UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

EXTRACCIÓN DE TÉRMINOS CATEGORIZADOS A TRAVÉS DE UN SERVICIO WEB

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

CRISTIÁN FELIPE SERPELL CARRIQUIRY

PROFESOR GUÍA:
CARLOS ALBERTO HURTADO LARRAÍN

MIEMBROS DE LA COMISIÓN:
GONZALO NAVARRO BADINO
RODRIGO ANDRÉS PAREDES MORALEDA

SANTIAGO DE CHILE
ENERO 2008

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERIO CIVIL EN COMPUTACIÓN
POR: CRISTIÁN SERPELL C.
FECHA: 14/01/2008
PROF. GUÍA: CARLOS HURTADO L.

"EXTRACCIÓN DE TÉRMINOS CATEGORIZADOS A TRAVÉS DE UN SERVICIO WEB"

El tema del acceso a la información es de especial interés dado el contexto actual de la web 2.0, donde se generan grandes volúmenes de información por parte de muchos usuarios. Es deseable categorizar y priorizar la información disponible para mejorar el acceso a ella. Para esto se han creado variadas interfaces basadas en términos que la caracterizan.

El trabajo consistió en el estudio de herramientas computacionales para extraer términos o palabras clave categorizados a partir de un texto en español, el diseño y construcción de un sistema que realice dicha acción y finalmente el diseño de una aplicación tipo servicio web que sirva para construir una interfaz que facilite el acceso a un conjunto de documentos a partir de los términos extraídos.

Para la realización de este trabajo, el alumno estudió herramientas existentes de minería de texto y extracción de información, de acuerdo a los requerimientos del sistema que extrae términos categorizados. Esto incluye distintos modelos y algoritmos estudiados a nivel teórico, además de implementaciones de los algoritmos elegidos. Se eligió el modelo de campos aleatorios condicionales como la base de la extracción de términos, y se estudiaron las distintas características del texto relevantes para dicho modelo.

Gracias a una serie de experimentos, se concluyó que la aplicación tipo servicio web propuesta cumple con los objetivos de generar una interfaz útil para el acceso a distintas colecciones de documentos, categorizados según nombres de personas. Se comprobó que la aplicación resulta efectiva incluso para colecciones de contenido profundamente distinto. Además, es suficientemente extensible como para ser aplicada en el futuro a otro tipo de categorías, como lugares, instituciones u otra, permitiendo tener más dimensiones para explorar los documentos.

Agradecimientos

En primer lugar, querría agradecer al profesor guía, señor Carlos Hurtado, por la dedicación con la que me ha apoyado durante la realización de este trabajo.

Agradezco a Fondecyt, que financió el presente trabajo a través del Proyecto Número 1050642 de Procesamiento y Análisis Semántico de Servicios Web.

Agradezco al profesor Rodrigo Paredes, por la gran cantidad de consejos constructivos que me otorgó para la completación de este trabajo.

Quisiera expresar mi agradecimiento a todos mis compañeros del Departamento de Ciencias de la Computación, por su apoyo y ayuda incondicionales que logran hacer de este un mejor trabajo.

Finalmente, agradezco a mi familia y a todos quienes me han ayudado a cumplir mis objetivos, junto con apoyarme cuando he decidido plantearme metas más altas.

Tabla de Contenidos

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

1.1. Tags y nubes de tags

Actualmente, gracias a Internet, existe un fácil acceso a grandes cantidades de información sobre prácticamente cualquier tema, generada por una gran cantidad de fuentes distintas, en idiomas diferentes. En este contexto, se han generado muchas aplicaciones que extraen y procesan automáticamente información de Internet, contenida en ciertos formatos estándares para compartir información.

Algunas de las tareas fundamentales son clasificar, catalogar y organizar tal cantidad de información. Un método cada vez más común para esto es asignar tags a la información. Los tags son términos relevantes asociados o asignados a un ítem de información, que lo describen y lo permiten clasificar. Éstos son usualmente elegidos por el autor o por alguien que ha revisado el ítem. Los tags son típicamente usados en la generación automática, flexible y dinámica de taxonomías para recursos en línea, tales como archivos, páginas web, imágenes y bookmarks. Generalmente, un ítem puede tener uno o más tags asociados. Actualmente los tags no sólo son generados por usuarios sino que son extraídos automáticamente por procedimientos basados en técnicas de extracción de información. Algunos ejemplos de páginas web que utilizan tags son: del.icio.us, YouTube, flickr, Gmail. En la Figura 1.1 se puede ver cómo los tags son parte de la interfaz que tiene el usuario para acceder a la información en estos sitios.

Usualmente los tags se agrupan en estructuras llamadas nubes de tags que han probado ser interfaces efectivas para acceder a la información. Al visualizar todos los tags, se muestran ordenados alfabéticamente, enfatizados con un tamaño de letra más grande o de otro modo los más frecuentes. De esta manera es posible encontrar un tag según nombre o según popularidad. Al seleccionar un tag de la nube de tags se muestra una colección de ítemes asociados con ese tag. Hay páginas web que utilizan nubes de tags dentro de su interfaz, por ejemplo: del.icio.us, last.fm, Ma.gnolia, technorati. En la Figura 1.2 se ve un ejemplo de nube de tags.

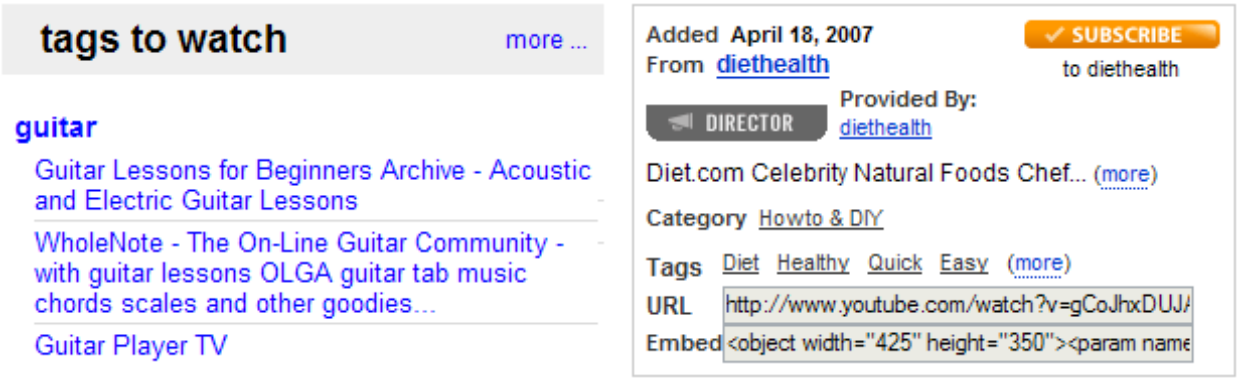

Figura 1.1: Uso de tags en interfaz del.icio.us y YouTube

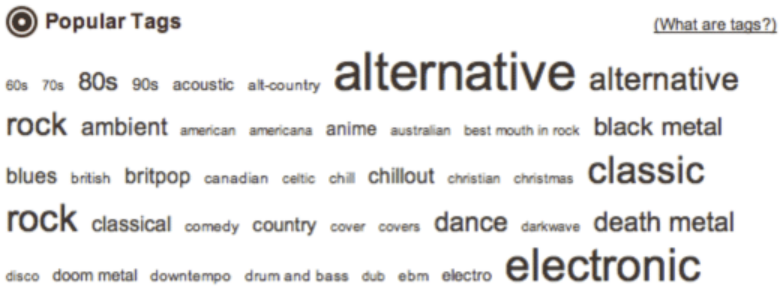

Figura 1.2: Nube de tags del sitio last.fm. Cada tag es un estilo musical

Este trabajo se centra particularmente en el análisis de tags asociados a documentos en la web. Se aplican técnicas de áreas como Extracción de Información, Clasificación de Textos, Minería de Textos para mejorar la organización y acceso a una colección de documentos en base a tags y nubes de tags.

1.2. Motivación

Un problema esencial de las nubes de tags es su falta de estructura. Los tags son términos relevantes de distintos tipos que mezclan dimensiones distintas de la información. Como ejemplo, tags que identifican personas, ciudades o países representan distintas dimensiones y éstas no se distinguen en las interfaces basadas en nubes de tags. En la Figura 1.3 se visualiza este problema.

Una estructura adicional podría ser muy útil para mejorar el acceso y organización de la información. Con una interfaz adecuada se podría acceder a una nube de tags asociada a una clase particular de una taxonomía de conceptos. Por ejemplo, se podría obtener una nube

Figura 1.3: Tags asociados a noticias en el sitio orbitando.com

de tags asociados a personajes públicos, otra a lugares, otra a instituciones, etc. Algunos de estos tags pueden no aparecer en la nube principal por tener baja tasa de aparición entre los documentos. En las Figuras 1.4 y 1.5 se muestran ejemplos de posibles interfaces que cumplen este propósito.

Figura 1.4: Mostrar distintas nubes según categoría seleccionada

Figura 1.5: Mostrar todos los tags juntos con distintos colores para cada tipo

Esto permitiría responder preguntas tales como: ¿cuáles son las noticias que tienen que ver con universidades?, ¿cuáles son las noticias donde hay cierto tipo de instituciones involucradas?, o ¿cuáles son las noticias donde hay políticos de derecha involucrados?

Para lograr la clasificación de tags, se requiere un análisis más profundo de qué son los tags, cómo se extraen o generan y el contexto que los rodea (por ejemplo, el texto que rodea la aparición de un tag en documento). Actualmente existe una gran variedad de herramientas de minería de textos que pueden ayudar a resolver el problema con distintos enfoques.

1.3. Servicios web

En el contexto actual, existe una tendencia a construir aplicaciones en la web que funcionan como servicio web. Esto significa que la aplicación puede utilizarse a través de Internet, e incluir su funcionalidad en la interfaz de otro sitio. De esta manera un sitio web puede incluir en su interfaz un elemento que es proveído por un servicio web, y no necesita instalar ni configurar la aplicación dentro de su servidor. Un ejemplo conocido de esto es el servicio provisto por Google Maps.

Es interesante abordar el problema planteado desde esta perspectiva, ya que son muchos los sitios que podrían necesitar proveer una interfaz con tags categorizados, pudiendo todos utilizar una aplicación tipo servicio web.

1.4. Objetivos

1.4.1. Objetivo general

El objetivo general del trabajo es el diseño, construcción y prueba de un sistema que extrae nombres de personas presentes en un conjunto de documentos y los procesa de manera que permitan construir una nube de tags útil para su comprensión y exploración.

1.4.2. Objetivos específicos

- Definir claramente la categoría en la que se clasificarán los términos.
- Diseñar un método para extraer términos de esta categoría.
- Construir y probar un sistema que siga dicho método.
- Diseñar una aplicación que permita utilizar dicho sistema como servicio web.

Capítulo 2

Marco teórico

2.1. Reconocimiento de entidades propias

Se definen las entidades propias como frases dentro de una oración o texto que representan nombres de personas, organizaciones, lugares, fechas y cantidades, entre otras. Por ejemplo, dentro del texto "Pedro, actualmente un periodista en Chile, jugó con De La Fuente a finales de los ochenta en el Real Madrid" se pueden destacar 5 entidades propias:

- Persona: *Pedro*.
- Lugar: *Chile*.
- Persona: *De La Fuente*.
- Fecha: *los ochenta*.
- Organización: *Real Madrid*.

Generalmente se utilizan cuatro categorías para clasificar las entidades que se extraen de un texto: Lugar, Persona, Organización y Misceláneo. Esta última agrupa todas las entidades que no entran en las demás categorías. La extracción y clasificación de entidades en estas categorías a partir de textos se ha llamado Reconocimiento de Entidades Propias (en inglés Named Entity Recognition).

Una manera comúnmente utilizada para representar formalmente el reconocimiento de entidades propias es la basada en palabras y etiquetas. Supongamos que se tiene un texto T, formado por una serie de n palabras $w_1\ w_2\ \cdots\ w_i\ \cdots\ w_n$, donde i representa la posición relativa dentro del texto. Cada palabra w_i posee una serie de características, por ejemplo puntuación o mayúsculas. Esto hace que una misma palabra, por ejemplo *paredes*, puede aparecer escrita de muchas maneras distintas, como *Paredes*, *PAREDES*, *paredes;* o *"Paredes*. La salida requerida para un sistema que reconoce entidades propias es una serie de etiquetas $y_i \in \Lambda$ asociadas a cada palabra w_i, donde Λ es el conjunto de tipos de entidades

posibles, además de un elemento que representa que no es entidad. Por ejemplo un posible Λ es $\{per, lug, org, mis, ninguna\}$ donde $y_i = ninguna$ representa que w_i no es parte de ninguna entidad propia.

Si Θ es el conjunto de todos los textos posibles, un sistema que realiza reconocimiento de entidades propias en el conjunto Λ puede representarse entonces por una función N que le asigna a cada palabra del texto una etiqueta:

$$N : \Theta \mapsto \Lambda^{\mathbb{N}}$$

Cada coordenada del vector de salida representa la etiqueta para la palabra correspondiente en el texto, el cual puede tener distintos largos. Se supone que si un texto es de largo n, los valores del vector de salida más allá de la coordenada n no tienen sentido para el problema. Cabe destacar que no es posible usar una notación del tipo $N : W \mapsto \Lambda$, con W el conjunto de palabras posibles, ya que el valor asignado a una palabra depende del contexto donde se encuentra. Una misma palabra puede ser parte de entidades de distinto tipo en distintos textos. Por ejemplo la palabra *Casas* puede representar una persona o un lugar, tal como en las siguientes oraciones: "Mañana veré a Don Armando Casas" y "Se está construyendo el conjunto habitacional Casas Rojas".

Se han probado diversos métodos para realizar esta actividad. Para poder compararlos se han realizado competencias llamadas CoNLL Shared Task: Language-Independent Named Entity Recognition. En ellas se pone a disposición de los participantes un conjunto de documentos clasificado manualmente, separado en un conjunto de entrenamiento, un conjunto de desarrollo y un conjunto de prueba. Las características específicas de estos documentos, los métodos utilizados por los diversos participantes y los resultados obtenidos están publicados en [8] y [9], enfocada la primera al idioma español y alemán, y la segunda enfocada a inglés y alemán.

2.2. Modelo de etiquetado

Usualmente, para construir una función N, se usa un modelo probabilístico, que dado un texto x, se define como

$$N(x) = \arg\max_{y \in \Lambda^{\mathbb{N}}} p(y|x)$$

En otras palabras, el clasificador N selecciona el vector de etiquetas y que es más probable, donde cada etiqueta está asociada a una palabra de la secuencia de palabras formada por el texto x. El problema se traduce entonces en encontrar un modelo para calcular las probabilidades $p(y|x)$.

2.2.1. Modelos generativos y discriminativos

Existen dos grandes tipos de modelos utilizados para calcular las probabilidades que se necesitan. Uno es el modelo *generativo*, el cual modela la probabilidad conjunta $p(y, x)$ de las etiquetas a generar y junto con la variable de entrada x, que en este caso sería el contenido de un texto. Este tipo de modelos se basa en la siguiente definición:

$$p(y|x) = \frac{p(y, x)}{p(x)}$$

Se tiene que para el problema de maximización, $p(x)$ es una constante, por lo que para utilizar este modelo basta definir cómo calcular las probabilidades $p(y, x)$.

El otro modelo es el *discriminativo*, el cual modela las probabilidades condicionales $p(y|x)$ directamente.

La mayor diferencia de manejar una distribución condicional $p(y|x)$ a modelar $p(y, x)$ radica en la dificultad en modelar esta última. Una alternativa para hacerlo es enumerar todas las posibles alternativas (y, x), a partir de datos de entrenamiento. Este enfoque es intratable para cualquier problema real, ya que generalmente los nombres buscados dentro de un texto x no han aparecido en el conjunto de entrenamiento, por lo que no se tendría información en el modelo y su probabilidad conjunta sería pequeñísima. Además, la cantidad de alternativas posibles es tan grande que sería difícil encontrar un conjunto de entrenamiento ya etiquetado que sirva como referencia para todas ellas.

Otra alternativa para modelar $p(y, x)$ es utilizar un modelo a partir de características de las palabras. Para etiquetar palabras no vistas es necesario entonces hacer uso de otras características de una palabra, como el uso de mayúculas, las palabras vecinas, sus prefijos y sufijos, la pertenencia a alguna lista predeterminada de palabras, u otras. Generalmente se incluyen en estos modelos varias características altamente dependientes. Para incluir estas características que dependen unas de otras en un modelo generativo, hay dos alternativas: profundizar el modelo para representar dichas dependencias en las entradas o hacer suposiciones de independencia. La primera de ellas es generalmente dificíl. Por ejemplo, es difícil imaginar cómo modelar la dependencia entre el uso de mayúsculas de una palabra y sus sufijos, y tampoco es deseable hacerlo, ya que de todas maneras esta dependencia se presentaría en las oraciones de entrenamiento. La segunda alternativa no es deseable ya que el modelo puede perder eficacia.

La principal ventaja de un modelo discriminativo es que está mejor dotado para incluir características interdependientes. Es ésta la razón por la que fue elegido un modelo de este tipo en este trabajo.

2.2.2. Modelos de grafo

Si bien al utilizar un modelo discriminativo es posible no suponer independencia entre las distintas variables utilizadas, es necesario que el modelo sea manejable y que se pueda entrenar en un tiempo razonable. Para esto, una alternativa es suponer que las probabilidades $p(y, x)$ o $p(y|x)$ se pueden factorizar como la multiplicación de un conjunto de funciones predefinidas.

Un modelo de grafo es una familia de distribuciones de probabilidad que se factoriza de acuerdo al grafo asociado. La idea principal es representar una distribución sobre una cantidad grande de variables aleatorias por un producto de funciones locales, que dependen sólo de un número reducido de variables.

Considerando y como el conjunto de variables aleatorias de salida, y x como las variables de entrada u observadas, se construyen los vértices del grafo como $V = x \cup y$. Dada una colección de subconjuntos $A \subset V$, se define el modelo de grafo asociado como el conjunto de todas las distribuciones que pueden escribirse de la forma

$$p(y, x) = \frac{1}{Z} \prod_{A} \Psi_A(x, y)$$

para cualquier elección de factores $\{\Psi_A\}$, donde cada Ψ_A es una función a valores reales positivos, y depende sólo de las variables aleatorias representadas por los vértices de A. Por esto la unión de todos los subconjuntos A debe cubrir todos los nodos del grafo. La constante Z es un factor de normalización que asegura que la distribución suma 1. Es definido como

$$Z = \sum_{x,y} \prod_{A} \Psi_A(x, y)$$

En la Figura 2.1 es posible ver un ejemplo de modelo de grafo. Según éste, la probabilidad $p(y, x)$ se puede factorizar como:

$$p(y, x) =$$
$$\Psi_{A_1}(y_1, x_1, x_2)\, \Psi_{A_2}(y_2, x_3, x_4)\, \Psi_{A_3}(y_2, x_2)\, \Psi_{A_4}(y_1)\, \Psi_{A_5}(y_2)\, \Psi_{A_6}(x_1)\, \Psi_{A_7}(x_2)\, \Psi_{A_8}(x_3)\, \Psi_{A_9}(x_4)$$

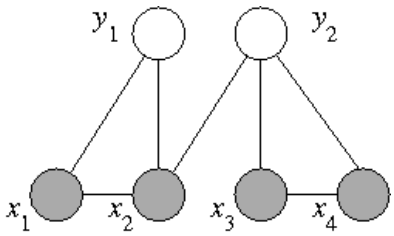

Figura 2.1: Grafo no dirigido. Los nodos blancos representan las variables de salida

Gracias a este ejemplo es posible notar que los conjuntos A explicados anteriormente quedan representados por los distintos cliques en el grafo.

2.2.3. Campos aleatorios condicionales

A continuación se presenta el modelo utilizado en este trabajo para realizar la tarea de reconocimiento de entidades propias, los campos aleatorios condicionales (en inglés Conditional Random Fields o CRF de ahora en adelante). La presentación y definiciones están basadas en [2], publicación donde se introdujo este modelo por primera vez orientado a etiquetación de secuencias.

En lo que sigue, x es una variable aleatoria sobre las secuencias de datos que serán etiquetadas, e y es una variable aleatoria sobre las correspondientes secuencias de etiquetas. Se supone que todos los componentes y_i de y que varían dentro del conjunto de etiquetas posibles Λ. Por ejemplo, x podría variar sobre oraciones en lenguaje natural. Las variables aleatorias x e y están distribuidas conjuntamente, pero en un contexto discriminativo se construye un modelo condicional $p(y|x)$ a partir de observaciones y secuencias etiquetadas asociadas, y no se modela explícitamente la distribución marginal $p(x)$.

Sea $G = (V, E)$ un grafo donde $y = (y_v)_{v \in V}$, tal que y está caracterizado por los vértices de G. Las variables x e y forman un campo aleatorio condicional si, condicionadas en x, las variables aleatorias y_v cumplen la propiedad de Markov con respecto al grafo:

$$p(y_v|x, y_w, w \neq v) = p(y_v|x, y_w, w \sim v)$$

donde $w \sim v$ significa que w y v están conectados por una arista en el grafo G. En otras palabras, la probabilidad de y_v depende sólo de las variables aleatorias y_w representadas por sus vecinos en el grafo, además de la secuencia observada x.

En la configuración más simple e importante para modelar secuencias, G es simplemente una cadena o línea, como la de la Figura 2.2, dada por

$$G = (V = \{1, 2, \ldots m\}, E = \{(i, i+1)\})$$

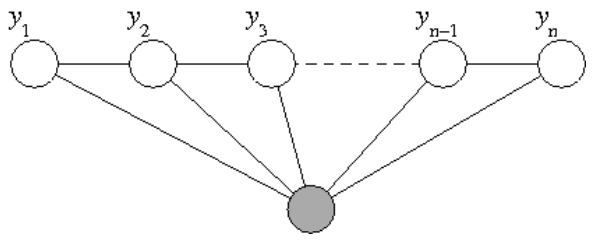

Figura 2.2: Modelo CRF de cadena. El nodo gris representa el texto x de entrada

En este caso x puede tener también una estructura natural de grafo, pero en general no es necesario suponer que x e y tienen la misma estructura de grafo. Incluso no es necesario suponer que x tiene alguna estructura de grafo.

De aquí en adelante se puede suponer una familia de factores

$$\Psi_A(x, y) = \exp\left(\sum_k \theta_{Ak} f_{Ak}(x, y)\right)$$

Así, la distribución condicional puede ser escrita como

$$\begin{aligned} p(y|x) &= \frac{1}{Z(x)} \prod_A \exp\left(\sum_k \theta_{Ak} f_{Ak}(x, y)\right) \\ &= \frac{1}{Z(x)} \exp\left(\sum_A \sum_k \theta_{Ak} f_{Ak}(x, y)\right) \end{aligned}$$

En el caso de un grafo $G = (V, E)$ de y con forma de árbol (dentro de los cuales una cadena es la expresión más simple), la distribución de la secuencia de etiquetas y dada la observación x tiene la forma

$$p_\theta(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{e \in E, k} \gamma_k f_k(y_{e_1}, y_{e_2}, x) + \sum_{v \in V, k} \mu_k g_k(y_v, x)\right)$$

Aquí y_{e_1} e y_{e_2} son las componentes de y asociadas con los vértices de la arista e, e y_v es la componente de y asociada al vértice v. Gráficamente, esto se puede ver en la Figura 2.2.

Se supone que las *características* f_k y g_k son dadas y fijas. Por ejemplo, una característica de vértice g_k podría ser una función que toma solo dos valores, siendo 1 si la palabra x_i está en mayúsculas y la etiqueta y_i es un nombre propio, y 0 si no. Se puede suponer que las características g_k son un subconjunto particular dentro de las funciones f_k. Así, en el caso de una secuencia las características son representadas por el mismo tipo de funciones Ψ_A mencionadas anteriormente, donde los conjuntos A contienen a los pares de vértices asociados a y_i e y_{i+1}.

El problema de estimación de parámetros es el de determinar los parámetros del modelo, que están dados por $\theta = (\gamma_1, \gamma_2, \ldots; \mu_1, \mu_2, \ldots) = (\theta_1, \theta_2, \ldots)$, a partir de datos de entrenamiento $D = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^N$. Estos N datos siguen una distribución empírica $\tilde{p}(y, x)$. Se define la función objetivo de log-verosimilitud como

$$\ell(\theta) = \sum_{k=1}^N \log\left(p_\theta\left(y^{(k)}|x^{(k)}\right)\right) \propto \sum_{x,y} \tilde{p}(y, x) \log\left(p_\theta(y|x)\right)$$

$$\ell(\theta) = \sum_{k=1}^{N} \left[\log\left(\frac{1}{Z\left(x^{(k)}\right)} \right) + \sum_j \theta_j f_j\left(x^{(k)}, y^{(k)}\right) \right]$$

Esta función representa el ajuste que tienen los parámetros del modelo a los datos de entrenamiento. Derivando esta última expresión, se tiene

$$\begin{aligned} \frac{\partial \ell(\theta)}{\partial \theta_j} &= \sum_{k=1}^{N} \left[f_j\left(x^{(k)}, y^{(k)}\right) - \sum_y f_j\left(y, x^{(k)}\right) p_\theta\left(y|x^{(k)}\right) \right] \\ &= E_{\tilde{p}(y,x)}\left[f_j(x,y)\right] - \sum_{k=1}^{N} E_{p_\theta\left(y|x^{(k)}\right)}\left[f_j\left(y, x^{(k)}\right)\right] \end{aligned}$$

donde $E_p[\cdot]$ denota la esperanza con respecto a la distribución p. Igualando esta última expresión a cero se logra la condición del modelo de máxima entropía: La esperanza de cada característica con respecto a la distribución del modelo es igual al valor esperado bajo la distribución empírica de los datos de entrenamiento.

No es posible determinar analíticamente los valores de los parámetros θ que maximizan la función de log-verosimilitud. Sin embargo, se ha probado que pueden ser efectivamente determinados utilizando alguna técnica iterativa de aproximación numérica.

Aunque se parecen mucho a los modelos de redes ocultas de Markov (en inglés *Hidden Markov Models* o HMM), la clase de CRF es mucho más expresiva, ya que permite dependencias arbitrarias en la secuencia de observación. Además, las características no necesitan especificar completamente un estado u observación, por lo que se puede esperar que el modelo sea estimado con menos datos de entrenamiento. Otra propiedad atractiva es la convexidad de la función objetivo ℓ. De hecho, los CRF comparten esta propiedad de convexidad de los modelos de máxima entropía en general. Es posible encontrar una introducción a los modelos HMM en [4].

2.2.4. Probabilidades CRF como computaciones matriciales

Como la tarea de extracción de entidades propias abordado se trata de una secuencia de palabras, supondremos de aquí en adelante que las dependecias de y, condicionadas a x, son una cadena. Para simplificar algunas expresiones, se agregan los estados especiales $y_0 = partida$ e $y_{n+1} = final$. Así, la estructura de grafo asociada es de la forma de la Figura 2.2. Para una estructura de cadena, la probabilidad condicional de una secuencia de etiquetas puede ser expresada de forma concisa en una forma matricial, la que es útil para explicar la estimación de parámetros y los algoritmos de inferencia.

Supongamos que $p_\theta(y|x)$ es un CRF. Para cada posición i en la secuencia de observación x, se define la variable aleatoria matricial $M_i(x) = [M_i(y', y''|x)] \in |\Lambda| \times |\Lambda|$ como

$$M_i\left(y', y''|x\right) = \exp\left(\sum_j \theta_j f_j\left(x, y|_{e_i} = (y', y'')\right)\right)$$

donde e_i es la arista con las etiquetas (y_{i-1}, y_i). En contraste con los modelos generativos, los CRF no necesitan enumerar todas las posibles secuencias de observación x, por lo que estas matrices pueden ser calculadas directamente al necesitarlas a partir de secuencias observadas de entrenamiento o prueba x y el vector de parámetros θ. La función de normalización $Z_\theta\left(x\right)$ es la componente $(partida, final)$ de la matriz producto de estas matrices:

$$Z_\theta\left(x\right) = \left(M_1\left(x\right) M_2\left(x\right) \cdots M_{n+1}\left(x\right)\right)_{partida,final}$$

Usando esta notación, la probabilidad condicional de una secuencia de etiquetas y se escribe, si $y_0 = partida$ e $y_{n+1} = final$, como

$$p_\theta\left(y|x\right) = \frac{\prod_{i=1}^{n+1} M_i\left(y_{i-1}, y_i|x\right)}{\left(\prod_{i=1}^{n+1} M_i\left(x\right)\right)_{partida,final}}$$

2.2.5. Inferencia con CRF

La labor de inferencia en CRF se refiere a encontrar, dada una observación x, el etiquetado más probable $y^* = \arg\max_y p\left(y|x\right)$. En un CRF de cadena, esta tarea puede ser realizada eficiente y exactamente por variantes de los algoritmos estándares de programación dinámica utilizados para HMM. Un conocido algoritmo para esto es el algoritmo de Viterbi, cuya adaptación para CRF de cadena es directa. La explicación de este algoritmo expuesta a continuación está basada en [2] y [7], donde se explica en mayor detalle la diferencia entre esta versión y la original para HMM.

Se define el vector $\alpha_t\left(x\right)$ de manera que la coordenada $j \in \left\{\lambda_0, \lambda_1, \ldots, \lambda_{|\Lambda|}, partida, final\right\}$ del vector represente la máxima probabilidad que la etiqueta y_t tome el valor j, tomando cualquier camino posible de estados para $y_0, \ldots, y_{t-1}$. De esta manera, siguiendo la notación matricial previa:

$$\left(\alpha_t\left(x\right)\right)_j = \max_{y^*|_0^{t-1}} \left[M_t\left(y_{t-1} = y^*_{t-1}, y_t = j|x\right) \prod_{i=1}^{t-1} M_i\left(y_{i-1} = y^*_{i-1}, y_i = y^*_i|x\right)\right]$$

$$\left(\alpha_0\left(x\right)\right)_j = \begin{cases} 1 & \text{si } j = partida \\ 0 & \text{si no} \end{cases}$$

Esta definición permite construir la siguiente recurrencia:

$$\left(\alpha_t\left(x\right)\right)_j = \max_{k \in |\Lambda|} \left[M_t\left(y_{t-1} = k, y_t = j|x\right)\left(\alpha_{t-1}\left(x\right)\right)_k\right]$$

Calcular la probabilidad de la secuencia de etiquetado más probable se reduce entonces en calcular $\alpha_{n+1}\left(x\right)$ y ver el valor asociado al estado $final$. Como lo que se busca es etiquetar

la secuencia x, es necesario además guardar en cada paso la etiqueta que se utilizó para calcular las componentes de cada vector $\alpha_i(x)$. Más formalmente, se define el vector $\beta_i(x)$ como:

$$(\beta_i(x))_j = \arg\max_{k \in |\Lambda|} [M_t(y_{t-1} = k, y_t = j|x)(\alpha_{t-1}(x))_k]$$

Estudiando la cantidad de tiempo necesario por el algoritmo, se deben primero calcular las matrices M_i para cada $i \in \{1, \ldots, n+1\}$. Cada matriz tiene $(|\Lambda|+2)^2$ componentes. Se puede suponer que para calcular cada componente se debe sumar la evaluación de a lo más C funciones características f_j. Esto da un costo inicial de $O\left((n+1)(|\Lambda|+2)^2 C\right)$. Luego, para calcular cada componente de un nuevo vector $\alpha_i(x)$ se necesita encontrar el máximo entre $|\Lambda|+2$ multiplicaciones. El costo total de calcular cada nuevo vector $\alpha_i(x)$ es entonces $O\left((|\Lambda|+2)^2\right)$. Es necesario calcular $n+1$ de estos vectores. Calcular los vectores $\beta_i(x)$ es despreciable, pues pueden ser calculados al mismo tiempo de calcular los $\alpha_i(x)$. Finalmente, el costo total del algoritmo para una secuencia de largo n es $O\left((n+1)(|\Lambda|+2)^2 C + (n+1)(|\Lambda|+2)^2\right) = O\left((n+1)(|\Lambda|+2)^2 (C+1)\right)$. Al ser la cantidad de estados $|\Lambda|$ y la cantidad de funciones características a evaluar C fijas, el costo es lineal con respecto al número de palabras de la secuencia a etiquetar.

2.3. Extracción de características

La elección de la técnica de aprendizaje que se utilizará para realizar una tarea de reconocimiento de entidades propias es muy importante. Sin embargo, como se puede comprobar en muchos de los sistemas que lo realizan, como los mencionados en [8] y [9], la elección de las caracterísicas a considerar por el modelo es aun más importante.

2.3.1. Categoría gramatical

La categoría gramatical (o parte de una oración, en inglés part of speech o POS) se refiere a una variable que puede tomar diferentes valores que condicionan la forma morfológica concreta de una palabra. La categoría gramatical de una palabra depende del contexto donde aparece, es decir de la función que cumple dentro de una oración. Un ejemplo de categorías gramaticales que tradicionalmente se usan en español sería:

- Artículo.
- Sustantivo.
- Pronombre.
- Verbo.
- Adjetivo.

- Adverbio.
- Preposición.
- Conjunción.

Cada una de estas categorías puede subdividirse en categorías más específicas, según género, número, etc.

Categoría gramatical como problema de etiquetación

Determinar la categoría gramatical de cada palabra en una oración es un problema de etiquetación de secuencias, por lo que existen diversos métodos para realizarlo. Hay una gran cantidad de etiquetadores para el idioma inglés, ampliamente probados y comparados entre sí.

Al cambiar de idioma inglés a español, hay algunos problemas específicos que se deben tomar en cuenta. Un problema surge al notar que en español existen muchas más variaciones de una misma palabra, lo que resulta en una cantidad más grande de parámetros que hay que afinar. Otro problema, en general, es la falta de grandes volúmenes de texto bien etiquetados, que puedan ser usados como entrenamiento para un sistema de etiquetado. Para superar estos problemas, se necesitan métodos que logren alta precisión con un volumen pequeño de entrenamiento.

Para construir un sistema que etiquete categorías gramaticales, se puede hacer una suposición del tipo Markov. Si t es la secuencia de etiquetado y w la secuencia de palabras observadas, se supone que:

$$p\left(w_1, \ldots, w_n, t_1, \ldots, t_n\right) = p\left(t_n | t_{n-2}, t_{n-1}\right) p\left(w_n | t_n\right) p\left(w_1, \ldots, w_{n-1}, t_1, \ldots, t_{n-1}\right)$$

Diversos métodos de n-gramas modelan la probabilidad de transición $p\left(t_n | t_{n-2}, t_{n-1}\right)$ de la siguiente manera:

$$p\left(t_n | t_{n-2}, t_{n-1}\right) = \frac{F\left(t_{n-2} t_{n-1} t_n\right)}{F\left(t_{n-2} t_{n-1}\right)}$$

donde $F\left(t_{n-2} t_{n-1} t_n\right)$ es la cantidad de ocurrencias del 3-grama $t_{n-2} t_{n-1} t_n$ en el conjunto de entrenamiento y $F\left(t_{n-2} t_{n-1}\right)$ es la cantidad de ocurrencias de la bigrama $t_{n-2} t_{n-1}$. Esta estimación produce problemas ya que muchas frecuencias son pequeñas, lo que hace que las probabilidades asociadas no puedan estimarse bien. Particularmente difícil es el caso de frecuencias nulas, pues no queda determinado si dicho trigrama es incorrecto o sólo muy improbable. Otro punto es que un etiquetador robusto debe ser capaz de trabajar con entradas más allá de lo correcto gramaticalmente.

Árbol de decisión

Una alternativa a la estimación con n-gramas es utilizar un árbol binario de decisión para estimar las probabilidades de transición. La probabilidad de una 3-grama dada es determinada siguiendo el camino correspondiente a través del árbol hasta llegar a una hoja. Por ejemplo, dado el árbol de la Figura 2.3, supongamos que se quiere determinar la probabilidad de un sustantivo precedido por un artículo y un adjetivo $p\,(NN|DET, ADJ)$, se debe responder primero la pregunta del nodo raíz. Como la etiqueta de la palabra previa es ADJ, se sigue el camino *sí*. La siguiente pregunta es verdadera también, por lo que se termina en un nodo hoja. En esta hoja hay una tabla que contiene la probabilidad de las distintas etiquetas siguientes. Basta mirar la entrada de la etiqueta NN.

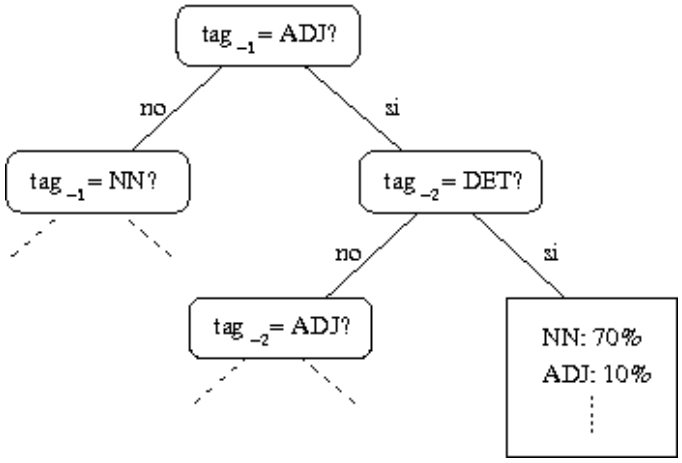

Figura 2.3: Árbol de decisión de ejemplo, con probabilidades de ejemplo para idioma inglés

Construcción del árbol de decisión

El árbol de decisión es construido recursivamente a partir del conjunto de 3-gramas de entrenamiento. En cada paso de la recursión, es creada una pregunta de manera que divida al conjunto de 3-gramas en dos subconjuntos lo más distintos posible, en cuanto a la distribución de probabilidad de la tercera etiqueta (predicha). La pregunta examina una de las dos etiquetas previas y verifica si es identica a una etiqueta t. Si T es el conjunto de posibles etiquetas, una pregunta tiene la forma siguiente:

$$etiqueta_{-i} = t : \; i \in \{1, 2\} \, ; \; t \in T$$

A cada paso de la recursión, todas las preguntas posibles son comparadas y la que da más información se agrega al nodo actual en el árbol de decisión. Entonces este nodo se expande recursivamente en cada uno de los dos subconjuntos del conjunto de entrenamiento, definidos por la pregunta. Los subárboles resultantes se agregan al nodo actual como subárboles *sí* o *no*.

Léxico

Un léxico contiene las probabilidades a priori de una etiqueta para cada palabra. Tiene tres partes: un léxico completo, un léxico de sufijos y una entrada por defecto.

Al buscar una palabra en el léxico, primero es buscada en el léxico completo. Si la palabra es encontrada, la probabilidad de la etiqueta correspondiente es entregada. Si no, se busca en el léxico de sufijos. Si ambas búsquedas fallan, se entrega la entrada por defecto.

El léxico completo es creado a partir de las palabras de un conjunto de entrenamiento. Se cuenta la cantidad de ocurrencias en él de cada par palabra/etiqueta. Aquellas etiquetas cuya frecuencia relativa para una palabra es menor a un 1 % son descartadas pues en general corresponden a errores en el conjunto de entrenamiento.

La segunda parte del léxico, el léxico de sufijos, se organiza en un árbol. Cada nodo del árbol (exceptuando la raíz) es etiquetados con un carácter. En cada nodo hoja, se agrega la lista de probabilidades para cada etiqueta. Buscar en el árbol de sufijos se refiere a seguir cada carácter de la palabra como una rama del árbol hasta llegar a un nodo hoja. En la Figura 2.4 se muestra un árbol de ejemplo.

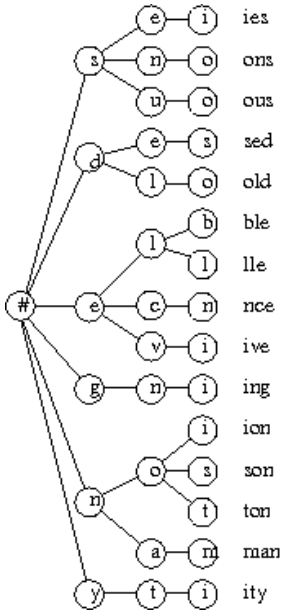

Figura 2.4: Un árbol de sufijos reversos de ejemplo para inglés

La entrada por defecto se construye restando la frecuencia de las etiquetas de todas las hojas del árbol de sufijos de las frecuencias de las etiquetas en el nodo raíz y normalizando las frecuencias resultantes, de manera que sumen 1.

TreeTagger

TreeTagger es una implementación open source en Java del modelo de árbol de decisión para etiquetar categorías gramaticales. Esta herramienta utiliza todos los conceptos recién vistos, y ha sido entrenada para distintos idiomas, entre ellos español. Para un análisis más detallado sobre el funcionamiento de TreeTagger, se puede revisar [5].

La siguiente es la lista de todas categorías gramaticales en que TreeTagger clasifica cada palabra de una oración.

```
ACRNM     Acronimo (ISO, CEI)
ADJ       Adjetivos (mayores, mayor)
ADV       Adverbios (muy, demasiado, como)
ALFP      Letra plural del alfabeto (As/Aes, bes)
ALFS      Letra singular del alfabeto (A, b)
ART       Articulos (un, las, la, unas)
BACKSLASH backslash (\)
CARD      Cardinales
CC        Conjuncion coordinante (y, o)
CCAD      Conjuncion coordinante adversativa (pero)
CCNEG     Conjuncion coordinante negativa (ni)
CM        coma (,)
CODE      Codigo alfanumerico
COLON     dos puntos (:)
CQUE      que (como conjuncion)
CSUBF     Conjuncion subordinante que introduce frases finitas (apenas)
CSUBI     Conjuncion subordinante que introduce frases infinitas (al)
CSUBX     Conjuncion subordinante subespecificada por tipo-subord (aunque)
DASH      menos (-)
DM        Pronombres demostrativo (esas, ese, esta)
DOTS      Etiqueta para "..."
FO        Formula
FS        Simbolos de puntuacion de detencion completa
INT       Pronombres interrogativos (quienes, cuantas, cuanto)
ITJN      Interjecciones (oh, ja)
LP        parentesis izquierdos ("(", "[")
NC        Sustantivos comunes (mesas, mesa, libro, ordenador)
NEG       Negacion
NMEA      Sustantivo de medida (metros, litros)
NMON      Nombre de mes
NP        Nombres propios
ORD       Ordinales (primer, primeras, primera)
PAL       al
```

PDEL	del
PE	Palabra extranjera
PERCT	porcentaje (%)
PNC	Palabra no clasificada
PPC	Pronombre personal clitico (le, les)
PPO	Pronombres posesivos (mi, su, sus)
PPX	Pronombres personales y cliticos (nos, me, nosotras, te, si)
PREP	Preposicion negativa (sin)
PREP	Preposicion
PREP/DEL	Preposicion compleja "despues del"
QT	Simbolo de cita (" ’ ‘)
QU	Cuantificadores (sendas, cada)
REL	Pronombres relativos (cuyas, cuyo)
RP	Parentesis derechos (")", "]")
SE	se
SEMICOLON	punto y coma (;)
SLASH	slash (/)
SYM	Simbolos
UMMX	Unidad de medida (MHz, km, mA)
VCLIger	Verbo gerundio clitico
VCLIinf	Verbo infinitivo clitico
VCLIfin	Verbo finito clitico
VEadj	Verbo estar. Pasado participio
VEfin	Verbo estar. Finito
VEger	Verbo estar. Gerundio
VEinf	Verbo estar. Infinitivo
VHadj	Verbo haber. Pasado participio
VHfin	Verbo haber. Finito
VHger	Verbo haber. Gerundio
VHinf	Verbo haber. Infinitivo
VLadj	Verbo lexico. Pasado participio
VLfin	Verbo lexico. Finito
VLger	Verbo lexico. Gerundio
VLinf	Verbo lexico. Infinitivo
VMadj	Verbo modal. Pasado participio
VMfin	Verbo modal. Finito
VMger	Verbo modal. Gerundio
VMinf	Verbo modal. Infinitivo
VSadj	Verbo ser. Past participio
VSfin	Verbo ser. Finito
VSger	Verbo ser. Gerundio
VSinf	Verbo ser. Infinitivo

2.3.2. Otras características

Además de la categoría gramatical, hay otras características que pueden usarse para el reconocimiento de entidades propias.

La siguiente lista presenta características de una palabra que se pueden extraer con búsqueda de expresiones regulares sobre ellas.

- Si es palabra (contiene sólo letras).
- Es una letra mayúscula acompañada de un punto.
- Es una letra mayúscula sola.
- Es un número (no contiene letras ni símbolos además de números).
- Es un punto.
- Es un coma.
- Es un caracter especial.
- Contiene un caracter especial.
- Sólo su letra inicial es mayúscula.
- Está escrita completa en mayúsculas.
- Está escrita completa en minúsculas.
- Es alfanumérica (letras y/o números).
- Termina con punto.
- Termina con coma.
- Termina con algún tipo de puntuación.
- Es puntuación sola.
- Contiene números.
- Es un dígito solo.
- Es dos dígitos.
- Es tres dígitos.
- Es cuatro dígitos.

- Es un rango de números (ejemplo *200-300*).
- Es una palabra separada por guión (ejemplo *auto-bomba*).
- Es una secuencia separada por guiones.
- Es url.
- Es email.

Como las funciones características pueden depender de las etiquetas de dos palabras contiguas, se consideran también una serie de características que toman en cuenta los pares de etiquetas posibles y la posición dentro de la secuencia completa.

Capítulo 3

Diseño de una solución

3.1. Servicio web

3.1.1. Casos de uso

En el contexto de una aplicación tipo servicio web, el diagrama de la Figura 3.1 presenta los casos de uso esperados. A continuación se explica en qué consiste cada uno.

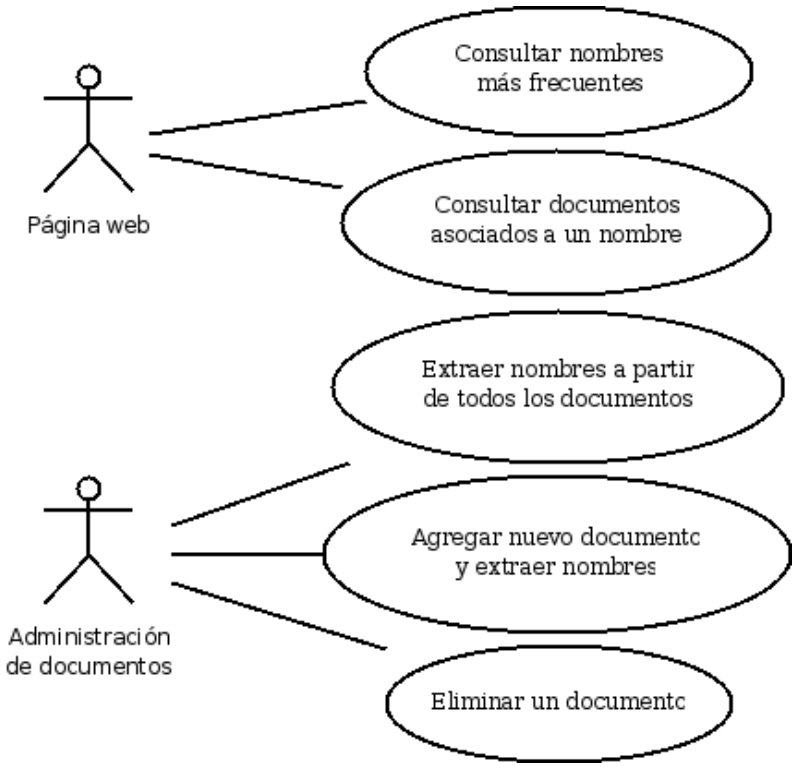

Figura 3.1: Casos de uso para la aplicación

- *Consultar nombres más frecuentes:* El usuario consulta la página web y desea ver la nube de nombres actual. Para generarla, la página web consulta al servicio web y éste le entrega los nombres más frecuentes junto con su frecuencia.
- *Consultar documentos asociados a un nombre:* El usuario eligió un nombre de la nube para ver los documentos asociados a dicho nombre. La aplicación entonces entrega la lista de documentos asociados a dicho nombre.
- *Extraer nombres a partir de todos los documentos:* Para inicializar el sistema, el administrador le entrega a la aplicación el conjunto completo de documentos. Ésta extrae los nombres de persona y guarda la asociación entre nombres y documentos.
- *Agregar nuevo documento y extraer nombres:* La administración agrega un nuevo documento al sistema. La aplicación extrae los nombres del nuevo documento y actualiza la asociación entre nombres y documentos.
- *Eliminar un documento:* La administración elimina un documento, por lo que la aplicación elimina toda la información que tenía del documento y los nombres asociados sólo a éste.

Si bien hay que hacer distinción entre los casos de uso tercero y cuarto, puede notarse que, desde el punto de vista de la aplicación, el tercer caso de uso se puede realizar si es posible realizar el cuarto. Partiendo de un conjunto de documentos al cual se quieren extraer nombres de personas, se puede ir agregando cada uno, uno por uno, hasta completar todos los documentos. De esta manera, la aplicación puede construirse de manera incremental, esto es que extraer nombres a partir de todos los documentos se realice agregando los documentos uno por uno.

3.1.2. Arquitectura y modelo de datos

Entendiendo esto, se propone la arquitectura de módulos de la Figura 3.2 para la aplicación. En esta arquitectura, cada módulo tiene una única función específica. Puede notarse que cuatro módulos corresponden a funciones esperadas del sistema completo y sólo uno de ellos no interactúa directamente con la interfaz de la aplicación. En este caso la interfaz se refiere simplemente a una interacción vía web con el sistema, por ejemplo acceder a una cierta url.

Esta arquitectura permite separar la funcionalidad esperada del servicio web con la parte de la aplicación que extrae realmente los nombres. Así, puede cambiarse radicalmente el método para extraerlos, construyendo un nuevo módulo, sin necesidad de cambiar esta arquitectura. Además, el extractor de nombres tampoco interactua con la base de datos, por lo que podría cambiarse la forma de manejar o de almacenar los datos, sin afectar la extracción de nombres.

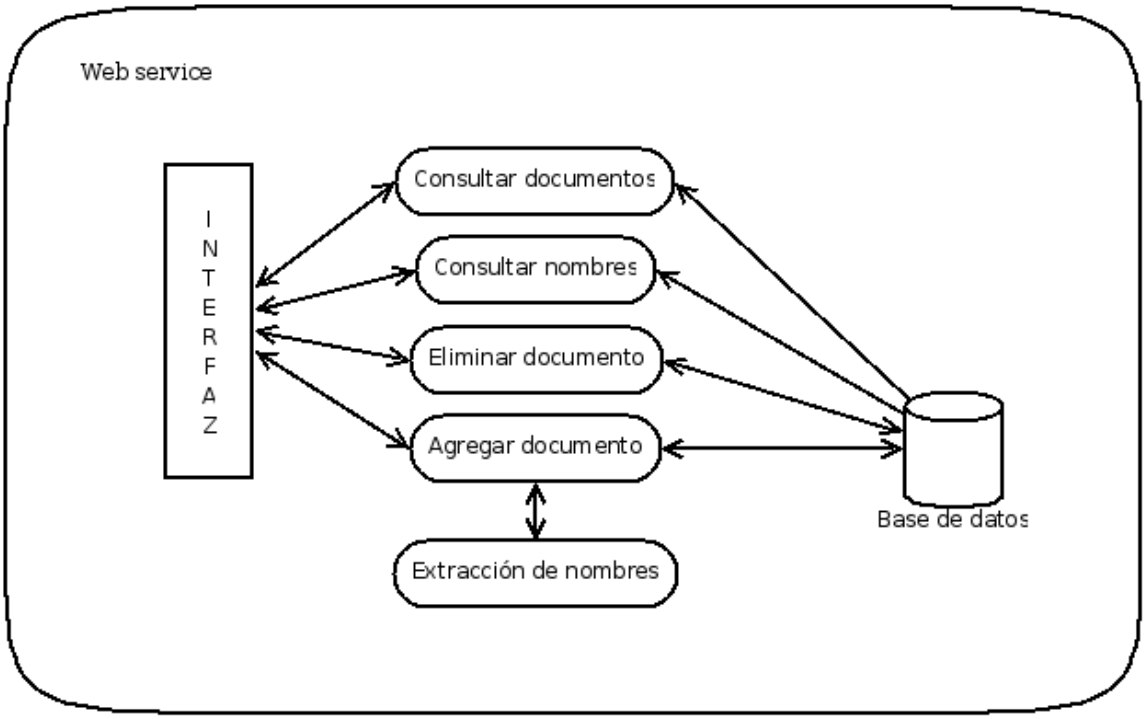

Figura 3.2: Diagrama de módulos de la aplicación

En las Figuras 3.3 y 3.4 se presenta el diseño lógico y físico de los datos. De esta manera, la base de datos puede mantener la información de los términos extraídos de documentos de distintos conjuntos. Cada ITEM representa un documento al cuál se le extrajeron nombres. No se guarda el texto del documento, sino sólo un identificador y una referencia al lugar de donde provino, por ejemplo una url. Un ITEM puede tener una serie de TAGs asociados, o ninguno. Sin embargo, se hace la suposición que si existe un elemento TAG, éste ha sido asociado a, al menos, un ITEM. Cada TAG es único en nombre y sitio. Es decir, puede haber TAGs con el mismo nombre, pero de distintos conjuntos de documentos. Además, para cada TAG se guarda la cantidad de documentos del sitio a los que fue asignado. Si bien esta información es redundante, permite realizar consultas rápidas sobre los nombres más frecuentes.

3.2. Módulos que interactuan con la interfaz

El diseño de estos módulos supone que reciben una cierta entrada a través de la interfaz. A partir de esta entrada, los dos módulos de consulta simplemente extraen datos desde la base de datos y la retornan de manera estructurada. Los otros dos módulos eliminan, ingresan o modifican datos en la base de datos y generan un valor de retorno.

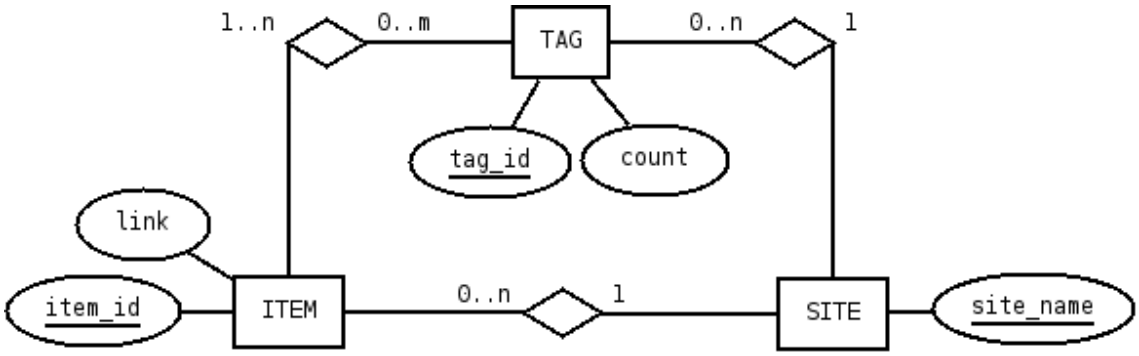

Figura 3.3: Modelo lógico de datos

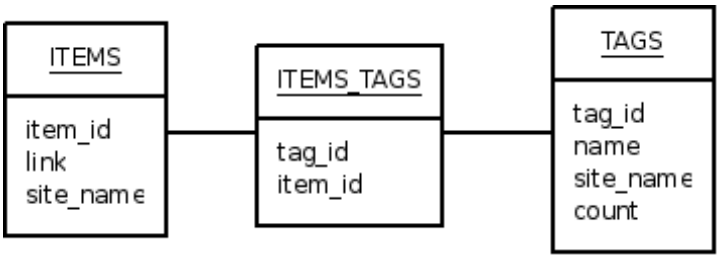

Figura 3.4: Modelo físico de datos

3.2.1. Consultar nombres

- *Entrada:* El nombre del sitio del cual se quieren saber los nombres más frecuentes.
- *Salida:* Una lista de los nombres más frecuentes junto con la frecuencia de aparición de cada uno en el conjunto de documentos de ese sitio, ordenados de mayor a menor frecuencia.

Un problema que se debe considerar, mostrado en el capítulo siguiente, es el caso de nombres formados por dos o más palabras. Éstos pueden aparecer varias veces en la lista, al aparecer también los tags formados por subsecuencias de las palabras que forman el nombre completo. Una posible solución sería que dado un tag formado por dos o más palabras, agrupar todos los subtags, si la frecuencia de aparición es relativamente parecida. Por ejemplo, si aparece el nombre *sanchez suarez* con una frecuencia muy similar a la frecuencia de aparición del nombre *sanchez*, significa que las apariciones de este último son parte de las que se contabilizaron para el nombre completo. En este caso se debería mostrar sólo *sanchez suarez*. Sin embargo, podría suceder que el nombre *suarez* aparece muchísimo más que el nombre *sanchez*, teniendo un porcentaje de aparición que no tiene relación al del nombre completo. En ese caso podría ser aconsejable mostrar también el nombre *suarez*. También puede ocurrir que aparezca otro tag como *miguel sanchez* con una frecuencia parecida a *sanchez suarez*, y que conjuntamente explican la frecuencia total de *sanchez*. En este caso lo ideal sería entregar los tags más largos y no el tag *sanchez*. De las veces en que aparece el

nombre *sanchez* solo, no acompañado de otra palabra, es muy probable que se esté refiriendo a alguno de los otros *sanchez* que fueron extraídos de los documentos, y que simplemente no se escribió su nombre completo, pues el contexto ayudaría al lector a saber de qué *sanchez* en particular se está hablando. De esta manera, si no se va a entregar el nombre *sanchez*, se requiere aumentar la frecuencia de los distintos tags que lo contienen, suponiendo que las apariciones de él sin otra palabra son proporcionales a la frecuencia con que aparece cada tag que está formado por él y otras palabras.

Se utilizó un criterio para determinar si un tag a debe ocultarse de la lista. Sea B el conjunto de tags que contienen al tag a, es decir que son más grandes, y sea S la suma de la frecuencia de aparición de todos los tags en B. El criterio oculta el tag a si se cumplen las dos condiciones siguientes:

- $\frac{S}{P(a)} > \mu$. Aquí $P(a)$ representa la frecuencia de aparición del tag a.
- Existe al menos un tag $c \in B$ tal que $\frac{P(c)}{P(a)} > \theta$.

En estas condiciones μ y θ son ciertos umbrales dados como parámetros. El primero indica que entre los tags más grandes que contienen al tag a, debe haber al menos una fracción μ de ellos distintos al tag a. Con esto se elimina el ruido proveniente de las veces en que al tag a se le agregan palabras contiguas como si fueran parte del nombre. El segundo indica que entre los tags de B debe existir al menos uno que acapare una fracción θ de las ocurrencias de a.

Al ocultarse un tag a, a todos los tags $b \in B$ que lo contienen se les asigna una nueva frecuencia $P'(b)$ como

$$P'(b) = P(b) + \frac{P(b)(P(a) - S)}{S} = \frac{P(b)P(a)}{S}$$

En la Tabla 3.1 se muestra un ejemplo. Cada vez que se decide eliminar un tag se utilizan las frecuencias originales, también para el cálculo de S. De esta forma, las nuevas frecuencias no afectan la decisión de eliminar un tag. Finalmente se entrega como salida la lista de los tags más frecuentes ordenada según su frecuencia modificada de aparición.

Tag	Frecuencia original	Frecuencia final
gonzalez	30	-
fernando gonzalez	10	$10 + \frac{10 \times 10}{20} = 15$
pedro gonzalez	5	$5 + \frac{10 \times 5}{20} = 7,5$
juan gonzalez	5	$5 + \frac{10 \times 5}{20} = 7,5$

Tabla 3.1: Ejemplo de modificación de frecuencias al eliminar tag *gonzalez*

Los nombres propios más comunes son muy frecuentes en el texto. Por ejemplo, el nombre *jose* aparece con una muy alta frecuencia en tres conjuntos de documentos mostrados

en el capítulo siguiente, siendo estos conjuntos profundamente distintos. Esto hace que este tipo de nombres no sean de relevancia para explorar los documentos, ya que hay muchísimos documentos donde aparece cada uno de éstos, al haber muchas personas con el mismo primer nombre. Para solucionar este problema, se agrega un filtro que elimina de la lista de tags generados los nombres formados por uno o más de estos nombres más comunes. Es necesario eliminar todas las posibles combinaciones de nombres comunes, ya que de otra manera podrían aparecer en la lista pares frecuentes como *jose maria* o *juan carlos*.

Es difícil construir una lista completa con todos los nombres de pila posibles, ya que en distintos idiomas hay una amplia gama de nombres. Al centrarse este trabajo en el idioma español, la lista de nombres utilizada debe incluir al menos los nombres comunes en los países de habla hispana. Se utilizó la lista de nombres que la legislación argentina permite para las personas nacidas en Argentina, que cumple con dicho requisito. Esta lista posee 4.124 nombres masculinos y femeninos distintos en total, incluyendo algunos nombres de otros idiomas y adaptaciones de ellos al español. La lista completa puede verse en el Apéndice B.

3.2.2. Consultar documentos

- *Entrada:* El nombre del sitio y el tag de los cuales se quieren saber los documentos asociados.
- *Salida:* Una lista de los documentos asociados al tag. La lista contiene la url almacenada en la tabla ITEMS de cada documento. Dentro de la salida de este módulo, puede incluirse una lista de tags similares al tag buscado. Éstos son los que lo incluyen (nombres formados por más palabras) y los que están incluidos en él (palabras que lo forman).

3.2.3. Eliminar documento

- *Entrada:* El nombre del sitio y el link del documento que se quiere eliminar del sistema.

Este módulo elimina todos los tags que estaban asociados sólo a dicho ITEM. Además, baja en 1 el contador COUNT de apariciones de los demás tags a los que también estaba asociado.

3.2.4. Agregar documento

- *Entrada:* El nombre del sitio y el link del documento que se quiere agregar al sistema.

Este módulo agrega la información del nuevo documento en la tabla ITEMS. Se extrae el contenido del documento desde su origen y dicho contenido se pasa al módulo de extracción de nombres. El resultado de éste se guarda en la tabla ITEMS_TAGS. Si se generó un tag que no se había extraído en ningún documento anterior, se agrega dicho tag a la tabla TAGS.

3.3. Módulo de extracción de nombres

- *Entrada:* El texto al cuál se quieren extraer nombres asociados.
- *Salida:* Lista de nombres asociados al texto.

Para extraer los nombres del texto, se utiliza un modelo de etiquetación CRF basado en palabras. Esto significa que cada palabra del texto se etiquetará como *persona* o *no persona*, y en base a dicha etiquetación, se generarán los tags asociados.

Este módulo presenta la mayor dificultad en este trabajo. Para poder enfrentarse a tal dificultad, se separa el trabajo en partes más pequeñas, más fáciles de abordar cada una por separado. Un esquema de los distintos bloques que conforman este módulo, junto con las dependecias de cada uno, puede verse en la Figura 3.5.

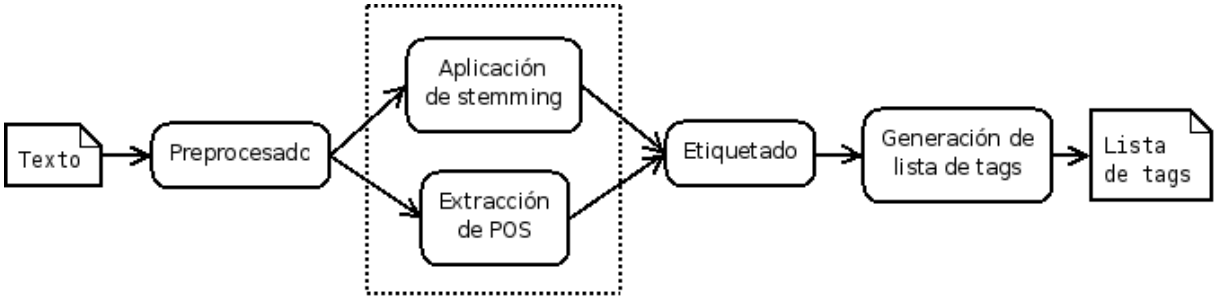

Figura 3.5: Diagrama de bloques del módulo de extracción de nombres

3.3.1. Preprocesado

En esta etapa se eliminan todos los caracteres "extraños", como distintos tipos de puntuaciones, parentizados, asteriscos, tabulaciones, saltos de línea y otros. Este tipo de caracteres son reemplazados por espacios, con lo que lo que estos caracteres se transforman en separadores de palabras. Así, cualquier tipo de puntuación que se presente con palabras a ambos lados se separa en dos palabras. Por ejemplo, si aparece "algo,ahora", se reemplaza por "algo ahora".

Además, se reemplazan las letras con acentos y eñes por la letra base, por ejemplo "Á" es reemplazada por "A" y "ñ" es reemplazada por "n". Al inicio se pensó que eliminar los acentos resultaría en una pérdida de información, por lo que no se consideró esta alternativa. Sin embargo, al no eliminarlos se generan varios problemas. Primero, distintas codificaciones de caracteres en distintos documentos pueden producir errores en las etapas posteriores. Además, la cantidad de palabras distintas que aparecen como universo de palabras posibles es mucho mayor, por lo que el sistema de etiquetado posterior tiene una dimensionalidad mucho mayor, haciendo su entrenamiento mucho más difícil. Otra razón es que este sistema

está pensado para enfrentarse a una amplia gama de documentos, dentro de los cuales puede haber muchas faltas de ortografía. Agregar los acentos agrava más este punto, pues aunque se producen muchas faltas de ortografía por otras razones, también se producen errores producto de los acentos.

Finalmente, como muchos documentos provienen de páginas web en formato *HTML*, se transforma el texto eliminando todos los tags de HTML y reemplazándolos por espacios.

3.3.2. Aplicación de stemming y extracción de POS

Stemming consiste en la reducción de una palabra a su raíz, sin perder información relevante para la tarea realizada. Por ejemplo, las conjugaciones de los verbos pueden reemplazarse por su infinitivo, las palabras que tienen distinto género pueden escribirse siempre en su versión masculina (o femenina). Todo este tipo de reducciones ayuda a reducir la dimensionalidad que tendrá el etiquetador automático, ya que el dominio de palabras se reduce. Los nombres generalmente no están formados por palabras comunes, por lo que realizar stemming no los modifica.

Si bien estas dos tareas son distintas, son realizadas en conjunto con la herramienta *TreeTagger* presentada en el capítulo anterior. TreeTagger etiqueta la parte de la oración para cada palabra, realizando primero la tarea de stemming. Un ejemplo de entrada/salida para TreeTagger, extraído de los documentos utilizados en el entrenamiento, es el siguiente:

```
Incluso paseamos por el clasico Barrio Rojo donde descubri que tengo un
negocio de ropa interior que retrata a la perfeccion mi personalidad

Incluso         ADJ     <unknown>
paseamos        VLfin   <unknown>
por             PREP    por
el              ART     el
clasico         NC      <unknown>
Barrio          NP      <unknown>
Rojo            VLfin   <unknown>
donde           ADV     donde
descubri        NC      <unknown>
que             CQUE    que
tengo           VLfin   tener
un              ART     un
negocio         NC      negocio
de              PREP    de
ropa            NC      ropa
interior        ADJ     interior
que             CQUE    que
```

```
retrata         VLfin   retratar
a               PREP    a
la              ART     el
perfeccion      VLfin   <unknown>
mi              PPO     m
personalidad    NC      personalidad
```

El texto de entrada ya está preprocesado, habiéndose eliminado los acentos. La columna del centro es la categoría de POS extraída y la columna de la derecha es el resultado del stemming. Cuando la palabra no es conocida por TreeTagger la salida del stemming es *<unknown>*. En este caso la palabra original se toma como resultado del stemming. Finalmente, para poder utilizar el resultado en la etapa siguiente, se pone énfasis en que cada par palabra-POS es un término distinto, que será utilizado. Para esto a cada palabra resultado del stemming se le agrega como sufijo su POS, siendo cada código de POS distinto. Como ejemplo, el resultado para el ejemplo sería:

```
InclusoADJ paseamosVLfin porPREP elART clasicoNC BarrioNP RojoVLfin dondeADV
descubriNC queCQUE tenerVLfin unART negocioNC dePREP ropaNC interiorADJ
queCQUE retratarVLfin aPREP elART perfeccionVLfin mPPO personalidadNC
```

3.3.3. Etiquetado

Esta etapa toma la salida de la parte anterior como entrada y genera un etiquetado para cada par palabra-POS, indicando si es nombre de persona o no.

Como implementación para la etiquetación de secuencias basada en CRF se utilizó el paquete CRF desarrollado por Sunita Sarawagi del ITT Bombay. Esta implementación es eficiente ya que se basa en las operaciones matriciales explicadas en el capítulo anterior y en el entrenamiento utiliza el algoritmo de optimización Quasi-Newton LBFGS. Este algoritmo es presentado en detalle en [3]. El código está escrito en java, y sigue una filosofía de orientación a objetos que lo hace extensible, permitiendo agregar nuevas características y nuevas funcionalidades al programa. La implementación se basa fundamentalmente en las descripciones de [2] y [6].

Al etiquetar, el programa genera una salida igual a la entrada, pero cada vez que ocurre un cambio de etiqueta presenta la etiqueta para todas las palabras anteriores a ese momento e introduce un salto de línea. Por ejemplo:

```
elART alcaldeNC dePREP SantiagoNP donNC RaulNP AlcainoNP noNEG seSE
presentarVLfin aPREP elART reeleccionNC

elART alcaldeNC dePREP SantiagoNP donNC |1
RaulNP AlcainoNP |2
noNEG seSE presentarVLfin aPREP elART reeleccionNC |1
```

En el texto de salida, el número del final indica la etiqueta de las palabras que lo preceden. 1 indica que es *no persona*, y 2 indica que es *persona*.

3.3.4. Generación de lista de tags

A partir de la salida del etiquetado se generan los nombres que se utilizarán como tags.

Algunos nombres pueden aparecer formados por varias palabras, por ejemplo *Ariel Arenas*, que son dos palabras. Sin embargo, como se eliminó la puntuación en el preprocesado, también pueden aparecer dos nombres que originalmente estaban separados por "," juntos, como por ejemplo *Daniel Macarena*. Como no es posible saber a priori cuál de estos casos es el que se presenta, para cada secuencia de palabras etiquetadas como nombre, se agrega como tag cada subsecuencia de palabras posible. Así, si se etiqueta la secuencia "Alberto Antonio Larrain" como nombre, se agregarán los tags *Alberto*, *Antonio*, *Larrain*, *Alberto Antonio*, *Antonio Larrain* y *Alberto Antonio Larrain*.

Finalmente, la unión de todas las subsecuencias de palabras para cada secuencia etiquetada como nombre propio se revisa para que no haya tags repetidos. Así, un tag se reporta sólo una vez para el documento, aunque aparezca varias veces en el texto. La lista resultante de tags es el resultado completo del módulo de extracción de nombres.

3.4. Trabajo realizado

Además del diseño completo presentado en este capítulo, se implementaron todos los módulos. Cada uno de estos se logró como un programa en lenguaje PHP distinto. La elección de este lenguaje se basó en la necesidad de ser utilizados posteriormente como parte de un servicio web, además de la flexibilidad que presenta PHP para interactuar con las entradas y salidas de los distintos componentes de la aplicación completa, incluyendo el paquete CRF y TreeTagger.

En cuanto al módulo de extracción de características implementado, el paquete CRF tuvo que ser modificado levemente, en especial para agregar las nuevas características de POS. TreeTagger se modificó en la parte del preprocesado y aplicación de stemming, para cumplir con los requerimientos de la extracción de nombres. El modelo CRF fue entrenado utilizando un conjunto de documentos de de los cuales se extrajeron nombres manualmente. Para entender en detalle este proceso se recomienda ver el capítulo siguiente.

Una vez construida y entrenada la aplicación se midió cómo afectan distintos factores en la efectividad de ésta. Finalmente, se compararon resultados obtenidos a nivel general utilizando la aplicación construida para generar nubes de tags para distintos conjuntos de documentos. Si bien se cuenta con el diseño para un servicio web, no se contruyó la interfaz web para la aplicación, por lo que no se puede hablar de un servicio web como tal.

Capítulo 4

Entrenamiento del sistema

4.1. RSS

Entre todos los ítemes posibles que se pueden tratar en el contexto de este trabajo, un grupo importante son las noticias. Gran cantidad de noticias se publican diariamente a través de Internet en los medios de prensa nacionales e internacionales. Además, muchos usuarios generan artículos con opiniones de diversos temas, a través de blogs. Esta información suele ser de caracter más relevante en el momento de publicación e ir disminuyendo con el tiempo.

Los documentos de este tipo suelen ser publicados a través de un feed en formato RSS[1]. El RSS consiste básicamente en una especificación para un formato de documentos electrónicos escritos según una sintaxis XML. Un usuario de contenido RSS se suscribe a un feed a través de un programa lector de feeds. En adelante, este programa puede revisar si hay algún contenido nuevo desde la última vez que se revisó. Si es así, obtiene el contenido y lo muestra al usuario. Se ve que la tendencia actual es a publicar siguiendo el formato RSS.

4.2. orbitando.com

orbitando.com es un sitio web desarrollado a principios del año 2006, el cual nació como un simple recolector de contenidos sindicados. Estos contenidos tenían la particularidad de ser contenidos creados en Chile, o bien, que hacían referencia a Chile: noticias, comentarios en blog, podcast, fotografías, videos. Inicialmente, se agregaron canales de manera explícita (principalmente los resultados de búsquedas predefinidas en sitios de noticias), y después,

[1]El significado de la sigla RSS puede ser distinto, dependiendo de la versión del formato al que se refiere. Hay tres significados principales:

- Really Simple Syndication (2.0)
- Rich Site Summary (0.91, 1.0)
- RDF Site Summary (0.9, 1.0)

se permitió a los usuarios agregar sus propios canales, como por ejemplo blogs. De esta manera, orbitando fue acumulando canales, y por lo tanto, acumulando una gran cantidad de artículos, los cuales son almacenados en una base de datos descrita a continuación.

Cada canal al cual está suscrito el sistema, se encuentra almacenado en una tabla CHANNELS, que contiene los datos de cada uno de los feeds(canales) que alimentan al sistema. Con respecto a los artículos, cada nuevo artículo RSS que llega al servidor es almacenado como un registro en la tabla ITEMS. Este ítem tiene asociados un conjunto de atributos, correspondientes a las partes que lo componen: titular, descripción, fecha de publicación, fecha de recepción, además de la referencia a su canal de origen, entre otros datos. A continuación, el sistema realiza algunas labores de extracción de información, o pre-procesado, tales como la extracción de términos, y la clasificación del artículo en categorías (que en el caso general, se obtiene simplemente a partir del canal de origen). El resultado de este paso, es un conjunto de metadatos, que son almacenados en la tabla ITEM_METADATA. Un breve esquema de la base de datos es el presentado en la Figura 4.1.

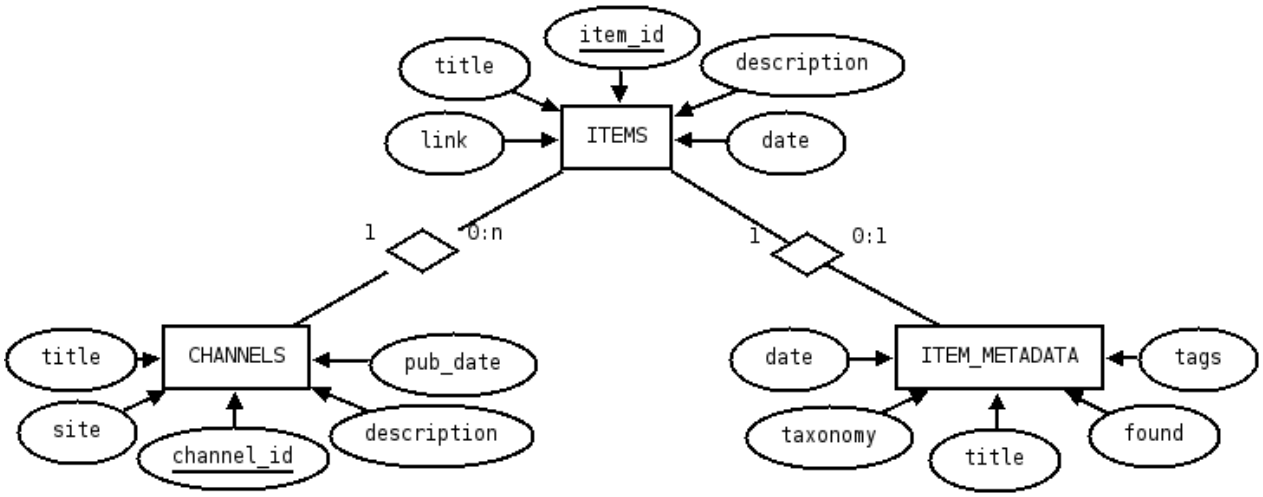

Figura 4.1: Breve esquema de la base de datos de orbitando.com

La base de datos original cuenta con una cantidad que asciende a cerca de 2.000.000 de artículos, hasta el mes de mayo de 2007. Esta es la cifra que arroja el conteo simple de entradas en la tabla ITEMS, la cual almacena los artículos existentes. Sin embargo, se observa que en esta tabla existe una gran cantidad de artículos repetidos, debido a errores producidos en una de sus principales fuentes: Google News. Se decidió tomar una muestra de la base de datos, recogiendo los artículos correspondientes a tres meses: enero, febrero y marzo de 2007. Esta muestra sería la utilizada para desarrollar el proyecto, con lo que finalmente, el universo de artículos con que se trabajó fue de 154.861, correspondientes a los artículos válidos de los tres meses mencionados. En la Figura 4.2 se puede apreciar la distribución temporal de los artículos, evidenciando una leve tendencia creciente, producto de la constante adición de nuevos canales al sistema.

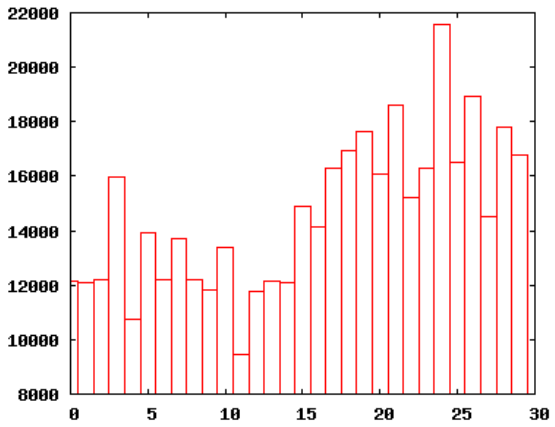

Figura 4.2: Cantidad de artículos cada tres días por un período de tres meses

En la actualidad, los canales que alimentan al sistema ascienden a más de 2.500 feeds. La principal fuente de artículos corresponde al conjunto de medios de prensa en línea, no sólo aquellos que publican contenidos sindicados, sino casi cualquier medio de prensa en internet que publique informaciones sobre Chile, gracias a los resultados recogidos por sitios como Google News, Topix.net, o MSN Search News, cuyas búsquedas pueden ser recogidas en formato RSS. Otras fuentes importantes de artículos son algunos medios de prensa nacional, tales como La Tercera online, y Emol.com, además de las búsquedas que arroja Flickr asociadas al tag Chile. La Tabla 4.1 muestra los canales más populares en la colección, es decir, los canales que aportan con la mayor cantidad de artículos.

Canal	Cantidad de artículos asociados
Flickr : chile - Everyone's Tagged Photos	36.462
La Tercera : Noticias del día	7.955
Topix : Search for "chile"	6.674
Google Noticias : tag chile	3.633
MSN Search News : Live Search News: chile	2.984
LA TERCERA	2.958
Google News Chile - Chile	2.479
Yahoo pipes: Chile News	2.277
EIN News: Chile News	1.785
Google News : chile	1.721

Tabla 4.1: Canales con más artículos en orbitando.com en el período enero - marzo 2007

Este trabajo está orientado a reconocer nombres de personas, en especial los que son más relevantes para el entendimiento del conjunto completo de documentos, los más frecuentes.

Por esta razón, los artículos provenientes de fuentes de fotografías, como *flickr* y *fotolog* no son relevantes, pues pueden hacer que el nombre de una persona en particular tenga mucho peso, al tener muchas fotos con su nombre como descripción, pero sin ser relevante su nombre en el contexto de todos los documentos. Para el proyecto no se consideraron los artículos provenientes de este tipo de fuentes. De la misma manera, se busca reconocer nombres en textos en español, por lo que no se consideran los artículos que han sido ingresados en la categoría *english* en el campo taxonomy. Parte de este primer filtro de la base de datos fue basado en el trabajo presentado en [1].

4.3. Conjunto de entrenamiento y prueba

Para entrenar el sistema automático etiquetador de nombres es necesario contruir un conjunto de textos de entrenamiento y prueba. Este conjunto debe estar etiquetado con los resultados esperados del sistema. Al ser los documentos provenientes de noticias y blogs en español, resulta fácil para un ser humano encontrar los nombres de personas dentro del texto, y diferenciarlos por ejemplo de nombres de instituciones o lugares. Se supone entonces a una persona como el mejor etiquetador, y se espera que el sistema automático se comporte de la misma manera. Así, el método que se utilizó para construir el conjunto de entrenamiento y prueba fue clasificar manualmente algunos documentos del total. Para esto, se contruyó una interfaz web que permite etiquetar marcando palabras dentro de los documentos anteriormente mencionados como personas. Dicha interfaz se muestra en la Figura 4.3. En ella se encuentran marcados los nombres *Chris Marker* y *Frederic Rossi*. Es posible notar que la puntuación y acentuación ya ha sido eliminada del texto antes de la etiquetación manual. Si bien esto dificulta la tarea al realizarla a mano, logra que las palabras que son clasificadas como nombres de personas sean efectivamente las mismas que le llegarían como entrada al módulo de extracción de nombres. Además, si no se realizara esto, podría haber errores como por ejemplo dos palabras que son mostradas como una sola pues están unidas por algún tipo de puntuación. Al etiquetar, quizás sólo una de ellas es un nombre de persona y no las dos. Esta interfaz se accedía a través de la url *http://alumnos.cadcc.cl/~cserpell/classify.php* y se mantuvo activa hasta el momento de entrega de este texto.

La manera de elegir qué documentos incluir en el conjunto de entrenamiento y prueba fue completamente al azar, dentro de los que pasaron el filtro explicado anteriormente. Los documentos son muy heterogéneos, por lo que el elegirlos al azar busca no sesgar los resultados hacia un tipo en particular de documentos. En total se extrajeron los nombres de personas de 400 documentos. El total de tags extraídos es de 3.878, siendo 2.113 de ellos distintos entre sí. En la Tabla 4.2 se muestran los nombres más frecuentes dentro de este conjunto.

Figura 4.3: Interfaz utilizada para clasificar manualmente

Nombre	Cantidad de artículos asociados
bachelet	32
michelle	21
michelle bachelet	19
carlos	18
jose	18
juan	17
gonzalez	13
antonio	12
luis	12
miguel	12
jorge	11
bush	10

Tabla 4.2: Nombres más frecuentes extraídos manualmente

Capítulo 5

Resultados

Este capítulo ha sido separado en dos secciones. En la primera de ellas se muestran los resultados y discusiones concernientes sólo al módulo de etiquetación automática de nombres. En la segunda sección se presenta un análisis de los resultados del sistema visto de manera amplia, en cuanto al objetivo de generar una nube de tags útil para el acceso a los documentos asociados a personas.

5.1. Etiquetador automático de nombres

El objetivo de esta sección es mostrar los distintos factores que están involucrados en el desempeño del sistema, y cómo afectan en los resultados.

Para medir la calidad de un reconocedor de entidades propias se puede comparar el comportamiento de un sistema N_1 con otro sistema N_2 utilizando un conjunto de prueba. Se definen los conceptos de *precisión* (Pre) y *recuperación* (Rec) para cada categoría $\lambda \in \Lambda$:

$$Pre\,(\lambda, N_1, N_2) = \frac{|N_1^i = N_2^i = \lambda|}{|N_1^i = \lambda|}$$

$$Rec\,(\lambda, N_1, N_2) = \frac{|N_1^i = N_2^i = \lambda|}{|N_2^i = \lambda|}$$

La precisión representa cuántas palabras de las que se eligieron como entidades del tipo λ fueron categorizadas igual que N_2. La recuperación representa cuántas palabras de las que N_2 categoriza en λ, son también categorizadas en λ por N_1. Estas dos medidas se utilizan en general comparando un sistema N con una clasificación realizada manualmente como conjunto de prueba. Esto supone que la clasificación manual es equivalente a la realizada por un clasificador perfecto. Una medida que incluye a las dos medidas previas con igual ponderación cada una es la siguiente [9]:

$$F = \frac{2 * Pre * Rec}{(Pre + Rec)}$$

Para entrenar el sistema, se decidió separar el conjunto de entrenamiento y prueba en dos: Un conjunto de entrenamiento con un 80 % de los documentos y otro con un 20 % como conjunto de prueba. Aunque sería posible hacer varias separaciones distintas del mismo conjunto para validar el sistema, el tiempo de entrenamiento es muy alto, por lo que se utilizó siempre la misma separación. La metodología utilizada para estudiar el comportamiento del sistema es medir la precisión y recuperación para las dos categorías posibles, que son *persona* y *no persona*. Se busca maximizar ambos valores, si bien generalmente no es fácil lograr que los dos sean altos a la vez. El total de palabras existentes en el conjunto de entrenamiento y prueba son 53.102, y de éstas 2.353 fueron etiquetadas como *persona*, lo que representa un 4,43 % del total.

A fin de comparar los resultados obtenidos por el sistema construido, se muestran en la Tabla 5.1 los valores de precisión y recuperación promedio obtenidos por clasificadores básicos o *tontos*. El primero clasifica cada palabra lanzando una moneda, con la misma probabilidad de entregar la etiqueta persona o no pesona para cada palabra. El segundo es un poco mejor, ya que en vez de ser la misma probabilidad para ambas categorías, esta probabilidad es el porcentaje de personas que hay en el total de documentos. Es decir etiqueta una palabra como persona con un 4,43 % de probabilidad. Éste resulta ser mucho mejor para la categoría no persona.

Clasificador	Equiprobable			Persona menos probable		
Categoría	Pre	Rec	F	Pre	Rec	F
No persona	95,38	49,58	65,25	95,60	95,80	95,70
Persona	4,44	48,27	8,13	5,23	5,00	5,11
Total	49,52	49,52	49,52	91,77	91,77	91,77

Tabla 5.1: Resultados obtenidos para clasificadores tontos

Existen cuatro características fundamentales del proceso de entrenamiento que pueden modificarse buscando mejorar la precisión y la recuperación:

- Cantidad de iteraciones del algoritmo LBFGS. Al maximizar la verosimilitud del modelo, el algoritmo realiza un proceso iterativo de aproximación de los parámetros del modelo.
- Utilización de stemming. El proceso de stemming presentado en el capítulo 3 es posible utilizarlo en distintas etapas del sistema.
- Utilización de características de POS extraídas por TreeTagger.
- Cantidad de documentos en el conjunto de entrenamiento y prueba.

5.1.1. Cantidad de iteraciones

En la Tabla 5.2 se puede observar que el aumentar la cantidad de iteraciones utilizadas mejora el comportamiento general del sistema. También se comprobó que agregar más iteraciones pasado un cierto umbral no mejora el desempeño del modelo. Para este experimento no fueron consideradas las características de POS. El proceso de stemming se realizó en el conjunto de entrenamiento y prueba. Considerando este resultado, se entrenó generalmente utilizando la mayor cantidad de iteraciones posibles.

Iteraciones	100			500 o más		
Categoría	Pre	Rec	F	Pre	Rec	F
No persona	97,28	99,83	98,54	97,55	99,89	98,71
Persona	89,77	34,27	49,61	92,81	35,83	51,70
Total	97,17	97,17	97,17	97,48	97,48	97,48

Tabla 5.2: Resultados al variar cantidad de iteraciones en algoritmo de optimización

5.1.2. Stemming

En cuanto a stemming, en la Tabla 5.3 se muestra un resumen de los resultados obtenidos utilizando stemming en el entrenamiento y en la entrada del etiquetador, sólo en el entrenamiento, o simplemente no utilizándolo. Al igual que al estudiar la cantidad de iteraciones, no fueron consideradas aquí las características asociadas a POS. La cantidad de iteraciones utilizada fue 100. Es posible notar que agregar stemming tanto en el conjunto de entrenamiento como en el conjunto de prueba mejora el desempeño del sistema, por lo que fue utilizado en la versión final.

Stemming	NO			Entrenamiento			Entrenamiento y prueba		
Categoría	Pre	Rec	F	Pre	Rec	F	Pre	Rec	F
No persona	97,22	99.84	98,52	97,27	99.81	98,52	97,28	99.83	98,54
Persona	89,88	32,75	48,01	88,20	34,06	49,14	89,77	34,27	49,61
Total	97,11	97,11	97,11	97,13	97,13	97,13	97,17	97,17	97,17

Tabla 5.3: Resultados al cambiar la utilización de stemming

5.1.3. POS

Como era esperado, agregar al etiquetador el uso de funciones características que toman en cuenta los POS generados por TreeTagger mejoró de gran manera el desempeño del sistema. En la Tabla 5.4 se muestran los resultados al usar o no dichas características, indicando que el uso de POS es fundamental para el etiquetado.

	Sin POS			Con POS		
Categoría	Pre	Rec	F	Prec	Rec	F
No persona	97,38	99,69	98,52	98,16	99,54	98,84
Persona	86,86	43,16	57,66	86,19	60,42	71,04
Total	97,14	97,14	97,14	97,78	97,78	97,78

Tabla 5.4: Resultados al agregar características de POS

5.1.4. Cantidad de documentos de entrenamiento

Los resultados presentados en la Tabla 5.5 indican que aumentar la cantidad de documentos utilizados en el entrenamiento mejora el desempeño del sistema. Sin embargo, agregar más documentos al conjunto de entrenamiento aumenta considerablemente el tiempo necesario para entrenar el sistema. En estos experimentos se tomaron en cuenta las características de POS.

Documentos	296			320		
Categoría	Pre	Rec	F	Pre	Rec	F
No persona	98,16	99,54	98,84	98,27	99,58	98,92
Persona	86,19	60,42	71,04	87,33	62,31	72,73
Total	97,78	97,78	97,78	97,93	97,93	97,93

Tabla 5.5: Resultados al agregar más documentos al conjunto de entrenamiento

El etiquetado con CRF presenta entonces dos características muy importantes en cuanto a su precisión y recuperación. La precisión es alta para la categoría *persona*, lo que significa que prácticamente todas las palabras que son marcadas como nombre de persona, efectivamente lo son. Se observa que la recuperación lograda no es tan alta como la precisión, lo que significa que habrá apariciones de nombres de personas que no serán detectados por el sistema. Esto puede parecer un mal resultado al principio, pero el objetivo de la aplicación es extraer los nombres de personas, en especial los que son más frecuentes. Así, un nombre frecuente puede no ser observado en algunas de sus apariciones, pero al ser éstas muchas en el conjunto de documentos, será detectado suficientes veces para ser considerado.

5.2. Visión amplia del sistema

Si bien medir la precisión y recuperación del etiquetador ayuda a mejorar el proceso de extraer los nombres, no se debe olvidar que el objetivo de la aplicación tipo servicio web es extraer los nombres de personas, de manera de poder construir una interfaz que haga más fácil el acceso a los documentos. Para eso es necesario estudiar los nombres extraídos y si

estos resultan útiles para dicho objetivo.

La Tabla 5.6 tiene como objetivo mostrar la diferencia entre los tags más frecuentes extraídos manualmente y por la aplicación. El porcentaje presentado es el porcentaje del total de documentos en que aparece cada palabra. En ella no se han eliminado los nombres formados sólo por nombres comunes. Claramente los resultados son muy similares, ya que el sistema fue entrenado con este mismo conjunto, de manera que la verosimilitud entre la etiquetación manual y la automática sea máxima. Se aprecia que los nombres aparecen con diferencias porcentuales muy bajas, y que, aunque en distinto orden, los nombres extraídos son casi los mismos. También puede notarse el efecto que tiene la recuperación sobre los resultados obtenidos: En general el porcentaje de apariciones de un nombre según la aplicación es menor que al considerar la etiquetación manual, ya que hay apariciones que no son detectadas. Sin embargo, esto sucede en general para todos los tags, por lo que no afecta qué nombres son los más frecuentes.

Manualmente		Aplicación	
Tag	*%*	*Tag*	*%*
bachelet	8,00	bachelet	8,00
michelle bachelet	4,75	michelle bachelet	4,50
gonzalez	3,25	gonzalez	3,00
bush	2,50	bush	2,50
chavez	2,00	chavez	2,00
morales	2,00	correa	1,75
evo	1,75	evo	1,75
evo morales	1,75	evo morales	1,75
fernandez	1,75	morales	1,75
rojas	1,75	rojas	1,75
calderon	1,50	arenas	1,50
chile	1,50	diaz	1,50
correa	1,50	calderon	1,25
diaz	1,50	fernandez	1,25
fujimori	1,50	fujimori	1,25

Tabla 5.6: Documentos asociados a nombres más frecuentes para conjunto de entrenamiento

En las Tablas 5.7 y 5.8 se muestran los nombres más frecuentes extraídos de distintos conjuntos de documentos antes de filtrarlos con el filtro que modifica las frecuencias mostradas explicado en el capítulo 3, junto con el porcentaje del total de documentos del conjunto en el que se encontraron. Los resultados filtrados se muestran a continuación. El primero es el conjunto completo de orbitando.com explicado en el capítulo anterior. Luego se utilizó la aplicación sobre un conjunto de artículos de noticias de la agencia de noticias española EFE, de mayo de 2000, con aproximandamente 8.000 artículos, y finalmente en todas las páginas

del sitio *plataformaurbana.cl*, a octubre de 2007, con aproximadamente 1.700 páginas. La manera de obtener este último conjunto de documentos fue bajar el sitio completo con el programa *wget*.

orbitando.com				*Agencia EFE*			
Con nom. com.		Sin nom. com.		Con nom. com.		Sin nom. com.	
Tag	*%*	*Tag*	*%*	*Tag*	*%*	*Tag*	*%*
bachelet	1,34	bachelet	1,34	jose	1,89	fernandez	0,69
juan	0,98	pinochet	0,91	juan	1,36	gonzalez	0,62
michelle	0,95	barnes	0,83	maria	1,22	lopez	0,43
jose	0,90	michelle bachelet	0,80	carlos	1,05	alvarez	0,39
pinochet	0,91	chavez	0,76	manuel	0,98	sanchez	0,37
barnes	0,83	ken barnes	0,75	luis	0,93	rodriguez	0,32
fernando	0,83	arenas	0,57	miguel	0,87	villanueva	0,30
maria	0,81	flores	0,55	francisco	0,80	martinez	0,29
michelle bachelet	0,80	castro	0,52	fernandez	0,69	aznar	0,28
ken	0,78	bush	0,50	antonio	0,65	rey	0,26
chavez	0,76	gonzalez	0,49	gonzalez	0,62	don	0,26
jorge	0,76	punta	0,46	fernando	0,51	jose maria aznar	0,26

Tabla 5.7: Tags más frecuentes extraídos con y sin nombres comunes (nom. com.)

En el caso de plataformaurbana.cl, es posible notar que hay nombres muy frecuentes, en comparación con los otros dos conjuntos. El primer factor que influye en esto es la cantidad de artículos. Al ser menor, la misma cantidad de apariciones de un nombre significa un porcentaje mucho más alto del total. Otro factor que influye es que el sitio tiene pocos usuarios que suben contenido, y cada página posee al autor dentro de su texto. Estos autores se presentan como muy frecuentes dentro de los nombres extraídos. Además, el tema del que trata el sitio es sobre urbanismo en general y en Chile. Muchos de los actores relevantes de esta esfera en Chile aparecen frecuentemente mencionados en los artículos. Finalmente, el contenido de las páginas fue entregado a la aplicación tal cual fue leído desde internet, teniendo mucho texto que no es relevante para la generación de tags, y generalmente muy repetido entre todas ellas, como parte de la interfaz, además de publicidad y vínculos a otras páginas. Una posible solución para esto es que un posible usuario del servicio web podría seleccionar qué parte del contenido de cada página entregar para la extracción de nombres. Esto no fue realizado ya que sería un trabajo específico para cada sitio.

En el lado derecho de las tablas se puede observar cómo el utilizar el filtro de nombres comunes hace aparecer en los nombres más frecuentes nombres de personas particulares y fácilmente distinguibles. Aún así, se aprecian muchos apellidos muy comunes que no distinguen personas particulares.

Con nombres comunes		Sin nombres comunes	
Tag	*%*	*Tag*	*%*
diego	58,74	diego portales	58,74
diego portales	58,74	portales	58,74
portales	58,74	made	58,62
made	58,62	paris made	58,62
paris	58,62	diego portales diplomado	58,50
paris made	58,62	diplomado	58,50
julio	58,56	portales diplomado	58,50
diego portales diplomado	58,50	jose llano	58,39
diplomado	58,50	llano	58,39
jose	58,50	sergio reyes	58,39
portales diplomado	58,50	salvad	46,20
jose llano	58,39	salvad veronica	46,20
llano	58,39	salvad veronica soza	46,20
reyes	58,39	soza	46,20
sergio	58,39	veronica soza	46,20

Tabla 5.8: Tags más frecuentes extraídos automáticamente para el sitio plataformaurbana.cl

Es posible modificar la lista de nombres generada para un sitio, dejando sólo los nombres que están formados por dos o más palabras. De esta manera una nube de tags tendría sólo nombres de personas particulares, y no apellidos compartidos por diferentes personas. En la Tabla 5.9 se pueden apreciar los resultados para los tres conjuntos estudiados, analizados a continuación.

Los resultados son bastante interesantes. En el caso de orbitando.com, una primera observación indica que tomar sólo nombres de dos o más palabras hace que aparezcan muchos más nombres particulares como los nombres más relevantes, lo que indica que es una buena idea. Al observar cómo varía el conjunto más frecuente para las noticias de EFE, no queda claro que haya habido una mejora. Es posible ver que se pierden varios apellidos que sí podían ser nexos a documentos con apariciones de alguna persona relevante, además de aparecer algunos nombres repetidos por formarse de tres palabras. Al notar la pérdida de los nombres dados por solo un apellido, se puede volver la atención al primer conjunto de documentos y notar que, a excepción de *saddam hussein*, todos los documentos donde aparecen las demás personas serían referenciados sólo por el apellido. El caso del sitio plataformaurbana.cl no es del todo claro, porque si bien las personas quedan más especificadas, no aparecen nuevos nombres.

En la Tabla 5.10 se puede apreciar lo importante que es aplicar el filtro que elimina algunos tags de la lista y modifica la frecuencia de los mismos, al consultar los nombres más

orbitando.com		Agencia EFE		plataformaurbana.cl	
Tag	%	*Tag*	%	*Tag*	%
michelle bachelet	0,80	jose maria aznar	0,26	diego portales	58,74
ken barnes	0,75	maria aznar	0,26	paris made	58,62
punta arenas	0,44	don juan	0,14	diego portales diplomado	58,50
augusto pinochet	0,43	don juan carlos	0,12	portales diplomado	58,50
hugo chavez	0,42	jean pierre	0,12	jose llano	58,39
fidel castro	0,32	jean pierre chevenement	0,12	sergio reyes	58,39
gabriela mistral	0,26	pierre chevenement	0,12	salvad veronica	46,20
saddam hussein	0,25	vazquez rana	0,12	salvad veronica soza	46,20
george w	0,24	alberto fernandez	0,11	veronica soza	46,20
w bush	0,24	vicente alvarez	0,11	construcc marcelo	43,65
george w bush	0,24	jaime mayor	0,10	address your	19,39
ricardo lagos	0,22	jaime mayor oreja	0,10	address your name	19,39

Tabla 5.9: Nombres de 2 o más palabras para distintos conjuntos de documentos

comunes en estos tres distintos sitios. Para este primer resultado, se tomó como criterio de eliminación, que existiera otro tag más grande cuya frecuencia de aparición fuera al menos un 15 % del primero.

Para los dos primeros resulta un éxito, ya que queda una lista con nombres que efectivamente resultan relevantes para el conjunto de documentos. Hay algunos tags que no fue posible eliminar, como *chilean* o *punta arenas*. Lamentablemente este problema proviene del etiquetado y no de los filtros que se puedan aplicar al consultar los nombres más frecuentes, ya que podrían suceder muchos casos imprevistos y fuera del alcance de dichos filtros. El resultado para el sitio plataformaurbana.cl no fue satisfactorio, por lo que es posible concluir que para este sistema no es posible entregar directamente las páginas completas como contenido de los documentos, sino que hay que extraer la parte que es texto relevante para la extracción de los nombres. Considerando esto último, se ejecutó la aplicación sobre otros dos nuevos conjuntos de documentos extraídos de blogs con *wget*, esta vez cuidando que el contenido de los documentos recuperado para la extracción de nombres sea el texto relevante. Estos sitios son *atinachile.cl*[1], con 1.239 artículos de temas diversos, y *elobservatodo.cl*[2], con

[1]Atina Chile se autodefine como un movimiento ciudadano cuyos principales propósitos son abrir un espacio de participación ciudadana activa en torno a la construcción del país, ser un agente de visión y cohesión de aquellas conversaciones necesarias y ausentes de la agenda nacional, y un movilizador como educador de las oportunidades que traen las nuevas tecnologías a la sociedad.

[2]El Observatodo se autodefine como un diario digital, que se enmarca dentro de la participación ciudadana en la creación de información. En él, sus reporteros escriben bajo una mirada participativa la noticia, sin tomar el papel de espectador objetivo que narra desde fuera una situación observada, sino un rol activo como testigo de un hecho. Se persigue instalar un estilo de periódico auténticamente participativo, donde la historia de una localidad sea escrita por muchos y no por pocos.

orbitando.com		Agencia EFE		plataformaurbana.cl	
Tag	*%*	*Tag*	*%*	*Tag*	*%*
ken barnes	0,83	alberto fernandez	0,69	paris made	58,62
michelle bachelet	0,80	gonzalez	0,62	jose llano	58,39
hugo chavez	0,76	lopez	0,43	construcc marcelo	43,65
augusto pinochet	0,68	sanchez	0,37	salvad veronica soza	30,80
punta arenas	0,57	villanueva	0,30	diego portales diplomado	29,25
fernando flores	0,55	geoffrey parker	0,26	it	19,44
fernando gonzalez	0,49	moreno	0,25	web stumbleupon	19,39
saddam hussein	0,35	pp carlos	0,24	patricio pone arqueros	19,39
steve jobs	0,34	hugo chavez	0,24	archivo	7,72
peter	0,33	mercado	0,17	junio	6,44
jorge schaulsohn	0,31	flores	0,15	mayo	6,21
evo morales	0,31	villarreal	0,15	michelle bachelet	5,57
rodriguez	0,30	guerrero	0,14	mercado	5,28
rafael correa	0,27	jose maria aznar	0,14	diciembre	4,70
chilean	0,27	manuel fraga	0,13	gabriela mistral	4,35
ii guerra	0,27	francisco munoz	0,12	noviembre	4,24
matias fernandez	0,27	perez	0,11	septiembre	4,24
gabriela mistral	0,27	don juan carlos	0,11	saludos	4,06
george w bush	0,25	ruiz polanco	0,10	febrero	3,83
jennifer lopez	0,25	vega	0,10	ricardo lagos	3,42
alvaro rojas	0,24	pinochet	0,10	salvador allende	2,96

Tabla 5.10: 21 tags más frecuentes aplicando filtro final

3.992 artículos mayormente relacionados con la cuarta región. Los artículos fueron recuperados a principios de noviembre de 2007.

Para terminar de ajustar la aplicación, es necesario definir el valor de los parámteros θ y μ definidos como umbrales para el filtro que elimina tags poco útiles. El valor de θ ha sido elegido empíricamente y se comprobó que los resultado son muy sensibles a su valor. Un valor muy grande hace que prácticamente no se eliminen tags y un valor muy pequeño hace que los tags se eliminen y queden sólo nombres muy largos, incluyendo palabras anexas al verdadero nombre y no aportando información relevante. Finalmente, se tomó el valor de $\theta = 15\,\%$, es decir que exista un tag más grande cuya frecuencia de aparición sea al menos un $15\,\%$ de la frecuencia del tag a ser eliminado.

Para elegir μ, podemos ver cómo varían los resultados para un conjunto tomando tres valores distintos en la Tabla 5.11. Es importante notar que un valor pequeño de μ no es eficaz si θ también es pequeño, ya que por ejemplo el nombre *michelle bachelet* podría apa-

recer siempre acompañado de palabras como *presidente*, *presidenta*, *mandataria*, las cuales podrían aparecer marcadas como persona en algunas oportunidades. Al sumarse todas estas veces, se elegiría eliminar el tag original, repartiendo su frecuencia de aparición en los demás tags más grandes. Así el tag se convertiría en una serie de tags con baja frecuencia en vez de un solo nombre, como es deseable. El resultado para $\mu = 0$ es el mismo que se tendría probando valores hasta un 15 %, ya que la condición dada por θ es más restrictiva en ese caso.

$\mu = 0$		$\mu = 25\,\%$		$\mu = 35\,\%$	
Tag	*%*	*Tag*	*%*	*Tag*	*%*
michelle bachelet	1,59	michelle bachelet	1,59	michelle bachelet	1,59
biografia	1,29	biografia	1,29	biografia	1,29
fernando flores	1,25	fernando flores	1,25	fernando flores	1,25
camilo herrera	0,97	camilo herrera	0,97	camilo herrera	0,97
astronomico		astronomico		astronomico	
jorge ianiszewski	0,40	jorge ianiszewski	0,40	jorge ianiszewski	0,40
jorge dominguez	0,40	jorge dominguez	0,40	jorge dominguez	0,40
andes	0,32	paula rojo	0,40	paula rojo	0,40
mercado	0,32	andes	0,32	andes	0,32
carlos montes	0,24	mercado	0,32	mercado	0,32
chavez	0,24	pinochet	0,32	pinochet	0,32
gabriel coloane	0,24	carlos montes	0,24	carlos montes	0,24
leonardo farkas	0,24	chavez	0,24	chavez	0,24
mercator ignacio lopez	0,24	gabriel coloane	0,24	gabriel coloane	0,24
george bush	0,20	leonardo farkas	0,24	leonardo farkas	0,24
punta arenas	0,20	mercator ignacio lopez	0,24	mercator ignacio lopez	0,24
isabel allende	0,19	peter alexander	0,22	peter alexander	0,22
enrique meza	0,18	george bush	0,20	george bush	0,20
aldea	0,16	punta arenas	0,20	punta arenas	0,20
lily perez	0,16	isabel allende	0,19	isabel allende	0,19
marcel	0,16	enrique meza	0,18	enrique meza	0,18
marcos kulka	0,16	aldea	0,16	aldea	0,16
saint pierre	0,16	lily perez	0,16	lily perez	0,16
sara larrain	0,16	marcel	0,16	marcel	0,16
steve jobs	0,16	steve jobs	0,16	steve jobs	0,16

Tabla 5.11: Tags extraídos para atinachile.cl para distintos valores de μ.

Con estos resultados se puede observar que efectivamente tener un valor de μ más alto que θ hace que no desaparezcan nombres con alta frecuencia, como se explicó anteriormente. Un ejemplo es *paula rojo*, *pinochet* y *peter alexander*. Además, es posible observar que el variar μ a valores más grandes no afecta mayormente la lista de tags entregada. Para valo-

res aún más grandes los resultados ya no son efectivos, por lo que no han sido expuestos aquí.

atinachile.cl		elobservatodo.cl	
Tag	*%*	*Tag*	*%*
michelle bachelet	1,59	michelle bachelet	4,44
biografia	1,29	rojas	3,76
fernando flores	1,25	mauricio guerrero	3,38
camilo herrera	0,97	leido	2,36
astronomico jorge ianiszewski	0,40	jorge olivares	1,90
jorge dominguez	0,40	enrique mursell	1,60
paula rojo	0,40	gabriela mistral	1,44
andes	0,32	gloria delucchi	1,43
mercado	0,32	perez	1,33
pinochet	0,32	don	1,30
carlos montes	0,24	colegio	1,23
chavez	0,24	augusto pinochet	1,12
gabriel coloane	0,24	juan alfaro	1,07
leonardo farkas	0,24	jorge ordenes	1,05
mercator ignacio lopez	0,24	contreras	1,03
peter alexander	0,22	sebastian castillo	0,98
george bush	0,20	hugo chavez	0,88
punta arenas	0,20	jorge bujalil	0,85
isabel allende	0,19	pinera	0,85
enrique meza	0,18	salinas	0,83
aldea	0,16	francisco oviedo	0,80
lily perez	0,16	mercado	0,78
marcel	0,16	china	0,75
steve jobs	0,16	evo morales	0,70
walter oliva	0,16	rey	0,70

Tabla 5.12: Tags más frecuentes como resultado completo de la aplicación, con $\mu = 25\,\%$

Finalmente, en la Tabla 5.12 se muestra además el resultado para el sitio elobservatodo.cl, indicando que la aplicación genera una lista de nombres útil para entender y explorar los documentos. Es posible notar que las dos listas tienen nombres bastante diferentes, relevantes para cada conjunto de documentos. Esto indica que el sistema es efectivo para conjuntos de documentos de contenidos radicalmente distintos. En la Figura 5.1 se muestra la forma de distribución de las frecuencias de aparición de los tags más frecuentes.

Es necesario aclarar que el eliminar tags que están contenidos en otros no hace que los documentos asociados a ellos se pierdan en cuanto al objetivo de explorar los documentos. Al buscar los documentos asociados a un tag formado por dos o más palabras, puede re-

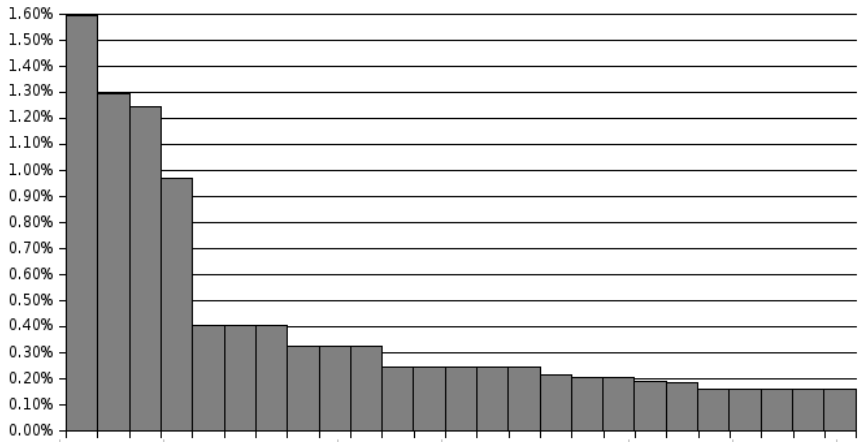

Figura 5.1: Distribución de frecuencia de aparición para tags en atinachile.cl

comendarse buscar también los documentos asociados a cada palabra por separado, con el fin de ampliar los resultados de la búsqueda. De la misma manera, si se elige un tag como *chavez*, sería posible elegir algún nombre específico que tenga este apellido, listando todos los tags que incluyen el tag más pequeño.

En el Apéndice A se incluye una lista con los nombres frecuentes que aparecen en la Tabla 5.12, junto con una breve explicación de cuál es su relevancia por la que podría aprecer en los resultados de las distintas páginas.

Capítulo 6

Conclusiones

6.1. Discusión

Se comprobó que la aplicación cumple el objetivo de extraer una lista de nombres relevantes para explorar los documentos, para conjuntos de tamaños y orígenes distintos. Por ejemplo, el conjunto de orbitando.com es mucho más grande que los demás. Sin embargo la lista de nombres generada no difiere en forma a la de los dos últimos blogs analizados, aun cuando el contenido de ellos sí es radicalmente distintos. Otra razón para concluir esto es el estilo de escritura de los artículos. Existe una gran cantidad de ruido proveniente de ortografía, además de existir diversas maneras distintas de redactar documentos en español, algunas de ellas muy distintas entre sí. La aplicación descrita logra extraer nombres efectivamente sin necesidad de tratar de manera especial estos casos, lo que comprueba el éxito logrado en esta materia.

Considerando la funcionalidad que tiene el servicio web diseñado en esta memoria, es importante destacar que aporta efectivamente a la web 2.0, pudiendo ser utilizado por distintos sitios, en especial blogs generados por usuarios generalmente anónimos. Sirve para construir interfaces más útiles para acceder a estos volúmenes de información que generalmente son de difícil navegación. De todas maneras, se comprobó que es necesario entregar como entrada a la aplicación sólo el texto relevante para la búsqueda de nombres y no las páginas completas. En efecto, al procesar las páginas provenientes del sitio plataformaurbana.cl no se tienen resultados concluyentes. Se puede mencionar que este servicio web puede ser utilizado no sólo para generar los términos categorizados, sino también para buscar dentro de términos o tags generados sin categorización, cuáles de ellos pertenecen a una categoría. Para esto, es posible comparar los términos generados por este sistema y los que se tenían anteriormente para un conjunto de documentos.

Durante el transcurso de este trabajo se profundizó en el conocimiento de herramientas de extracción de información, en particular campos aleatorios condicionales (CRF). El uso de esta herramienta necesitó su comprensión y adaptación al problema abordado, lográndose

resultados satisfactorios.

Además de la extracción de términos categorizados, fue necesario idear una heurística para modificar las frecuencias de éstos, de manera de generar una lista útil. Esta heurística le da un enfoque nuevo al problema de generación de nubes de tags, ya que no sólo se centra en la extracción de estos términos, sino en cómo ellos resultan relevantes para generar la nube.

Gran parte del sistema no toma en cuenta que la categoría de la que se extraen términos son nombres de personas. Esto significa que la aplicación puede ser reentrenada para otra nueva categoría, como por ejemplo lugares o marcas. De esta manera, la herramienta construida en esta memoria es lo suficientemente flexible, presentando un vasto potencial para ser aplicada en otro tipo de documentos y categorías.

6.2. Trabajo futuro

En este trabajo se implementó la parte funcional para un servicio web, en el lenguaje PHP. Sin embargo, aunque ha sido explicado todo el sistema como una interfaz vía web, sólo fue probado a nivel local, y no fue construido para poder correr realmente como servicio web. Un posible trabajo posterior sería llevar a cabo esta implementación final, dejando a disposición un sistema robusto que extraiga términos categorizados.

Si bien como se mencionó el sistema es flexible, pudiendo ser entrenado para otra categoría además de nombres de personas, se necesitaría un cuidado especial al realizar esto, ya que no se contaría con la lista de nombres comunes, y algunos parámetros o características podrían variar. Un trabajo que podría realizarse en el futuro es entrenar el sistema para otra categoría y medir su eficacia para generar nubes de tags para esta categoría.

En cuanto al postprocesado al que son sometidos los términos generados para entregar una lista útil, la elección de los parámetros μ y θ fue realizada empíricamente. Podría idearse un método para poder comparar cuantitativamente distintas nubes de tags generadas, y con ello poder elegir de manera más rigurosa y formal estos dos parámetros.

Bibliografía

[1] GÓMEZ MARTÍNEZ, D. 2007. *Diseño Editorial Social en Portales de Información Utilizando Técnicas de Minería de Datos*. Memoria para optar al título de ingeniero civil en computación, Universidad de Chile.

[2] LAFFERTY, J., MACCALLUM, A. PEREIRA, F. 2001. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. Proceedings of the Eighteenth International Conference on Machine Learning. Páginas 282 a 289.

[3] LIU, D. C., NOCEDAL, J. 1989. *On the Limited Memory BFGS Method for Large Scale Optimization*. Mathematical Programming B. Volumen 45. Páginas 503 a 528.

[4] RABINER, L. R. 1989. *A Tutorial on Hidden Markov Models and Selected Applications on Speech Recognition*. Proceedings of the IEEE. Páginas 257 a 286.

[5] SCHMID, H. 1994. *Probabilistic Part-of-Speech Tagging Using Decision Trees*. Proceedings of International Conference on New Methods in Language Processing.

[6] SHA, F., PEREIRA, F. 2003. *Shallow Parsing with Conditional Random Fields*. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology. Volumen 1. Páginas 134 a 141.

[7] SUTTON, C., MCCALLUM, A. 2006. *An Introduction to Conditional Random Fields for Relational Learning*. Introduction to Statistical Relational Learning. Edited by Lise Getoor and Ben Taskar. MIT Press.

[8] TJONG KIM SANG, E. F. 2002. *Introduction to the CONLL-2002 Shared Task: Language-Independent Named Entity Recognition*. Proceeding of the sixth conference on Natural language learning. Volumen 20. Páginas 1 a 4.

[9] TJONG KIM SANG, E. F., DE MEULDER, F. 2003. *Introduction to the CONLL-2003 Shared Task: Language-Independent Named Entity Recognition*. Proceedings of the seventh conference on Natural language learning at HLT-NAACL. Volumen 4. Páginas 142 a 147.

[10] WALLACH, H. M. 2004. *Conditional Random Fields: An Introduction*. Technical Report MS-CIS-04-21. Department of Computer and Information Science, University of Pennsylvania.

[11] ZHANG, L., PAN, Y., ZHANG, T. 2004. *Focused Named Entity Recognition using Machine Learning*. Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. Páginas 281 a 288.

Apéndices

C . Descripción de nombres frecuentes

Nombre y descripción
michelle bachelet Presidenta de la República de Chile, período 2006-2010
fernando flores Senador de la República de Chile, período 2002-2010
mauricio guerrero Periodista. Editor periodístico, corresponsal y director de El Observatodo
camilo herrera Gerente del Colegio Altamira y creador del portal Educandonos.cl
jorge ianiszewski Escritor y divulgador científico. Editor de Círculo Astronómico
jorge olivares Activo colaborador y corresponsal de El Observatodo
jorge dominguez Vicepresidente ejecutivo, coordinador de Atina Chile
enrique mursell Activo colaborador y corresponsal de El Observatodo
paula rojo Gerente general de la Fundación Mercator. Fundadora de Atina Chile
gabriela mistral Destacada poetisa, diplomática y profesora. Premio Nobel de Literatura 1945
gloria delucchi Abogado. Magister en comercio internacional. Representante de Atina Iquique
carlos montes Diputado socialista de la República de Chile, período 2006-2010
augusto pinochet Comandante en jefe, presidente y senador de la República de Chile
gabriel coloane Participante del 1er concurso internacional de poesía libre Atina Chile

Nombre y descripción
juan alfaro Corresponsal de El Observatodo
leonardo farkas Empresario minero, presidente de la minera Santa Bárbara
jorge ordenes Alcalde de Andacollo
ignacio lopez Director de proyectos, Fundación Mercator
peter alexander Participante del 1er concurso internacional de poesía libre Atina Chile
sebastian castillo Corresponsal de El Observatodo
george bush Presidente de Estados Unidos, período 2001-2009
hugo chavez Presidente de Venezuela, período 1999-2013
jorge bujalil Corresponsal de El Observatodo
isabel allende Escritora y dramaturga
enrique meza Empresario, fundador de ICCOM. Colaborador de Atina Chile
francisco oviedo Corresponsal de El Observatodo
lily perez Secretaria General de Renovación Nacional
steve jobs Empresario e informático de Estados Unidos, presidente de Apple Inc.
evo morales Presidente de Bolivia, desde 2006
walter oliva Sexto vicepresidente y tesorero de Democracia Cristiana

D . Lista de nombres comunes utilizada

aaron	abaco	abad	abancuy
abati	abbi	abbot	abby
abdallah	abdias	abdo	abdon

abdul	abel	abelard	abelardo
abele	abi	abigail	abner
abo	abra	abraham	abram
abril	absalom	absalon	acab
acacio	achill	acnin	ada
adabella	adair	adalbaro	adalbert
adalberto	adalgisa	adalgiso	adalia
adalrico	adaluz	adalvino	adam
adan	adara	adassa	adda
ade	adela	adelaida	adelaide
adelardo	adelfo	adelhard	adelia
adelina	adelino	adelio	adelma
adelmar	adelmaro	adelmo	adelqui
ademar	ademaro	adena	adhelmar
adhemar	adiel	adilia	adimar
adina	adolfo	adolphus	adonai
adonias	adonis	adoracion	adria
adriadna	adrian	adriana	adriano
adriel	adulfo	afra	afrodita
agalia	agamenon	agape	agapita
agapito	agar	agata	agatha
agaton	agenor	ageo	agesislao
aggie	agila	aglae	agnano
agnelo	agnes	agnese	agnus
agop	agostina	agripina	agripino
agüeda	agus	agustin	agustina
ahmed	aicardo	aida	aidano
aide	aidee	aien	aike
ailen	ailicec	ailin	ailizett
aillen	aime	aimon	ain
aina	ainara	ainoa	ainoha
aisha	aitana	aitor	aixa
akemi	akira	aladino	alaia
alaide	alain	alan	alana
alano	alaor	alardo	alarico
alba	albana	albano	albert
albertina	alberto	albina	albino
albretch	alceo	alcia	alcibiades
alcides	alcira	alcmena	alcuino
ald	alda	aldana	aldano
aldino	aldo	aldous	ale

alec	aleck	alegra	aleida
alejandra	alejandrina	alejandrino	alejandro
alejo	aleksander	aleksandra	alem
alen	ales	alesia	alesio
alessandro	alessia	alessio	alethia
alex	alexa	alexander	alexandra
alexandre	alexia	alexis	aleydis
alf	alfa	alfio	alfonsa
alfonsina	alfonso	alfred	alfredo
algiso	alhueche	ali	alice
alicia	alida	alide	alidia
alin	alina	aline	alipio
alira	alison	alix	alize
alma	almandos	almendra	almira
almudena	alondra	alonso	alphonse
alsan	altair	altea	alterio
alucan	alucio	alueche	aluhe
alulay	alumine	alvar	alvaro
alvero	alvina	ama	amada
amadeo	amadis	amado	amador
amaia	amal	amalia	amalio
amalsinda	aman	amancai	amancay
amancio	amanda	amandio	amando
amankaya	amantzi	amapola	amara
amaranta	amaranto	amarilia	amarilis
amarilla	amaro	amaru	amatista
amaya	ambar	ambroise	ambrosia
ambrosio	amelia	amelio	america
americo	amerigo	ami	amiel
amilca	amilcar	amin	amina
aminta	amintor	amir	amira
ammia	ammiano	ammiel	amneris
amon	amos	amparo	ampelio
amy	ana	anabella	anacleto
anael	anahi	anahid	anaias
analia	ananias	ananquel	anarda
anastasia	anastasie	anastasio	anat
anatilde	anatolio	anaxagoras	andre
andrea	andreas	andreina	andres
andresa	androcles	andromeda	andronico
andy	ane	anelida	anelina

anelisa	anelsy	anfos	angel
angela	angeles	angelica	angelina
angelino	angelo	angie	angus
ania	anian	aniano	anias
anibal	anicet	aniceto	aniela
aniketa	aniria	anisia	anisio
ann	anna	annabel	anolfo
anouk	anquises	ansaldo	anselma
anselmo	antares	antenor	anthony
antia	antigona	antigono	antipas
antoinette	antolin	anton	antonella
antonia	antonieta	antonino	antonio
antoshika	antu	anuar	anunciacion
anush	anxela	aparicio	apeles
apia	apolinar	apolinario	apolito
apolo	apolonio	aquiles	aquilesia
aquilino	ara	arabel	arabela
arabia	araceli	aracelia	arador
aram	arami	arandu	arantza
aranzazu	arapey	araxi	arcadia
arcadio	arcangel	arcangela	arcelia
archibaldo	ardalion	ardon	arduino
are	areb	arebela	areli
ares	argenis	argenta	argentina
argentino	argento	argeo	argimiro
argus	ari	aria	ariadna
ariadne	arian	ariana	ariane
arianna	ariano	ariel	ariela
arielle	aristarco	aristeo	aristides
aristobulo	aristocles	aristoteles	aritofanes
arlet	armanda	armando	armen
armentario	arminda	armonia	arnaldo
arnoldo	arnulfo	arquelao	arquimedes
arsenia	arsenio	artemio	artemisa
arthur	artura	arturo	ary
arydea	asa	asael	asaf
ascension	ascla	asdrubal	asela
asenat	aser	ashley	asier
astolfo	astor	astra	astrea
astrid	astryd	asuncion	atahualpa
atala	atenea	athina	athos

atica	atila	atilano	atilio
atzin	aucan	auda	audomaro
audrey	augusta	augustine	augusto
aura	aurele	aurelia	aureliano
aurelio	auristela	aurora	austin
auxano	avelina	avelino	axel
ayax	ayelen	aylen	aylin
aymara	aynkan	ayrton	aysha
azalea	azanias	azarias	azariel
azas	azrael	azucena	azul
baal	bab	babbete	babs
bacchus	baco	bahia	bahiana
balbina	balbino	balbo	baldomero
baldovin	baldovino	balduino	baldwin
baltasar	balthasar	barbara	barbea
barbelo	barbie	barlaan	barnie
bart	bartelemy	bartie	bartley
bartolo	bartolome	baruc	baruj
basa	basemat	basil	basileo
basilia	basilio	bastien	batilde
baudelio	baudilia	baudilio	bautista
bayard	beat	beatrice	beatrix
beatriz	beattie	beda	bee
begga	begona	bela	belen
belinda	belisa	belisaria	belisario
bella	belmiro	belona	beltran
ben	benedetta	benedicta	benedicto
benet	benicio	benigna	benigno
benilda	benildo	benito	benjamin
bennie	benny	benon	berenguer
berenice	berna	bernabe	bernabela
bernabeu	bernarda	bernardina	bernardino
bernardita	bernardo	beronike	berta
bertha	bertilda	bertoldo	bertran
bessie	betania	betiana	betina
betsabe	betsy	bettina	betty
bianca	bianei	bianey	bibiana
bibiano	bicor	bienvenida	bienvenido
bill	birgitta	bixintxo	blanca
blanche	blandina	blas	blasco
bob	bobby	boleslao	bona

bonanova	bonfilia	bonifaci	bonifacia
bonifacio	boreas	boris	borja
branco	brandon	branko	braulia
braulio	brenda	brendano	breogan
brian	briana	bricio	bridget
brigida	brigitte	brisa	briseida
britanic	bru	bruna	brunela
brunella	brunilda	bruno	brutus
buenaventura	cadmo	caetano	caifas
cain	caitan	cala	calanit
caleb	calfu	calfucir	caligula
calimaco	calinica	caliope	calistrato
calixta	calixto	calquin	camelia
camil	camila	camille	camilo
canan	cancio	candela	candelaria
candice	candida	candido	canela
cannan	canumil	caren	caridad
carim	carina	carisa	carl
carla	carles	carlo	carlomagno
carlos	carlota	carmela	carmelo
carmen	carmin	carmina	carmine
carol	carola	carolina	carpo
carysa	casandra	casandro	casdoa
casia	casiana	casiano	casilda
casildo	casimira	casimiro	casio
cassandre	casta	castalia	casto
castor	cataldo	catalina	caterina
catherine	cato	caton	catrian
catriel	catulo	cayetano	cayo
ceadas	cebeles	cecile	cecilia
cecilio	ceferina	ceferino	celedonio
celerino	celest	celeste	celestino
celia	celica	celide	celina
celinda	celio	celmira	celsa
celso	cencio	cenobio	centola
cerca	cesar	cesare	cesaria
cesarina	cesario	cesarion	cesia
cesira	ceumar	chalten	chandra
chantal	charles	charlotte	charo
chenoa	chiara	chloe	chris
christian	christina	christine	christopher

cibeles	cibran	ciceron	cid
cielo	cilinia	cindy	cinthia
cintia	cintio	cipriano	cira
circe	cirenia	ciriaco	ciril
cirila	cirilo	cirinea	cirineo
ciro	ciset	clara	clarabella
clare	claribel	clarisa	claro
claudette	claudia	claudina	claudino
claudio	claus	clea	cleandro
clelia	clemencia	clemente	clementina
clementino	cleo	cleodora	cleofas
cleofe	cleopatra	cleto	clide
clidia	climaco	climene	clinio
clio	clodoaldo	clodomiro	clodovea
clodoveo	cloe	clorinda	clorindo
cloris	clotilde	clovis	cochi
colin	collipal	colomba	colon
concepcion	concordia	cono	conrad
conrado	consolacion	constance	constancio
constantino	constanza	consuelo	cora
coral	cordelia	corina	cornelia
cornelio	corrado	cosimus	cosme
covadonga	crescencio	crimilda	cripin
crisanto	crisipo	crisoforo	crisol
crisologo	crisostomo	crispina	crispo
cristal	cristel	cristhian	cristian
cristina	cristo	cristobal	cristopher
cruz	cuasimodo	cumelen	cunibaldo
cuniberto	cupido	custodia	custodio
cuyen	cynthia	cyprien	cyril
cyrille	dacia	dacil	dacio
daff	dafna	dafne	dagma
dagmar	dagoberto	daiana	daila
daira	daisy	dalal	dalia
dalila	dalma	dalmacia	dalmacio
dalmazio	dalmiro	damaris	damaso
damia	damian	damiana	damocles
dan	dana	danae	danel
daniel	daniela	daniele	danila
danilo	danisa	danny	dante
dara	dardo	daria	dariel

dario	dativa	david	davina
davor	day	dayanira	de
debora	deborah	debra	dedalo
deidamia	dejanira	de la cruz	de la paz
de las mercedes	de las nieves	del	del carmen
delfin	delfina	delfor	delia
delicia	del lujan	delma	de los angeles
de los milagros	del pilar	del rosario	del sagrado corazon
del sol	de lujan	del valle	delvis
demeter	demetria	demetrio	demian
democrito	demostenes	denis	denisa
denise	dennis	deodato	deolinda
deonilde	derek	desdemona	desiderato
desideria	desiderio	desiderius	desiree
deyanira	diadema	diana	dibe
dick	diderot	didier	didimo
didio	diego	digna	dimas
dimitri	dimpna	dina	dinah
dino	dinora	dinorah	diocles
diogenes	diomedes	dion	dionel
dionisia	dionisio	dios	dioscoro
diva	divina	doelia	dolly
dolores	domenec	domenica	domiciano
domicio	domikene	dominga	domingo
dominica	dominique	domitila	domma
dommina	donaldo	donardo	donatela
donatila	donato	donina	donna
donosa	dora	dorcas	doria
dorian	dorina	doris	dorotea
doroteo	dositeo	douglas	doyel
drusila	duarte	dubraska	dugen
duilio	dulas	dulce	dulcinea
duncan	dunstan	dunstano	dustin
dylan	dyonis	eawinda	ebe
eber	eberardo	ebo	ecio
eco	eda	edco	edda
edelberto	edelia	edelio	edelira
edelma	edelmar	edelmira	edelmiro
edelweiss	eden	edesio	edgar
edgarda	edgardo	edilia	edilio
edilma	edipo	edit	edita

edith	edmar	edmund	edmundo
edna	eduarda	eduardo	edurne
eduviges	edward	edwin	efebo
efraim	efrain	efrein	efren
egberto	egda	egeo	egeria
egidia	egidio	egisto	egle
eider	eilal	eileen	einar
eira	eiru	eitan	ekaterina
ela	eladia	eladio	elais
elal	elba	elbio	elcira
elda	elea	eleazar	electra
elena	elenio	eleodora	eleodoro
eleonor	eleonora	eleuterio	elia
elian	eliana	eliane	elias
eliazar	elida	elido	eliecer
eliel	eliezer	eligio	elihu
elin	elina	elinda	elio
elis	elisa	elisabet	elisandro
elisea	elisenda	eliseo	elizabeth
ellis	elmer	elodia	eloisa
eloy	elpidia	elpidio	elsa
elsira	elsy	eluhuei	eluney
elvia	elvina	elvio	elvira
elvis	elvisa	ema	emanuel
emelia	emelina	emelinda	emerenciana
emerio	emerita	emerson	emeterio
emigdio	emil	emilia	emilian
emiliana	emiliano	emilio	emillen
emilse	emily	emir	emma
emmanuel	emna	emperatriz	ena
encarnacion	endike	eneas	eneida
eneyen	engracia	enoc	enon
enos	enric	enrica	enrico
enrique	enriqueta	enzo	epicuro
epifanio	erakil	erardo	erasmo
erato	erberto	ercilia	eri
eric	erica	erico	erik
erika	erina	erlinda	ermelinda
ermelindo	erminia	ernesta	ernestina
ernesto	ernie	eros	errolan
ervina	ervino	erwin	es

esau	escipion	escolastica	escolastico
esculapio	esdras	esmerada	esmeralda
esmirna	esopo	espartaco	esperance
esperanza	estanislao	esteban	estefania
estela	estelinda	ester	esterina
esther	estrella	etel	etelinda
etelvina	ethel	etienne	ettiene
eudosia	eudoxia	eufemia	eufemio
eufrasia	eugen	eugenia	eugenio
eulalia	eulalio	eulogia	eulogio
eulogius	eunice	eurico	euridice
eusebia	eusebio	eustacio	eustaquio
eustasio	eva	evando	evangelina
evangelino	evarista	evaristo	evasio
evelia	evelina	evelio	evelyn
evodia	exal	exaltacion	exequiel
eyen	ezequias	ezequiel	ezer
ezilda	ezio	ezra	fabia
fabian	fabiana	fabiano	fabio
fabiola	fabricia	fabricio	facunda
facundo	falco	fani	fanny
fantino	fanuel	fara	faraon
farid	fariol	farisa	fatima
faust	fausta	fausto	favio
fayruz	fazzio	fe	febe
febes	febo	federica	federico
federigo	fedor	fedra	fedro
felicia	feliciano	felicitas	felipa
felipe	felisa	felisardo	felix
ferdinando	fergus	fermin	fermina
fernan	fernanda	fernando	fiamma
fidel	fidela	fidelia	filadelfo
filademo	fileas	filebert	filelio
filemon	filiberto	filis	filomena
fina	fiona	fiorella	fito
flaminia	flaminio	flavia	flaviano
flavie	flavio	flor	flora
floreal	florencia	florencio	florentino
florian	floriana	florinda	florio
floro	folco	fortuna	fortunata
fortunato	franca	francesc	francesca

francesco	francine	francis	francisca
francisco	franco	frank	fred
fredel	frederic	fredeswinda	fresia
freya	frida	friedrich	frine
fritz	froilan	fructuoso	frutos
fucsia	fulbert	fulco	fulgencio
fulk	fulvia	fulvio	fulxencio
fusca	fusiano	gabelo	gabina
gabino	gabriel	gabriela	gabriele
gabriella	gabrielle	gaby	gad
gadiel	gaetan	gaia	gail
gal	gala	galatea	galeaso
galeno	galia	galilah	galileo
galit	galo	gamal	gamaliel
gandolfo	ganix	garcia	garcilaso
gardenia	gardine	garibaldo	garoa
gaspar	gaston	gatty	gaudencio
gavina	gaxan	gayane	gea
gedeon	gelasio	gema	genara
genaro	genciana	generosa	generoso
genesis	genoveva	geoffrey	geordie
george	georgia	georgina	geppe
geppetto	geraldina	geraldine	geraldo
geranio	gerardo	gerda	geremias
gerlac	germain	german	germana
germinal	geronimo	gerson	gertrudis
gervasio	gesualdo	getulio	ghita
giacometta	gian	giancarla	giancarlo
gianfranco	gianira	gianluca	gianmarco
gianna	gianni	giannina	gibert
gigi	gil	gilberta	gilberto
gilda	gildo	gillen	gimena
gina	ginebra	gines	ginette
ginnie	giobbe	gioconda	gioia
giordano	giorgio	gioseppina	giovanna
giovanni	giraldo	girzie	gisberto
gisela	giselle	gislena	giulia
giuliana	giuliano	giulio	giunia
giuseppe	giusto	gladis	gladys
glauc	glauco	glenda	glicina
gloria	godfred	godofredo	goio

goliard	goliat	gonzalo	goratze
gordon	gotardo	goyo	grace
gracia	gracian	graciana	graciano
gracias	graciela	graham	grato
grau	gravida	grazian	grecia
greer	gregor	gregorina	gregorio
greta	gretel	grisel	griselda
grizel	guadalberto	guadalupe	gualberto
gualtar	gualterio	guaraci	guido
guillermina	guillermo	guilleuma	guinerve
guiomar	gulmen	gumersindo	gundelinda
gundenia	gunter	gus	gustav
gustavo	guy	guzman	gwenole
gyorgy	habib	habid	habrilia
hada	hadda	hadrian	hadwig
haide	haidee	haig	haizea
halima	haman	hamanchay	hamlet
hannah	hans	haroldo	harry
hartman	harumi	hassan	hayde
haydee	haziel	hebe	heber
heberto	hector	heda	heidi
heidy	heinz	helda	heldo
helen	helena	helga	heli
helia	heliana	heliena	heliodoro
heloisa	helvecia	helvia	hemilce
henedina	henoch	henry	heraclea
heraclito	heraldo	herbet	hercilia
hercules	heriberto	hermalindo	herman
hermelinda	hermelindo	hermenegildo	hermes
hermilda	herminda	herminia	herminio
hermione	hermogenes	hernan	hernando
herodes	herodiade	herodoto	hersilia
herundina	herve	higinia	higinio
hilaria	hilario	hilary	hilda
hilde	hildegarda	hildegunda	hildelita
hilen	hipolita	hipolito	hippolyte
hiram	hobart	homero	honorata
honorato	honoria	honorina	honorio
horacio	horangel	hortensia	hortensio
hossana	huaman	huapi	huara
hubert	huberto	huenu	hugo

hugolina	hugolino	huichal	huilen
hullen	humbert	humberto	husai
hygin	ia	iael	iago
iair	ian	ianina	iara
ibar	ibel	iber	iberia
iberico	iberio	ibero	ibi
ibrahim	icaro	iciar	ida
idalia	idalina	idara	idelia
idelina	idumea	idumeo	ifigenia
igal	ignacia	ignacio	igone
igor	iguazel	ilana	ilanit
ilda	ildefonso	ildegunda	ileana
ilia	iliana	illcapil	illed
illel	ilona	iluminada	ilva
ilve	imanol	imelda	imperio
ina	inacayal	inaki	inalen
inan	inaqui	inaxio	inca
incul	inda	indalecio	indes
indiana	indibil	indira	indro
ines	ingmar	ingrid	inigo
inmaculada	inocencio	inti	ioav
iole	ion	iona	ionatan
ione	iosef	iracema	iraida
irati	iratze	iren	irene
ireneo	iriel	irina	irineo
iris	irma	irmina	irta
irupe	isaac	isabel	isabelina
isabella	isabelle	isachar	isadora
isaias	isaldina	isanqui	isaura
isberga	iselda	isidora	isidoro
isidre	isidro	isis	ismael
ismelda	ismenia	isod	isolda
isolina	israel	israela	itaete
italina	italo	itamar	itatay
itati	itsaso	ittmar	itziar
iva	ivan	ivana	ivanka
ivanna	ivany	iverna	ives
ivon	ivonne	iwan	iziar
jaasiel	jabel	jacint	jacinta
jacinto	jack	jackie	jacob
jacobo	jacqueline	jacques	jael

jaia	jaime	jair	jairo
jalil	james	jamila	jan
jana	jane	janice	jannifer
jano	janvier	jaqueline	jasmine
jason	javier	javiera	jayma
jazmin	jean	jeannette	jehiel
jehova	jemima	jemina	jenara
jenaro	jenifer	jennifer	jenny
jenofonte	jeremiah	jeremias	jeremy
jeronima	jeronimo	jerry	jerusalen
jesabel	jesica	jessica	jesualdo
jesus	jesusa	jeuel	jezabel
jimena	jimeno	joab	joad
joan	joana	joanes	joaquin
joaquina	joav	job	jocelin
jocelyn	joe	joel	joey
johana	johann	johanna	john
jon	jonah	jonas	jonatan
jonathan	jordan	jordana	jordi
jordina	jorge	jorgelina	josafat
joscio	jose	josefa	josefina
joselin	josemaria	josep	joseph
josephine	joshua	josias	josue
jova	jovita	joyce	juan
juana	juanelo	juanita	juanjo
juanma	judas	judit	judith
juez	jules	julia	julian
juliana	juliano	julieta	julio
juno	jupiter	justa	justin
justina	justiniano	justino	justo
juven	juvenal	juvencia	juvencio
juventina	juventino	kaiane	kaitan
kaled	kalen	kalid	kalil
kamar	kamil	kamille	kapriel
karem	karen	karenina	karim
karin	karina	karitte	karl
karol	karumanta	kate	katharina
katherine	katia	katja	katu
kaukel	kay	keila	keith
kemal	kemberley	ken	kenji
kenneth	kensel	kenti	keren

kevin	khalil	kiara	kiliano
killa	kim	kipa	kirian
kirios	kirk	kistine	kitty
klaus	konrad	kore	krin
kristine	kurt	kusi	kussi
kuyen	laban	ladio	ladislao
ladolfo	laelia	laercio	laertes
lahual	lahuen	laia	laila
lain	lair	lais	lamberto
landerico	landolf	landolfo	landrada
lanfranco	lanin	laodamia	laodicea
lara	larisa	larry	lastenia
laszlo	laumer	laura	laureana
laureano	laurelino	laurence	laurencia
laurencio	laurent	laurentina	laurentino
laurindo	lauro	lautaro	laviana
lavinia	lawrence	laya	lazaro
lea	leal	leandra	leandre
leandro	learco	leda	ledicia
leila	leire	leixandre	leiza
lelia	lelica	lelio	lelis
lemuel	len	lena	leneo
leo	leocadia	leocadio	leocricia
leon	leonarda	leonardo	leoncia
leoncio	leonel	leonela	leonelo
leonid	leonidas	leonides	leonila
leonilda	leonor	leonora	leontina
leontino	leopolda	leopoldina	leopoldo
lesbia	leslie	lesmes	let
leticia	letizia	leto	leuco
leuter	lev	levi	levon
lewis	leylen	lia	liafdag
liam	lian	liana	libano
libe	liber	libera	liberal
liberata	liberato	libertad	libia
libio	libitina	libna	liboria
liborio	lican	licas	licerio
licia	licinio	licio	licurgo
lida	lide	lidia	liduvina
lie	lien	ligia	lihue
lihuel	lil	lila	lilia

lilian	liliana	lilibet	lina
lincoln	linda	lindor	linneo
lino	linus	lioba	lionel
lionela	lior	lis	lisa
lisandra	lisandro	lisardo	lisette
lisias	lisistrato	liu	livia
livino	livio	liza	llanca
llanque	loana	loida	lola
lolly	longinos	lope	lorea
loreley	lorena	lorenz	lorenza
lorenzo	loreta	loreto	lorna
loruhama	lorujama	lot	lotario
louis	lourdes	loyola	lua
luana	luano	luca	lucas
lucelia	lucero	lucho	lucia
luciana	luciano	lucila	lucina
lucio	lucrecia	lucrecio	lucy
ludmila	ludovica	ludovico	ludwig
luigi	luis	luisa	luisina
lujan	luken	luminosa	luna
lupe	luperco	lupo	lutero
lutgarda	luvencio	luz	lydia
lydie	mabel	macabeo	macaire
macarena	macaria	macario	macedonio
maciel	maciela	maclovio	macra
macrina	madelaine	madelon	madox
mael	mafalda	magali	magdalena
maggie	magin	magno	magnolia
mahoma	maia	maialen	maica
maico	maida	maile	mailen
mailin	maimara	mainque	maira
maisa	maisie	maitane	maite
maiten	maitena	maku	malaquias
malco	malcolm	malcom	malcon
malena	malguen	malisa	malka
malte	malva	malvina	mamerto
mampu	manases	manela	manfredo
manila	manlio	manon	manque
manric	manuel	manuela	manzur
mapril	mara	maral	marana
maravillas	marc	marcela	marceliana

marcelina	marcelino	marcelo	marcia
marcial	marcio	marco	marcos
marga	margalit	margarita	margot
maria	maria begona	maria belen	maria de la concepcion
maria de la paz	maria de las nieves	maria de las victorias	maria del mar
maria de los angeles	maria de los milagros	maria del pilar	maria del rosario
maria fatima	maria gracia	maria guadalupe	maria ines
maria inmaculada	maria jesus	maria jose	maria lourdes
marian	mariana	marianela	mariangeles
marianne	mariano	maria noel	maria nuria
maria sol	maria soledad	maribel	maricel
maricruz	marie	mariel	mariela
marien	marieta	marilda	marilena
marilina	marilu	marilyn	marin
marina	marine	marino	mario
mariola	marion	mariquena	maris
marisa	marisabel	marisel	marisela
marisol	marite	mariu	marlene
marlon	maro	maron	marta
martha	martial	martin	martina
martiniana	martiniano	martino	martzel
maruja	marvel	marvela	marvin
mary	marziabo	massimo	mateo
mateos	mathias	matias	matilde
matteo	matty	maud	maura
maureen	mauricio	maurizio	mauro
max	maxelinda	maxima	maximiano
maximiliana	maximiliano	maximino	maximo
maya	mayda	mayra	medarno
medea	mederico	meinard	melani
melania	melanie	melany	melas
melba	melchior	melchor	melea
melecio	melibea	melibeo	melin
melina	melinda	meline	melisa
melissa	melito	meliton	melitona
melody	melquiades	melquisedec	melusina
melvin	memmon	menajem	menandro
menas	mendel	menelao	meneo
menna	mennos	menqui	mentor
merced	mercedes	merces	mercurio
meredith	meris	meritxell	merle

merlin	merlina	merlino	mery
metran	meulen	meuris	mi
mia	micael	micaela	michael
michela	michele	michelle	mickey
micky	micol	midas	miguel
miguelina	mijael	mijail	mijal
mikel	mila	milagros	milan
milba	milburga	milca	milciades
mildreda	milena	milenka	miles
mileva	millan	millaray	milton
milva	milwida	mina	minerva
minotauro	miqueas	miqueo	mirabel
mirana	miranda	mirari	mireille
mirella	miren	mireya	miriam
mirko	mirna	mirta	mirtha
misael	misky	mitra	mixel
miyen	modesta	modesto	mohamed
moira	moises	monica	montserrat
mora	morena	morfeo	morgana
moria	morris	moshe	munir
munira	muredac	muriel	mustafa
myra	myriam	myrna	naara
nabil	nabila	nabor	nabucodonosor
nacho	nadal	nadia	nadina
nadine	nadir	nahara	nahir
nahuel	nahum	naiara	naim
naimid	naiquen	naira	nais
najla	nale	nambi	namuncura
nana	nancy	nandor	nantilde
naomi	napoleon	nara	narcis
narcisa	narciso	narcisse	narela
narellla	narkis	naroa	nasha
nasya	natacha	natal	natali
natalia	natalio	nataly	natan
natanael	nataniel	natasha	nathanel
natividad	natzari	navila	nayara
nayla	nayme	nayra	nazar
nazarena	nazareno	nazaret	nazareth
nazario	neandro	nedar	neera
neftali	nehemias	nehueln	nehuen
neiber	nela	nelda	nelida

nelly	nelo	nelson	nemesia
nemesio	neptuno	nera	nerea
nereida	nereo	nereu	neri
nerida	nerina	nerio	neron
nestor	netanel	neus	nevio
neyen	nicandro	nicanor	nicasio
niceas	niceforo	niceto	nicholas
nick	nicky	nicodemo	nicolas
nicolasa	nicole	nicomedes	nicon
nidia	nieves	nikole	nilce
nilda	nilo	nils	nimfa
nimia	nimsi	nina	ninfa
nino	niobe	niranjana	nirma
nisim	noam	noe	noel
noelia	noelino	noelio	noemi
nofre	nolan	nolasco	nolberto
nominanda	nontue	nora	norah
norali	norberta	norberto	noreia
noris	norma	norman	normando
nothelmo	nubar	nubia	nuil
numa	numas	numeria	numilen
nuncia	nuncio	nunila	nur
nuri	nuria	nuriel	nurit
nuriya	nydia	obadia	obdulia
obdulio	obed	oberto	octavia
octaviano	octavio	oda	oderico
odette	odila	odilon	odin
odo	odoacro	odon	ofelia
ofir	olaf	olalla	olaya
olegario	oleguer	olga	olimpia
olimpo	olina	olinda	oliver
oliverio	olivia	olmo	omar
omaro	omer	omero	ona
onan	ondina	onelia	onesimo
onofre	oralia	orangel	orencio
orestes	orfeo	orfilia	oria
oriana	oriel	orieta	orietta
origenes	orion	orlando	orly
ornella	orosco	orquidea	oscar
oseas	osias	osman	osmar
osmaro	osmin	osmundo	osvaldo

oswald	oswaldo	otelo	othmaro
othon	otilde	otilia	otilio
otniel	oton	otoniel	otto
otton	ovidia	ovidio	owen
ozana	ozias	oziel	pabil
pablo	pace	paciente	pacifico
pacomio	paddy	paine	palaciada
palas	palemon	palixena	palma
palmira	paloma	pamela	pampa
pampin	panambi	panbil	pancracia
pancracio	pandora	panfilo	pantaleon
paola	paolo	paris	parmenides
parmenio	partenia	pascua	pascual
pascualina	paskasi	pastor	pastora
pat	patricia	patricio	patrick
patsy	patty	paul	paula
paulette	paulina	paulino	paulo
paun	pavel	paz	pearl
pedro	peer	pegaso	peggy
pehuen	pelagia	pelagio	pelayo
pelegrino	penelope	penen	penny
perceval	percival	perfecto	periandro
pericles	perla	perpetuo	perseo
petra	petrona	petronio	petruos
petrus	pia	pichi	piedad
piera	pierina	piero	pigmalion
pilar	pilato	pilmayquen	pimpinela
pio	piperion	piren	pirra
pirro	piuque	placida	placido
platon	plauto	plinio	plotino
plubio	pluton	pola	policarpo
polidora	polidoro	polifemo	polixena
pompei	pompeo	pompeyo	pompilio
ponce	poncio	ponpey	popea
porcia	porfirio	poseidon	prantxes
praxedes	preciosa	presentacion	priamo
prilidiano	primavera	primitiva	primo
princesa	prinio	prisc	prisca
priscila	procopio	prometeo	proserpina
prospero	prudencia	prudenciana	prudenciano
prudencio	prudens	ptolomeo	publia

publio	pulqueria	pulqui	pura
purificacion	purlan	pusaki	queila
queremon	querian	querima	querubin
querubina	queta	quichauel	quico
quiliano	quillen	quimey	quinciano
quint	quinta	quintilian	quintiliano
quintilio	quintilo	quintin	quinto
quionia	quique	quirico	quirina
quirino	quiteria	rabulas	rachel
radames	radegunda	rafa	rafael
rafaela	rafel	raffi	raguel
raidis	raifroid	raimon	raimundo
rainero	raingarda	raiquen	ralph
ram	ramiro	ramon	ramona
ramses	rancul	randolfo	raphaela
raquel	raquildis	ratrudis	rauel
raul	ray	rayen	raymi
raymond	raynaldo	rebeca	recaredo
regina	reginaldo	regino	regis
regulo	reina	reinaldo	reinardo
relinda	remedios	remigio	remo
renaldo	renan	renata	renato
renau	rene	renee	renzo
restituto	reuben	reuquen	rex
reyes	reynaldo	rhode	ricarda
ricardo	richard	rick	rigel
rigoberto	rina	rinaldo	rita
rivi	rob	robert	roberta
robertino	roberto	robin	robinson
robustiano	rocco	rocio	rocky
rodas	rode	rodolfo	rodrigo
rogelia	rogelio	roger	rogerio
rolan	rolando	roman	romana
romanela	romano	romelio	romeo
romero	romildo	romina	romualdo
romula	romulo	ronald	ronan
roni	roque	roquelina	rosa
rosalba	rosalia	rosalie	rosalina
rosalinda	rosamunda	rosana	rosario
rosaura	rosenda	rosendo	rosicler
rosilda	rosina	rosinda	rosmari

rosmira	roswinda	rotrauda	roxana
roy	ruben	rubi	rubina
rudecindo	rudolf	rufino	rufo
ruperto	rut	ruth	rutilda
ruy	ruyan	saba	sabacio
sabas	sabel	sabela	sabelia
sabelio	sabina	sabino	sabrina
sacha	sadoc	sadoth	safira
safo	sagar	salaberga	sally
salome	salomon	salustio	salvador
salvadora	salvator	salvatore	salvia
salviana	salviano	salvina	salvino
salvio	salvo	sam	samanta
samantha	samar	samara	samir
samira	samuel	sancho	sandor
sandra	sandro	sanson	santiago
santina	santino	santo	santos
sara	sasha	saskia	sathya
saturio	saturnino	saturno	sauken
saul	saula	saulo	saustin
saveria	saverio	sayi	seaghdha
sean	sebasten	sebastian	sebastiana
secundina	sefora	segismunda	segismundo
segunda	segundino	segundo	sein
selemias	selene	selenia	selesio
selim	selma	selva	semarias
seminaris	semiramis	sempronio	senan
seneca	septimia	septimio	septimo
serafin	serafina	seraphin	serapio
serena	sereno	sergei	sergia
sergio	servanda	servando	servia
servio	serxio	set	seth
severa	severino	severo	shadia
shaiel	sharif	sharon	sheila
shirley	shirly	shulamit	shulinen
siagria	sibila	sidney	sigfrido
siglinda	sigrid	silo	silvana
silvano	silverio	silvestre	silvia
silvina	silvino	silvio	simeon
simon	simona	simplicio	sincletica
sinesio	sinforiano	sinforosa	sinforoso

sintiques	sir	sira	sirio
siro	sisebuto	sixta	sixto
socorro	socrates	sofanor	sofia
sofiel	sofocles	sol	solana
solange	solano	soledad	solon
sonia	sonsoles	sophe	sophie
soraya	sotero	spiro	stanislao
stefan	stefania	stefano	stella
stephanie	steven	styalianos	sulamita
sultana	sunta	suray,	surisadai
susana	suyai	suyay	suzette
suzzane	sylvester	sylvius	tabare
tabatha	tabita	tabitha	taciana
taciano	tacio	tacito	tadeo
taffy	taiana	taiel	tais
takara	tali	talia	talin
tamar	tamara	tancredo	tania
tarcila	tarquino	tarsicia	tarsicio
tarsilia	tatiana	tatiano	tea
tecla	ted	tehuel	telemaco
telma	telmo	temis	teo
teobaldo	teocrito	teodelina	teodequilda
teodolinda	teodomiro	teodor	teodora
teodorico	teodoro	teodosia	teodosio
teofania	teofano	teofila	teofilo
teolinda	tercio	terencio	teresa
teresita	terpsicore	terry	tertulio
teseo	tesira	tetis	thaiel
thais	thelma	theo	theodor
thiago	thomas	tiago	tibalt
tiberio	tibor	tiburcio	ticiana
ticiano	tico	tilo	timotea
timoteo	tiquico	tirsa	tirso
tita	tito	tiziana	tiziano
tobias	toby	tolomeo	tom
tomas	tomasa	tome	torbellino
torcuato	toribio	toscana	traful
transito	triana	trinidad	tristan
tristana	trix	troilo	troya
tubal	tubau	tulia	tulio
tupac	turquesa	tusnelda	tzipora

ubaldina	ubaldo	uberto	uciel
udolfo	ulfrido	ulises	ulla
ulpiano	ulpio	ulrico	umbelina
umberto	urania	urbana	urbano
urias	uriel	ursina	urso
ursula	ursulina	ursy	utka
uxia	uxio	uziel	valburga
valda	valdemar	valdo	valdrada
valentin	valentina	valentino	valeria
valeriano	valerio	valeska	valfredo
valquiria	vanda	vanesa	vanina
vaniria	vanna	vannina	varinia
veda	velia	vella	venancia
venancio	venceslao	ventura	venus
vera	verbena	veredigna	verena
verna	vernon	vero	verona
veronica	veronique	vertan	vesna
vespasiano	vesta	vic	vicencio
vicenta	vicente	victor	victoria
victoriano	victorino	victorio	vidal
vilfredo	vilma	vin	vinicio
viola	violeta	violetta	violette
virgilio	virginia	virginio	virna
visitacion	vita	vital	vitalia
vitaliano	vitalicio	vito	vitoldo
viv	viviana	viviano	vladimir
vladimiro	vulcano	vulpiano	wagner
walberto	walda	waldemar	waldino
waldo	walkiria	walkyria	walter
waltruda	wanda	wara	washington
waskar	wayca	wenceslao	wendi
wenzel	wereburga	werner	wilfredo
william	willie	willka	wilma
wilson	winifreda	winni	wulfilde
xabat	xacob	xalome	xalvat
xan	xarles	xavier	xaviera
xenia	xerman	xian	xilda
ximena	xiomara	xob	xochiel
xochilt	xochtiel	xoel	xose
xulio	xusto	yaco	yadira
yael	yago	yaguati	yaima

yain	yair	yaiza	yaku
yakue	yamal	yamel	yamil
yamila	yamile	yanella	yanet
yanina	yanine	yanira	yannick
yanquiman	yara	yasmin	yasmina
yazmin	yemina	yeneko	yerimen
yeruti	yesica	yexalen	yilia
yoana	yoav	yocasta	yoconda
yoel	yolanda	yole	yone
yosef	yrko	yudit	yuki
yulan	yunca	yunka	yuqui
yuri	yve	yves	yvette
yvonne	zaba	zacarias	zach
zacky	zahir	zahira	zaida
zair	zaira	zaqueo	zara
zarina	zeeb	zelma	zelmar
zelmira	zemira	zenadia	zenobia
zenobio	zenon	zenzo	zilla
zina	zita	zite	zoe
zoila	zoilo	zoraida	zosimo
zuleica	zulema	zulima	zulma
zunilda	zuria		