay dos opciones:

   a) El elemento e aparece frecuentemente en M_i, lo cual quiere decir que $\text{EFN}(e, M_i)$ será mayor a comparación de un elemento que no aparece frecuentemente. Entonces el producto $\text{rel}(e,\zeta)p_i$ seguirá siendo cercano a 1, lo cual le asigna un índice de relevancia cercano a 1 al elemento e en el multiconjunto M_i.

   b) El elemento e aparece no frecuentemente en M_i, por lo que $\text{EFN}(e, M_i)$ se acercará más a 0 al multiplicar por p_i. Esto quiere decir que el elemento e es importante a nivel global, por la cantidad de información que aporta, pero en particular para el multiconjunto M_i no lo es, ya que no lo describe.

Ya se tiene una medida de relevancia para elementos de un multiconjunto M_i. Para obtener los descriptores de M_i, únicamente debe calcularse $\text{EFN}(e, M_i)\ \forall e \in M_i$, y obtener los k elementos más relevantes, con k el número de descriptores que se necesita obtener.

En la siguiente sección, se muestra una aplicación de *EFN* para asignar índices de relevancia a palabras en conjuntos de documentos.

5.2. *EFN* para medir relevancia de palabras

El índice de relevancia *EFN* puede utilizarse para obtener descriptores robustos de conjuntos. Si consideramos al universo U como el vocabulario de algún idioma y a documentos en su modelo *bolsa de palabras* como multiconjuntos, donde la multiplicidad de una palabra es su frecuencia en el documento, podemos aplicar esta medida para obtener palabras relevantes en conjuntos de documentos.

Por ejemplo, veamos el histograma de la figura 5.1 que corresponde a la distribución de probabilidades de que la palabra "the" en un conjunto de documentos T que describen distintos animales, donde el texto T_1 es un artículo de Wikipedia acerca de elefantes.

Ahora veamos el histograma de la figura 5.2, que tiene la distribución de probabilidades de la palabra "elephant" en el mismo conjunto de documentos descrito anteriormente.

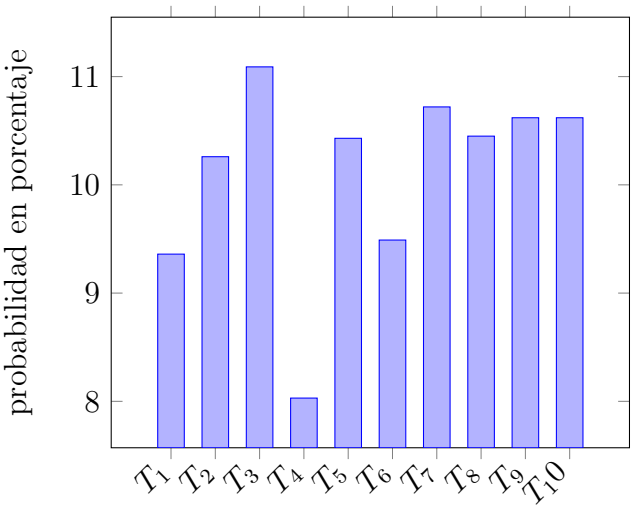

Figura 5.1: Histograma de la distribución de probabilidades de la palabra “the”

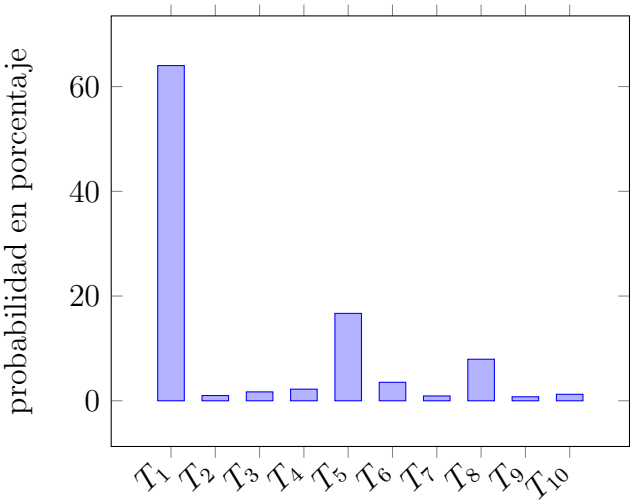

Figura 5.2: Histograma de la distribución de probabilidades de la palabra “elephant”

Del histograma de la figura 5.2 podemos notar dos cosas: que la entropía de la distribución de probabilidades de esa palabra será baja, dando como resultado que la relevancia global $rel(\text{elephant}, T)$ sea alta. Lo segundo que hay que notar, es que la palabra “elephant” es relevante para el documento T_1, dando como resultado que la entrada 1 de la distribución de probabilidades del conjunto, sea alta también. Por lo que al ser multiplicada por $rel(\text{elephant}, T)$, obtenemos un valor de relevancia alto.

Es importante notar un efecto interesante en esta medida. Si tenemos un conjunto de documentos que tratan acerca del mismo tema, entonces el vocabulario (el universo) será pequeño y palabras que sí son relevantes en el idioma podrían no serlo dentro de la colección.

Analicemos el siguiente conjunto de documentos. Este conjunto es acerca del mismo tema que es “estructuras de datos” en computación. Lo que va a pasar es que la palabra “computer” aparecerá homogéneamente en el conjunto de documentos lo que hará que la

palabra no sea tan relevante en el conjunto de documentos. El histograma de la figura 5.3 muestra la distribución de probabilidades de la palabra “computer” en un conjunto de 17 documentos que hablan acerca de estructuras de datos en computación.

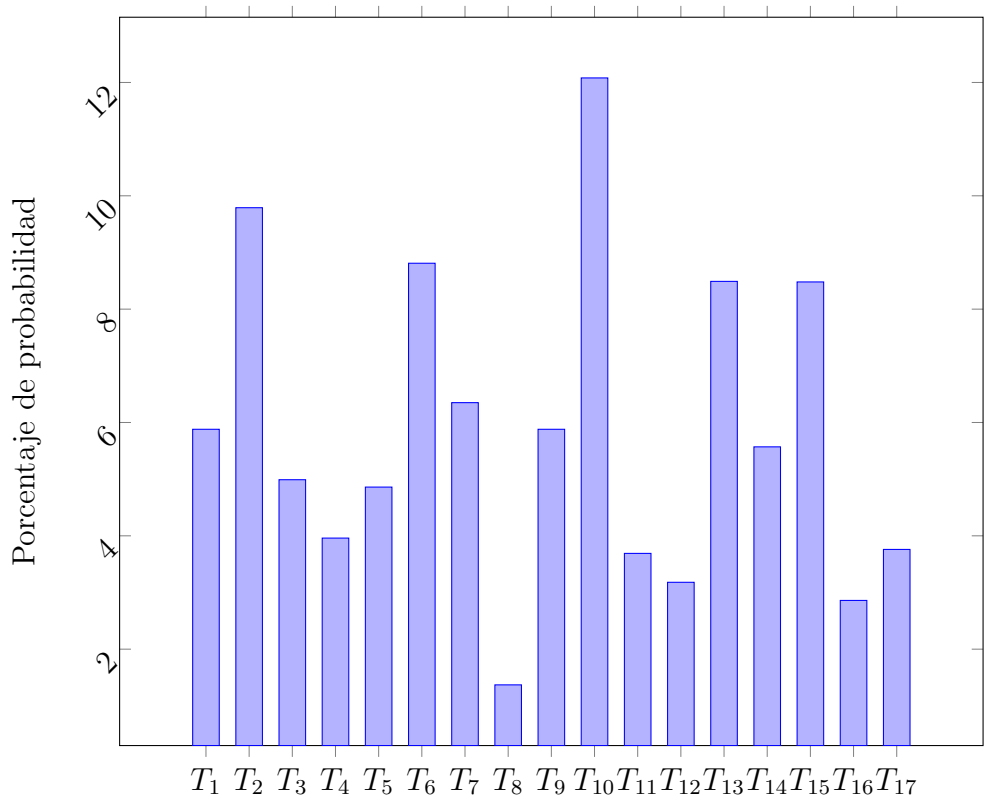

Figura 5.3: Histograma de la distribución de probabilidades de la palabra “computer” en un conjunto de documentos de “estructuras de datos”

Cómo podemos ver, la palabra “computer” no será relevante en el conjunto de documentos, ya que aparece homogéneamente en todos los documentos. De esto podemos concluir que la exactitud de *EFN* depende de qué tan amplio es el vocabulario y cómo se distribuye la palabra en el conjunto de documentos.

Particularmente la palabra “computer” en el idioma inglés es una palabra que aporta más información que palabras comunes como “in”, “the”, “up”, etc., pero en el ejemplo anterior, dado que el vocabulario de la colección anterior es pequeño y todos los documentos son acerca del mismo tema, “computer” no será relevante en el conjunto de documentos. Si tuviéramos un vocabulario más amplio y la palabra “computer” no apareciera de manera homogénea en todos los documentos, podría determinarse que “computer” es relevante para el idioma inglés.

Para determinar la relevancia de las palabras en el idioma, se puede construir un conjunto de documentos con un vocabulario amplio que sea representativo del idioma y que sea de distintos temas para que las palabras se distribuyan de manera heterogénea en el conjunto de documentos.

Si de alguna forma tuviéramos calculados los índices de relevancia de las palabras en un conjunto de documentos con las características descritas y quisiéramos calcular los

índices de relevancia de un documento nuevo, que no se encuentra en la colección, la solución intuitiva sería agregar el documento nuevo a la colección y volver a calcular los índices de relevancia de todas las palabras.

Siguiendo la misma idea pero para conjuntos de multiconjuntos, en la siguiente sección se propone una mejora algorítmica para el cálculo de los índices de relevancia de un multiconjunto nuevo. La mejora que se propone, además de calcular los índices del multiconjunto nuevo sin tener que actualizar los índices ya calculados, mejora la calidad de la selección de los descriptores.

5.3. Una mejora algorítmica para *EFN*

Lo que se propone con esta mejora, es que dado un conjunto de multiconjuntos, se pueda obtener índices de relevancia de elementos en un multiconjunto que no se encuentra en el conjunto original sin tener que volver a calcular todos los índices. Esta mejora también mejora la calidad de la selección de los descriptores.

Consideremos a un conjunto de multiconjuntos $\zeta = \{M_1, ..., M_n\}$ sobre un universo U, el cual tiene a los elementos relevantes del universo distribuidos de manera heterogénea en los multiconjuntos, al cual llamaremos *conjunto representativo del universo*. Sea $e \in U$, sea $nfvector(e, \zeta)$ el vector de frecuencias normalizadas del elemento e en el conjunto ζ, sea $dpvector(\text{nfvector}(e, \zeta))$ el vector de la distribución de probabilidades del vector de frecuencias normalizadas y sea $H(e, \zeta)$ la entropía del elemento e en el multiconjunto ζ.

Dado un multiconjunto $M_{n+1} \notin \zeta$, la entropía del elemento $e \in M_{n+1}$ puede calcularse sin tener que calcular de nuevo los vectores de frecuencias normalizadas y de distribución de probabilidades del elemento e en el conjunto ζ.

Tenemos que el vector de frecuencias normalizadas del elemento e en el conjunto ζ está definido como

$$\text{nfvector}(e, \zeta) = \left(\frac{\text{mul}(e, M_1) + 1}{|M_1|}, ..., \frac{\text{mul}(e, M_n) + 1}{|M_n|} \right) = (f_1, ..., f_n)$$

Sea $\zeta' = \zeta \cup \{M_{n+1}\}$, el vector de frecuencias normalizadas del elemento e en el conjunto ζ' se define como

$$\text{nfvector}(e, \zeta') = \left(\frac{\text{mul}(e, M_1) + 1}{|M_1|}, ..., \frac{\text{mul}(e, M_n) + 1}{|M_n|}, \frac{\text{mul}(e, M_{n+1})}{|M_{n+1}|} \right) = (f_1, ..., f_n, f_{n+1})$$

Tenemos que el vector de distribución de probabilidades del elemento e para el conjunto ζ se define como

$$\text{dpvector}(\text{nfvector}(e, \zeta)) = \left(\frac{f_1}{total}, ..., \frac{f_n}{total} \right) = (p_1, ..., p_n)$$

donde $total = \sum_{i=1}^{n} f_i$ y la entropía del elemento e en el multiconjunto ζ es

$$H(e, \zeta) = -\sum_{i=1}^{n} p_i \log(p_i)$$

El vector de distribución de probabilidades del elemento e en el conjunto ζ' puede ser obtenido a partir del vector de distribución de probabilidades del elemento e en el conjunto ζ de la siguiente manera

$$\begin{aligned}\text{dpvector}(\text{nfvector}(e,\zeta')) &= \left(\frac{f_1}{total+f_{n+1}},...,\frac{f_n}{total+f_{n+1}},\frac{f_{n+1}}{total+f_{n+1}}\right)\\ &= \left(\frac{p_1 total}{total+f_{n+1}},...,\frac{p_n total}{total+f_{n+1}},\frac{f_{n+1}}{total+f_{n+1}}\right)\\ &= (q_1,...,q_n,q_{n+1})\end{aligned}$$

Entonces tenemos que $q_i = p_i z$ con $z = \frac{total}{total+f_{n+1}}$ para todo $i = 1,...,n$. Ahora, la entropía del elemento e en el multiconjunto ζ' está definida como

$$\begin{aligned}H(e,\zeta') &= -\sum_{i=1}^{n+1} q_i \log(q_i)\\ &= -q_1\log(q_1) - ... - q_2\log(q_2) - q_{n+1}\log(q_{n+1})\\ &= -p_1 z\log(p_1 z) - ... - p_n z\log(p_n z) - q_{n+1}\log(q_{n+1})\\ &= -z(p_1\log(p_1 z) + ... + p_n\log(p_n z)) - q_{n+1}\log(q_{n+1})\\ &= -z(p_1\log(p_1) + p_1\log(z) + ... + p_n\log(p_n) + p_n\log(z)) - q_{n+1}\log(q_{n+1})\\ &= -z(\sum_{i=1}^{n} p_i\log(p_i) + \sum_{i=1}^{n} p_i\log(z)) - q_{n+1}\log(q_{n+1})\\ &= -z(-H(e,\zeta) + \log(z)\sum_{i=1}^{n} p_i) - q_{n+1}\log(q_{n+1})\end{aligned}$$

$\sum_{i=1}^{n} p_i$ es una distribución de probabilidad discreta, por lo que

$$\begin{aligned}&= -z(-H(e,\zeta) + \log(z)1) - q_{n+1}\log(q_{n+1})\\ &= zH(e,\zeta) - z\log(z) - q_{n+1}\log(q_{n+1})\end{aligned}$$

Por lo que para calcular la nueva entropía únicamente deben tenerse en cuenta los valores *total* y $H(e,\zeta)$ sin necesidad de volver a calcular el vector de distribución de probabilidades. La relevancia del elemento en el conjunto ζ' es

$$\text{rel}(e,\zeta') = 1 - \frac{H(e,\zeta)}{\log(n+1)}$$

y finalmente, *EFN* para el elemento e en el multiconjunto M_{n+1} es

$$EFN(e,M_{n+1}) = \text{rel}(e,\zeta')q_{n+1}$$

De esta manera, se puede establecer un conjunto sobre un universo representativo que ayudará a discriminar de mejor manera elementos que sí sean relevantes de elementos

que no lo son. Es importante notar que los valores de relevancia de los elementos en el conjunto ζ no cambiarán. El cálculo de la nueva entropía se realiza de manera virtual para obtener el índice *EFN*. De esta forma se logra que la relevancia de los elementos en el universo no se altere con la llegada de multiconjuntos que podrían tener una estructura en la que algún elemento, que es importante en el universo, aparezca constantemente en dichos multiconjuntos y haga que su histograma se vea homogéneo.

Particularmente, en el ejemplo de los documentos, esta mejora se puede lograr creando un conjunto representativo del universo de documentos que contenga un vocabulario muy variado, documentos de longitud larga y con las palabras distribuidas de manera heterogénea. Algunos resultados se muestran en el capítulo 7.

6

Clasificación mediante encajes en espacios vectoriales

El trabajo que se describe en las siguientes secciones, es una forma de utilizar los encajes vectoriales de cualquier objeto para clasificar multiconjuntos bajo un conjunto de categorías de forma no supervisada; la única condición es que el encaje vectorial cumpla con ciertas propiedades. La clasificación se lleva a cabo de forma no supervisada ya que no se da un modelo a partir de un conjunto de entrenamiento, lo único que se da es un conjunto de categorías para etiquetar a los multiconjuntos.

En el capítulo 4 se habló acerca de representar las palabras de un vocabulario como vectores. Pero estos vectores no sólo eran una representación de las palabras en el espacio vectorial, también tienen la característica de que palabras parecidas semánticamente aparecen cerca en el espacio vectorial. Con esta representación, pueden derivarse clases de palabras, encontrar palabras parecidas a una palabra dada, etc. Incluso documentos completos pueden ser representados cómo un sólo vector con fines de clasificación.

Con base en lo anterior, se busca realizar una clasificación parecida a la de los documentos en la que se etiquetan documentos con categorías. La diferencia es que se hace de manera general. Es decir, la clasificación funciona no sólo para documentos, sino para cualquier tipo de objeto que cumpla las propiedades que se describen más adelante.

Supongamos que podemos encajar cualquier tipo de elemento de un conjunto U en un espacio vectorial y que el objetivo del encaje es capturar algún tipo de similitud entre los elementos. Es decir, si dos elementos $u, v \in U$ están relacionados bajo alguna medida de similitud, entonces los vectores de u y v deben aparecer cerca en el espacio vectorial.

Lo que se describe a continuación, es la definición de la medida de similitud y qué condiciones debe cumplir el encaje vectorial de los objetos para poder llevar a cabo las clasificaciones que se describen en las siguientes secciones.

Definición 15. *Sea U un conjunto de elementos. Una función de similitud* $\mathrm{sim} : U \times U \to [0, 1]$ *tiene las siguientes propiedades:*

1. $\forall u, v \in U$ *tenemos que* $\mathrm{sim}(u, v) = 1$ *si y sólo si* $u = v$
2. $\forall u, v \in U$ *tenemos que* $\mathrm{sim}(u, v) = \mathrm{sim}(v, u)$
3. *Sean* $u, v, w \in U$*, si* $\mathrm{sim}(u, v) \sim 1$ *y* $\mathrm{sim}(v, w) \sim 1$ *entonces* $\mathrm{sim}(u, w) \sim 1$

Ahora definamos qué es un *buen* encaje vectorial del conjunto U.

Definición 16. *Un encaje de elementos en el espacio vectorial es* bueno *si y sólo si dados los elementos u, v, $w \in U$ y la medida de similitud* sim*, tales que* $\text{sim}(u, v) < \text{sim}(u, w)$ *casi siempre se cumple que* $\text{dist}(\vec{u}, \vec{v}) > \text{dist}(\vec{u}, \vec{w})$.

Un *buen* encaje manda elementos con similitud alta a vectores con distancia pequeña y viceversa. Notemos que no necesariamente se cumple para toda terna de elementos que si $\text{sim}(u, v) < \text{sim}(u, w)$ entonces $\text{dist}(\vec{u}, \vec{v}) > \text{dist}(\vec{u}, \vec{w})$, es decir, estamos suponiendo que para una pequeña cantidad finita de pares no se cumple esta propiedad.

Para las clasificaciones que se verán en las siguientes secciones, definimos lo siguiente:

- U un conjunto base de elementos.
- La función $\text{vec} : U \to \mathbb{S}^d$ que encaja a los elementos de U en la esfera unitaria $\mathbb{S}^d$ de dimensión d, tal que $\forall x \in U$ tenemos que $|\,\text{vec}(x)| = 1$ y además vec tiene la propiedad de ser un *buen* encaje. Denotemos a $\text{vec}(x)$ como $\vec{x}$.
- $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ un conjunto de multiconjuntos tal que $\forall e \in M_i$ tenemos que $e \in U$.
- $R_i = \{r_1, r_2, ..., r_m\}$ el conjunto de los m más elementos relevantes de M_i obtenidos con la medida *EFN*, descrita en el capítulo 5 y $\text{rel}_\text{i}(r_j)$ el índice de relevancia del elemento r_j respecto al multiconjunto M_i.

Partiendo de lo anterior, se desarrollaron dos clasificaciones de multiconjuntos que se explican en la siguientes secciones.

6.1. Clasificación plana (ranking)

El objetivo de esta clasificación es clasificar al multiconjunto M_i respecto a un conjunto de categorías $C = \{c_1, c_2, ..., c_p\}$ tal que $c_i \in U$. Las categorías no tienen ninguna relación entre sí y lo que se quiere es ordenar a las categorías de forma que las primeras categorías etiqueten a M_i y las últimas categorías sean las que menos se relacionan con M_i.

Esta clasificación funciona a través de "votos" que los elementos relevantes de M_i brindan a las categorías de C. La idea intuitiva es que si una categoría recibe muchos votos entonces M_i pertenece a esa categoría.

Definamos el *voto ponderado* que un elemento relevante r brinda a la categoría c

$$\text{voto\_ponderado}(r, c) = (\vec{c} \cdot \vec{r})\, \text{rel}_\text{i}(r)$$

con $\text{rel}_\text{i}(r_j)$ el índice de relevancia del elemento r_j respecto al multiconjunto M_i. Ahora definamos el *voto* que brinda el multiconjunto M_i a la categoría c

$$\text{voto}(M_i, c) = \sum_{r \in R_i} \text{voto\_ponderado}(r, c)$$

Ya que estamos suponiendo el uso de un *buen* encaje, el criterio para medir qué tan similares son la categoría c y el elemento relevante r, es la distancia entre $\vec{c}$ y $\vec{r}$, que se define como

$$\text{dist}(\vec{c}, \vec{r}) = |\vec{c} - \vec{r}|$$

y como estamos suponiendo que U está encajado en $\mathbb{S}^d$, tenemos que $\forall \vec{v} \in \mathbb{S}^d$ se cumple que $|\vec{v}| = 1$.

Consideremos la medida similitud coseno, que mide el coseno del ángulo entre dos vectores. Tenemos que como todos los vectores están en la esfera unitaria, entonces se cumple $\forall\ \vec{u}, \vec{v}$ y $\vec{w} \in \mathbb{S}^d$ que si $\text{dist}(\vec{u}, \vec{v}) < \text{dist}(\vec{u}, \vec{w})$ entonces $\cos(\vec{u}, \vec{v}) > \cos(\vec{u}, \vec{w})$. De lo anterior podemos decir entonces que si $\text{sim}(u, v) < \text{sim}(u, w)$ entonces $\cos(\vec{u}, \vec{v}) < \cos(\vec{u}, \vec{w})$, por lo que en lugar de comparar distancias euclidianas podemos comparar similitudes coseno.

Ya que los vectores están normalizados, se tiene que $|\vec{c}| = |\vec{r}| = 1$, por lo que

$$\cos(\vec{c}, \vec{r}) = \frac{\vec{c} \cdot \vec{r}}{|\vec{c}||\vec{r}|} = \frac{\vec{c} \cdot \vec{r}}{1 \cdot 1} = \vec{c} \cdot \vec{r}$$

por ello, en el *voto ponderado* se calcula el producto punto entre $\vec{c}$ y $\vec{r}$.

Ya que la función coseno está acotada, tenemos que $-1 \leq \vec{c} \cdot \vec{r} \leq 1$, es decir, el producto punto de los vectores unitarios está acotado y además, vectores cercanos tendrán un producto punto cercano a 1.

Consideremos a la categoría que tiene el *voto* mayor, digamos c_l. La categoría c_l es la que etiquetará al multiconjunto M_i. Ahora veamos por qué esta categoría es a la que corresponde el multiconjunto.

Recordemos que se está suponiendo *fuertemente* que el encaje vectorial es *bueno*, es decir, que casi toda pareja de elementos cumple que si son similares entonces la distancia entre sus encajes es pequeña y suponemos también que el conjunto de elementos relevantes R_i describe al multiconjunto M_i.

Supongamos que una categoría c es más similar al elemento relevante r que alguna otra categoría c'. Es decir, $\text{sim}(c, r) > \text{sim}(c', r)$. Ya que estamos suponiendo el uso de un encaje *bueno*, la similitud coseno entre los vectores $\vec{c'}$ y $\vec{r}$ será menor que la similitud coseno entre $\vec{c}$ y el $\vec{r}$. Al multiplicar por la relevancia, la relación se mantiene. Es decir, si tenemos

$$\cos(\vec{c'}, \vec{r}) < \cos(\vec{c}, \vec{r})$$

al multiplicar por la relevancia del elemento se mantiene que

$$\cos(\vec{c'}, \vec{r})\, \text{rel}_\text{i}(r) < \cos(\vec{c}, \vec{r})\, \text{rel}_\text{i}(r)$$

Por otro lado, si se tienen dos elementos relevantes r y r' tales que r es más relevante que r', es decir $\text{rel}_\text{i}(r') < \text{rel}_\text{i}(r)$ y $\text{sim}(r', c) < \text{sim}(r, c)$ entonces

$$\cos(\vec{c}, \vec{r'})\, \text{rel}_\text{i}(r') < \cos(\vec{c}, \vec{r})\, \text{rel}_\text{i}(r)$$

lo cual quiere decir que un elemento menos similar a la categoría, aunque sea multiplicado por su relevancia seguirá siendo menor que un elemento con más relevancia y más similar a la categoría. De lo anterior se puede decir que los elementos más relevantes tienen más peso en su voto.

Lo importante del *voto ponderado* es que el voto que brindan los elementos relevantes a las categorías, está pesado por el índice de relevancia de los elementos relevantes. Si

tenemos un elemento relevante r con índice de relevancia bajo y resulta ser muy similar a una categoría c que no es similar a los demás elementos con más relevancia, es decir $\text{sim}(r_j, c) < \text{sim}(r, c)\ \forall r_j \in R_i \setminus \{r\}$, dicho voto no afecta en el ranking, ya que r no es tan relevante y su voto no tiene tanto peso.

6.2. Clasificación jerárquica (taxonómica)

Para la clasificación descrita en la sección anterior, suponemos que las categorías no tienen ninguna relación entre sí. Ahora suponemos que el objetivo es realizar una clasificación en una jerarquía de categorías, es decir, la clasificación irá desde lo más general hasta lo más particular.

La jerarquía de categorías se puede representar con un árbol, donde para cualquier nodo padre, la categoría general es él mismo y sus hijos son subcategorías de la categoría general que además están relacionadas entre sí.

Lo que se busca en esta clasificación, es dar peso a los nodos del árbol de categorías para encontrar un camino que lleve de la categoría más general a la más particular, tratando de encontrar el nodo en el árbol que mejor describa al multiconjunto. Una vez encontrado ese nodo, la clasificación jerárquica del multiconjunto es el camino desde ese nodo hacía las hojas, siguiendo los nodos con mayor peso.

La idea es que si muchas subcategorías de hijas de una categoría de la jerarquía reciben muchos votos, entonces la categoría padre también recibirá un voto alto.

Para esta clasificación también suponemos el conjunto $\mathcal{M}$ de multiconjuntos, M_i el multiconjunto a clasificarse, R_i el conjunto de elementos relevantes de M_i y $\text{rel}_\text{i}(r_j)$ el índice de relevancia del elemento r_j respecto al multiconjunto M_i.

Sea τ un árbol de categorías, tal que $V(\tau) = \{v_1, v_2, ..., v_p\}$ es el conjunto de todos los nodos del árbol y $v_i \in U$, es decir, $V(\tau)$ es un conjunto de categorías. El peso de la raíz de un subárbol de τ que corresponde a una categoría v_i, de define como el promedio de la suma de los pesos de los vértices del subárbol.

Definamos la función tam_subarbol que devuelve el tamaño del subárbol de cualquier nodo incluido el mismo

$$\text{tam\_subarbol}(v) = \begin{cases} 1 & \text{si } v \text{ es una hoja} \\ \sum\limits_{u \in \text{hijos}(v)} \text{tam\_subarbol(u)} + 1 & \text{si } v \text{ es un nodo interior} \end{cases}$$

y el peso de cada nodo $v \in V$ se define como

$$\text{peso}(v) = \begin{cases} \text{voto}(M_i, v) & \text{si } v \text{ es una hoja} \\ \dfrac{\text{voto}(M_i, v) + \sum\limits_{u \in \text{hijos}(v)} \text{peso(u)}}{\text{tam\_subarbol}(v)} & \text{si } v \text{ es un nodo interior} \end{cases}$$

Notemos que el peso de las hojas es únicamente el voto que las palabras relevantes le dieron a la categoría, lo cual podría verse como una clasificación plana: la categoría con el mayor número de votos es a la que pertenece M_i.

Cuando el nodo v no es una hoja, se suman los pesos de los nodos del subárbol y se divide entre el número de nodos que tiene el subárbol, esto se hace para que el peso de los nodos que no son hojas esté normalizado para evitar la siguiente situación: supongamos que se tiene a v un nodo interno padre de muchas categorías, las cuales no están tan relacionadas con los elementos de R_i y se tiene a otro nodo interno u, cuyos hijos están muy relacionados con los elementos de R_i pero son menos categorías que los hijos de v.

Si sólo se sumaran los pesos de los hijos de los nodos internos, aunque las categorías de los hijos de v no sean tan similares a los elementos de R_i probablemente la suma sea mayor que la de los hijos de u que sí son similares a los elementos de R_i ya que son más categorías, es por eso que debe normalizarse la suma. De esta forma, el nodo interno con el mayor peso, será la categoría que mejor describa al multiconjunto M_i.

Sea c_{max} el nodo con mayor peso en τ y $P_1 = \text{root}, c_1, .., c_{max}$ el camino desde la raíz hasta c_{max}. Sea $P_2 = c_{max}, c'_1, ..., c'_l$ el camino de nodos desde c_{max} hasta alguna hoja, tales que c'_i es el nodo con mayor peso cuyo padre es c'_{i-1}. Es decir, c'_1 es el hijo con mayor peso de c_{max}, c'_2 es el hijo con mayor peso de c'_1, etc. y c'_l es una hoja. La clasificación jerárquica del multiconjunto M_i es el camino $P = P_1 \cup P_2$.

6.3. Clasificación plana como agrupamiento

En el capítulo 3 se habló acerca de la clasificación de documentos de manera no supervisada, que es mejor conocida como agrupamiento. La idea era representar a los documentos como vectores y seleccionar a los vectores de algunos documentos, de forma que formaran *centroides* y después, dependiendo el tipo de agrupamiento que se quisiera hacer, empezar a formar grupos respecto a su cercanía con centroides.

Ahora, podemos pensar en un agrupamiento para multiconjuntos. El problema podría traducirse a agrupar al multiconjunto al grupo mas cercano, donde los grupos tienen como centroide al vector que representa a la categoría. Para lograr esto, el multiconjunto debería estar representado como un vector también.

Cabe mencionar que el algoritmo de agrupamiento presentado en esta sección no es estrictamente de "agrupamiento". La diferencia es que aquí no se trata de agrupar al conjunto de multiconjuntos a diferencia de los algoritmos presentados en la sección 3.3.2. También debe notarse que esta forma de plantear el problema no es más que eso, un planteamiento. No es diferente a la clasificación plana propuesta, sólo otra forma de verlo y el resultado final es el mismo.

Para la agrupación también suponemos el conjunto $\mathcal{M}$ de multiconjuntos, M_i el multiconjunto a agruparse, R_i el conjunto de elementos relevantes de M_i, $C = \{c_1, c_2, ..., c_p\}$ el conjunto de categorías y $\text{rel}_\text{i}(r_j)$ el índice de relevancia del elemento r_j respecto al multiconjunto M_i.

La representación en un vector del multiconjunto M_i se define cómo

$$\vec{M_i} = \sum_{r \in R_i} \vec{r} \cdot \text{rel}_\text{i}(r)$$

que se define como el centroide del grupo al que pertenece originalmente el multiconjunto M_i. El criterio para medir la similitud entre el grupo de la categoría y el grupo del multiconjunto M_i, es el producto punto de sus vectores, es decir

$$\text{similitud}(\vec{c}, \vec{M_i}) = \vec{c} \cdot \vec{M_i}$$

El multiconjunto M_i pertenece al grupo de la categoría c_j cuya similitud sea mayor

$$\text{clasificacion}(M_i, C) = max\{\text{similitud}(\vec{c_j}, \vec{M_i}) \forall c_j \in C\}$$

Teorema 1. *$\forall c \in C$ se cumple que*

$$\text{similitud}(\vec{c}, \vec{M_i}) = \text{voto}(M_i, c)$$

Demostración. Tenemos que el vector de M_i está definido con base en sus elementos relevantes y la relevancia de estos, es decir

$$\vec{M_i} = \text{rel}_\text{i}(r_1) \cdot \vec{r_1} + ... + \text{rel}_\text{i}(r_m) \cdot \vec{r_m}$$

Sea $c \in C$ una categoría, entonces la similitud está definida como

$$\begin{aligned}
\text{similitud}(\vec{c}, \vec{M_i}) &= \vec{c} \cdot \vec{M_i} \\
&= \vec{c} \cdot (\text{rel}_\text{i}(r_1) \cdot \vec{r_1} + ... + \text{rel}_\text{i}(r_m) \cdot \vec{r_m}) \\
&= \vec{c} \cdot \text{rel}_\text{i}(r_1) \cdot \vec{r_1} + ... + \vec{c} \cdot \text{rel}_\text{i}(r_m) \cdot \vec{r_1} \\
&= \text{rel}_\text{i}(r_1)(\vec{c} \cdot \vec{r_1}) + ... + \text{rel}_\text{i}(r_m)(\vec{c} \cdot \vec{r_m}) \\
&= \text{voto_ponderado}(r_1, c) + ... + \text{voto_ponderado}(r_m, c) \\
&= \text{voto}(M_i, c)
\end{aligned}$$

□

En la sección anterior se justificó por qué la votación funciona. De la misma manera podemos concluir que la similitud definida aquí, funciona de la misma forma.

6.4. Aplicaciones de clasificación plana

La clasificación plana puede utilizarse para clasificar cualquier objeto que pueda ser encajado vectorialmente y que cumpla con la condición de ser un *buen encaje*. Por ejemplo, imaginemos que estamos en alguna ciudad T y tenemos a todos los supermercados que están ahí y nos gustaría saber en qué se especializan, es decir, supongamos que el supermercado m_1 se especializa en vender carne, ropa y medicinas, el supermercado m_2 se especializa en vender ropa y electrónicos, etc.

Con el inventario que el supermercado tiene, puede hacerse una clasificación. Primero, se tendría que definir la relación de similitud para poder realizar el encaje de los productos que se venden. Digamos que dos artículos son similares si corresponden a la misma categoría, es decir, "jamón" y "queso" son más similares más que "jamón" y "playera", por ejemplo.

Una vez encajados los productos, el inventario del supermercado puede ser visto como un multiconjunto de artículos, donde la multiplicidad del artículo es la cantidad de existencias con las que cuenta el supermercado. De esta forma, obtenemos con la medida EFN los artículos más relevantes para el supermercado y las categorías se definen como el tipo de artículos que se venden en el supermercado, es decir $C = \{$*cárnicos, electrónicos, ropa, medicinas*$\}$.

Otro de los problemas que pueden resolverse con la clasificación propuesta, es la clasificación de documentos. Contamos con el encaje vectorial de word2vec, que fue visto en el capítulo 4, que cumple con las propiedades de ser un buen encaje: dos palabras w y v similares semánticamente aparecen cerca en el espacio vectorial. El algoritmo se muestra en la siguiente sección.

6.5. Aplicación de la clasificación plana a conjuntos de documentos

Sea $T = \{D_1, ..., D_n\}$ un *conjunto representativo del universo* de documentos, es decir, un conjunto con un vocabulario amplio y con documentos de longitud relativamente grande y $C = \{c_1, ..., c_p\}$ un conjunto de categorías bajo las cuales se quiere clasificar a algún documento. Los documentos están definidos sobre algún idioma, el cuál será nuestro universo U y $\forall w \in U$, $vec(w)$ es el encaje vectorial obtenido con el esquema algorítmico word2vec. Para clasificar al documento D_{n+1} se hace lo siguiente:

1. Obtener con la medida *EFN* las k palabras más relevantes del documento D_{n+1}. Llamemos $R_{n+1} = \{r_1, ..., r_k\}$ al conjunto de palabras más relevantes del documento D_{n+1} y $\text{rel}_{n+1}(r_j)$ el índice de relevancia de la palabra r_j.

2. Calcular $voto(D_{n+1}, c)$ $\forall c \in C$.

3. Etiquetar al documento D_{n+1} con la categoría que tenga mayor voto.

La elección de la k es un parámetro libre. Experimentalmente se ha visto que elegir entre 15 y 20 palabras da un mejor resultado. Algunos ejemplos se muestran en el capítulo 8.

Para medir la calidad de la clasificación, podemos considerar que cada documento D_i está etiquetado manualmente con m_i categorías. Digamos que las etiquetas del documento D_i son $C_{D_i} = \{c_1^i, ..., c_{m_i}^i\}$ y que el conjunto de categorías C se define como

$$C = \bigcup_{i=1}^{n} C_{D_i}$$

La prueba consiste en realizar una clasificación plana (ranking) sobre C. Al realizar la clasificación plana sobre el documento D_i con el conjunto de categorías C, obtenemos a las categorías ordenadas por voto de mayor a menor, por lo que obtenemos una permutación de C. Llamemos C'_{D_i} a esa permutación. Lo que nos interesa es obtener las primeras m_i entradas de C'_{D_i} sin importar el orden, para poder compararlas con C_{D_i}.

La exactitud de la clasificación se mide tomando la intersección de las primeras m_i entradas de C'_{D_i} con C_{D_i}. Si la intersección resulta en C_{D_i} entonces decimos que la clasificación fue exacta.

Recordemos que nuestra relación de similitud entre los elementos del universo es la similitud semántica de las palabras y, por lo visto en el capítulo 4, podemos concluir que el encaje que produce word2vec es un *buen encaje*.

También notemos que se pide un *conjunto representativo del universo* de documentos para que la selección de descriptores (palabras relevantes) sea de mejor calidad, ya que cada palabra seleccionada debe describir correctamente al documento para que a la hora de medir la similitud semántica de las palabras por medio de la votación, el resultado sea correcto.

En el capítulo 8 se muestran algunos resultados de la clasificación plana en conjuntos de documentos.

IV

Caso de uso de las aportaciones

7

Obtención de palabras relevantes con Entropía (de frecuencias) Normalizada

En este capítulo se muestran algunos resultados de la obtención de elementos relevantes con el índice de relevancia *EFN* (definido en el capítulo 5) aplicado a conjuntos de documentos.

Los resultados que se muestran a continuación, fueron probados en varios tipos de conjuntos de documentos, desde colecciones de artículos obtenidos de Wikipedia[1] hasta colecciones de libros electrónicos obtenidos vía la biblioteca electrónica "Proyecto Gutenberg"[2]. Los artículos y libros están en inglés porque es más fácil su procesamiento y porque la mayoría de los libros electrónicos en texto plano están en ese idioma.

Las primera sección de pruebas consiste en comparar la selección de palabras con el índice *EFN* contra la selección del índice *TF-IDF*. Las segunda sección consiste en obtener las 15 palabras más relevantes de libros o artículos en colecciones de longitud variada. En algunos casos se consideró un conjunto representativo del universo de documentos. En todos los cuadros las palabras se muestran ordenadas por índice de relevancia decreciente.

Por su naturaleza subjetiva, no es factible tener una medida de calidad formal de relevancia de palabras, lo cual tiene que evaluarse de manera subjetiva por personas. Para la evaluación de ambas pruebas se le pidió a un grupo de 5 personas que juzgara si la selección de palabras era adecuada.

Para los resultados de la primera prueba tenían que seleccionar de entre los conjuntos de palabras producidos por *EFN* y *TF-IDF* cuál describía mejor al documento y si el orden de las palabras por relevancia era correcto. En todos los casos se determinó que la selección realizada por *EFN* era mejor. Para los resultados de la segunda prueba, tenían que determinar si la selección de palabras era adecuada y en todos los casos la respuesta fue positiva.

[1] `https://en.wikipedia.org/`

[2] `https://www.gutenberg.org/`

7.1. *EFN* vs *TF-IDF*

Para la primera prueba se utilizó un conjunto de 19 artículos de Wikipedia acerca de estructuras de datos en computación, es decir, arreglos, pilas, colas, árboles, etc. El artículo de menor longitud cuenta con 776 palabras, el de mayor longitud tiene 6,432 palabras y el vocabulario tiene 4,294 palabras. También se probó tomando como conjunto representativo del universo una colección de 73 libros del "proyecto Gutenberg", donde el libro con mayor longitud tiene 248,341 palabras, el libro con menor longitud tiene 3,729 palabras y el vocabulario tiene 80,759 palabras.

La tabla del cuadro 7.1 muestra la selección de las 15 palabras más relevantes ordenadas por relevancia decreciente. Estas palabras fueron seleccionadas con *TF-IDF* y con *ENF* con y sin conjunto representativo del universo.

Artículo	**Palabras relevantes TF-IDF**	**Palabras relevantes EFN**	**Palabras relevantes EFN conjunto representativo**
Array	dimensional	dimensional	array
	row	row	arrays
	column	indices	dimensional
	arrays	address	index
	vector	array	indices
	address	arrays	data
	indices	column	vector
	dimension	elements	row
	element	vector	stored
	elements	index	address
	formula	element	linear
	major	major	element
	dope	dimension	dimension
	matrices	consecutive	column
	array	addressing	major

Computer Science	engineering study computation software computers computational scientific disciplines theory systems svg discipline academic calculator field	computer science software engineering computation study computational theory computers systems computing design field fields humans	computer software computational computation computing engineering computers data science svg algorithms programming systems disciplines digital
List	lists append monad monoid cons list lisp nil constructor monadic dialects brackets monomorphic abstract lua	lists list abstract lisp append nil type programming monad monoid cons linked types languages singly	list lists data linked programming lisp arrays nil abstract implemented append array monoid implementations monad

Cuadro 7.1: Comparación entre *TF-IDF* y *EFN* en una colección de 19 artículos de Wikipedia

En el caso del artículo "Array", la selección de palabras realizada por *TF-IDF* no tiene en los primeros lugares a las palabras "array" o "arrays". En cambio '*EFN* sí. Esto se debe a que *TF-IDF* no mide la relevancia de la palabra en en el conjunto de documentos, únicamente mide si aparece o no en los demás documentos del conjunto. Al ser un conjunto de documentos del mismo tema, "array" aparece muchas veces, lo cual produce que la función *idf* le de un peso pequeño.

Ahora veamos la selección de palabras de los mismos artículos en una colección que cuenta con 50 artículos de Wikipedia acerca de varios tópicos como animales, fenómenos naturales, biología, matemáticas y tiene contenida a la colección de 19 artículos de computación. En la colección de 50 artículos, el artículo con menor longitud tiene 252 palabras, el que tiene mayor longitud tiene 11,778 palabras y el vocabulario es de 13,731 palabras. Los resultados se muestran en la tabla del cuadro 7.2.

Artículo	**Palabras relevantes TF-IDF**	**Palabras relevantes EFN**	**Palabras relevantes EFN conjunto representativo**
Array	array dimensional row arrays elements indices address index element column vector dimension memory addressing dope	array dimensional arrays elements indices row element index address vector column memory data dimension linear	array arrays dimensional index indices data vector row stored address linear element dimension column major
Computer Science	computer engineering science computation software computational computers computing disciplines design svg systems academic calculator theory	computer science software theory computation engineering computational computing computers systems design formal study information disciplines	computer software computational computation computing engineering computers data science svg algorithms programming systems disciplines digital

List	list	list	list
	lists	lists	lists
	append	abstract	data
	lisp	type	linked
	abstract	linked	programming
	programming	lisp	lisp
	nil	programming	arrays
	monad	operation	nil
	monoid	data	abstract
	data	nil	implemented
	cons	languages	append
	languages	arrays	array
	arrays	append	monoid
	type	elements	implementations
	operation	value	monad

Cuadro 7.2: Comparación entre *TF-IDF* y *EFN* en una colección de 19 artículos de Wikipedia

Como se puede observar ahora, que la colección ahora es heterogénea, es decir, de distintos temas, la función *TF-IDF* cambió de manera positiva la selección de sus palabras al igual que *EFN*. Los únicos resultados que se mantuvieron igual fueron los de *EFN* con el conjunto representativo del universo.

Para el caso de *TF-IDF*, aunque la selección mejoró, nuestros sujetos de prueba estuvieron de acuerdo con que tanto la selección como el orden con el que fueron seleccionadas las palabras, sigue sin ser mejor que la de *EFN* con el conjunto representativo del universo.

7.2. Selección de palabras con *EFN*

Para esta primera prueba no se consideró un conjunto representativo del universo. En el cuadro 7.3 se muestra el resultado de la obtención de las palabras más relevantes de dos libros en una colección de 10 libros y en una colección de 73 libros. En la colección de 10 libros, el libro con mayor longitud cuenta con 113,161 palabras, el libro más corto cuenta con 9,804 y el tamaño del vocabulario es de 23,519 palabras. En la colección de 73 libros el libro con mayor longitud tiene 248,341 palabras, el libro con menor longitud tiene 3,729 palabras y el vocabulario tiene 80,759 palabras.

Notemos que los resultados en ambas columnas son similares, casi una permutación en el orden. Esto se debe a que a pesar de que la colección de 10 libros es pequeña, el vocabulario de la colección es amplia y la longitud de los libros también. En la tabla 7.4 se muestran más resultados sobre la colección de 73 libros.

Libro	Palabras relevantes colección de 10	Palabras relevantes colección de 73
"Dreams" de Henri Bergson	memories dreams dream sensations waking dreamer memory sensation images visual dreaming perceptions consciousness observer slumber	dream memories dreams waking sensations dreamer memory sleep sensation dreaming images visual observer effort perceptions
"Town Geology" de Charles Kingsley	coal slate beds limestone rocks slates shells sandstone pebbles mud lime coral geology snowdon clay	coal beds mud coral limestone slate clay ice rocks pebbles slates geology sandstone shells lime

Cuadro 7.3: 15 palabras relevantes en colecciones de distintas longitudes

Libro	Palabras relevantes	Libro	Palabras relevantes
"The Story of alchemy and the beginnings of chemistry" de M. M. Pattison Muir	alchemists alchemical substances metals alchemy lavoisier mercury phlogiston radio alchemist substance properties metal compounds atoms	"A history of witchcraft in England from 1558 to 1718" de Wallace Notestein	witchcraft witches witch ibid elizabeth accused acquitted middlesex trials pamphlet london hopkins stearne hanged confessions

Cuadro 7.4: 15 palabras relevantes en colección de 73 libros del "Proyecto Gutenberg"

Ahora, veamos tres colecciones más. La primera colección cuenta con 19 artículos de Wikipedia acerca de estructuras de datos, la misma colección que en la primera prueba de la sección anterior. El artículo de menor longitud cuenta con 776 palabras, el de mayor longitud tiene 6,432 palabras y el vocabulario tiene 4,294 palabras. La segunda colección cuenta con 50 artículos de Wikipedia acerca de varios tópicos como animales, fenómenos naturales, biología, matemáticas y tiene contenida a la colección de 19 artículos de computación. En la colección de 50 artículos, el artículo con menor longitud tiene 252 palabras, el que tiene mayor longitud tiene 11,778 palabras y el vocabulario es de 13,731 palabras.

La tercera colección tiene como conjunto representativo del universo la colección de 73 libros del "proyecto Gutenberg". Primero se calculó la relevancia del vocabulario de 80,759 de la colección de 73 libros y después se obtuvo la relevancia de las palabras de los artículos de estructuras de datos. Las palabras relevantes se muestran en el cuadro 7.5.

Artículo	Palabras relevantes colección de 19	Palabras relevantes colección de 50	Palabras relevantes con conjunto representativo
Array	dimensional row indices address array arrays column elements vector index element major dimension consecutive addressing	array dimensional arrays elements indices row element index address vector column memory data dimension linear	array arrays dimensional index indices data vector row stored address linear element dimension column major
Binary Tree	key binary left subtree right search tree root min node we child priority parent trees	binary tree key node search right root subtree trees left we child value priority parent	node binary key search tree subtree nodes root insertion traversal items delete trees parent data

Cuadro 7.5: 15 palabras relevantes en colecciones de distintas longitudes

Notemos que en el caso particular del artículo “Array”, en la colección de 19 artículos “array” no aparecía en primer lugar. Esto se debe a que la presencia de la palabra “array” en el conjunto de artículos de estructuras de datos es muy común, pero al hacer más grande la colección y el vocabulario entonces la palabra se vuelve más relevante en la colección y por lo tanto en el documento también. Aún así, lo que se esperaría es que “array” y “arrays” fueran las dos palabras más relevantes, y aunque el vocabulario de la colección de 50 es más grande, no se logra ese resultado.

En cambio, cuando utilizamos el conjunto representativo del universo de documentos con un vocabulario bastante amplio, “array” y “arrays” son las palabras más relevantes en el artículo de “Array”.

Ahora analicemos una colección interesante. La colección original "Reuters" cuenta 21,578 documentos y consiste en artículos de noticias publicados en el año 1987. Usualmente la colección se separa en dos conjuntos: uno de entrenamiento y uno de pruebas. Para la tabla que se muestra en el cuadro 7.6 se utilizó un subconjunto del conjunto de pruebas que cuenta con 2745 artículos. La mayoría de los artículos cuenta con 100 palabras en promedio y el tamaño del vocabulario es de 14,622 palabras. En la tabla también se muestra la selección de palabras utilizando el conjunto representativo del universo del "Proyecto Gutenberg".

Artículo 1 colección Reuters	**Artículo 1 conjunto representativo**	**Artículo 2 colección Reuters**	**Artículo 2 conjunto representativo**
exchange	exchange	trade	japan
rubber	rubber	japan	trade
trading	trading	exports	dlrs
coffee	coffee	to	exports
are	nainggolan	of	tariffs
was	palm	the	billion
nainggolan	tonnes	tariffs	imports
palm	rupiah	japanese	hong
start	saleh	billion	businessmen
he	mln	imports	electronics
in	fob	said	taiwan
five	traded	in	japans
up	commodity	electronics	kong
trade	officials	hong	semiconductors
oil	traders	businessmen	japanese
Artículo 3 colección Reuters	**Artículo 3 conjunto representativo**	**Artículo 4 colección Reuters**	**Artículo 4 conjunto representativo**
wmc	wmc	they	nsw
mine	pct	nsw	ports
about	tonnes	ports	shipping
will	mln	port	port
gold	goodall	shipping	disrupted
be	dlrs	pay	cargo
it	kms	the	disruption
pct	bundey	disruption	ban
at	reuter	disrupted	handling
is	wmng	movements	today
by	leach	handling	sources
mt	corp	ban	dispute
leach	holdings	dispute	reuter
goodall	gra	cargo	containers
bundey	adelaide	action	kembla

Cuadro 7.6: 15 palabras relevantes de distintos artículos de la colección Reuters

Podemos ver que si sólo se usa la colección "Reuters" como base, la selección de palabras no es acertada. Es decir, las palabras relevantes no son descriptores robustos de los textos ya que tenemos a palabras "on", "in", "is", "up", etc. que son palabras comunes en inglés y que no aportan mucha información acerca del texto. Esto sucede por dos cosas: el tamaño del vocabulario es pequeño con respecto a de la colección y la longitud de los artículos es pequeña también, lo que resulta en que el vocabulario de los artículos sea muy pequeño. Por ejemplo, la colección de 73 libros del "Proyecto Gutenberg" tenía un

vocabulario de 80,759 palabras, lo cual es casi 6 veces más grande que el vocabulario de "Reuters" y al ser utilizada como conjunto representativo del universo, la selección de palabras mejora dramáticamente.

La tabla que se muestra en el cuadro 7.7 contiene un cuadro comparativo de las características de las colecciones utilizadas en este capítulo para análisis.

Colección	**Tema colección**	**Número de documentos**	**Documento con mayor longitud**	**Documento con menor longitud**	**Tamaño del vocabulario**
Proyecto Gutenberg	Varios	10	113,161	9,804	23,344
Proyecto Gutenberg	Varios	73	248,341	3,729	80,759
Wikipedia	Estructuras de datos	19	6,432	776	4,294
Wikipedia	Varios	50	11,778	252	13,731
Reuters	Noticias mundiales	2,745	1,115	7	14,622

Cuadro 7.7: Resumen de las características de las colecciones

De los análisis anteriores podemos concluir dos cosas:

1. Las palabras obtenidas con *EFN* en colecciones de documentos con un vocabulario chico con respecto al tamaño de la colección y documentos de longitud corta (que llevan a tener un vocabulario pequeño por documento) no son tan relevantes, es decir, no son palabras que ayuden a distinguir al documento, como en la colección "Reuters" sin un conjunto representativo del universo.

2. Sin importar la longitud de la colección, si el vocabulario es grande y la longitud de los documentos (y por lo tanto la del vocabulario también), la selección de palabras relevantes es más atinada, es decir, no obtenemos palabras muy frecuentes en el conjunto de documentos pero sí palabras que ocurren frecuentemente en el documento que ayudan a describirlo.

3. El uso de un conjunto representativo del universo de documentos, mejora no la selección de palabras relevantes, también mejora el índice de relevancia que le es asignada a la palabra. Ejemplo: la palabra "array" en el artículo "Array" de Wikipedia.

De las pruebas anteriores, podemos concluir que el índice de relevancia *EFN* selecciona descriptores robustos para los documentos y cuando se cuenta con un conjunto representativo del universo de documentos, la calidad mejora mucho.

Por otro lado, cuando se realizaron las pruebas, el programa que utilizaba el conjunto representativo del universo era mucho más rápido. Esto es porque se preprocesaba la entropía de las palabras del vocabulario. Para obtener la relevancia de las palabras de los nuevos documentos, únicamente se calculaba la nueva entropía sin tener que volver a calcular de nuevo la distribución de probabilidad y la entropía del nuevo conjunto.

La mejora propuesta, además de mejorar la calidad de las palabras, mejoró notablemente el desempeño del programa que calculaba *EFN*.

8

Clasificación de documentos

En este capítulo se muestran los resultados de las clasificaciones propuestas en el capítulo 6 aplicadas a conjuntos de documentos.

Recordemos que para poder llevar a cabo la clasificación es necesario contar con una relación de similitud, un *buen encaje*, una medida de relevancia y un conjunto de multiconjuntos. Nuestro conjunto de multiconjuntos serán documentos, por lo que los elementos del universo serán palabras de algún vocabulario y nuestra relación de similitud es la similitud semántica entre palabras. Finalmente, nuestro *buen encaje* es el que produce word2vec y la medida de relevancia es *EFN*.

El conjunto de documentos está en inglés ya que el encaje de word2vec que se utilizó en estas pruebas, es un conjunto de vectores entrenado en un conjunto de documentos de "Google News" en inglés con al rededor de 100 millones de millones de palabras [21] y que cuenta con un vocabulario de 575,433 palabras distintas. Todas estas palabras son atómicas, es decir, no son palabras compuestas ya que por lo visto en la sección 4.3, el entrenamiento para frases es ligeramente distinto.

Para los resultados que se muestran a continuación se utilizaron 3 colecciones de documentos. La primera colección cuenta con 59 libros de distintas áreas: biología, historia, botánica, geografía, música, literatura, etc. Los libros fueron obtenidos vía el "Proyecto Gutenberg"[1]. El proyecto es una biblioteca electrónica con una cantidad considerable de libros electrónicos para consulta. Al igual que en las bibliotecas físicas, los libros del "Proyecto Gutenberg" están acomodados por categorías para su consulta. La colección que se clasificó fue etiquetada manualmente con las categorías que tenían en el "Proyecto Gutenberg". La mayoría de los documentos cuenta con 3 etiquetas en promedio y el total de categorías distintas es 67. En estas pruebas fueron utilizadas 15 palabras para describir al documento.

[1] https://www.gutenberg.org/

Libro	**Categorías originales**	**Categorías votación en orden descendiente**	**Porcentaje de coincidencia**
"The history of alchemy and the beginnings of chemistry" de M. M. Pattison Muir	[alchemy, chemistry]	alchemy chemistry	100 %
"The Declaration of Independence of The United States of America"	[history, america, political]	law political ethics	33.3 %
"The Age of Reason"de Thomas Paine	[philosophy, psychology, religion, atheism]	christian religion atheism poetry	50 %
"BEETHOVEN: A Character Study together with Wagner's Indebtedness to Beethoven" de George Alexander Fischer	[music]	music	100 %
"Dreams" de Henri Bergson	[philosophy, psychology, psychoanalysis]	fairytale romance folklore	0 %

Cuadro 8.1: Algunos resultados de la clasificación plana

Porcentaje de coincidencia	**Número de documentos**	**Porcentaje de coincidencia**	**Número de documentos**
100 %	18	33 %	8
75 %	4	25 %	1
66.6 %	7	20 %	1
60 %	1	0 %	3
50 %	16		

Cuadro 8.2: Porcentajes de coincidencia de todos los documentos

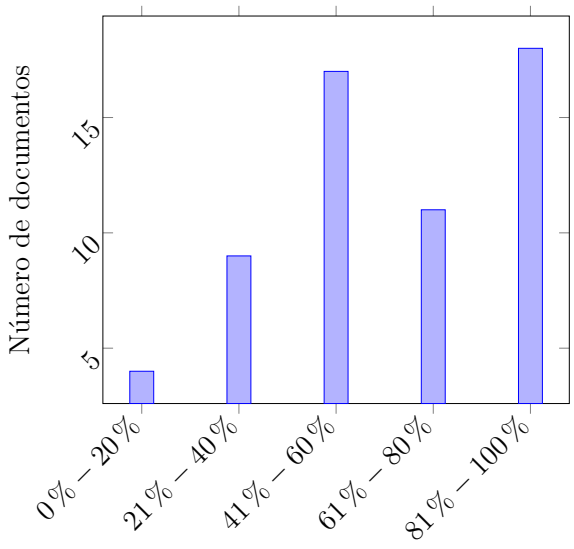

Figura 8.1: Histograma de porcentajes de coincidencia de la colección "Proyecto Gutenberg"

Como podemos ver, más del 50 % de los documentos tiene una clasificación por arriba del 50 % de coincidencia con las etiquetas originales.

En el caso particular del libro "Dreams", sucede que las palabras relevantes (tabla 7.3) describen mejor a las categorías "fairytale", "romance" y "folklore" que a las categorías que tenía asignadas manualmente. Esto sucede por el conjunto de datos con el que fue entrenado word2vec.

Ahora, analicemos los resultados de otra colección. Esta colección cuenta con 50 artículos de Wikipedia[2] de distintos temas como naturaleza, biología, computación, animales extintos, etc. Para esta clasificación, se le pidió a un grupo de personas que etiquetaran a los documentos con el conjunto de tópicos principales de Wikipedia[3] y sus subcategorías.

La tabla que se muestra en el cuadro 8.4, muestra los porcentajes de coincidencia de la clasificación de la colección de 50 artículos de Wikipedia. Para esos resultados, fueron usadas 10 palabras como descriptores. Para los resultados de la tabla del cuadro 8.5 fueron usadas 20 palabras como descriptores por cada documento sobre el mismo conjunto.

[2] `https://en.wikipedia.org/`

[3] `https://en.wikipedia.org/wiki/Category:Main_topic_classifications`

Artículo	**Categorías originales**	**Votación 10 descriptores**	**Votación 20 descriptores**	**Porcentaje de coincidencia**
Array	[science, mathematics, computer, logic]	number computer geometry technology	computer logic technology geometry	25 % / 50 %
Queue	[number, computer, mathematics, logic]	statistics logic technology number	logic number computer technology	50 % / 75 %
Lion	[biology, land, nature, africa, planet, animal, earth]	statistics nature fire oceania prehistory africa animal	biology land nature oceania prehistory africa animal	42 % / 71 %
Wild Boar	[biology, nature, africa, animal, earth]	water sea land prehistory animal	biology sea land prehistory animal	20 % / 40 %

Cuadro 8.3: Algunos resultados de la clasificación plana

Porcentaje de coincidencia	**Total de documentos**
81 % - 100 %	4
61 % - 80 %	17
41 % - 60 %	22
21 % - 40 %	6
0 % - 20 %	1

Cuadro 8.4: Clasificación de 50 artículos de Wikipedia con 10 palabras como descriptores

Porcentaje de coincidencia	**Total de documentos**
81 % - 100 %	7
61 % - 80 %	19
41 % - 60 %	20
21 % - 40 %	4
0 % - 20 %	0

Cuadro 8.5: Clasificación de 50 artículos de Wikipedia con 20 palabras como descriptores

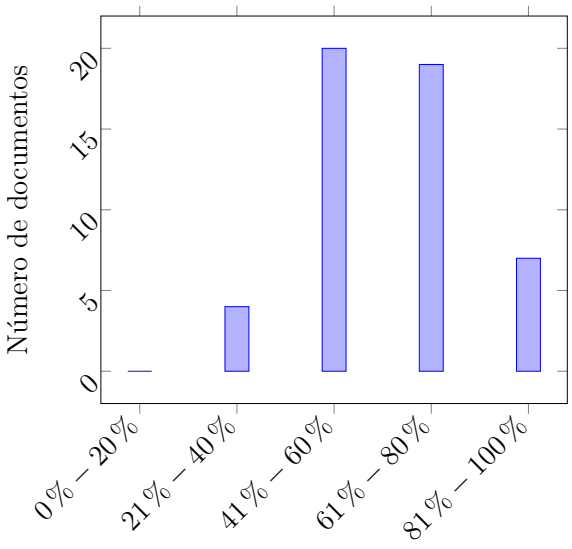

Figura 8.3: Histograma de porcentajes de coincidencia de la colección Wikipedia con 20 descriptores

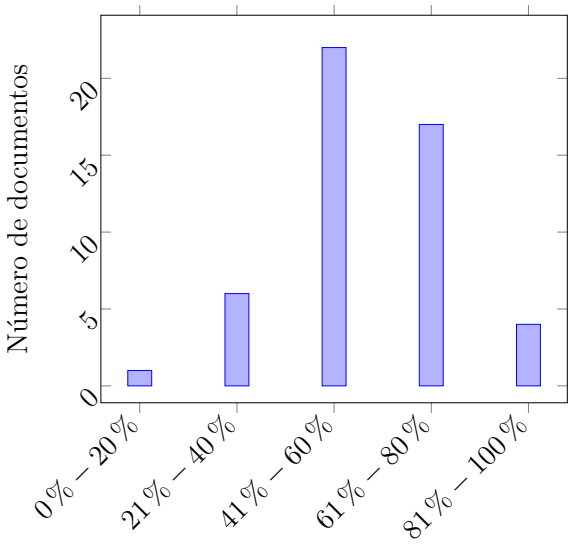

Figura 8.2: Histograma de porcentajes de coincidencia de la colección Wikipedia con 10 descriptores

Como podemos ver, la selección de más descriptores mejoró la calidad de la votación. Recordemos que cada documento cuenta con un vocabulario finito. No pueden escogerse todas las palabras como descriptores porque tenemos palabras que no describen al documento y que además, podrían afectar la votación. Al realizar las pruebas, se observó que el número óptimo de descriptores es entre 15 y 30.

Por último, veamos la colección "Reuters". Esta colección, como se dijo en el capítulo anterior, cuenta con 21,578 documentos. "Reuters" es la colección con la que se mide la efectividad de las clasificaciones de documentos. Consiste en dos conjuntos de documentos, un conjunto de entrenamiento y un conjunto de pruebas y cada uno de los documentos

del conjunto está etiquetado con categorías.

Uno de los artículos se muestra a continuación:

Chase Corp Ltd ¡CHCA.WE¿said it will make an offer for all fully-paid shares and options of ¡Entregrowth International Ltd¿it does not already own. Chase, a property investment firm, said it holds 48 pct of Entregrowth, its vehicle for expansion in North America. It said agreements are being concluded to give it a beneficial 72.4 pct interest. The offer for the remaining shares is one Chase share for every three Entregrowth shares and one Chase option for every four Entregrowth options. Chase shares closed on Friday at 4.41 dlrs and the options at 2.38. Entregrowth closed at 1.35 dlrs and options at 55 cents. Chase said the offer for the remaining 27.6 pct of Entregrowth, worth 34.2 mln dlrs, involved the issue of 5.80 mln Chase shares and 3.10 mln Chase options. Chase chairman Colin Reynolds said the takeover would allow Entregrowth to concentrate on North American operations with access to Chase's international funding base and a stronger executive team. He said there also would be benefits from integrating New Zealand investment activities. Chase said the offer is conditional it receiving accptances for at least 90 pct of the shares and options. REUTER

Existen varias razones por las cuales no podemos utilizar "Reuters" para dar un punto de referencia para esta clasificación a pesar de que es la colección más usada para medir la calidad de clasificaciones. Las razones son:

1. Muchas categorías con las que están etiquetados los artículos son nombres compuestos, por ejemplo "palm-oil", "money-supply", "soy-oil". El conjunto de vectores que utilizamos no contempla palabras compuestas

2. La longitud de los artículos de la colección es pequeña, el promedio es de 100 palabras por artículo, lo cual produce un vocabulario pequeño que no ayuda a distinguir bien descriptores del documento.

3. Los artículos cuentan con muchas siglas y abreviaciones, por ejemplo "wmc", "pct", "dlr","mln", "nsw", lo cual no está encajado en el espacio. Por lo que cuando este tipo de palabras son seleccionadas como descriptores, no aportan ningún tipo de información acerca del artículo.

8.1. Clasificación de documentos jerárquica

Al igual que la clasificación plana, la clasificación jerárquica también puede ser utilizada para clasificar documentos. Se tiene que construir un árbol de categorías para esta clasificación.

Se propone el árbol de la categoría "Science". Según Wikipedia, la ciencia se divide en 4 ramas principales: "natural", "formal", "social" y "aplicada". Cada una de estas categorías también cuenta con sus ramas, por ejemplo "formal" tiene como subramas a "logica", "matemáticas", "estadística" y "computación". En la figura 8.4 se muestra el árbol de categorías.

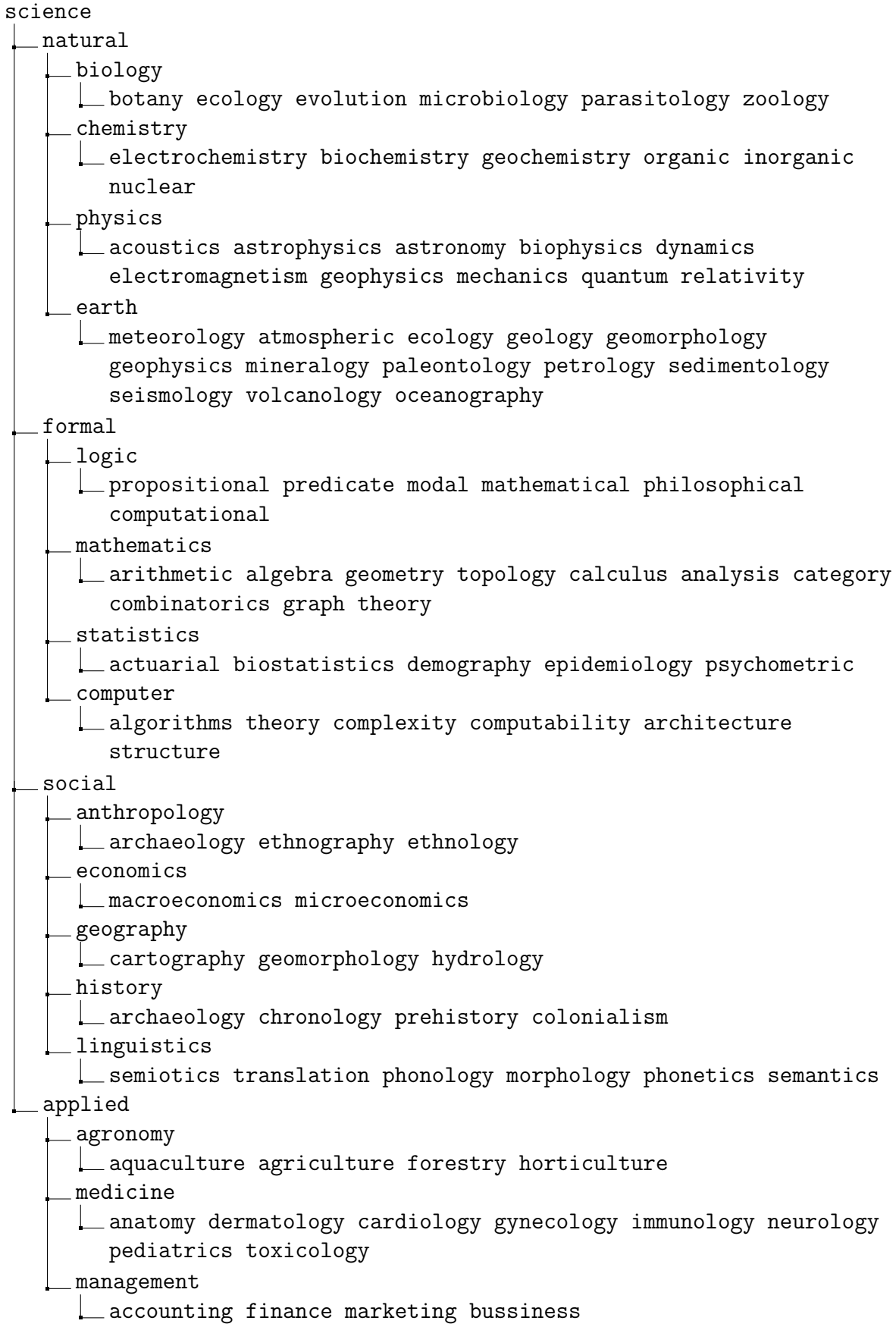

Figura 8.4: Árbol jerárquico de categorías

En el cuadro 8.6 se encuentran los resultados de la clasificación jerárquica de algunos artículos de Wikipedia. La trayectoria que se presenta, consiste en tomar el camino desde la raíz hasta el nodo que mayor peso tuvo. A partir de ese nodo, se busca a los hijos que más peso tuvieron hasta llegar a las hojas.

Artículo	**Nodo más pesado**	**Trayectoria desde la raíz**
Earthquake	Earth	science natural **earth** geophysics
Ring (Algebra)	Mathematics	science formal **mathematics** algebra
Cycle (Graph Theory)	Mathematics	science formal **mathematics** graph
Magma	Earth	science natural **earth** mineralogy
Thylacine	Biology	science natural **biology** zoology

Cuadro 8.6: Resultados de la clasificación jerárquica

El nodo con más peso es la raíz del subárbol que mejor describe al documento. La trayectoria completa produce un etiquetamiento jerárquico para el documento.

V

Conclusiones

9

Conclusiones

El objetivo de este trabajo fue proponer una clasificación que funcionara para cualquier tipo de objeto en un universo. La propiedad que deben de cumplir los objetos del universo es que se pueda definir una relación de similitud y esta relación pueda encajarse en un espacio.

La clasificación propuesta en este trabajo, es una generalización del problema de clasificación de documentos, la cual se puede hacer de manera supervisada o no supervisada. La clasificación se lleva a cabo sobre conjuntos de multiconjuntos y un conjunto de categorías sobre el cual se quiere realizar la clasificación.

La clasificación de algún multiconjunto consiste en tomar elementos destacados estadísticamente, es decir, elementos que puedan ayudar a describir al multiconjunto con base en su frecuencia a los cuales les llamamos descriptores. Los descriptores realizan una votación sobre el conjunto de categorías y el peso de su voto se basa también en la relevancia que tienen en el multiconjunto. Es decir, elementos que tienen relevancia alta tienen un voto más pesado que elementos con menor relevancia.

Se pensó en multiconjuntos porque son estructuras a las cuales se les puede asociar un histograma, donde la frecuencia de un elemento es su multiplicidad en el multiconjunto. Los multiconjuntos son una abstracción del modelo *bolsa de palabras* para representar documentos en la que únicamente se almacenan tuplas de la forma *(palabra,frecuencia)*.

La obtención de descriptores puede realizarse de muchas maneras. En este trabajo también se propuso un índice de relevancia de elementos en multiconjuntos para obtener los descriptores. También se mostró que con un pequeño cálculo se pueden obtener índices de relevancia de elementos a partir de conjuntos base que cumplieran ciertas características.

Para la realización de las pruebas de obtención de descriptores en conjuntos de documentos, es decir, palabras relevantes, el cálculo de la entropía en un conjunto representativo del universo significó una mejora en la calidad de la selección de las palabras y también significó una mejora en tiempo en la ejecución del programa.

La diferencia radica en que ya no se tiene que calcular la entropía de todo el alfabeto cada vez que se ejecuta el programa. En cambio, preprocesar el cálculo de la entropía de todas las palabras del vocabulario y sólo almacenar un par de valores más, fueron una mejora notable en el tiempo que el programa tardaba en ejecutarse. Podría pensarse que de alguna forma, la máquina está aprendiendo algo acerca del vocabulario, pero en realidad no. Únicamente está preprocesando el vocabulario y obteniendo estadísticas con base a multiconjuntos robustos, con un vocabulario amplio y cada vez que llegan documentos

nuevos, no se actualizan los valores anteriores.

A pesar de que ya se cuenta con algunas medidas de relevancia como *TF-IDF*, la medida *EFN* demostró ser un buen discriminador de elementos relevantes. Como trabajo futuro, podría probarse también con algún tipo de multiconjunto que no sean documentos para observar los resultados.

El éxito de la clasificación, además de depender de la buena selección de descriptores, también depende de la calidad del encaje. En particular, el encaje que proporciona word2vec es de los más populares y con vocabulario más grande que se encuentra en la literatura. El único inconveniente con wor2vec es que aunque explica muy bien cómo es que encaja las palabras y qué hace para preservar la similitud semántica entre sus vectores, no explica cómo es que logra capturar relaciones semánticas.

La captura de relaciones semánticas de wor2vec marcó tendencia en los nuevos encajes, es decir, cuando se trata de crear un nuevo encaje se tiene en mente que para que este sea exitoso, debe capturar relaciones semánticas también, las cuales pueden (deben) ser extraídas con una operación entre vectores.

Con las pruebas de clasificación realizadas, puede verse que no en todos los casos la clasificación coincide con el 100 % de las etiquetas propuestas. Esto se debe en parte al conjunto de entrenamiento y al tipo de entrenamiento que fue usado. En algunos casos, una etiqueta compuesta hubiera descrito mejor al documento y algunas palabras son de algún campo especializado. Por ejemplo, nosotros asociamos las palabras “renglón” y “columna” a las palabras matrices o arreglos, pero el encaje encontraba que “arquitectura” es la palabra más cercana. Pero esto lo sabemos porque en nuestro campo lo asociaríamos a eso, pero un arquitecto por ejemplo, lo asociaría a “arquitectura”.

La calidad del encaje que utilizamos es buena, pero para poder tener una clasificación de mayor calidad, probablemente el conjunto de entrenamiento de word2vec deba contener a los documentos que se quiere clasificar, o al menos documentos que estén en el mismo contexto.

Bibliografía

[1] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[2] Geza Schay, *Introduction to probability with statistical applications* Springer Science and Business Media, 2007.

[3] Robert M. Gray, *Entropy and Information Theory*, Springer-Verlag, 2013.

[4] Blizard, Wayne D. *Multiset theory*, Notre Dame J. Formal Logic 30, 1988.

[5] *Deep Learning Tutorial*, URL: `http://ufldl.stanford.edu/tutorial/`

[6] Daniel Jurafsky and James H. Martin, *Speech and Language Processing*, Prentice Hall, Draft of August 24, 2015.

[7] Dan Gusfield, *Algorithms on strings, trees and sequences: Computer Science and Computational Biology*, Cambridge University Press, 2nd edition, 1997.

[8] Carlos Gershenson, *Artificial neural networks for beginners*, arXiv preprint cs/0308031, 2003.

[9] Quoc Le and Tomas Mikolov, *Distributed Representations of Sentences and Documents*, ICML. Vol. 14, 2014.

[10] Stephen Robertson, *Understanding inverse document frequency: on theoretical arguments for IDF*, Journal of documentation, 2004.

[11] Stuart Russell and Peter Norving, *Artificial Intelligence A Modern Approach*, Pearson, 3rd edition, 2010.

[12] Kalita and Deepshikha, *Supervised and Unsupervised Document Classification-A survey*, International Journal of Computer Science and Information Technologies, 2015.

[13] Patrick Trinkle, *An introduction to unsupervised document classification*, 2009.

[14] Daniel Jurafsky and James H. Martin, *Speech and Language Processing*, Prentice Hall, Draft of September 28, 1999.

[15] Jeffrey Pennington, Richard Socher and Christopher D. Manning. *GloVe: Global Vectors for Word Representation*, 2014.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient estimation of word representations in vector space*, CoRR, abs/1301.3781, 2013.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. *Distributed representations of words and phrases and their compositionality*, In Advances in Neural Information Processing Systems, 2013.

[18] Xin Rong, *word2vec parameter learning explained*, arXiv preprint arXiv:1411.2738, 2014.

[19] Zellig Harris. *Distributional structure*, Word, 10(23):146–162, 1954.

[20] Omer Levy, Yoav Goldberg. *Neural word embedding as implicit matrix factorization*, In Advances in Neural Information Processing Systems, 2014.

[21] word2vec *Tool for computing continuous distributed representations of words*, URL: `https://code.google.com/archive/p/word2vec/`

[22] David Dolan Lewis, *Reuters-21578 test collection*, URL: `http://www.daviddlewis.com/`