UNIVERSIDAD POLITÉCNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Multilingualidad en Reconocimiento de Texto Manuscrito

Trabajo
presentado por Miguel Ángel del Agua Teba
supervisado por
D. Nicolás Serrano Martínez Santos y
Dr. Alfons Juan Císcar

8 de septiembre de 2010

PRÓLOGO

Actualmente vivimos en una sociedad en la que el acceso, manipulación y distribución de la información, cobra una importancia vital para el desarrollo cultural y económico. Estas tareas se han automatizado gracias al auge de las Telecomunicaciones, de Internet y de la Informática, como herramientas de procesamiento de datos.

En este contexto, dado que existen enormes colecciones de documentos históricos en bibliotecas, museos y archivos, surge la necesidad de digitalizarlos con el objetivo de garantizar su preservación y beneficiarse de las nuevas tecnologías, como puede ser su distribución a través de grandes bibliotecas digitales on-line.

El principal objetivo, sin embargo, no es simplemente proporcionar acceso a imágenes en bruto de documentos digitalizados, sino anotarlas con su contenido informativo real y en particular con transcripciones de texto e incluso con su traducción, para que toda la información que contienen sea lo más accesible posible. Desafortunadamente, el estado del arte en transcripción automática, difícilmente pueden tratar con texto manuscrito o incluso con texto impreso antiguo.

Aún así, existen prototipos que tratan de reducir estas barreras y que son capaces de dar apoyo en la transcripción de texto, reduciendo notablemente la carga de trabajo que esta tarea conlleva. En este ámbito, uno de los muchos inconvenientes a los que un transcriptor se enfrenta, es que algunos libros antiguos están escritos en más de un idioma.

Esto tampoco pasa desapercibido para un sistema de transcripción automática y es aquí donde se centra el presente trabajo. Realizaremos una primera aproximación en transcripción automática de documentos multilíngües. Para ello desarrollaremos y experimentaremos con distintos sistemas de reconocimiento.

El documento se organiza como se describe a continuación: En el primer capítulo se repasarán los distintos algoritmos y métodos utilizados en el trabajo. En el segundo capítulo se describirá la base de datos GERMANA. En el tercer capítulo, se desarrollará el sistema monolíngüe donde se considerará como si todo el documento estuviera escrito en un mismo idioma. En el cuarto capítulo, se describirá el sistema multilíngüe, donde se entrena un sistema distinto para cada idioma. En el quinto capítulo, se estudiará un sistema que predice el idioma de la línea a reconocer. Y por último, en el sexto capítulo resumiremos el trabajo y extraeremos las conclusiones del mismo.

ÍNDICE GENERAL

CAPÍTULO 1

PRELIMINARES

1.1. Reconocimiento de Formas

En este primer apartado se dará un breve repaso a lo que se entiende por sistema de reconocimiento de formas(de ahora en adelante RF), así como a la teoría necesaria para seguir la metodología aplicada en este documento. Cabe destacar que no se espera que estos preliminares sean una descripción detallada sobre el tema que se trata, sino un breve resumen. Para más información puede referirse a [1].

1.2. Sistemas de Reconocimiento de Formas

Un sistema de RF puede verse básicamente como dos procesos separados: un módulo de aprendizaje y un módulo de clasificación. El módulo de aprendizaje recibirá como entrada, datos en bruto correspondientes a muestras del mundo real, donde mediante diversos algoritmos abstraerá la información necesaria de la entrada para aprender los patrones de las muestras, que le servirán para clasificar en el siguiente modulo. El módulo de clasificación con la información adquirida del modulo de aprendizaje, recibirá datos en bruto y retornará una categoría, clase o etiqueta correspondiente a la clasificación. Ahora bien, concretando podemos descomponer aún más estos módulos:

- **Preproceso**, esta etapa está formada por una serie de algoritmos que tratarán los datos en bruto (imágenes de vídeo, audio correspondiente a una grabación, etc...), convirtiéndolos en valores manejables y estructurados, que tratarán el

resto de procesos. La eliminación de ruido o de redundancia suelen ser las metas más comunes de esta etapa.

- **Extracción de características**, con los datos previamente tratados, se puede extraer la información relevante para nuestro sistema. Que será aquella que consiga distinguir bien los elementos de diferentes categorías y englobe de forma correcta los individuos de las mismas. Un ejemplo de este proceso sería, pasar de una imagen de un árbol al número de frutas que en ella se encuentran.
- **Aprendizaje**, a partir de las características de nuestras muestras, el sistema adquirirá la información necesaria para clasificar por medio de procesos estadísticos.
- **Clasificación**, una vez entrenado el sistema, tomará una muestra y devolverá la etiqueta de una determinada clase.

1.3. Teoría de la decisión de Bayes

Bajo ciertas asunciones, el mejor clasificador que se puede obtener es el llamado clasificador de Bayes. A partir de una muestra x_i y una serie de clases para esas muestras $C = \{c_1, c_2, ..., c_m\}$, el mejor clasificador es el que asigna a cada muestra la clase que maximiza su probabilidad a posteriori. Pero no podemos obtener la probabilidad a posteriori exacta porque necesitaríamos todas las muestras existentes de nuestro problema. Por tanto, podemos aplicar la regla de Bayes a partir de la expresión, para obtener algo que podamos calcular:

$$\underset{c \in C}{\operatorname{argmax}}\, p(c \mid x_i) = \underset{c \in C}{\operatorname{argmax}}\, \frac{p(x_i \mid c)p(c)}{p(x)} = \underset{c \in C}{\operatorname{argmax}}\, p(x_i \mid c)p(c) \tag{1.1}$$

Donde $p(c)$ es la probabilidad a priori de c y en la última fórmula experimentalmente podemos estimar $p(c)$ y $p(x_i \mid c)$, que pueden ser estimados por medios estadísticos y matemáticos.

1.4. Modelos de Markov de capa oculta

Los modelos de Markov de Capa Oculta, que de ahora en adelante se nombrarán como HMMs(del inglés Hidden Markov Models), son modelos paramétricos que dan la probabilidad de secuencias de caracteres simbólicos o vectores numéricos. De forma más concreta, un HMM es una máquina de estados finitos que emite una serie de símbolos o vectores numéricos. Los HMMs se encuentran formados por dos procesos bien definidos. Por un parte se tiene un comportamiento oculto, que es el correspondiente a la transición entre estados con unas ciertas probabilidades y por otra parte tenemos el comportamiento visible, que es el de emisión de las salidas que se pueden emitir cada uno con una determinada probabilidad.

Los HMMs son ampliamente utilizados en el RF dado que son una herramienta para poder clasificar secuencias de un número indeterminado de componentes. Como

puede ser el habla o en el caso que ocupa este documento, imágenes de texto manuscrito. En este documento se distinguirán dos tipos de HMMs, los que emiten símbolos y los que emiten vectores numéricos. De esta manera formalmente un HMM se define como una tripleta de cuatro elementos $M = (Q, \Sigma, \pi, B)$ donde:

- **Q** es un conjunto de estados, que incluye un estado inicial I y un estado final F que no emiten símbolos.
- **E** es en el caso de HMMs que emiten símbolos, el conjunto de símbolos observables que pueden se emitidos por los estados. Mientras que en el caso de HMMs que emiten vectores numéricos, sería el espacio real d-dimensional $E \subseteq \mathbb{R}^d$.
- $\pi \in \mathbb{R}^{\mathbf{Q}\times\mathbf{Q}}$ es la matriz con las probabilidades correspondientes a las transiciones entre los estados.
- **b**, es la función de emisión de los estados, en el caso de HMMs discretos se corresponderá con una matriz que indica la probabilidad de emisión de uno de los posibles símbolos de salida en un determinado estado. Mientras que en el caso de HMMs continuos, esta función de salida modela la densidad de probabilidad de emitir un vector de E en un determinado estado, mediante una mixturas de gaussianas.

$$b_{j\in Q}(x_t) = \begin{cases} B_{j,x_t} & \text{en el caso discreto} \\ \sum_{m=1}^{M_j} c_{jm} N(\mu_{jm}, \Sigma_{jm}) & \text{en el caso continuo} \end{cases} \tag{1.2}$$

Adicionalmente debe cumplir las siguientes restricciones:

- Las probabilidades iniciales deben cumplir:

$$0 \leq \pi_{Iq} \leq 1, \sum_{q\in(Q-F)} \pi_{Iq} = 1, \pi_{IF} = 0 \tag{1.3}$$

- Las probabilidades de transición entre estados deben cumplir:

$$\forall q, q' \in (Q - I), 0 \leq \pi_{q,q'} \leq 1, \sum_{q'\in Q} \pi_{q,q'} = 1, A_{F,q} = 0 \tag{1.4}$$

- En el caso de HMMs discretos la función b, será equivalente a un matriz B que cumplirá que:

$$\forall q \in (Q - \{I, F\}) \wedge x \in E, 0 \leq B_{q,x} \leq 1, \sum_{x\in E} B_{q,x} = 1, B_{F,x} = 0, B_{I,x} = 0 \tag{1.5}$$

- En el caso de HMMs continuos la función b debe cumplir que:

$$\int_{x\in X} b(q_i, x)dx = 1, \forall q_i \in (Q - \{I, F\}) \tag{1.6}$$

El último punto a destacar es la topología del modelo. Lo topología es la forma que tiene el grafo subyacente al modelo, que viene dada por las transiciones que es posible realizar. Algunos ejemplos de estas topologías son:

- **Ergódico**, se trata de un grafo completo, es decir donde todas las transiciones posibles tienen un valor mayor que cero.
- **Izquierda-derecha**, es aquel modelo cuyo grafo subyacente se trata, de un grafo acíclico dirigido, donde cada estado tiene transiciones posibles a el mismo o a estados superiores.
- **Ferguson**, es un caso particular del tipo anterior, donde cada estado tiene transiciones a: el mismo, el siguiente estado y el estado final.

1.5. Algoritmo Forward

Con tal de averiguar la probabilidad de emitir una secuencia de símbolos (en el caso de HMMs discretos) o vectores (en el caso de modelos continuos), se puede usar el algoritmo Forward. Se define una función α_{ntq} que indica la probabilidad de emitir un prefijo de la secuencia $\boldsymbol{x}_n = \{\boldsymbol{x}_{1n}, ..., \boldsymbol{x}_{tn}\}$ en el instante de tiempo t estando en el estado q.

$$\alpha_{ntq} = \begin{cases} \pi_{Iq} p(\boldsymbol{x}_{n1} \mid \boldsymbol{\theta}, q) & t = 1 \\ p(\boldsymbol{x}_{nt} \mid \boldsymbol{\theta}, q) \sum\limits_{q' \in Q} \alpha_{nt-1q'} \pi_{q'q} & 1 < t \leq T \end{cases} \tag{1.7}$$

Donde $\pi_{qq'}$ es la probabilidad de la transición de q a q' y θ son los parámetros de la función de emisión de los estados. Así la probabilidad de que la secuencia x sea emitida por el modelo de Markov con parámetros $\boldsymbol{\theta}$ es:

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = p(\boldsymbol{x}_{n1}\boldsymbol{x}_{n2}...\boldsymbol{x}_{nT} \mid \boldsymbol{\theta}) = \alpha_{nTF} = \sum_{q \in (Q-F)} \alpha_{nTq} \cdot \pi_{qF} \tag{1.8}$$

La complejidad temporal de este algoritmo es $\Theta(\mid Q \mid^2 \times T)$. Aunque si se usan topologías de *izquierda-derecha*, entonces el coste temporal es $\Theta(\mid Q \mid \times T)$.

1.6. Algoritmo Backward

El algoritmo calcula la probabilidad β_{ntq}, que es la probabilidad de emitir el sufijo de la muestra x_n, del instante $t+1$ al final, sabiendo que en el instante t se estaba en el estado q. De esta manera el valor β_{ntq}:

$$\beta_{ntq} = \begin{cases} 1 & t = T \\ \sum\limits_{q' \in Q} p(\boldsymbol{x}_{nt+1} \mid \boldsymbol{\theta}, q') \pi_{qq'} \beta_{nt+1q'} & 1 \leq t < T \end{cases} \tag{1.9}$$

Donde $\pi_{qq'}$ es la probabilidad de la transición del estado q a q' y $\boldsymbol{\theta}$ son los parámetros de la función de emisión de los estados. Así la probabilidad de que la secuencia x sea emitida por el modelo de Markov con parámetros $\boldsymbol{\theta}$ es:

$$p(x \mid \boldsymbol{\theta}) = p(\boldsymbol{x}_1\boldsymbol{x}_2...\boldsymbol{x}_T \mid M) = \beta_{N0I} = \sum_{q' \in Q} \pi_{Iq'} p(\boldsymbol{x}_{n1} \mid \boldsymbol{\theta}, q')\beta_{n1q'} \tag{1.10}$$

La complejidad temporal de este algoritmo es $\Theta(\mid Q \mid^2 \times T)$. Aunque si se usan topologías de *izquierda-derecha*, entonces el coste temporal es $\Theta(\mid Q \mid \times T)$.

1.7. Algoritmo de Viterbi

El algoritmo de Viterbi es una variación del algoritmo Forward. Si en el algoritmo Forward hallábamos la probabilidad de una secuencia x como el sumatorio de las probabilidades obtenidas por los distintos caminos. En el algoritmo de Viterbi se elige únicamente la mayor. De esta manera con sustituir el sumatorio por un máximo tenemos:

$$VIT_{ntq} = \begin{cases} \pi_{Ij} p(\boldsymbol{x}_{n1} \mid \boldsymbol{\theta}, q) & t = 1 \\ p(\boldsymbol{x}_{nt} \mid \boldsymbol{\theta}, q) \max\limits_{q' \in Q} VIT_{nt-1q'} \pi_{q'q} & 1 < t \leq T \end{cases} \tag{1.11}$$

Donde $\pi_{qq'}$ es la probabilidad de la transición del estado q a q' y $\boldsymbol{\theta}$ son los parámetros de la función de emisión de los estados. Así la probabilidad de que la secuencia x sea emitida por el modelo de Markov con parámetros $\boldsymbol{\theta}$ es:

$$VIT_{nTq} = \max_{q' \in (Q-F)} VIT_{nq'T}\pi_{q'F} \leq \sum_{q' \in (Q-F)} \alpha_{nTq'} \cdot \pi_{q'F} = \alpha_{nTq} \tag{1.12}$$

Que como se puede observar, es un cota inferior del algoritmo Forward. El coste temporal es $\Theta(\mid Q \mid^2 \times T)$, aunque si se usan topologías de *izquierda-derecha*, entonces el coste temporal es $\Theta(\mid Q \mid \times T)$

1.8. Algoritmo "Forward-Backward"

El algoritmo "Backward-Forward" o "Baum-Welch" se utiliza para la estimación de los parámetros que definen el HMMs continuo formado, en este caso por mixturas de gaussianas: $\boldsymbol{\theta} = \{\pi_{q'q}, p_{iq}, \boldsymbol{\mu}_q \text{ y } \Sigma_q\}$. Donde $\pi_{qq'}$ es la matriz que indica la probabilidad entre transiciones; p_{iq}[a] es el peso de cada componente de la mixturas; $\boldsymbol{\mu}_q$ es el vector de medias y Σ_q la matriz de covarianzas. Sea $x = \{x_1, x_2, ..., x_N\}$ un conjunto de N secuencias de vectores, utilizadas para estimar los parámetros.

De esta manera el modelo descrito se define como:

[a] En este apartado se ha usado P para denotar la probabilidad de un suceso, para no confundirla con p_{iq}.

$$P(\boldsymbol{x}_n) = \sum_{\boldsymbol{s}_n \in S_n} \sum_{\boldsymbol{z}_n \in Z_n} \prod_{q=1}^{Q} \pi_{Iq}^{S_{n1tq}} \prod_{t=2}^{T} \prod_{q'q}^{Q} \pi_{q'q}^{(s_{nt-1q'} s_{ntq})} \prod_{t=1}^{T} \prod_{q=1}^{Q} \prod_{i=1}^{I} (p_{qi} \cdot P(x_{nt} \mid q, i, \boldsymbol{\theta}))^{s_{ntq} z_{ntqi}} \tag{1.13}$$

Donde $P(x_{nt} \mid q, i, \boldsymbol{\theta})$ es la probabilidad para una normal de media $\boldsymbol{\mu}_{qi}$ y matriz de covarianza Σ_{qi}. Así los parámetros de el modelo se estiman como:

$$s_{ntq}^{(k)} = \frac{P(s_{ntq} = 1, \boldsymbol{x}_n \mid \boldsymbol{\theta}^{(k)})}{P(\boldsymbol{x}_n \mid \boldsymbol{\theta}^{(k)})} = \frac{\alpha_{ntq}^{(k)} \cdot \beta_{ntq}^{(k)}}{\sum_{q \in Q} \alpha_{nTq}^{(k)}} \tag{1.14}$$

$$\begin{aligned} (s_{nt-1q'}, s_{ntq})^{(k)} =& \frac{P(s_{nt-1q'} = 1, s_{ntq} = 1, \boldsymbol{x}_n \mid \boldsymbol{\theta}^{(k)})}{p(\boldsymbol{x}_n \mid \boldsymbol{\theta}^{(k)})} = \\ & \frac{\alpha_{nt-1q'}^{(k)} \cdot \pi_{q'q} \cdot P(\boldsymbol{x}_{nt} \mid q) \cdot \beta_{ntq}}{\sum_{q \in Q} \alpha_{nTq}^{(k)}} \end{aligned} \tag{1.15}$$

Donde $s_{ntq}^{(k)}$, es la probabilidad de en el instante t, emitir el t-ésimo vector de la muestra n, estando en el estado q. Y $(s_{nt-1q'}, s_{ntq})^{(k)}$, es la probabilidad de usar la transición de q' a q, en el instante t emitiendo el t-ésimo vector de la muestra n al llegar al estado q.

Así según el HMM propuesto con dichos parámetros, la probabilidad de una muestra dada, vienen determinada por:

Las transiciones del estado inicial a los demás estados, se expresan como:

$$\pi_{Iq} = \frac{\sum_{n=1}^{N} s_{n1q}^{(k)}}{\sum_{i=1}^{N} \sum_{q=1}^{Q} s_{n1q}^{(k)}} = \frac{\sum_{n=1}^{N} s_{n1q}^{(k)}}{N} \tag{1.16}$$

Mientras que las transiciones entre los demás estados se expresan como:

$$\pi_{q'q} = \frac{\sum_{n=1}^{N} \sum_{t=2}^{T} (s_{nt-1q'}, s_{ntq}^{(k)})}{\sum_{n=1}^{N} \sum_{t=2}^{T} s_{nt-1q'}^{(k)}} \tag{1.17}$$

Para un HMM continuo de mixturas de gaussianas los parámetros se reestiman mediante el factor $(s_{ntq} \cdot z_{ntqi})^{(k)}$ donde i es la componente de la mixtura, se formula como sigue:

$$\begin{aligned} (s_{ntq} \cdot z_{ntqi})^{(k)} =& \frac{P(s_{ntq} = 1, z_{ntqi}, \boldsymbol{x}_n \mid \boldsymbol{\theta}^{(k)})}{P(\boldsymbol{x}_n, \boldsymbol{\theta}^{(k)})} = \\ =& \alpha_{ntq}^{(k)} \cdot p_{qi}^{(k)} \cdot \frac{P(\boldsymbol{x}_{nt} \mid z_{ntqi} = 1, s_{ntq} = 1, \boldsymbol{\theta}^{(k)})}{P(\boldsymbol{x}_{nt} \mid s_{ntq} = 1, \theta^{(k)})} \\ & \cdot \frac{1}{p(\boldsymbol{x}_n, \boldsymbol{\theta}^{(k)})} \cdot \beta_{ntq}^{(k)} \end{aligned} \tag{1.18}$$

Donde este factor es la probabilidad de emitir, en el instante t de la componente i-ésima del estado q para la muestra n. Así definimos p_{qi}, como el peso de una componente de la mixtura y que la siguiente expresión:

$$p_{qi}^{(k+1)} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}(s_{ntq} \cdot z_{ntqi})^{(k)}}{\sum_{n=1}^{N}\sum_{t=1}^{T} s_{ntq}^{(k)}} \tag{1.19}$$

Se define $\boldsymbol{\mu}_{qi}$ como el vector de medias en el estado q y la componente i y que la siguiente expresión:

$$\boldsymbol{\mu}_{qi}^{(k+1)} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}(s_{ntq} \cdot z_{ntqi})^{(k)}\boldsymbol{x}_{nt}}{\sum_{n=1}^{N}\sum_{t=1}^{T}(s_{ntq} \cdot z_{ntqi})^{(k)}} \tag{1.20}$$

Por último Σ_{qi} es la matriz de covarianza del estado q y la componente i y tiene la siguiente expresión:

$$\Sigma_{qi}^{(k+1)} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}(s_{ntq} \cdot z_{ntqi})^{(k)}(\boldsymbol{x}_{nt} - \boldsymbol{\mu}_{qi})(x_{nt} - \boldsymbol{\mu}_{qi})^{t}}{\sum_{n=1}^{N}\sum_{t=1}^{T}(s_{ntq} \cdot z_{ntqi})} \tag{1.21}$$

El coste de este algoritmo es $\Theta(R\times \mid Q \mid^2 \times T)$: pero si el modelo M es de izquierda-derecha es $\Theta(R\times \mid Q \mid \times T)$.

1.9. Modelos de lenguaje, basados en N-gramas

En el campo de RF con tal de aproximar la estructura sintáctica y semántica de un lenguaje se usa lo que se conoce como un modelo de lenguaje. Estos lo que hacen básicamente, es calcular la probabilidad de que una serie de palabras del lenguaje aparezcan en cierto orden. Existen varias maneras de crear un modelo de lenguajes desde autómatas finitos deterministas a modelos basados en cadenas de Markov.

Los llamados modelos de N-gramas son una de las posibilidades para parametrizar un modelo de lenguaje. Un modelo de N-gramas nos dice la probabilidad que tiene un elemento de aparecer dependiendo de los N elementos anteriores. Existen diversos algoritmos que construyen el modelo a partir de una muestra del lenguaje, así como otros que tratan el problema de la aparición de elementos que no existían al entrenar. Para mayor comodidad se referirá de ahora en adelante a un modelo formado por palabras, pero podría bien formarse al igual por caracteres.

Dada una secuencia de palabras $\mathbf{w} = \{w_1, w_2, ..., w_m\}$ la probabilidad de que aparezca la misma se define como:

$$p(\mathbf{w}) = p(w_1)p(w_2 \mid w_1)...p(w_n \mid w_1, w_2, ..., w_n - 1) \tag{1.22}$$

Dado que el número de palabras que puede aparecer en una línea puede ser elevado, la estimación directa de la formula anterior resulta impracticable. Es por ello que se utilizan los N-gramas donde aproximaríamos cada probabilidad anterior, asumiendo

dependencia de tan solo sus *N-1* elementos anteriores. De esta forma la probabilidad ahora se expresaría como:

$$p(c) \approx \prod_{i=1}^{m} p(w_i \mid w_{i-N+1}, ..., w_{i-1}) \tag{1.23}$$

Los modelos de *N*-gramas pueden representarse mediante autómatas estocásticos de estados finitos, que desde ahora se designarán como AEEFs. Los AEEFs son una variación de los autómatas de estados finitos, donde a cada transición se le asigna una probabilidad. Así formalmente se puede definir un AEEF como una séxtupla $(Q, \Sigma, P, q_0, a, F)$ donde:

- $\mathbf{Q}$ es un conjunto de estados.
- $\mathbf{\Sigma}$ es un conjunto de símbolos de entrada.
- $\mathbf{P} \subset Q \times \Sigma \times Q$ es un conjunto de transiciones de la forma $\delta(q, c, q')$ que indica una transición del estado $\mathbf{q}$ a q' mediante el símbolo c.
- $\mathbf{q_0}$ es el estado inicial.
- $\mathbf{a}$ es una función de probabilidad de transición $a : P \to \mathbb{R}^+$.
- $\mathbf{F}$ es una función de probabilidad del estado final $F : Q \to \mathbb{R}^+$.

Adicionalmente debe cumplir algunas restricciones como:

$$\forall q \in Q \quad F(q) + \sum_{\forall (p,c,p'), p=q} a(p, c, p') = 1 \tag{1.24}$$

Una vez definidos se pueden representar el modelo formado por los distintos los unigramas, bigramas, ..., *N*-gramas como un AEEF. De esta manera cada *N*-grama se correspondería con un estado y las transiciones entre ellos serían la probabilidad de ese *N*-grama.

Con tal de estimar la probabilidad de los distintos *N*-gramas del modelo así como la probabilidad de final de línea a partir de ese *N*-grama, se deberá partir de los conteos de cada evento del texto con el que se entrena y optimizar la distribución que siguen dichos eventos. Que siguen una distribución multinomial. Por máxima verosimilitud se obtiene:

$$\hat{p}(c \mid q) = \frac{c(qc)}{c(q)} \qquad \forall (q, c, q') \in R \tag{1.25}$$

$$\hat{F}(q) = \frac{c_f(q)}{c(q)} \qquad \forall q \in Q \tag{1.26}$$

Al estimar un modelo de lenguaje uno de los principales problemas que aparecen es el llamado problema del suavizado. Este es el problema que aparece al estimar un modelo con un conjunto de palabras, mientras que en la práctica aparece alguna palabra desconocida. Hecho que no indica que su probabilidad de aparición sea cero.

Por ello lo que se hace es, si se tiene un modelo de *N*-gramas, y aparece alguna serie de N-palabras que no están en el modelo, se aproxima su probabilidad usando el modelo de (*N-1*)-gramas. Las técnicas más conocidas son el descuento y back-off, aunque básicamente ambas se basan en la redistribución de la masa de probabilidad, para aceptar eventos no observados.

El descuento es el proceso por el cual se reemplazan las cuentas originales de ocurrencia de los eventos (*N*-gramas) con cuentas modificadas, de forma tal que se redistribuye la masa de probabilidad de los eventos más comúnmente vistos a los menos frecuentes y no vistos. En cuanto al back-off, esta técnica consiste en redistribuir la masa de probabilidad de un modelo de *N*-gramas entre los sucesos que no aparecen inicialmente en el entrenamiento, utilizando un modelo de (*N-1*)-gramas ponderado con un back-off. De esta manera la probabilidad $\tilde{p}$ de un modelo donde se ha aplicado "back-off" queda como:

$$\tilde{p}(c \mid q) = \begin{cases} d(qc)\hat{p}(c \mid q) & \text{si } c(qc) > 0 \\ \alpha(q)\tilde{p}(c \mid q^*) & \text{si } c(qc) = 0 \end{cases} \tag{1.27}$$

donde $d(qc)$ es un descuento aplicado al N-grama qc, $\tilde{p}(c \mid q^*)$ es la probabilidad del modelo suavizador(en el caso de trigramas, sería el bigrama) y el back-off α es:

$$\alpha(q) = \frac{1 - \sum_{\forall c: c(qc)>0} d(qc)\hat{p}(c \mid q)}{1 - \sum_{\forall c: c(qc)>0} \tilde{p}(c \mid q^*)} \tag{1.28}$$

1.10. Esquema probabilístico

Disponemos de 3 modelos claramente diferenciados: Modelado a nivel de caracteres, a nivel de palabra y a nivel de línea. El proceso de reconocer letras a partir de las imágenes, se realiza mediante un modelo de caracteres. El sistema deberá segmentar las imágenes en las distintas letras que la forman. Así pues se utilizarán HMMs continuos que emiten vectores de características que siguen una distribución de mixturas de gaussianas, para modelar cada carácter que puede aparecer en el entrenamiento.

Con tal de modelar palabras se usa un modelo léxico, dicho modelo expresa con qué combinación de letras es posible representar palabras. Esto es debido a que puede que una palabra sea representada en minúsculas o mayúsculas, por distintos símbolos aunque represente la misma palabra.

Por último con tal de modelar líneas se usa un modelo de lenguaje. Este modelo expresa la probabilidad de que unas palabras vayan a continuación de otras o en determinado contexto.

Si se unen los tres modelos en uno solo, se puede integrar todo el proceso en el mismo sistema. Así, tomando el HMMs de cada carácter c , si partimos de una imagen formada por un número p de vectores de características o ventanas, nuestra muestra se representa como $x = x_1 x_2, ..., x_p$. El problema de distinguir una serie de caracteres $c = c_1 c_2 ... c_n$ donde n es el número de caracteres de que forman la línea, se puede ver el proceso de reconocimiento como un problema de maximización:

$$\hat{c} = \underset{c}{\operatorname{argmax}}\, p(c \mid x) = \underset{c}{\operatorname{argmax}}\, \frac{p(x \mid c)p(c)}{p(x)} = \underset{c}{\operatorname{argmax}}\, p(x \mid c)p(c) \tag{1.29}$$

Donde $p(c \mid x)$ se modelará como HMMs continuo de mixturas de gaussianas y $p(c)$ se modela como un modelo de lenguaje. Ahora bien falta incluir el proceso de segmentación en la formulación, pero como no entra en el ámbito de este documento, lo incluiremos resumidamente. Así, el proceso de reconocimiento de caracteres con segmentación integrada puede verse como:

$$\hat{c} \approx \underset{c}{\operatorname{argmax}} \underset{b,\Phi \in d(c)}{\operatorname{máx}} \prod_{i=1}^{n} p(x_{b_{i-1}}^{b_i} \mid c_i)p(q_{i-1}, c_i, q_i)F(q_n) \tag{1.30}$$

Donde los $b_1 b_2 ... b_i ... b_N$ son las distintas segmentaciones de los caracteres. Donde dada un secuencia de caracteres $c = c_1 c_2 ... c_N$ la secuencia de transiciones de uno a otro, puede expresarse como autómata de estados finitos $(q_0, c_1, q_1), (q_1, c_2, q_2), ...,$ (q_{N-1}, c_N, itq_N), con distintos caminos de principio a fin y denominados estos caminos con $\Phi(c)$.

CAPÍTULO 2

LA BASE DE DATOS GERMANA

2.1. Introducción

En esta sección, presentamos la base de datos GERMANA. GERMANA es el resultado de digitalizar y anotar un manuscrito de 764 páginas, escritas en su mayor parte en castellano, con el título *"Noticias y documentos relativos a Doña Germana de Foix, última Reina de Aragón"*. Fue escrito por Vicent Salvador, el marqués de Cruïlles en 1891. Tiene aproximadamente 21,000 líneas anotadas y transcritas por expertos paleógrafos. Para más información puede consultar [2].

2.2. Descripción

Se puede considerar que la transcripción de GERMANA es una tarea sencilla por las siguientes razones; La primera es que es un libro escrito por un único autor, de una temática bastante específica; La vida de *Germana de Foix* (1488-1538), nieta del rey Luis XII de Francia y segunda esposa de Fernando el Católico de Aragón. Además, el manuscrito original está muy bien conservado y la mayoria de las páginas contienen texto escrito con buena caligrafía y gran espacio entre líneas. Por otra parte, el manuscrito comprende alrededor de $217K$ palabras extraídas de un vocabulario de $30K$, lo que a priori es una cantidad de datos suficiente para el modelado de lenguaje.

Ni que decir tiene que la extracción de lineas y el reconocimiento de texto manuscrito sobre GERMANA sea precisamente una tarea sencilla. GERMANA tiene características própias de un documento histórico que la hacen especialmente difícil:

manchas, caracteres y palabras inusuales, etc. Además, el manuscrito incluye muchas notas y documentos que están escritos en diferentes lenguas tales como catalán, francés o latín.

En conjunto, pensamos que GERMANA implica una complejidad acorde a la cantidad de datos que proporciona. Que se sepa, es la primera base de datos pública de estas características para la investigación de texto manuscrito en castellano. Su tamaño es comparable al de otras bases de datos, como puede ser IAM [3]. Debido a su estructura secuencial, también es adecuada para la evaluación realista de sistemas de reconocimiento de texto manuscrito interactivos. Por otra parte, se puede usar también para el estudio de técnicas de identificación de idiomas y adaptación a la escritura.

A continuación, describiremos el manuscrito y la base de datos en las secciones 2.3 y 2.4 respectivamente.

2.3. El manuscrito

Tal y como se dijo en la introducción, GERMANA es el resultado de la digitalización y anotación de un manuscrito español del año 1891 sobre la vida de Germana de Foix. El manuscrito original se conserva en la colección de Nicolau Primitiu, en la Biblioteca Valenciana [4] . Comprende 764 páginas de las cuales, de acuerdo con su índice, se dividen en 17 secciones.

Por simplicidad, distinguiremos solo 7 partes del manuscrito:

1. *Parte inicial (pp 1-6):* Un subtítulo, un título y el retrato de Doña Germana de Foix.
2. *Los capítulos(pp 7-180):* 174 páginas divididas en 6 capítulos, cada uno de los cuales está basado en un periodo distinto de la vida de Germana.
3. *Notas (pp 181-282):* 290 notas numeradas, referenciadas en los diferentes capítulos.
4. *Notas biográficas (pp 283-302):* De 8 personas relevantes mencionadas en la segunda parte.
5. *Documentos (303-540):* Copias de texto manuscrito de 71 documentos históricos relacionados con la vida de Germana.
6. *Ilustraciones (pp 541-716):*4 documentos con su propio pie de imagen final.
7. *Parte final (pp 717-764):* Varios índices e imagenes.

La mayoría de páginas solo contienen texto manuscrito, alineado con una pauta horizontal en una plantilla simple de 24 líneas (pp 1-180 y 729-764) o 32 líneas (pp 181-728). Como ejemplo de los dos tipos de plantillas más usuales que hay en GERMANA, mostramos la página 29 de 24 líneas en la figura 2.1 en la página 15 y la página 190 de 32 líneas en la figura 2.2 en la página 16. También podemos observar, que la escritura es fácilmente legible y el alineado a la pauta horizontal es bastante preciso. Por otra parte, mostramos también otros tipos de páginas del GERMANA en la figura 2.3 en la página 17, donde se pueden ver títulos, notas, páginas con ilustraciones, etc.

Hasta la página 180, el manuscrito está escrito únicamente en lengua Castellana. A partir de ésta, el lector puede encontrar también Catalán, Francés, Latín, Alemán e Italiano. En la tercera parte, hay 33 notas, la mayoría de las cuales están escritas en Catalán (4, 47, 50, 73, 78, 79, 81, 82, 84, 85, 87-91, 94-96, 134, 177, 194, 205, 209, 214, 227, 229, 236, 238, 261, 266-268 y 270); 18 en Francés (1, 2, 15, 22, 23, 25, 29, 44-46, 71, 109, 110, 119, 155, 170, 257 y 280); y 1 en Alemán (180). Además, también hay 24 documentos en la quinta parte escritos en Catalán (7, 8, 27, 29, 31-33, 36-40, 44, 48-54, 59, 64, 68 and 69); 10 Latín (2, 4-6, 12, 24, 34, 42, 43, 70); 1 y Francés (7); 1 en Alemán (25); y 1 en Italiano (65). La biografía, las notas y las ilustraciones están escritas en Castellano, aunque también hay algo de Catalán (un breve pasaje de 13 líneas, comenzando en la última línea de la página 300; las notas 39, 47 y 61 de la ilustración C; y la nota 17 de la ilustración D). Al lector interesado se le remite a[5] para un estudio más profundo sobre el manuscrito, desde el punto de vista de un historiador.

2.4. La base de datos

El manuscrito fue cuidadosamente escaneado por expertos de la Biblioteca Valenciana, a una resolución de 300dpi en color verdadero. Como ocurre con los documentos históricos en general, las páginas escaneadas contienen ruido en forma de manchas, gotas de agua y transparencia del lado contrario. Además, el documento muestra el efecto "warping"debido a la encuadernación del libro. Aún así, el manuscrito se puede leer con bastante facilidad y por eso mismo, decidimos no aplicar ningún preprocesado para corregir estos defectos, a la hora de marcar las líneas de texto.

La anotación de las líneas de texto en GERMANA consiste en dos partes. Por una parte, todos los bloques de texto fueron marcados con rectángulos de mínima inclusión y dentro de cada uno de ellos, cada línea de texto fue marcada con líneas base (rectas). Todo esto se hizo de manera semi-automática mediante la ayuda del programa *GNU Image Manipulation Program* (GIMP) [6] y el prototipo GIDOC [7] desarrollado específicamente para la anotación de bloques y líneas de GERMANA. Todos los bloques y líneas detectados automáticamente fueron también supervisados manualmente y corregidos en su caso.

Por otro lado, todo el manuscrito fue transcrito línea a línea, por expertos paleógrafos. El proceso de transcripción no comenzó de cero, sino de una transcripción parcial producida por expertos de la Biblioteca Valenciana en 2002. Esta transcripción cubría la mayor parte del manuscrito (75 %), pero no era aprovechable para fines de investigación, sobre todo porque no incluía la página original y los saltos de línea. Aún así, sirvió como base para realizar una transcripción final allá por 2007 de acuerdo a las normas siguientes:

- Los saltos de línea y página se copian tal cual.
- Los espacios en blanco solo sirven para separar palabras.
- No se corrigen los errores ortográficos.
- No se harán cambios de capitalización o acentuación.

Idioma	Páginas	Líneas	Palabras (K)	Léxico Tamaño (K)	Léxico Sing. (%)	Conjunto caractrs.
Español	595	16599	176,8	19,9	55,6	111
Catalan	87	2417	26,9	4,6	63,2	86
Latín	29	951	8,3	3,4	69,2	87
Francés	8	266	3,0	1,1	71,1	82
Alemán	8	228	1,5	0,6	52,7	71
Italiano	2	68	0,8	0,3	67,3	59
Ninguno	35	0	0,0	0,0	0,0	0
Todos	764	20529	217,2	27,1	57,4	115

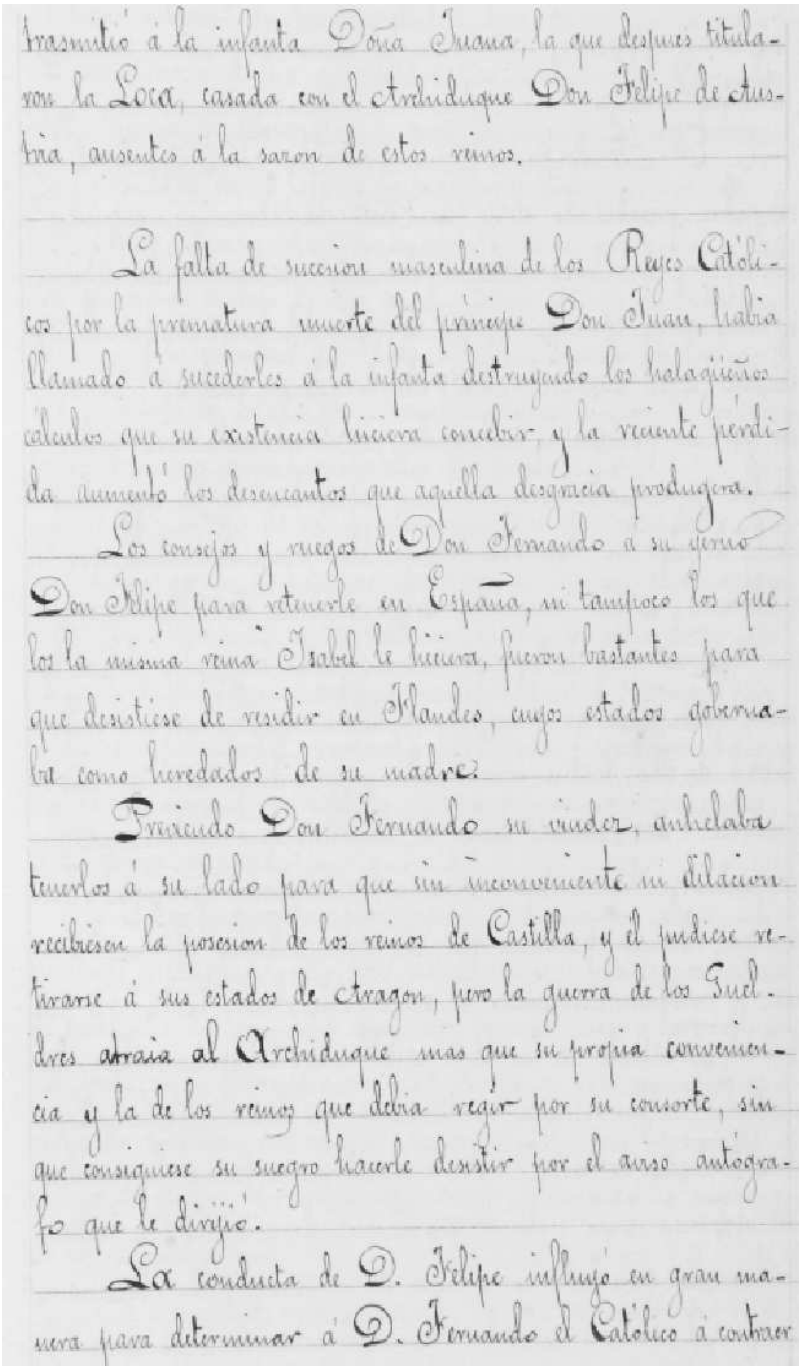
Cuadro 2.1: Estadísticas básicas de GERMANA(Sing=Singletons, palabras que solo ocurren una vez).

- Los signos de puntuación se copian tal como aparecen.
- Las abreviaciones se copian literalmente, excepto los subíndices y superíndices, que serán escritos con la notación de LaTeX, como `_{sub}` y `^{super}`, respectivamente. Después, se les acompañará por su correspondiente significado entre llaves. Como por ejemplo `Dª.` se transcribe como `D^{a}.[Doña]`.

Además, para facilitar el procesamiento del idioma del manuscrito, cada línea transcrita fue etiquetada manualmente de acuerdo con su lengua dominante. El tiempo total requerido para transcribir el manuscrito por un solo experto, se estimó en 232 horas; esto es aproximadamente 30 minutos por página en promedio.

La tabla 2.1 contiene algunas estadísticas básicas, extraídas de nuestras transcripciones de GERMANA. A tener en cuenta que la parte escrita en Castellano comprende alrededor de $17K$ líneas de texto y $177K$ palabras (running words) con un léxico de $20K$, el cual es comparable en tamaño al de otras bases de datos, como puede ser IAM[3]. Un dato bastante llamativo, es el hecho de que el 56 % de las palabras solo aparecen una vez en el texto (singletons).

La base de datos está diponible en la web del grupo PRHLT (`prhlt.iti.es`) para usos no comerciales. Además, se puede encontrar una transcripción impresa del manuscrito, no orientada al reconocimiento de texto manuscrito en [5].



trasmitió á la infanta Doña Juana, la que despues titularon la Loca, casada con el Archiduque Don Felipe de Austria, ausentes á la sazon de estos reinos.

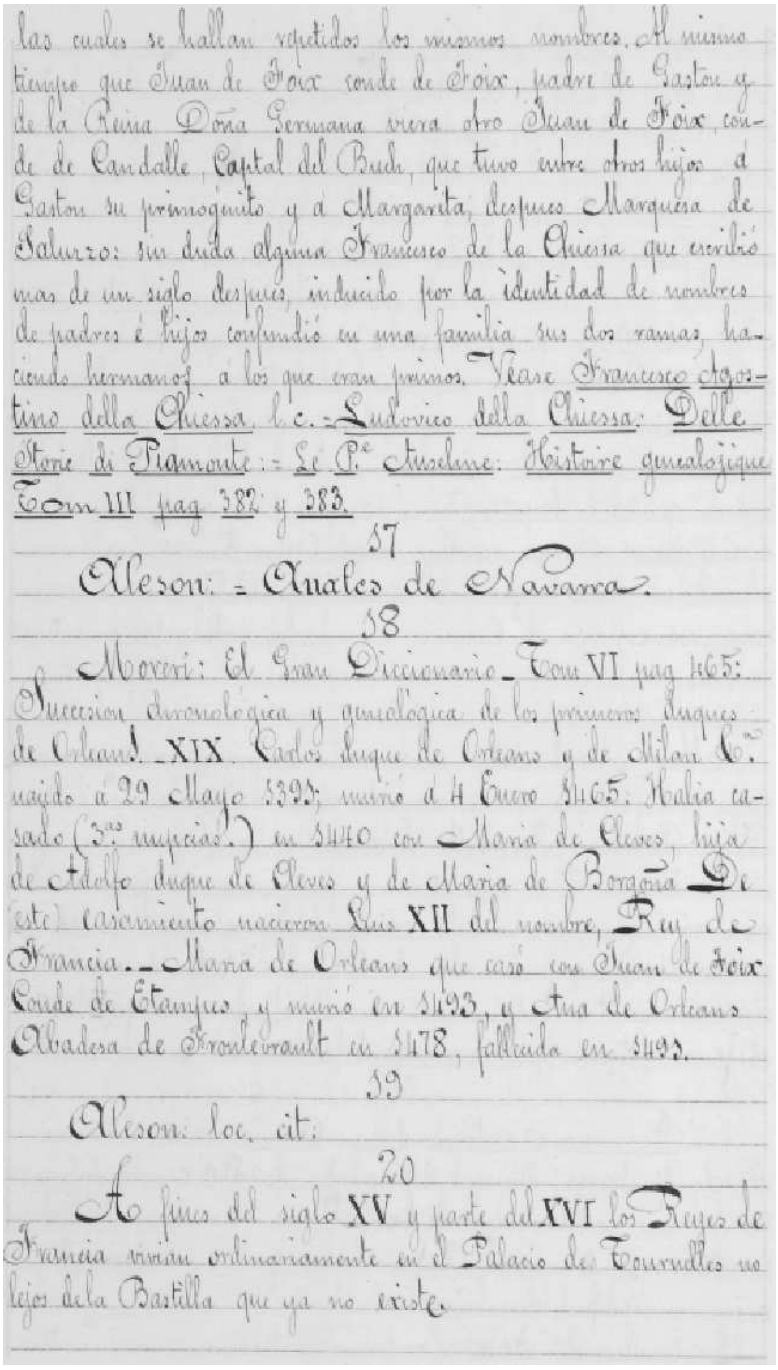
La falta de sucesion masculina de los Reyes Católicos por la prematura muerte del príncipe Don Juan, habia llamado á sucederles á la infanta destruyendo los halagüeños cálculos que su existencia hiciera concebir y la reciente pérdida aumentó los desencantos que aquella desgracia produgera.

Los consejos y ruegos de Don Fernando á su yerno Don Felipe para retenerle en España, ni tampoco los que los la misma reina Isabel le hiciera, fueron bastantes para que desistiese de residir en Flandes, cuyos estados gobernaba como heredados de su madre.

Deseando Don Fernando su vinder, anhelaba tenerlos á su lado para que sin inconveniente ni dilacion recibiesen la posesion de los reinos de Castilla, y él pudiese retirarse á sus estados de Aragon, pero la guerra de los Suelderes atraia al Archiduque mas que su propia conveniencia y la de los reinos que debia regir por su consorte, sin que consiguiese su suegro hacerle desistir por el auxo autógrafo que le dirijio.

La conducta de D. Felipe influyó en gran manera para determinar á D. Fernando el Católico á contraer

Figura 2.1: Página 29 de GERMANA. Ejemplo de plantilla de 24 líneas.


las cuales se hallan repetidos los mismos nombres. Al mismo tiempo que Juan de Foix conde de Foix, padre de Gaston y de la Reina Doña Germana viera otro Juan de Foix, conde de Candalle, Capital del Buch, que tuvo entre otros hijos a Gaston su primogenito y a Margarita, despues Marquesa de Saluzzo: sin duda alguna Francesco de la Chiessa que escribió mas de un siglo despues, inducido por la identidad de nombres de padres é hijos confundió en una familia sus dos ramas, haciendo hermanos a los que eran primos. Vease Francesco Agostino della Chiessa, l.c. = Ludovico della Chiessa: Delle Storie di Piamonte: = Le P.e Anselme: Histoire genealojique Tom III pag 382 y 383.

57

Aleson: = Anales de Navarra.

58

Moreri: El Gran Diccionario - Tom VI pag 1165: Succesion cronologica y genealogica de los primeros Duques de Orleans. - XIX Carlos Duque de Orleans y de Milan C.e nacido a 29 Mayo 1394: murió a 4 Enero 1465. Maria casado (3.os nupcias.) en 1440 con Maria de Cleves, hija de Adolfo duque de Cleves y de Maria de Borgoña. De este Casamiento nacieron Luis XII del nombre, Rey de Francia. - Maria de Orleans que casó con Juan de Foix Conde de Etampes y murió en 1493, y Ana de Orleans Abadesa de Fontebrault en 1478, fallecida en 1493.

59

Aleson: loc. cit:

20

A fines del siglo XV y parte del XVI los Reyes de Francia vivian ordinariamente en el Palacio de Tournelles no lejos de la Bastilla que ya no existe.

Figura 2.2: Página 190 de GERMANA. Ejemplo de plantilla de 32 líneas.

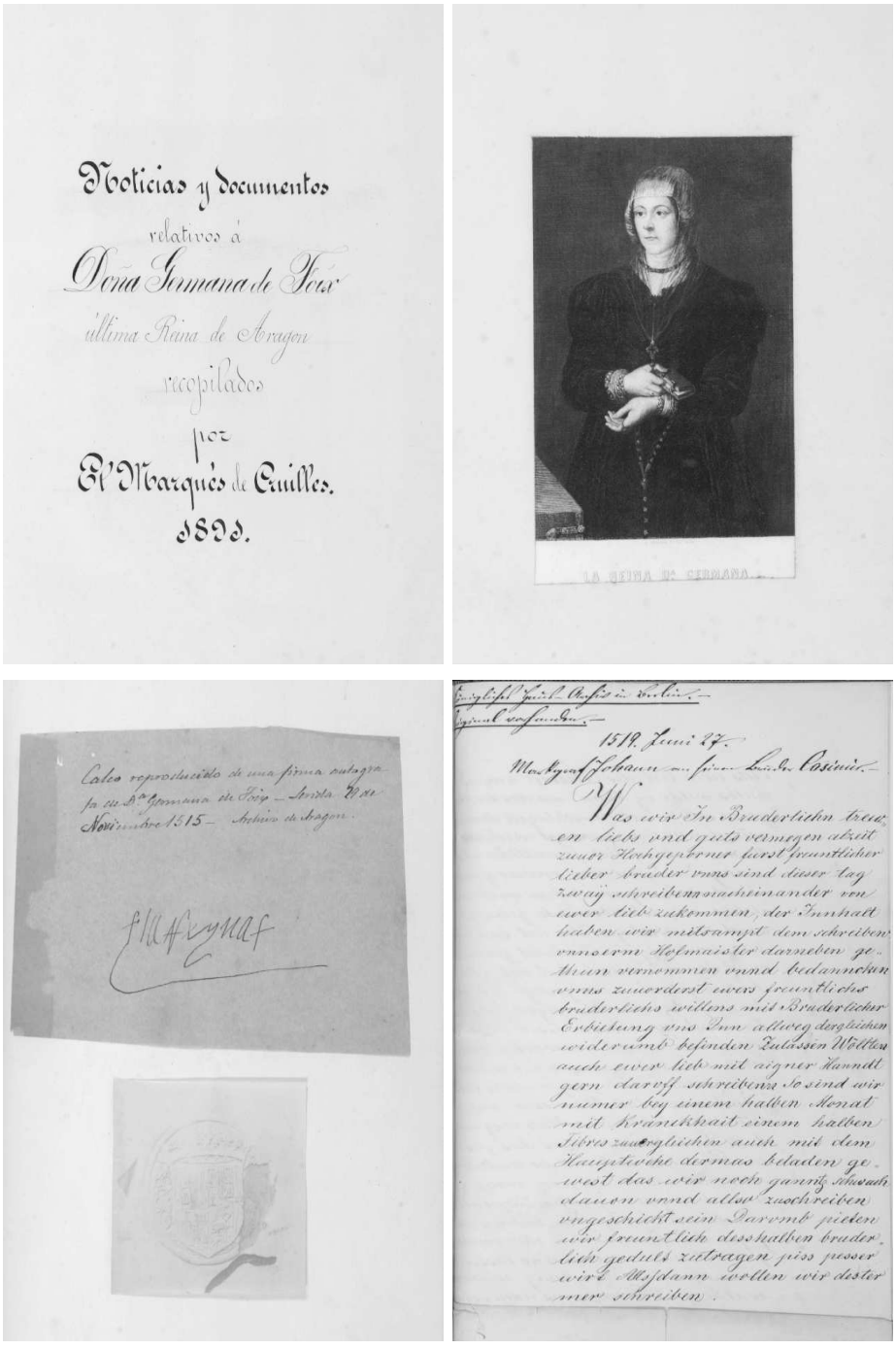


Figura 2.3: Páginas de GERMANA que no cumplen las normas de estilo.

CAPÍTULO 3

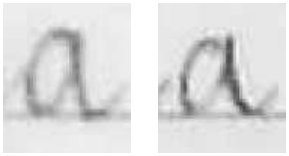
SISTEMA MONOLINGÜE

3.1. Introducción

Este capítulo se centra en el *reconocimiento de texto manuscrito continuo* (de ahora en adelante *RTM*), a partir del trabajo presentado en el artículo *"The GERMANA Database"* [2]. En él se habla sobre la base de datos GERMANA, resultado de la digitalización y anotación de un manuscrito de 1891 con el título *"Germana de Foix, última Reina de Aragón"*. Escrito por Vicent Salvador, el marqués de Cruïlles.

Tal y como podemos leer en el capítulo 2, el libro se divide en un total de 7 secciones. Las dos primeras nos relatan la vida de *Germana de Foix* y el resto son un compendio de notas, notas biográficas, documentos e ilustraciones cuyo contenido puede ser muy variable (tablas, listas o incluso imágenes). Con la dificultad añadida de que aparecen en idiomas tales como Latín, Catalán, Italiano, Francés o Alemán. Por estos motivos, los experimentos llevados a cabo en el citado artículo solo trabajan con las 2 primeras secciones (hasta la página 180), ya que son las más uniformes y están escritas íntegramente en Castellano.

El enfoque dado al RTM tiene mucho que ver con el empleado en el *reconocimiento automático del habla (RH)*. Básicamente, radica en la inter-cooperación de diferentes fuentes de conocimiento, cada una con un diferente grado de percepción: carácter, palabra y línea. Es decir, estas fuentes de conocimiento interactúan conjuntamente proveyendo un entendimiento global de la línea. Estas fuentes de conocimiento pueden ser apropiadamente modeladas mediante el uso, por ejemplo, de modelos de estados finitos tales como los HMMs (para los caracteres), gramáticas o autómatas (para las palabras) y modelos de lenguaje (para las líneas). Todos estos modelos, pueden ser



(a) Castellano (b) Latín

Figura 3.1: Comparación de caligrafía de la letra 'a' en idiomas diferentes.

integrados en un solo gran modelo de estados finitos que pueda hacer frente a la tarea global del reconocimiento.

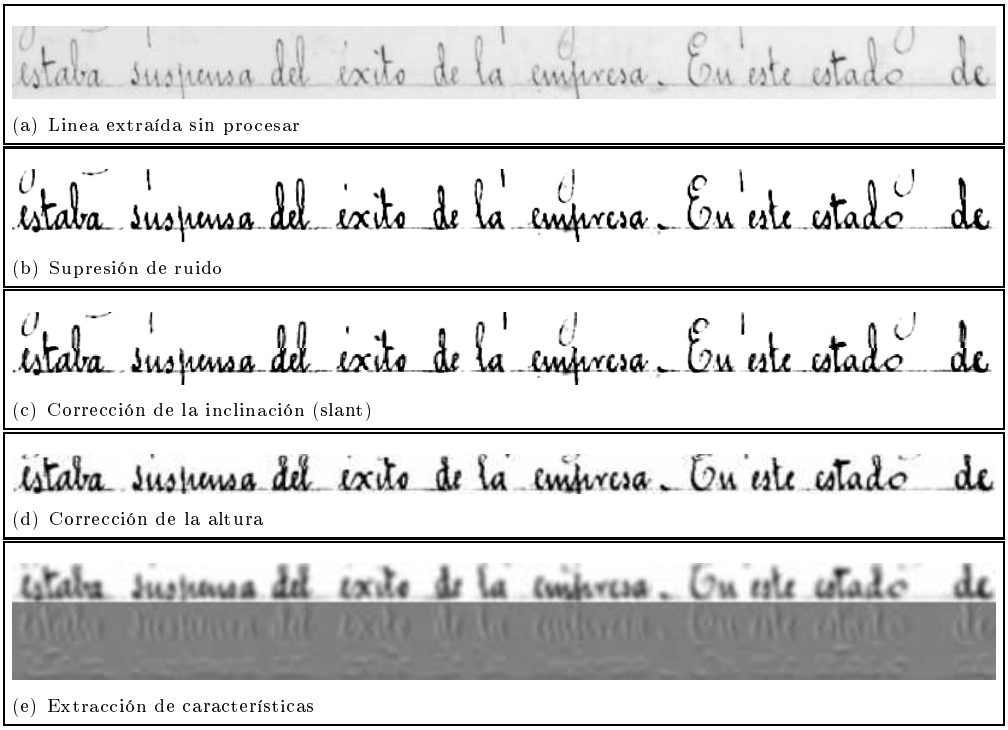
En el caso que nos ocupa, el modelo a nivel de carácter se ha implementado mediante HMMs *izquierda-derecha* continuos. Cada carácter se representa mediante un HMM, que se puede definir como una máquina de estados finitos estocástica que modela secuencias de vectores de características extraídos de instancias de estos caracteres a lo largo del eje horizontal. Por otro lado, las sentencias están formadas por concatenación de palabras que han sido modeladas empleando un *modelo de lenguaje* (ML). Para más información, puede referirse a [1].

Teniendo en cuenta que el libro ha sido escrito por un único autor, es evidente que los modelos a nivel de carácter son iguales para todos los idiomas. Como se puede observar en la figura 3.1, no existen diferencias significativas en la caligrafía. A partir de esta premisa, nos hemos planteado utilizar un modelo léxico y de lenguaje globales para todos los idiomas. Pese a lo que pudiera parecer no es una idea descabellada, ya que cuando el sistema se encuentre con un idioma nuevo, tendrá los modelos de los idiomas previamente entrenados con los que trabajar. Además como el modelo de lenguaje tiene poca historia, no estará muy influenciado por los idiomas que haya entrenado hasta el momento. En cuanto al vocabulario global del sistema, la llegada de un nuevo idioma supondrá nuevas palabras que añadir, al igual que ocurriría si hubiese un vocabulario por idioma.

La estructura del capítulo consistirá en una descripción donde hablaremos sobre cómo se ha llevado a cabo el preproceso, la extracción de características y el entrenamiento de los modelos. Después, presentaremos unos experimentos de referencia, para asentar las bases sobre las que vamos a trabajar. Y a continuación, comentaremos los diferentes experimentos llevados a cabo para comprobar el funcionamiento del sistema monolingüe. Por último, comentaremos los resultados y concluiremos sopesando los pros y contras del uso de un sistema monolingüe para abordar un problema cuya naturaleza es multilingüe.

3.2. Descripción

A partir del corpus de GERMANA, el primer paso consiste en separar las líneas de texto manuscrito y aplicar el preproceso a las imágenes con el fin de simplificar la tarea de reconocimiento. Se aplican correcciones a nivel de ruido, de inclinación (slant) y altura. Para más información sobre este proceso puede referirse a [1].



(a) Linea extraída sin procesar

(b) Supresión de ruido

(c) Corrección de la inclinación (slant)

(d) Corrección de la altura

(e) Extracción de características

Figura 3.2: Ejemplo de preproceso y extracción de características de una línea del GERMANA.

Una vez tenemos las imágenes preprocesadas, el siguiente paso consiste en extraer una secuencia de vectores de características de dimensión fija. Para hacer esto, la imagen preprocesada se divide en una rejilla de celdas cuadradas cuyo tamaño es una pequeña fracción de su altura. Después, cada celda se caracteriza por su nivel de gris normalizado, su derivada horizontal y su derivada vertical. Un ejemplo de todo este proceso lo podemos ver en la figura 3.2 y para más información, puede referirse a [8] donde se describe con más detalle el proceso.

La estimación de los parámetros de los HMMs se ha llevado a cabo con el software *Hidden Markov Model Toolkit* (HTK) [9], que pese a estar destinado a tareas de reconocimiento del habla, puede adaptarse con bastante facilidad al reconocimiento de caracteres aislados. HTK puede usar modelos con una amplia gama de opciones, pero en el caso que se desarrolla se han hecho algunas suposiciones, como puede ser el uso de modelos de izquierda a derecha, puesto que es el sentido de escritura natural. Cada modelo está formado por un número determinado de estados, que en el proceso de entrenamiento se adaptarán a un determinado número de celdas de la rejilla de la imagen, como puede observarse en la figura 3.3 en la página siguiente.

El modelo a nivel léxico, como se comentó en el apartado 3.1 en la página 19, consiste en modelar cada palabra del vocabulario mediante un *Autómata Estocástico de*

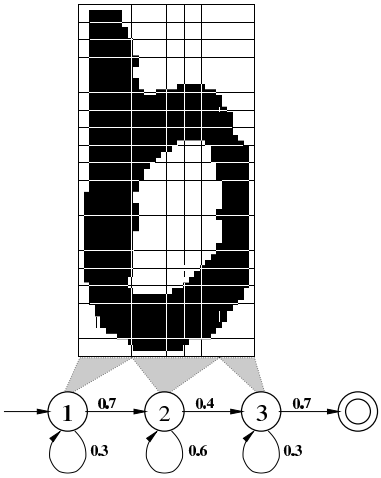


Figura 3.3: HMMs de tres estados que modela la letra 'b'

Estados Finitos (AEEF) que establece las secuencias (permitidas) de caracteres que la conforman. Estos AEEFs se generan automáticamente a partir de un diccionario en el que se lista cada una de las palabras del vocabulario y su correspondiente secuencia de caracteres.
En cuanto al modelo de lenguaje, en primer lugar se preprocesan las transcripciones para aislar caracteres especiales (sobre todo signos de puntuación) con el fin de simplificarlo. Después, se construye un modelo de 2-gramas a partir de las transcripciones preprocesadas mediante el software *SRI Language Modeling Toolkit* (SRILM) [10, 11] con descuento *Kneser-Ney* [12] (en caso de tener una muestra muy pequeña con la que entrenar, se aplica otra variante del mismo o en su defecto el descuento de *Witten-Bell*).

Debido a la estructura secuencial del libro, la tarea básica a realizar consiste en transcribirlo de principio a fin. A diferencia de lo que hacen en otros trabajos como [2] y [8], hemos trabajado con las 20,000 líneas de las que se compone GERMANA. De esta forma, analizaremos el comportamiento del sistema cuando trabajamos con varios idiomas sin realizar ningún tipo de distinción entre ellos.

3.3. Experimentos de referencia

Con el fin de asentar las bases sobre las que vamos a trabajar, en esta sección vamos a realizar unos experimentos de referencia, que no son más que una reproducción de los llevados a cabo en [2]. En cuanto a los parámetros utilizados, podríamos dividirlos en 2 grupos. Por un lado tendríamos los que afectan a la etapa de entrenamiento, como pueden ser el número de estados de los HMMs, el número de componentes por mixtura de los HMMs y el número de iteraciones de entrenamiento en cada división de las mixturas. En nuestro caso hemos utilizado 4 estados, con componentes 64 por

mixtura y 4 iteraciones. Y por otro lado, los parámetros que influyen en la etapa de reconocimiento son el *Grammar Scale Factor* (GSF) y el *Word Insertion Penalty* (WIP) que afectan al peso del modelo de lenguaje sobre los HMMs. En nuestro caso tenemos un GSF de 40 y un WIP de -20. Cabe aclarar que estos valores fueron los utilizados en el citado artículo.

Debido a la estructura secuencial del libro, la tarea básica a realizar consiste en transcribirlo de principio a fin. Puesto que las dos primeras partes de GERMANA, son las únicas que están escritas íntegramente en Castellano, nos centraremos en ellas para su transcripción. Empezando por la página 3, dividimos GERMANA en 9 bloques de 20 páginas cada uno ($3-22$, $23-42$, ..., $163-180$). Después, desde el bloque 2 al 9, cada bloque fue transcrito por el sistema habiendo entrenado con los bloques previos. Por otro lado, utilizamos herramientas y técnicas de uso común para el preproceso, extracción de características, modelado de imagen mediante HMMs y modelado de lenguaje [1]. Los resultados podemos verlos en la figura 3.4 en términos de *Word Error Rate* (WER) por bloque. El WER es el número medio de operaciones de edición necesarias para convertir la línea reconocida en la línea de referencia.

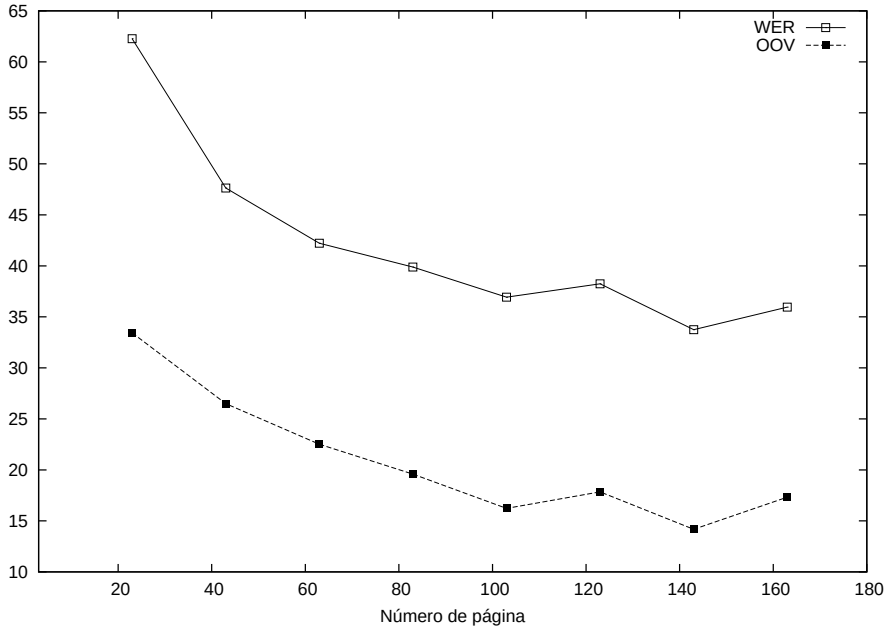


Figura 3.4: *Word Error Rate* (WER) resultado de la transcripción de bloques de páginas de GERMANA. Por cada bloque, el sistema de transcripción se entrena con las páginas de los bloques previos. Además, se incluye el *out-of-vocabulary* words, para analizar el impacto de éste sobre el WER.

Como se esperaba, el WER se decrementa conforme la muestra de entrenamiento aumenta. En particular, el sistema alcanza el 34 % de WER para los dos últimos

bloques, lo que no es un mal resultado si tenemos en cuenta que la mayor parte del error se debe a la ocurrencia de *out-of-vocabulary* (OOV) words. De hecho, hablando en términos relativos, esta parte tiene mucha importancia a medida que avanzamos en la transcripción. Así, podemos observar que en el primer bloque el 54 % de WER se debe a OOV words, mientras que en el último este porcentaje es del 48 %.

3.4. Experimentos sobre la base de datos GERMANA completa

A continuación analizaremos como se comporta el sistema, tratando a todos lo idiomas como si de uno solo se tratara. Para ello lo que hicimos fue entrenar de manera incremental por bloques de 500 líneas (500, 1000, ...,19000, 19500) y obtuvimos el *Word Error Rate* (WER), resultado de reconocer el bloque siguiente ($501 - 1000$, $1001 - 1500$, etc.). Además, para poder apreciar como influye la aparición de nuevos idiomas, se calculó la parte proporcional del WER perteneciente a cada uno. Como resultado tenemos la gráfica de la figura 3.5. Tal y como se puede apreciar, el WER

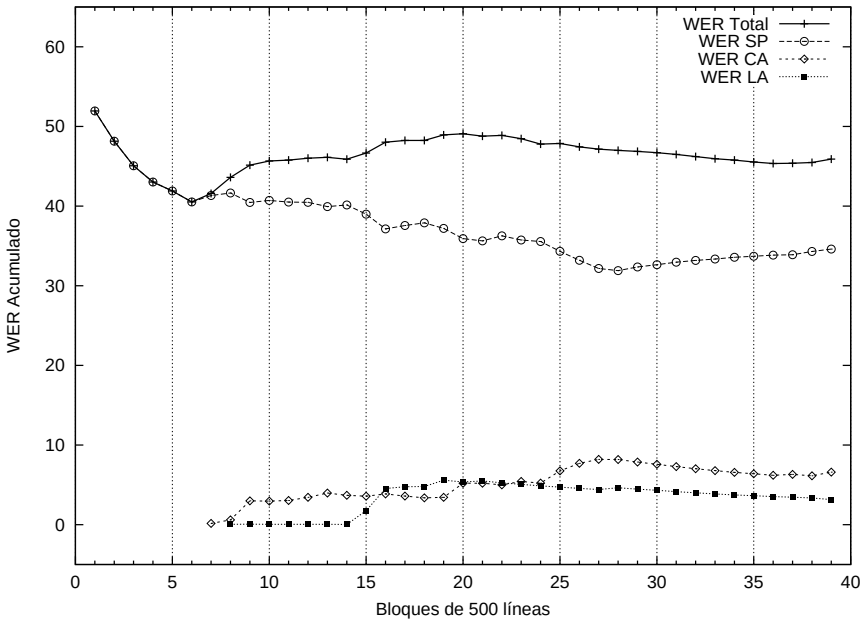


Figura 3.5: WER en función de los bloques de líneas utilizados en el entrenamiento (línea contínua y remarcada). El WER obtenido es sobre el bloque siguiente y acumulado al anterior. Además se incluye el WER proporcional a cada uno de los idiomas Castellano, Catalán y Latín (se han obviado Alemán, Italiano y Francés para mayor claridad y por su insignificante aportación al WER total).

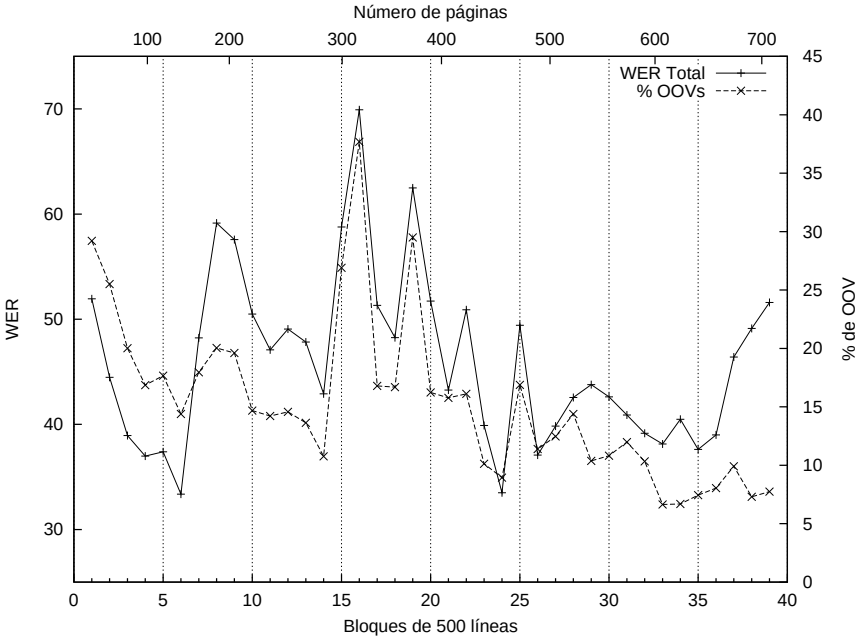


Figura 3.6: WER y porcentaje de WER debido a palabras fuera del vocabulario en función de los bloques de líneas utilizados en el entrenamiento.

va bajando progresivamente hasta el séptimo bloque. A partir de entonces aparece el Catalán y al aportar palabras que seguramente no están en el vocabulario, provoca una subida del WER, cosa que también ocurre con la aparición del Latín. Por otra parte, si nos fijamos en la evolución del WER en Castellano, podemos apreciar como el sistema aprende hasta el bloque 28 y a partir de entonces deja de mejorar, por efecto del sobre-entrenamiento, del reconocimiento de páginas cuyo contenido puede ser más complicado y de la degradación del modelo por la aparición de otros idiomas.

Paralelamente, en la gráfica 3.6, podemos ver como afectan las palabras fuera del vocabulario (OOV) al sistema. Con este fin, se ha optado por dibujar el WER no acumulado para ver como varia en cada momento. Hasta la línea 3500 (bloque 7), se puede apreciar una clara correlación entre el WER y OOV, ya que a medida que se incluyen las palabras fuera del vocabulario en el diccionario, se obtienen mejores resultados. Sin embargo, a partir de entonces, la aparición del Catalán y Latín, provoca un incremento del WER no solo influenciado por OOV, sino también por un incremento en los fallos durante el reconocimiento.

Con el objetivo de explicar con mayor precisión este aumento del WER, se presenta la gráfica 3.7 en la página siguiente, centrada en la región del bloque 7 al 15, donde podemos ver el WER de la gráfica anterior (resultado de entrenar hasta el bloque b y reconocer el bloque $b+1$), junto con el WER resultado de reconocer cada 10 líneas. Es decir, se entrena al igual que antes por bloques de 500 líneas (500, 1000, ..., 19000,

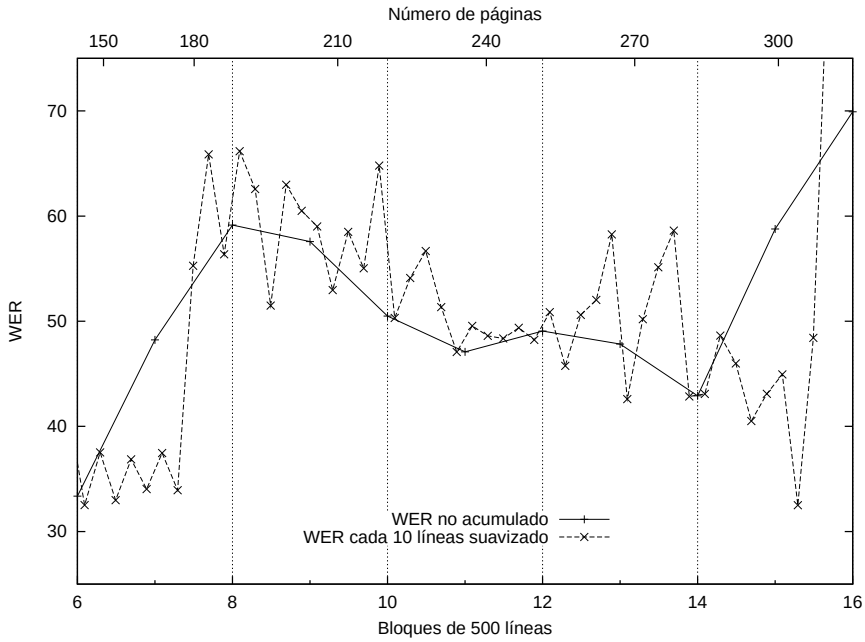


Figura 3.7: WER en función de los bloques de líneas utilizados en el entrenamiento y WER obtenido al reconocer cada 10 líneas sobre el bloque siguiente. Si por ejemplo entrenamos hasta el primer bloque (500 líneas), después obtendremos el WER al reconocer las líneas, $501-511$, $511-521$, ... , $981-991$.

19500), pero a la hora de reconocer, no lo hacemos del bloque siguiente, sino cada 10 líneas durante el bloque siguiente (si hemos entrenado 500 líneas, obtendremos el WER al reconocer desde la $501-511$, después de $511-521$, etc). Además se ha suavizado para mejorar su visualización. Con todo esto lo que se obtiene es una gráfica donde podemos ver con total precisión, qué regiones del libro son más complicadas para el reconocedor.

Tal y como se puede apreciar, el aumento de WER se situa en el intervalo que va desde la página 180 a la 290 aproximadamente. Si miramos el capítulo 2, podemos ver que además de que coincide con la sección *Notas* (páginas 181 a 282), también coincide con que el autor empieza a utilizar la plantilla de 32 líneas. Se podría pensar en un primer momento que no tiene porqué influir en el reconocimiento. Pero si nos fijamos en la imagen 3.8, donde mostramos dos ejemplos de líneas extraídas sobre una plantilla de 24 líneas y otra de 32 respectivamente, podemos ver que el autor emplea una caligrafía más apretada. Además de que al haber más líneas en el mismo espacio, cada una de ellas es más estrecha, por lo que puede aumentar el ruido provocado por las otras líneas que la rodean.

Volviendo a la gráfica 3.6, hasta el bloque 28 se explica el WER con bastante precisión por medio de las palabras fuera del vocabulario. Principalmente, hay dos subidas

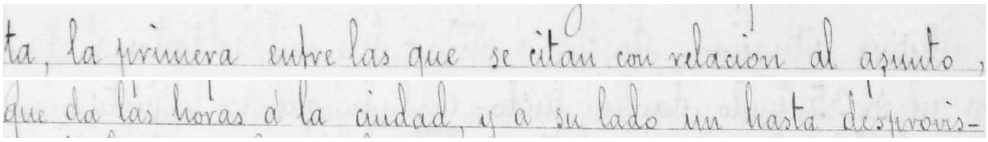


Figura 3.8: Líneas 18 y 20 de las páginas 153 y 181 de GERMANA. Ambas líneas están a la misma escala que nos podemos encontrar en el libro.

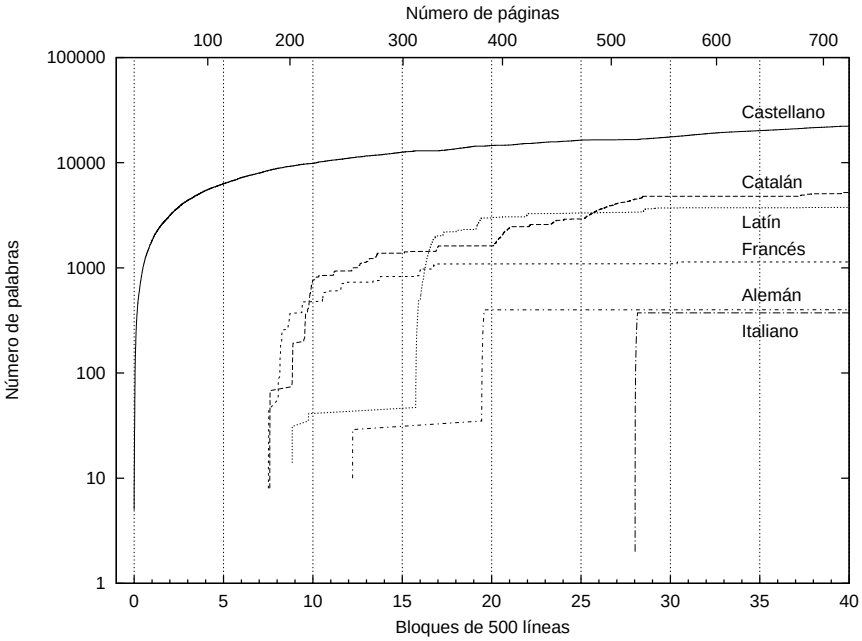
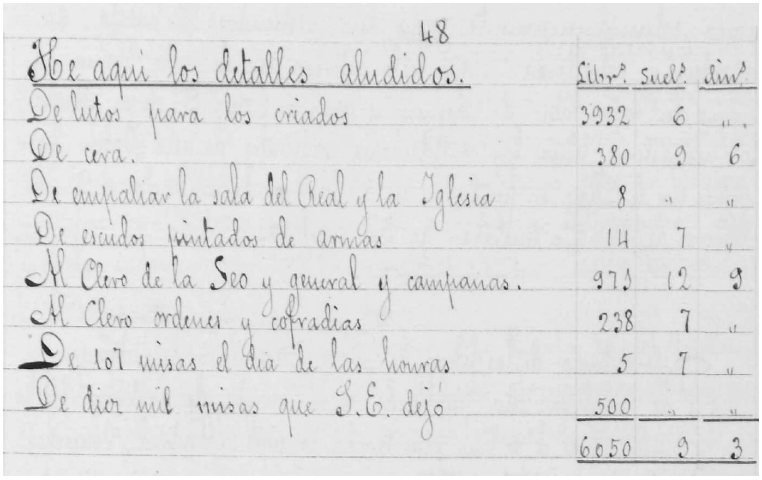


Figura 3.9: Número de palabras de cada idioma en función de los bloques de líneas de GERMANA.

en el bloque 16 y 19, debido a la aparición del Latín y Alemán, tal y como se puede contrastar con la gráfica 3.9. Donde se muestra la evolución del número de palabras de cada idioma y por consiguiente también podemos apreciar en qué momento aparece cada uno. Como decíamos, en este intervalo se relaciona con bastante precisión el aumento del WER con el número de palabras fuera del vocabulario, sin embargo no se mantiene en el bloque 22. En este bloque (página 440 aproximadamente) nos encontramos en la sección de *Documentos* y más concretamente, en estas páginas el autor copia una serie de cartas donde se abusa constantemente de frases que ocupan dos líneas y también realiza constantes cambios en el estilo de la caligrafía para destacar ciertas palabras. A partir del bloque 28 (página 540 aproximadamente), se vuelve a perder la relación con las palabras fuera del vocabulario, ya que es donde comienzan las dos últimas secciones (*Ilustraciones y Parte Final*). A lo largo de las mismas, nos

encontramos con muchas listas, como por ejemplo la de la imagen 3.10 y documentos aislados (como cartas o notas) 3.11, con el consiguiente cambio de caligrafia.

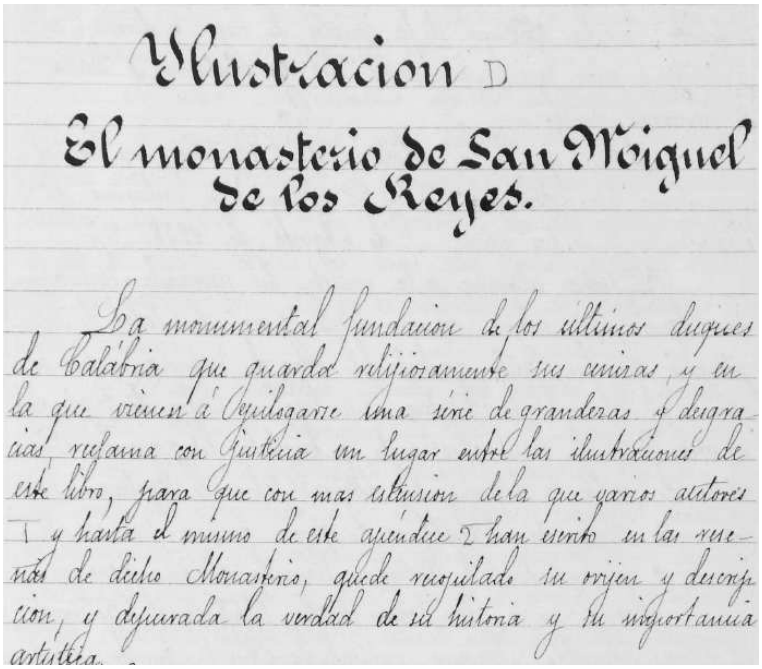


48

Hoz aqui los detalles aludidos.

	Libr.s	Suel.s	din.s
De lutos para los criados	3932	6	"
De cera.	380	9	6
De empaliar la sala del Real y la Iglesia	8	"	"
De escudos pintados de armas	14	7	"
Al Clero de la Seo y general y campanas.	975	12	9
Al Clero ordenes y cofradias	238	7	"
De 107 misas el dia de las honras	5	7	"
De dicr mil misas que S.E. dejó	500	"	"
	6050	9	3

Figura 3.10: Lista de ejemplo presente en la página 681.



Ylustracion D

El monasterio de San Miguel de los Reyes.

La monumental fundacion de los ultimos duques de Calabria que guarda religiosamente sus cenizas, y en la que vienen á perpetuarse una série de grandezas y desgracias, reclama con justicia un lugar entre las ilustraciones de este libro, para que con mas estension de la que varios autores I y hasta el mismo de este apéndice 2 han escrito en las reseñas de dicho Monasterio, quede recopilada su origen y descripcion, y dilucidada la verdad de su historia y su importancia artistica.

Figura 3.11: Caligrafía de otros autores. Página 687.

	Adaptación de parámetros		Cálculo de WER	
Iteración	Entrenamiento	Validación	Entrenamiento	Test
1	-	-	Bloque 1	Bloque 2
2	Bloque 1	2	Bloques 1 y 2	Bloque 3
3	Bloques 1 y 2	3	Bloques 1, 2 y 3	Bloque 4
...	...	...	...	...
39	Bloques 1, ..., 37	Bloque 38	Bloques 1, ...,39	Bloque 40

Cuadro 3.1: Desarrollo del experimento adaptativo.

3.5. Adaptación de GSF y WIP

Otro de los experimentos que hemos realizado ha consistido en adaptar los parámetros *Grammar Scale Factor* (GSF) y *Word Insertion Penalty* (WIP) a medida que avanzamos en la transcripción. Dado que la estructura de las líneas va variando a lo largo de todo el libro, puede resultar interesante ir modificando estos parámetros para alterar el peso de los HMMs sobre el modelo de lenguaje.

Al igual que antes, hemos trabajado por bloques de 500 líneas. Para adaptar los parámetros, lo hemos hecho de la manera más representativa posible. Es decir, vamos entrenando de manera incremental por bloques (500, 1000, ..., 19000, 19500) habiendo adaptado sobre el último bloque entrenado, es decir, si debemos entrenar hasta el bloque 3, adaptamos utilizando como bloque de validación el bloque 3. Después, se obtiene el WER resultado de reconocer el bloque siguiente ($501-1000$,$1001-1500$,etc) con esos parámetros previamente adaptados. En la tabla 3.1 se muestra claramente el proceso explicado y como se puede ver, en la primera iteración no hay adaptación de parámetros, ya que se utilizan los del experimento base.

El resultado de todo esto lo tenemos en las gráficas 3.12 y 3.13, donde comparamos el sistema base y el adaptativo para apreciar la mejora obtenida.

Gracias a la adaptación del GSF y WIP, se reduce progresivamente el WER obtenido. Hasta el bloque 14, la mejora es muy pequeña porque los parámetros apenas varian con respecto a los de partida, pero a partir de este punto es cuando aparece la plantilla de 32 líneas. De esta manera, conforme avanzamos en la transcripción, estos parámetros se adaptan a las nuevas características del texto, alterando el peso de los HMMs sobre el modelo de lenguaje para reducir el WER cometido. Con lo que se consigue una mejora gradual que llega a los 1,6 puntos respecto al original.

3.6. Conclusiones

Los experimentos mostrados en este capítulo, nos sugieren que el uso de un sistema monolingüe para una tarea de naturaleza multilingüe, no funciona mal. El principal inconveniente con el que nos encontramos, es la heterogeneidad de GERMANA, ya que a partir de la página 180 el autor cambia tanto la plantilla de escritura, lo que conlleva un cambio en la caligrafía y ruido por el contexto en el que se encuentra cada línea, como el cambio del contenido, ya que nos encontramos con listas, tablas,

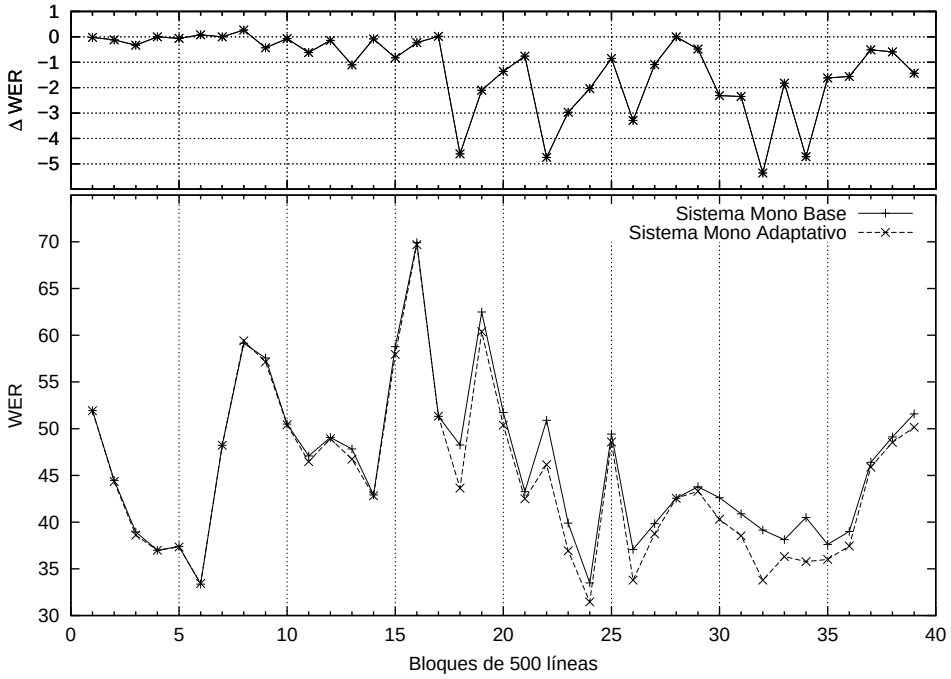


Figura 3.12: Abajo: WER para el sistema mono base y el sistema mono adaptativo, en función de los bloques de líneas utilizados en el entrenamiento. Arriba: Diferencia de WER en cada instante de los dos sistemas. Dado que el sistema adaptativo mejora, tenemos un incremento negativo.

aportaciones de otros autores, etc. Desde este punto de vista, los resultados obtenidos con GERMANA son fiables, en el sentido de que en una tarea real nos podemos encontrar con este tipo de problemas.

Por otro lado, los experimentos para la adaptación automática de parámetros, han dado resultados muy esperanzadores, sobretodo porque las mejoras obtenidas han ido en aumento cuanto más grande se hacia la muestra de entrenamiento y porque además, es totalmente aplicable a una tarea real.

En el próximo capítulo, analizaremos el comportamiento del sistema multilingüe y veremos entonces, si se mejoran los resultados obtenidos y en su caso, tendremos que ver si compensa esa mejora respecto al coste de entrenar un modelo diferente por lenguaje.

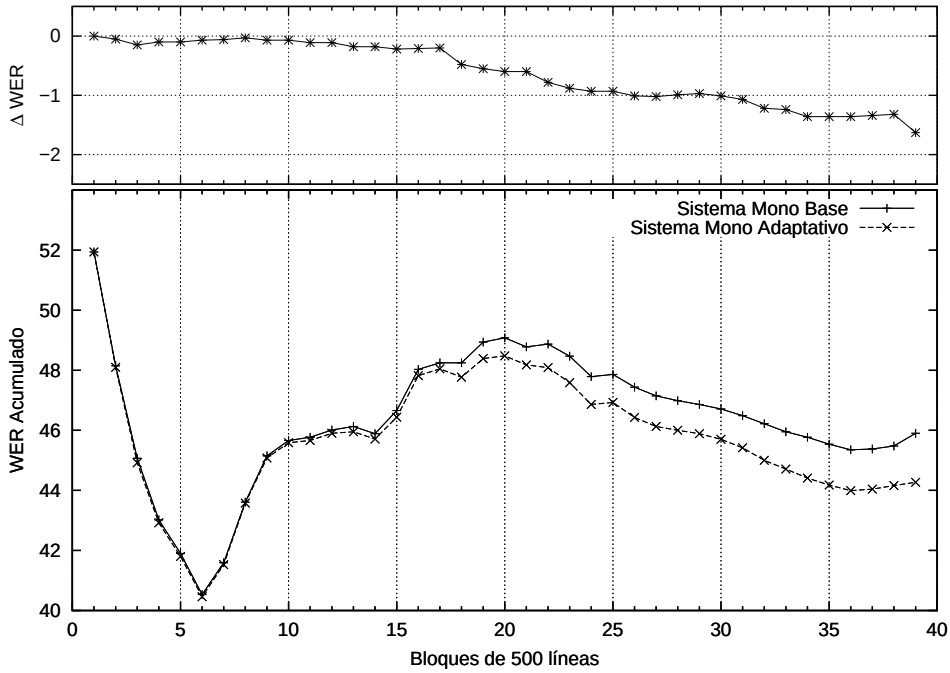


Figura 3.13: Abajo: WER (acumulado) para el sistema mono base y el sistema mono adaptativo, en función de los bloques de líneas utilizados en el entrenamiento. Arriba: Diferencia de WER en cada instante de los dos sistemas. Dado que el sistema adaptativo mejora, tenemos un incremento negativo.

CAPÍTULO 4

SISTEMA MULTILINGÜE

4.1. Introducción

En este capítulo, nos enfrentaremos a la tarea presentada en el capítulo 3 desde un punto de vista diferente. Dado que el libro está escrito en varios idiomas, asumiremos esta premisa y los trataremos a cada uno de ellos por separado. Como ya se comentó en los capítulos 2 y 3, el libro está escrito por un solo autor, por tanto los modelos que se entrenan a nivel de carácter no tienen porqué diferir entre idiomas. Sin embargo, dado que cada lengua tiene un vocabulario distinto, sí que diferirán en el modelo léxico y aún es más, como las líneas se componen de manera diferente, también tendrán un modelo de lenguaje único. Esto nos lleva a realizar un análisis del comportamiento del sistema tratando cada idioma por separado. La teoría sugiere que los modelos se adaptarán de una manera más ajustada y por tanto, deberían mejorar los resultados. No obstante, tendremos que valorar el coste del entrenamiento de varios modelos por idioma, frente a la hipotética mejora del sistema con respecto al sistema monolingüe.

La estructura del presente capítulo es muy similar a la del anterior. Comentaremos someramente la etapa de preproceso, extracción de características y entrenamiento de los modelos. Después, desarrollaremos los diferentes experimentos llevados a cabo para poner a prueba el sistema, siempre con el objetivo de comparar los resultados con los del sistema monolingüe. Y para finalizar, analizaremos las ventajas y los inconvenientes de ambos.

4.2. Descripción

Nuestro principal objetivo es decidir cuál de los dos sistemas se comporta mejor a la hora de transcribir GERMANA. Por este motivo, las etapas de preproceso y extracción de características son idénticas para ambos sistemas 3.2. En lo que se refiere al entrenamiento, como el libro está escrito por un solo autor, utilizaremos un mismo modelo a nivel de carácter para todos los idiomas ya que la caligrafía no cambiará, tal y como puede apreciarse en la imagen 3.1 en la página 20. En cuanto al modelo léxico, emplearemos uno diferente por cada idioma, pues es obvio que no tienen porqué compartir el mismo vocabulario. Y en lo que al modelo de lenguaje se refiere, cada idioma construye de manera diferente las frases, por tanto también usaremos un modelo único para cada uno.

La tarea básica a realizar en este capítulo consistirá también en transcribir las 20,000 líneas de GERMANA, pero con las restricciones anteriormente comentadas. Además, cabe mencionar el hecho de cómo averiguar el idioma de cada una de las líneas. A este respecto, supondremos que este dato es conocido (en el corpus están etiquetadas las líneas con su idioma). Con lo que obtendremos unos resultados optimistas, es decir, una cota inferior del *Word Error Rate* (WER) al que podemos llegar suponiendo que el proceso de detección del idioma es perfecto.

4.3. Experimentos sobre la base de datos GERMANA completa

Los parámetros iniciales utilizados en este sistema fueron los mismos que se utilizaron en el capítulo anterior; 4 estados, 64 componentes para las mixturas, 4 iteraciones de entrenamiento en cada división de las mixturas, 40 de GSF y -20 de WIP. Para mayor claridad, dividimos el corpus de GERMANA en bloques de 500 líneas. El proceso de entrenamiento consistió (al igual que en el sistema monolingüe) en entrenar incrementalmente cada bloque (500, 1000, ..., 19000, 19500) y reconocer el siguiente ($501-1000$, $1001-1501$, ..., $19501-20000$). Es decir, el WER obtenido en la iteración x se corresponde con haber entrenado hasta el bloque x y reconocido el bloque $x+1$. Además, como se comentó en la sección anterior, vamos a suponer que el idioma de cada línea es conocido. Por tanto, los resultados obtenidos en este capítulo serán una cota inferior de los resultados que se podrían obtener en un sistema que predijese el idioma. Aún así, no distarán mucho de la realidad, pues una posibilidad para predecir el idioma seria suponer que el idioma de una línea, es el de la línea anterior, con lo que se obtendría un porcentaje de acierto muy elevado.

Los resultados los podemos ver en la gráfica 4.1 en la página siguiente, en términos de WER acumulado para los idiomas Castellano, Catalán y Latín. Se han omitido Alemán, Italiano y Francés por una mayor claridad al presentar los resultados. Si nos fijamos detenidamente, el reconocimiento del Catalán y Latín, comienza una iteración más tarde con respecto al sistema monolingüe. Esto es porque en ese sistema, la primera vez que aparece un idioma diferente del Castellano, es reconocido utilizando los modelos entrenados hasta el momento (son comunes a todos los idiomas). Sin em-

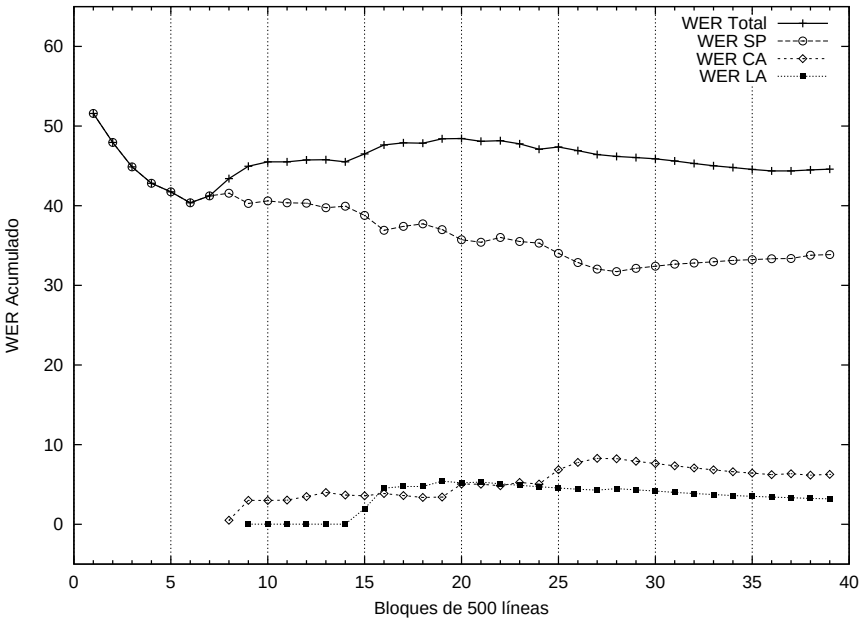


Figura 4.1: WER en función de los bloques de líneas utilizados en el entrenamiento (línea contínua y remarcada). El WER obtenido es sobre el bloque siguiente y acumulado al anterior. Además se incluye el WER proporcional a cada uno de los idiomas Castellano, Catalán y Latín (se han obviado Alemán, Italiano y Francés para mayor claridad y por su insignificante aportación al WER total).

bargo, en el sistema multilingüe, en el momento en que aparece por primera vez un idioma, no se puede llevar a cabo la tarea de reconocimiento, ya que no se dispone de los modelos entrenados del mismo. En cuanto a la gráfica, podemos observar como la evolución del WER sigue un patrón muy similar al del sistema monolingüe. En lo que respecta al WER total, el sistema sigue una tendencia decreciente hasta el bloque 7, a partir del cual, el autor comienza a utilizar una plantilla diferente (de 32 líneas) y aparecen otros idiomas como Catalán o Latín lo que provoca una penalización en el WER. Por otra parte, si nos fijamos en la evolución del WER en Castellano, el sistema aprende hasta el bloque 28, y a partir de entonces se satura por el sobre-entrenamiento y por la aparición de líneas cuyo contenido puede resultar más complicado de reconocer. En los demás idiomas, existe una tendencia similar, aunque al no tener una muestra tan grande como la del Castellano, en algunos casos no llega a saturarse.

En la gráfica 4.2 en la página siguiente, presentamos la relación de WER (no acumulado) frente al procentaje de WER debido a *out-of-vocabulary* words (OOV). Como se puede apreciar, se sigue una evolución muy similar a la presentada en el sistema monolingüe, pero con ciertas diferencias. Hasta el bloque 14 el patrón es el mismo, porque tenemos un decremento de WER acorde al decremento de OOVs

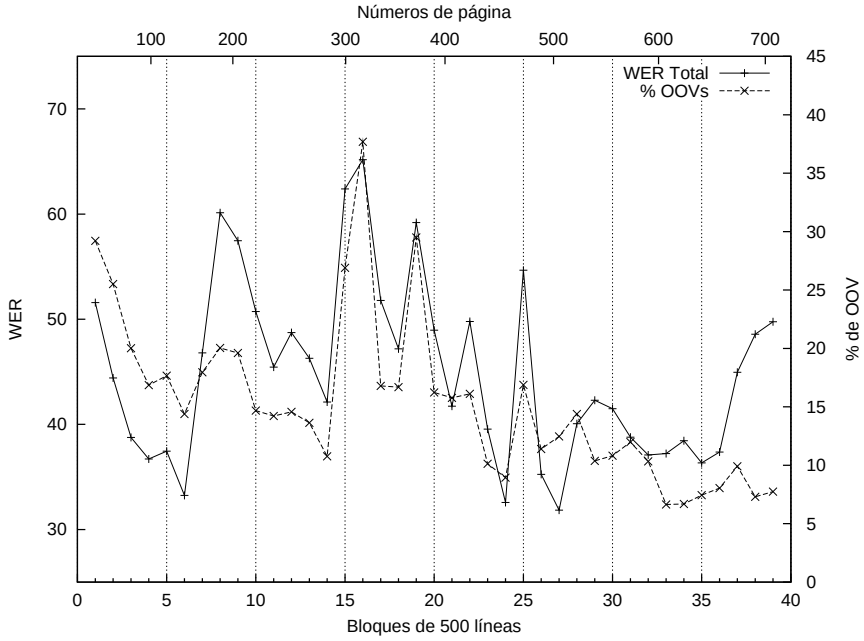


Figura 4.2: WER y porcentaje de WER debido a palabras fuera del vocabulario en función de los bloques de líneas utilizados en el entrenamiento.

hasta el bloque 7 y después de éste, tenemos una subida muy pronunciada debido principalmente (como ya dijimos en el capítulo 3 en la página 19) al cambio de la plantilla sobre la que escribe el autor, que pasa de 24 a 32 líneas y a la aparición de nuevos idiomas. Por otra parte, tenemos dos grandes subidas en los bloques 16 y 19 debidas principalmente a las OOV words, pero son menos acentuadas que en el sistema monolingüe porque coincide con un incremento notable del peso de otros idiomas, como el Latín (bloque 16) y Alemán (bloque 19) que al ser reconocidos por sus propios modelos, se ajustan mejor y se obtienen mejores resultados. Hasta el bloque 28 se mantiene otra vez la relación de WER con las palabras fuera del vocabulario, pero de ahí hasta el final esto se pierde porque es donde comienzan las secciones *Ilustraciones* y *Parte Final*. En ellas, la tarea de reconocimiento es mucho más complicada porque nos encontramos con muchas listas, tablas y aportaciones de otros autores (con su correspondiente cambio de caligrafia) lo que nos lleva a cometer muchos errores en la transcripción.

Llevando a cabo un análisis más detallado del sistema, presentamos la figura 4.3 en la página siguiente donde se muestra una comparativa de la evolución del WER acumulado obtenido por los dos sistemas, de los 3 idiomas con más presencia en GERMANA. En Catalán y Latín tenemos una evolución muy similar, el sistema multilingüe comienza peor que el monolingüe porque apenas ha sido entrenado (muestra de entrenamiento muy pequeña) y el monolingüe puede reconocer con los modelos en-

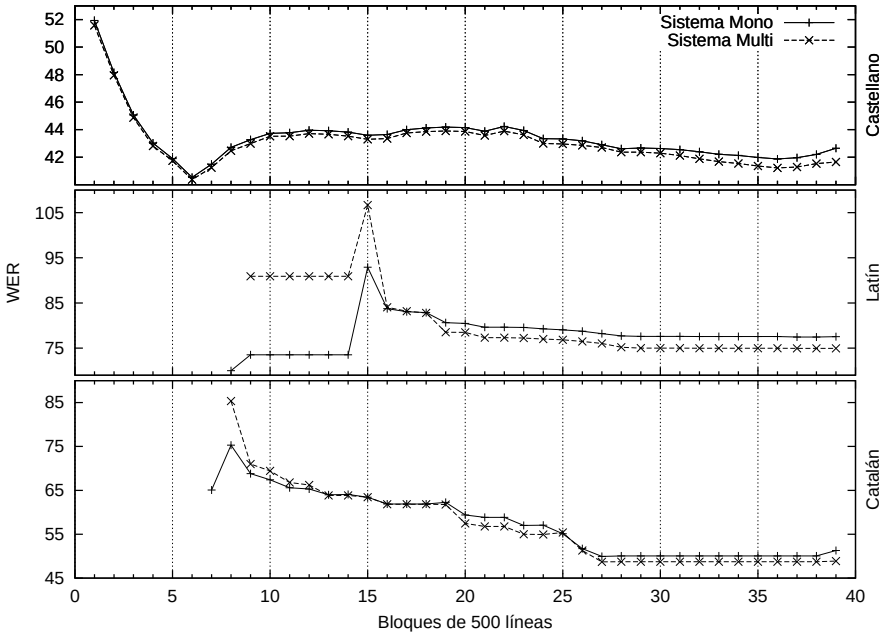


Figura 4.3: Comparativa del Sistema Monolingüe frente al Sistema Multilingüe en términos de WER (acumulado) por idioma (Castellano, Latín y Catalán).

trenados por otros idiomas (muestra de entrenamiento mayor). Sin embargo, conforme aportamos nuevas transcripciones al conjunto de entrenamiento, el sistema multilingüe obtiene mejores resultados ya que su muestra se ajusta más al idioma que se está reconociendo. En el caso del Castellano, también funciona mejor el sistema multilingüe, pero al principio los dos sistemas dan resultados muy similares porque al ser la primera lengua en aparecer, sus muestras de entrenamiento son idénticas. El resultado global de ambos sistemas lo podemos ver en la figura 4.4 en la página siguiente, en términos de WER acumulado. Tal y como se esperaba, el sistema multilingüe funciona mejor, llegando a reducir el WER final en 1,31 puntos (de 45,9 en el sistema mono, hemos pasado a 44,59 en el sistema multi).

4.4. Adaptación de GSF y WIP

En la sección previa, hicimos la transcripción de todo el corpus de GERMANA con los parámetros fijados a unos valores determinados. Estos valores fueron los mismos que se utilizaron en el artículo [2], donde se adaptaron para las primeras páginas del GERMANA. Sin embargo, puesto que el libro va variando su estructura y contenido, puede resultar interesante adaptar estos parámetros continuamente. Más concretamente, adaptamos el *Grammar Scale Factor* y el *Word Insertion Penalty* que son los

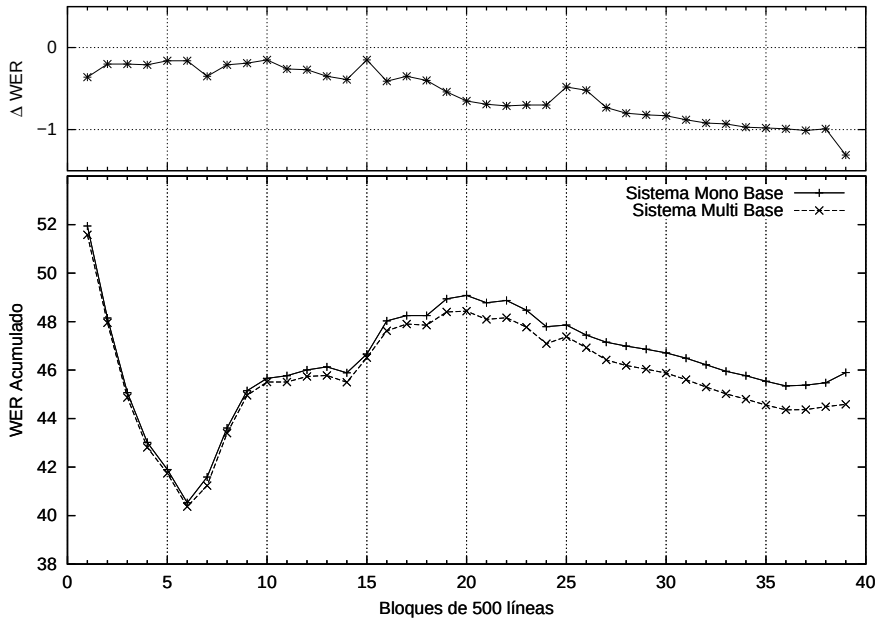


Figura 4.4: Abajo: Comparativa del Sistema Monolingüe frente al Sistema Multilingüe en términos de WER (acumulado) en función del los bloques de líneas entrenados. Arriba: Incremento de WER producido por el Sistema Multilingüe frente al Monolingüe.

que intervinen en la fase de reconocimiento.

Al igual que en los experimentos previos, hemos trabajado a nivel de bloques de 500 líneas. Para adaptar los parámetros vamos entrenando incrementalmente por bloques (500, 1000, ..., 19000, 19500) habiendo adaptado sobre el último bloque entrenado. Es decir, si tenemos que entrenar hasta el bloque x, utilizamos desde el bloque 1 hasta el $x-1$ como entrenamiento y el x para validar. Una vez obtenidos los valores que mejor resultado han dado, entrenamos normalmente hasta el bloque x y obtenemos el WER al reconocer el bloque siguiente con esos parámetros, para mayor claridad puede consultar la tabla 3.1 en la página 29. El resultado del sistema multilingüe adaptativo frente al sistema multilingüe base, lo podemos ver en la figura 4.5 en la página siguiente en términos de WER no acumulado. Tal y como se puede apreciar en la parte superior de la figura, el sistema tiene una tendencia clara a reducir el WER con respecto al sistema base. Sin embargo, por cuestiones de coste computacional en la fase de adaptación, se limitó en un orden de magnitud el número de estados activos con los que trabajaba el reconocedor, lo que provocó el empeoramiento reflejado en los bloques 7, 8 y 9.

En líneas generales, como se puede apreciar en la gráfica 4.6 en la página 40, el uso de un sistema adaptativo es muy positivo ya que permite obtener una mejora gradual

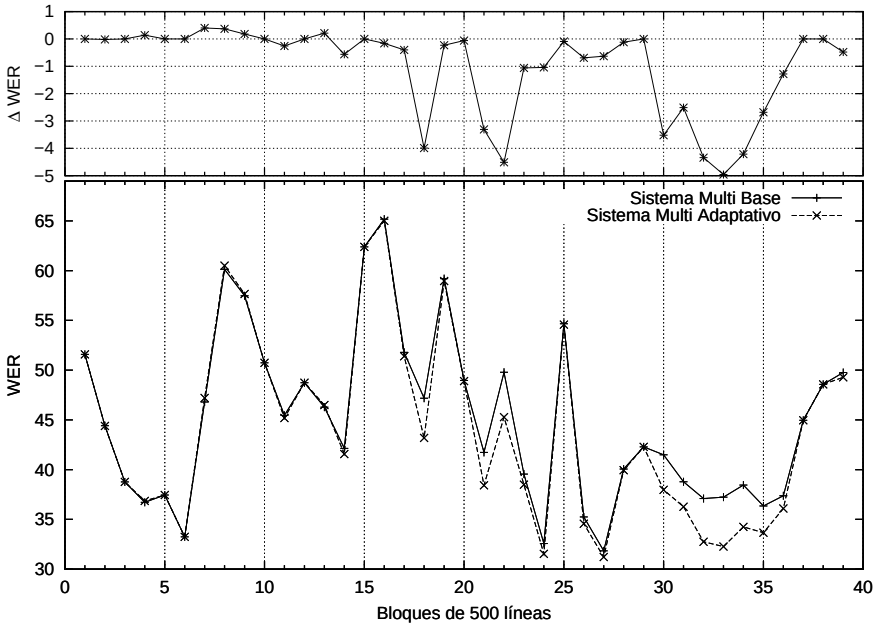


Figura 4.5: Abajo: WER para el sistema multi base y el sistema multi adaptativo, en función de los bloques de líneas utilizados en el entrenamiento. Arriba: Diferencia de WER en cada instante de los dos sistemas. Dado que el sistema adaptativo mejora, tenemos un incremento negativo.

de los resultados. En nuestro caso, hemos alcanzado una mejora de hasta 1,09 puntos sobre el original, obteniendo un WER final de 43,5.

Por otra parte, otro punto interesante es comparar los dos sistemas en sus versiones adaptativas. En la gráfica 4.7 en la página 41 tenemos los resultados de los dos sistemas adaptativos en términos de WER acumulado. Como se puede ver, el sistema multilingüe adaptativo funciona ligeramente mejor que el sistema monolingüe. Seguramente se debe a que al utilizar modelos de lenguaje diferentes para cada idioma, los hace más ajustados y son capaces de transcribir con más precisión. En parte impulsado por el hecho de que disponemos de vocabularios independientes más reducidos, en lugar de un solo vocabulario global.

4.5. Conclusiones

Los resultados obtenidos en este capítulo, sugieren que el sistema multilingüe funciona ligeramente mejor que el sistema monolingüe. Sin embargo, estos resultados no son más que una cota inferior de lo que se obtendría con un sistema multilingüe real, ya que como comentamos en la introducción, hemos supuesto que el idioma de cada línea es conocido. Es evidente por tanto, que si el idioma no es conocido, se

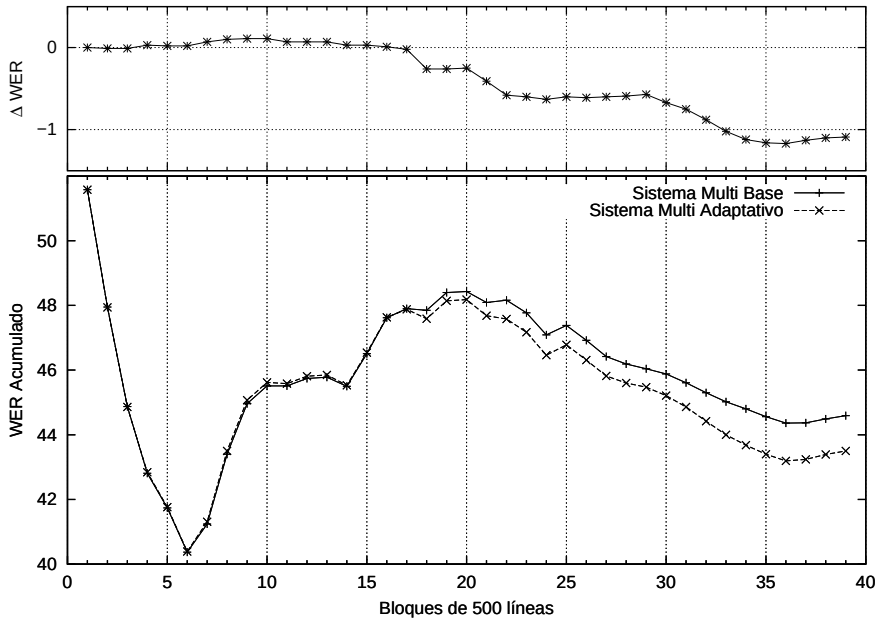


Figura 4.6: Abajo: WER (acumulado) para el sistema multi base y el sistema multi adaptativo, en función de los bloques de líneas utilizados en el entrenamiento. Arriba: Diferencia de WER en cada instante de los dos sistemas. Dado que el sistema adaptativo mejora, tenemos un incremento negativo.

obtendrían resultados más deficientes y será este el tema del siguiente capítulo.

Por otra parte, al igual que sucedió en el sistema monolingüe, la adaptación dinámica de los parámetros abre una vía para la mejora de los resultados. Hemos conseguido mejorar el sistema adaptando solamente el *Grammar Scale Factor* y el *Word Insertion Penalty*, por lo que si intentamos ampliar la adaptación a más parámetros, seguramente se puedan conseguir mejores resultados de los presentados.

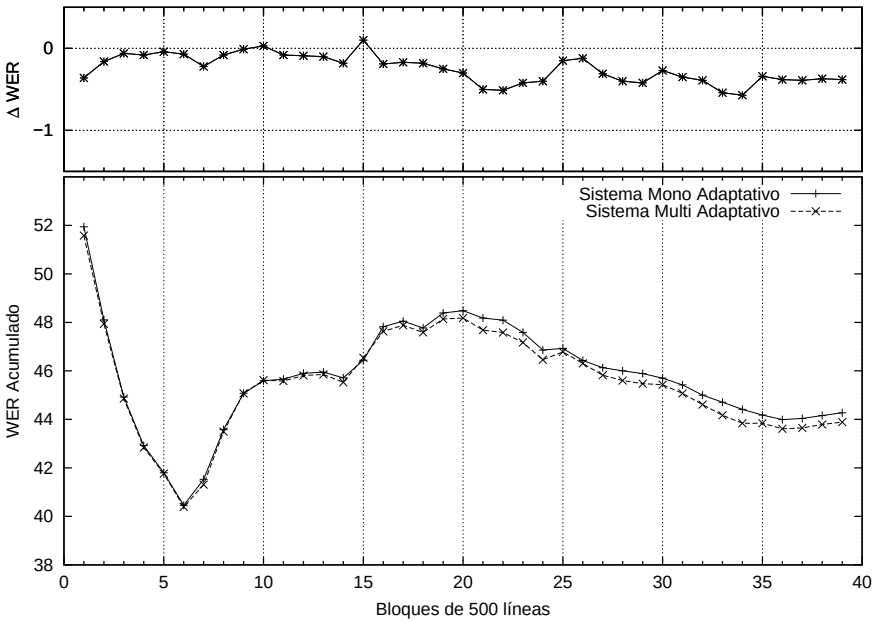


Figura 4.7: Abajo: Comparativa de WER (acumulado) entre el sistema mono adaptativo y el sistema multi adaptativo, en función de los bloques de líneas utilizados en el entrenamiento. Arriba: Incremento de WER en cada instante entre los dos sistemas. Como el sistema multi adaptativo es mejor, tenemos un incremento negativo.

CAPÍTULO 5

PREDICCIÓN DEL IDIOMA

5.1. Introducción

En el capítulo anterior obtuvimos una serie de resultados en el Sistema Multilíngüe, bajo la suposición de que el idioma de cada frase era conocido. Pues bien, dentro de este contexto, nos podríamos cuestionar qué ocurriría si este dato fuera desconocido, dado que la anterior suposición no es realista. En el artículo [13] proponen utilizar una serie de clasificadores bien conocidos: El vecino más cercano, el prototipo más cercano, naive Bayes y Support Vector Machines para averiguar el idioma de documentos web. En este marco, intentaremos poner a prueba alguno de los modelos propuestos, para ver cómo funcionan en nuestro caso.

La estructura de este capítulo aboradará principalmente la predicción basada en el idioma de la línea anterior y la predicción basada en naive Bayes. Presentaremos ambos modelos y compararemos los resultados obtenidos con los del sistema multilingüe del capítulo anterior (idioma conocido) y con los del sistema monolingüe.

5.2. Descripción

Nuestro objetivo es repetir los experimentos realizados en el capítulo 4, pero con la dificultad añadida de la predicción del idioma. En cuanto a las etapas de preproceso y extracción de características, seguiremos el mismo procedimiento que en los sistemas previos. Por otro lado, entrenaremos los modelos de la misma forma en que lo hicimos para el sistema multilingüe, es decir, utilizando un modelo a nivel de carácter idéntico

para todos los idiomas, y un modelo léxico y de lenguaje diferente para cada uno.

La principal novedad en este capítulo será la inclusión de un modelo predictor que estime el idioma de cada frase, que se encargará de decidir qué modelos aplicar en la etapa de reconocimiento según el idioma estimado.

5.3. Predicción basada en la línea anterior

Dada la estructura del corpus de GERMANA, una primera aproximación para la estimación del idioma en caso de proceder a su transcripción secuencial (tal y como habíamos hecho hasta ahora), es suponer que el idioma de cada línea es el de la línea anterior. Con esto pretendemos obtener un porcentaje de acierto muy elevado, ya que la frecuencia de cambio de idioma en un texto de estas características es muy baja. Cabe aclarar que los experimentos expuestos de ahora en adelante trabajan con los mismos parámetros utilizados en la sección 4.3; HMMs de 4 estados, 4 iteraciones de entrenamiento en cada división de las mixturas, 64 componentes para las mixturas, 40 de GSF y -20 de WIP. El experimento realizado en esta sección ha consistido en entrenar el sistema por bloques de 500 líneas de manera incremental (500, 1000, ..., 19000, 19500) y después obtener el WER al reconocer sobre el bloque siguiente en cada iteración ($501-1000$, $1001-1501$, ..., $19501-20000$). A la hora de averiguar el idioma, como comentamos anteriormente, supusimos que el idioma de cada nueva línea era el de la línea anterior. Con todo esto hemos obtenido un porcentaje de acierto en la predicción del idioma de un 97,6 %, y su efecto sobre el sistema queda reflejado en la gráfica 5.1 en la página siguiente.

Tal y como se puede observar, comienza a empeorar el sistema a partir del bloque 7 (línea 3500) que es cuando aparecen Catalán y Latín. Desde el bloque 23 al 28 (líneas 11500 a 14000) el sistema registra el mayor aumento de WER porque en esta sección se alternan con bastante frecuencia Castellano y Catalán sobre todo, aunque también tenemos Latín, Francés y Alemán. Para explicar este aumento puede referirse a la gráfica 5.2 en la página 46 donde se muestra el número de cambios de idioma por bloque, donde podemos ver que efectivamente, en el rango de bloques comentado tenemos el mayor número de cambios de idioma.

Por otro lado, en la gráfica 5.3 en la página 47 tenemos la comparativa del WER acumulado para los dos sistemas. Se puede ver que el sistema con predicción del idioma solo empeora en 0,36 (WER final de 44,95) puntos sobre el sistema ideal (el que conoce el idioma) que sigue siendo inferior al resultado obtenido por el sistema monolingüe (45,9).

5.4. Predicción por naive Bayes

Otro de los experimentos realizados ha consistido en predecir el idioma con el modelo clasificador de texto basado en naive Bayes, que es conocido por ser un modelo robusto, ligero y fácil de actualizar. En este sentido, el idioma de una frase f se puede predecir tal y como se presenta en la ecuación 5.1 en la página siguiente. Donde L es el conjunto de idiomas en el conjunto de entrenamiento, N_{f,t_j} es la frecuencia del

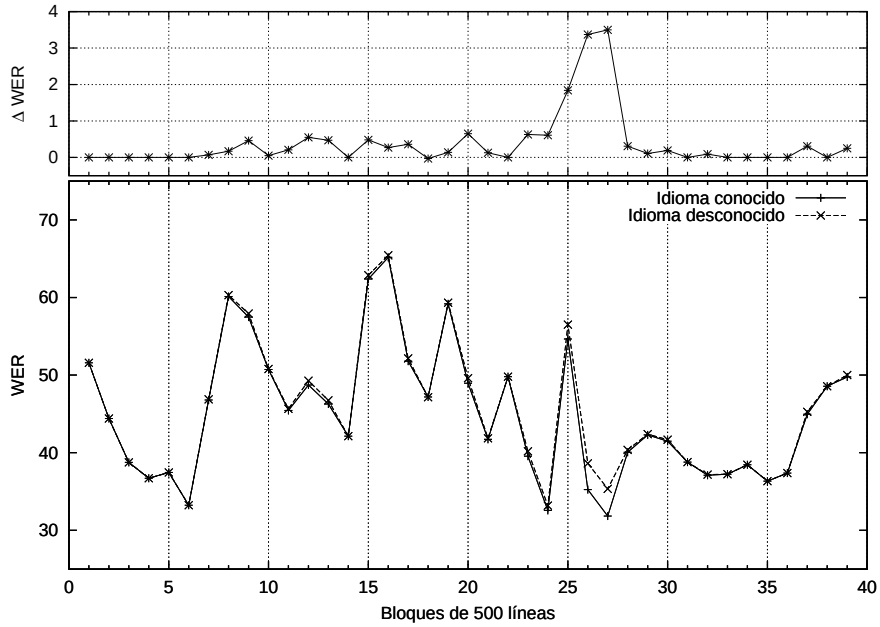


Figura 5.1: Abajo: Comparativa de WER entre el sistema multilingüe con idioma conocido y el sistema multilingüe con idioma desconocido (el idioma de cada frase es el de la frase anterior), en función de los bloques de líneas entrenados. Arriba: Incremento de WER producido por el sistema al predecir el idioma.

término *j-ésimo* en f, W es el conjunto de palabras del vocabulario, y $P(t \mid l_i)$ lo tenemos en 5.2. Para más información puede referirse a [14].

$$\hat{l}(f) = \underset{l_i \in L}{\operatorname{argmax}} P(l_i) \prod_{j=1}^{|W|} \frac{P(t_j \mid l_i)^{N_{f,t_j}}}{N_{f,t_j}!} \tag{5.1}$$

$$P(t \mid l_i) = \frac{1 + \sum_{k=1}^{|F|} N_{k,t} P(l_i \mid f_k)}{\mid W \mid + \sum_{j=1}^{|W|} \sum_{k=1}^{|F|} N_{k,t_j} P(l_i \mid f_k)} \tag{5.2}$$

Para entrenar este modelo, hemos utilizado la implementación de naive Bayes multinomial del software `rainbow` [15]. El experimento ha consistido en entrenar el sistema por bloques de 500 líneas de manera incremental (500, ..., 19500) y obtener el WER al reconocer sobre el bloque siguiente estimando el idioma ($501 - 1501$, ..., $19501 - 20000$).

Se ha obtenido un 81,96 % de acierto en la predicción del idioma, y los efectos sobre el sistema los podemos ver en la gráfica 5.4 en la página 48. En este caso, hemos tenido alguna zona (bloque 8) donde pese a estimar erróneamente el idioma, el WER ha sido menor. Esto se debe a que si el predictor de idioma dice que una línea está en

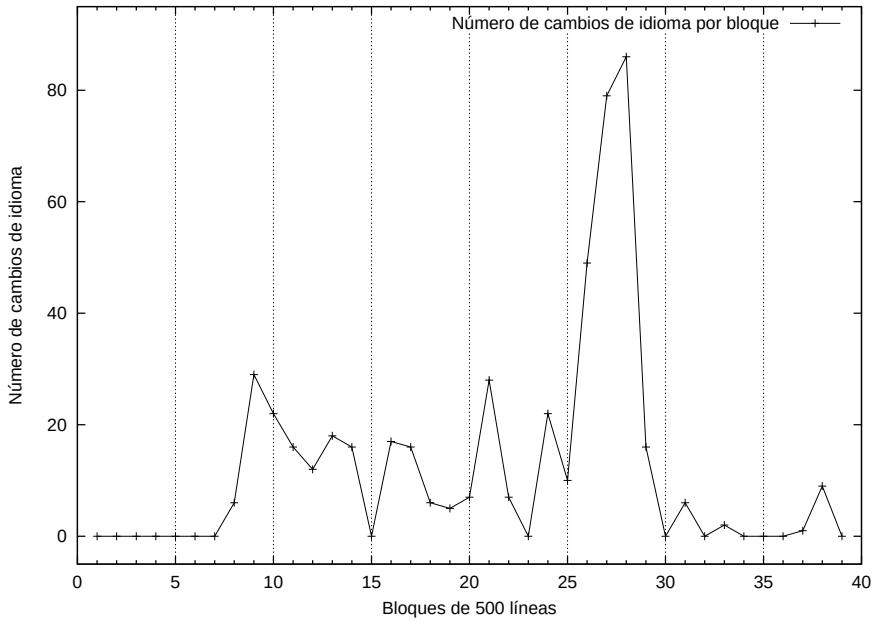


Figura 5.2: Número de veces que cambia el idioma de una frase a otra en función del número de bloques.

Castellano cuando en realidad está en Catalán, puede darse el caso de que los modelos de Castellano reconozcan mejor el Catalán que los propios modelos del Catalán, por estar mucho más entrenados. Por otra parte, este modelo predictor comete más error que el anterior durante toda la transcripción.

El resultado final lo mostramos en la gráfica 5.5 en la página 49 donde se ve claramente la pérdida obtenida con este sistema en forma de WER acumulado. Como se puede apreciar, se obtiene un WER de 47,74, es decir, una pérdida de 3,15 puntos sobre el sistema multilingüe ideal, y de 1,84 puntos sobre el sistema monolingüe.

5.5. Conclusiones

En este capítulo hemos analizado el efecto que tiene sobre el sistema multilingüe, el hecho de predecir el idioma. De entre las dos soluciones, la que mejor ha funcionado y no por ello la más complicada, ha sido suponer que el idioma de una frase es el mismo que el de la frase anterior. Ha funcionado razonablemente bien, gracias a la estructura secuencial de GERMANA y a que en un texto de estas características, la frecuencia de cambio de idioma es relativamente baja. En cuanto al modelo basado en naive Bayes, hay que pensar que está pensado para averiguar el idioma de un documento aleatorio. Contexto en el que nuestra primera aproximación obtendría un error mucho más alto, debido a su estructura no secuencial.

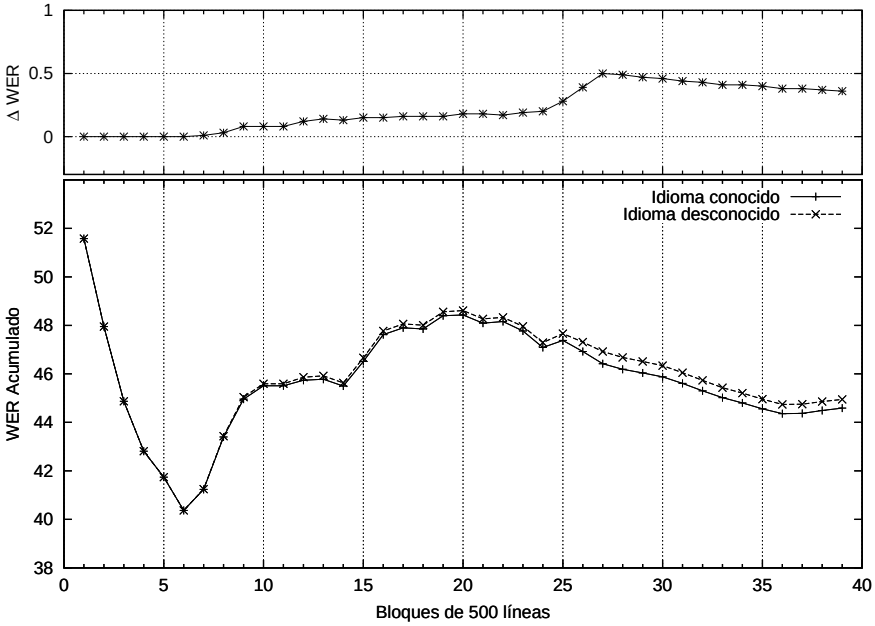


Figura 5.3: Abajo: Comparativa de WER (acumulado) entre el sistema multilingüe con idioma conocido y el sistema multilingüe con idioma desconocido (el idioma de cada frase es el de la frase anterior), en función de los bloques de líneas entrenados. Arriba: Incremento de WER producido por el sistema al predecir el idioma.

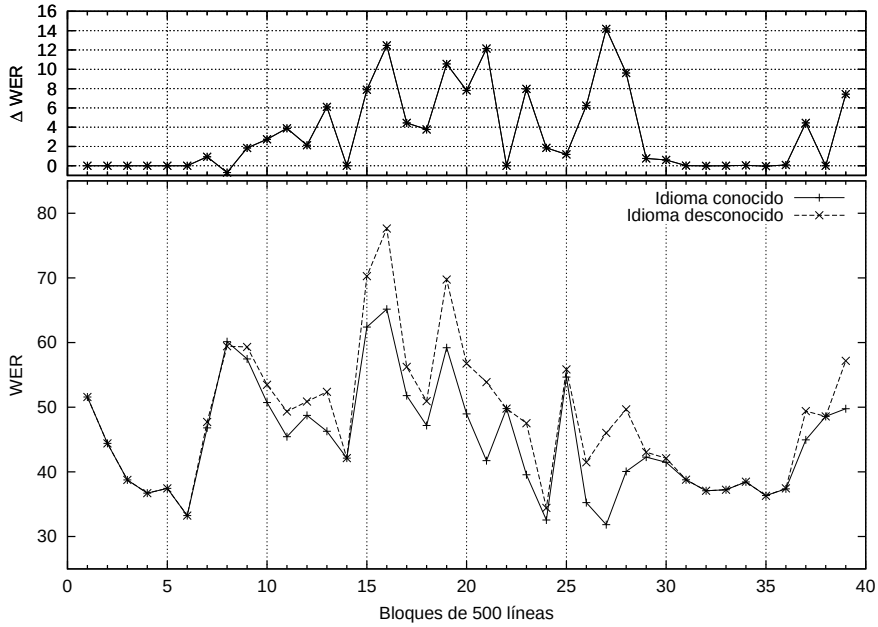


Figura 5.4: Abajo: Comparativa de WER entre el sistema multilingüe con idioma conocido y el sistema multilingüe con idioma desconocido (el idioma de cada frase se estima por naive Bayes), en función de los bloques de líneas entrenados. Arriba: Incremento de WER producido por el sistema al predecir el idioma.

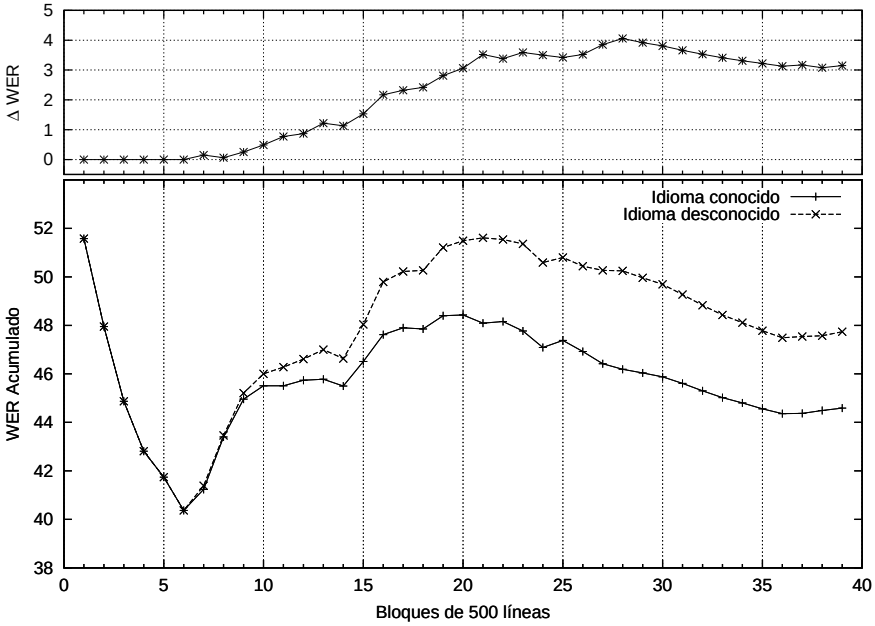


Figura 5.5: Abajo: Comparativa de WER (acumulado) entre el sistema multilingüe con idioma conocido y el sistema multilingüe con idioma desconocido (el idioma de cada frase se estima por naive Bayes), en función de los bloques de líneas entrenados. Arriba: Incremento de WER producido por el sistema al predecir el idioma.

CAPÍTULO 6

CONCLUSIONES

En este trabajo hemos realizado una comparativa entre dos sistemas de reconocimiento de texto manuscrito, para analizar el impacto de trabajar con idiomas diferentes a la hora de realizar la transcripción automática. Nos hemos basado en el corpus de GERMANA, presentado en el capítulo 2, ya que se compone de hasta 6 idiomas diferentes.

En el capítulo 3, presentamos el sistema monolingüe que se basaba en la premisa de suponer que todo el texto de entrada pertenecía a una única lengua. Los resultados fueron muy prometedores y como punto final al capítulo, se realizó la adaptación automática de parámetros que ayudó a rebajar el *Word Error Rate* en varios puntos.

En el siguiente capítulo se mostró el sistema multilingüe mediante el que tratábamos a cada lengua de manera diferente. Con este sistema se rebajó aún más el WER gracias a que los modelos se especializaron para cada lengua, pero a su vez introdujo la problemática sobre la predicción del idioma de una línea. En un principio se supuso conocido y además también se realizó la adaptación de parámetros, obteniéndose unos resultados optimistas que mejoraban a los del sistema monolingüe.

En el capítulo 5, con el afán de dar unos resultados más realistas para el sistema multilingüe, se introdujeron unos modelos básicos para la predicción del idioma basados en el idioma de la línea anterior y naive Bayes. Dada la estructura secuencial del libro, la predicción basada en el idioma de la línea anterior fue la que mejor funcionó y pese a empeorar ligeramente los resultados del sistema multilingüe, aún eran mejores que los del sistema monolingüe.

En el futuro se podría optar por adaptar un mayor número de parámetros (solo se ha hecho con el *Grammar Scale Factor* y el *Word Insertion Penalty*), ya que la

estructura de un libro como GERMANA es variable a lo largo de todos sus capítulos. Además, también se podrían utilizar modelos predictores de idioma más avanzados, para acercarnos a los resultados optimistas obtenidos en el capítulo 4. Y por otro lado, también cabría estudiar el comportamiento de un sistema basado principalmente en el reconocimiento a nivel a de carácter con un modelo de lenguaje de mayor complejidad (9-gramas por ejemplo).

BIBLIOGRAFÍA

[1] A. H. Toselli, *Reconocimiento de Texto Manuscrito Continuo*. PhD thesis, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Valencia (Spain), March 2004. Advisor(s): Dr. E. Vidal and Dr. A. Juan (in Spanish).

[2] D. Pérez, L. Tarazón, N. Serrano, F. Castro, O. R. Terrades, and A. Juan, "The germana database," *In Proc. of ICDAR*, pp. 301–305, 2009.

[3] U. V. Marti and H. Bunke, "The IAM-database: an English sentence database for off-line handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, 2002.

[4] "Biblioteca Valenciana." http://bv.gva.es/.

[5] E. Belenguer, ed., *Germana de Foix, última reina de Aragón*. Univ. de Valencia (Spain), 2007.

[6] "GNU Image Manipulation Program (GIMP)." http://www.gimp.org/.

[7] `prhlt.iti.es/gidoc.php`., 2009.

[8] D. P. i Cardona, "Preparació de corpus i desenvolupament de prototips en reconeixement de text manuscrit," November 2009. Advisor(s): Dr. Alfons Juan i Císcar and Dr. Moisés Pastor i Gadea.

[9] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. A. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book for HTK V3.4*. Cambridge University Press, Cambridge and UK, 2006.

[10] A. Stolcke, "Srilm – an extensible language modeling toolkit." http://www.speech.sri.com, 2002.

[11] A. Stolcke, "SRILM - An Extensible Language Modeling Toolkit," in *Proccedings of 7th the International Conference on Spoken Language Processing (ICSLP 2002)*, pp. 901–904, 2002.

[12] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling.," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 181–184, 1995.

[13] T. Baldwin and M. Lui, "Language Identification: The Long and the Short of the Matter," in *Human Language Technologies: The 2010 Anual Conference of the North American Chapter of the ACL*, pp. 229–237, 2010.

[14] A. McCallum and K.Nigam, "A comparison of Event Models of Naive Bayes Text Classification," *AAAI-98 Workshop on "Learning for Text Categorization"*., 1998.

[15] A. K. McCallum, "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering." http://www.cs.cmu.edu/ mccallum/bow, 1996.

ÍNDICE DE FIGURAS

ÍNDICE DE CUADROS