UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO FIN DE CARRERA

RECONOCIMIENTO DE ESCRITOR INDEPENDIENTE DE TEXTO BASADO EN CARACTERÍSTICAS DE TEXTURA

Susana Pecharromán Balbás
Octubre 2007

PROYECTO FIN DE CARRERA

Título: *Reconocimiento de escritor independiente de texto basado en características de textura*

Autor: *Dº. Susana Pecharromán Balbás*

Tutor: *D. Fernando Alonso Fernández*

Tribunal:

Presidente: *Joaquín Rodríguez González*

Vocal: *Juan Alberto Sigüenza Pizarro*

Vocal Secretario: *Javier Ortega García*

Fecha de lectura:

Calificación:

RECONOCIMIENTO DE ESCRITOR INDEPENDIENTE DE TEXTO BASADO EN CARACTERÍSTICAS DE TEXTURA

AUTOR: Susana Pecharromán Balbás
TUTOR: Fernando Alonso Fernández

Área de Tratamiento de Voz y Señales
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Octubre 2007

Resumen

En este proyecto se estudia, implementa y evalúa un sistema automático de identificación y verificación de escritor independiente de texto basado en características de textura. Como base de datos para la experimentación se emplea la IAM, que es una base de datos electrónica y de libre acceso a la comunidad científica. La identificación y verificación es off-line, es decir, se realiza sobre textos ya escritos y escaneados.

Tras una introducción a la biometría y al estado del arte en reconocimiento de escritor, se efectúa un estudio exhaustivo de las características que se pueden extraer de un texto escrito detallando la información que aporta cada una. Es necesario realizar un preprocesado de las imágenes con los textos de muestra antes de calcular las funciones de distribución de probabilidad y la autocorrelación, es decir, las características para formar patrones con los que evaluar el rendimiento del sistema.

Para la parte experimental se ha modificado la base de datos original con el fin de obtener otra base de datos que se adapte mejor a los requerimientos del sistema. Los cambios realizados consisten en equiparar el número de muestras de cada escritor almacenado en la base de datos para que todos posean dos textos distintos.

Por último se evalúa el rendimiento en identificación y verificación de escritor desarrollado tanto en modo monomodal como multimodal, se estudia la influencia que tiene sobre el funcionamiento del sistema la disponibilidad de muestras con más o menos líneas de texto y para identificación se comprueba cómo afecta el tamaño de la lista Top N y el número de escritores del conjunto de test. Finalmente, se presentan las conclusiones y se proponen líneas de trabajo futuras.

Palabras clave

Biometría, identificación y verificación de escritor independiente de texto, IAM, función de distribución de probabilidad, autocorrelación, característica, fusión de características.

Abstract

In the present project, a text-independent writer identification and verification system using textural features has been studied, implemented and evaluated. The database used in the experimental part is the IAM-database which is freely available to researches upon request. Scanned handwriting images are used for identification and verification.

After a brief introduction to biometric systems and to writer recognition systems, we study in detail the features that characterize writer individuality. A succession of image preprocessing steps are applied before computing probability distribution functions and autocorrelation. These features allow us to build patterns in order to evaluate the performance of the system.

Before testing the developed system the original IAM-database is modified to always contain two samples per writer.

In the experimental part, it is evaluated the performance of the implemented writer identification and verification system using on the one hand feature combinations and on the other hand individual features. It is also studied how the availability of more or less amount of text lines of handwritten material influences in the final results and how the identification performance depends on the number of writers contained in the test data set and on the numbers of writers of the Top N rank. Finally, conclusions are drawn and future lines of work are proposed.

Key words

Biometric, text-independent identification and verification, IAM, probability distribution function, autocorrelation, feature, feature combinations.

Agradecimientos

En primer lugar quiero agradecer a mi tutor Fernando Alonso Fernández toda la ayuda que me ha prestado a lo largo del proyecto, ya que sin su apoyo y consejos no hubiera sido posible su realización.

También quiero agradecer a Javier Ortega la posibilidad que me ha brindado de colaborar en el ATVS y a todos los miembros del grupo por su colaboración y apoyo cuando han surgido imprevistos.

Por último, agradecer a todas las personas que directa o indirectamente me han apoyado y a las que quedo muy agradecida.

Susana Pecharromán Balbás
Octubre 2007

Este proyecto ha sido realizado en el Área de Tratamiento de Voz y Señales (ATVS) en la Escuela Politécnica superior de la Universidad Autónoma de Madrid. El proyecto ha sido financiado parcialmente por el Ministerio de Defensa y el Ministerio de Educación y Ciencia a través del proyecto TEC2006-13170-C02-01.

Índice de contenidos

Índice de figuras

Índice de tablas

Glosario

Autenticar: en biometría, la palabra *autenticación* suele usarse como sinónimo genérico de *identificación* y *verificación.*

Base de datos: recopilación de uno o más archivos computerizados. En el caso de sistemas biométricos, estos archivos pueden ser lecturas del sensor biométrico, plantillas, resultados de coincidencias, información sobre el usuario final, etc.

Biometría: ciencia de reconocimiento de individuos basándose en sus características físicas o de comportamiento. Estudia cuantitativamente la variabilidad individual de los seres vivos utilizando métodos estadísticos.

Captura: proceso de recopilación de un rasgo biométrico de un individuo mediante un sensor.

Características: características matemáticas distintivas calculadas a partir de un rasgo biométrico, y utilizadas para generar un patrón de referencia.

Comparación: proceso en el que se confronta un patrón de entrada con patrones almacenados en la base de datos con anterioridad, para tomar una decisión sobre identificación o verificación.

Curva DET: representación de la tasa de falsa aceptación frente a la tasa de falso rechazo en un eje normalizado, obteniéndose una única curva para determinar el punto de trabajo.

Dependiente de texto: sistema de *autenticación* de escritor en el que el contenido del texto escrito es conocido por el sistema.

Decisión: acción a seguir (automática o manual) que resulta de la comparación de la puntuación obtenida por el usuario con la escala del sistema.

Extracción: proceso de conversión de un rasgo biométrico capturado por un sensor en datos biométricos que puedan ser comparados con un patrón referencia.

Fusión de características: proceso de combinación de características individuales.

Identificación: tarea en la que el sistema reconoce a un usuario comparando sus rasgos biométricos con los patrones de todos los usuarios almacenados en la base de datos. Se realiza una comparación de uno a varios.

Independiente de texto: sistema de *autenticación* de escritor en el que el sistema no conoce el contenido del texto escrito.

Patrón: representación digital de las características distintivas de un individuo, que contiene la información extraída de un rasgo biométrico. Los patrones se utilizan durante la autenticación biométrica como base de comparación.

Rasgo biométrico: cualquier característica fisiológica o de conducta del ser humano que reúna las siguientes condiciones: universalidad, unicidad, estabilidad, evaluabilidad, rendimiento, aceptabilidad y fraude.

Sistema biométrico: consiste en un sistema reconocedor de patrones cuyo modo de operación es el siguiente: captura un rasgo biométrico, extrae un conjunto de características y las compara con varios patrones almacenados en una base de datos para decidir si los datos de entrada pertenecen a un individuo determinado o a un impostor.

Sistema biométrico multimodal: sistema biométrico que emplea múltiples rasgos biométricos.

Sistema biométrico unimodal: sistema biométrico que emplea un único rasgo biométrico.

Tasa de falsa aceptación (FAR): área bajo la curva de impostores que queda por encima del umbral y que indica la probabilidad de que un impostor sea aceptado. Se utiliza para medir el rendimiento biométrico durante la tarea de verificación.

Tasa de falso rechazo (FRR): área bajo la curva de usuarios válidos que queda por debajo del umbral y que indica la probabilidad de que un usuario registrado no sea aceptado por el sistema. Se utiliza para medir el rendimiento biométrico durante la tarea de verificación.

Tasa de igual error (EER): punto de la curva DET en el que la FAR y la FRR son iguales. Se emplea como medida del rendimiento en verificación**.** Por lo general, cuánto más bajo sea su valor, mayor será la precisión del sistema biométrico.

Umbral: valor predeterminado para las tareas de verificación o identificación de grupo abierto en los sistemas biométricos. La decisión de aceptación o rechazo depende de si el resultado de coincidencia del patrón de entrada y los de la base de datos se encuentra por encima o por debajo del umbral respectivamente. Esta escala es ajustable de modo que el sistema biométrico puede ser más o menos estricto según los requisitos de cada aplicación biométrica.

Verificación: tarea durante la cual el sistema biométrico intenta confirmar la identidad declarada de un individuo, al comparar la muestra suministrada con uno o más patrones almacenados en la base de datos.

Capítulo 1

Introducción

1 Introducción

1.1 Motivación del proyecto

En la actualidad el uso de técnicas de reconocimiento y autenticación biométrica está cobrando gran relevancia dado el creciente número de entornos y aplicaciones que requieren verificar la identidad de sus usuarios, desde forenses o policías hasta el control de acceso a instalaciones [1]. La biometría supone una forma sencilla y segura de identificación de personas. Una de sus principales ventajas consiste en que los rasgos biométricos en general son más difíciles de duplicar o falsificar, ya que a diferencia de los métodos comúnmente utilizados, no se basan en lo que cada individuo *posee* (por ejemplo el DNI o una llave) o *recuerda* (por ejemplo un PIN) para confirmar o establecer su identidad. La biometría se basa en algo que el individuo *es*. En la Figura 1 se muestran algunos ejemplos de aplicación de los sistemas biométricos.

Figura 1. (a), (b), (c) y (h) Sistemas de verificación de huella dactilar usados para seguridad en red, cajeros y apertura/cierre de puertas. (d) y (f) Sistemas de verificación basados en la geometría de la mano. (e) Sistema de reconocimiento de iris. (g) Sistema de reconocimiento facial.

Los rasgos biométricos se pueden clasificar en rasgos biométricos fisiológicos y rasgos biométricos de comportamiento o conducta [1]. Entre los rasgos biométricos fisiológicos se encuentran el iris, la huella dactilar, la geometría de la mano, la retina y el ADN, entre otros. Su principal característica es su reducida variabilidad a lo largo del tiempo, pero su adquisición es más invasiva. Por el contrario, los rasgos biométricos de comportamiento o conducta, como pueden ser la voz, la firma o la escritura, son menos invasivos aunque la exactitud de la identificación es menor debido a la variabilidad de los patrones de comportamiento. La diferencia principal entre unos y otros es que los fisiológicos están siempre presentes, mientras que en los de comportamiento es necesario hacer una realización (por ejemplo firmar o hablar).

El tema principal de este proyecto, la autenticación de personas basada en imágenes escaneadas de escritura (autenticación de escritor off-line) ha suscitado un gran interés en los últimos años debido a sus aplicaciones en el campo forense y en el análisis de documentos históricos [4]. Además constituye una amplia área de estudio dentro del campo de investigación de la biometría del comportamiento [3], [6] y [9]. Estos sistemas se basan en considerar que la variación del estilo de escritura entre diferentes escritores es mayor que la variación intrínseca de cada escritor considerado de manera aislada, como se muestra en la Figura 2.

Figura 2. Comparación de caracteres escritos y palabras de tres escritores diferentes. La variación entre escritores excede la variabilidad intrínseca de cada escritor.

1.2 Objetivos y enfoque

El presente proyecto se centrará en el estudio de una serie de características que permitan identificar a las personas en base a su escritura independientemente del contenido del texto escrito. Se asumirá que las muestras de escritura han sido tomadas de forma natural, es decir, sin alterar el estilo en el que el individuo suele escribir y manteniendo la curvatura y forma de las letras junto con su separación original. Para desarrollar el estudio se utilizarán una serie de características que operan en el nivel de

análisis de textura [6]. Las características del nivel de textura proporcionan información referente a la forma habitual de cada individuo de coger el bolígrafo y la inclinación preferente de los trazos a la hora de escribir, junto con la curvatura.

Como punto de partida, en el capítulo 2, se realiza una introducción a la biometría en la que se clasifican e introducen los rasgos biométricos en función de sus características, se hace una presentación de los sistemas biométricos y se comenta su aceptación en la sociedad.

En el capítulo 3 se realiza una exposición de los sistemas automáticos de reconocimiento. Se explica la estructura y las etapas de las que consta cualquiera de estos sistemas junto con sus posibles modos de operación y se comentan las diferentes formas de medir el rendimiento de un sistema de este tipo. Los criterios empleados para evaluar el rendimiento de los sistemas de verificación pueden ser [2] la representación de las curvas de falso rechazo y falsa aceptación, las curvas DET o las curvas ROC. En este proyecto se utilizarán las curvas DET como medida de evaluación del rendimiento del sistema desarrollado.

Es importante distinguir entre reconocimiento de escritura, identificación y verificación de escritor, conocer las diferencias entre los métodos de identificación de escritor dependiente e independiente de texto y los posibles factores de variabilidad de la escritura junto con las características que le confieren su individualidad [7]. Para ello en el capítulo 4 se realiza un breve resumen del estado del arte en el que se explican todos estos conceptos.

El objetivo final de este proyecto es estudiar, desarrollar, implementar y documentar un sistema automático de identificación y verificación de escritor independiente de texto. Los algoritmos que conforman el sistema están programados en un PC utilizando Matlab y el sistema programado [6] se describe en el capítulo 5. Para lograr la identificación y verificación de escritor se analiza la escritura de cada texto y se extraen una serie de características que permiten calcular funciones de distribución de probabilidad y autocorrelaciones con las que construir patrones. Emplear un método independiente de texto presenta varias ventajas dado que no es necesario conocer el contenido semántico del mismo y requiere una mínima intervención humana. Además cuenta con una mayor aplicabilidad que los sistemas basados en métodos dependientes de texto, caso que se explica en el capítulo 4.

Los experimentos llevados a cabo se exponen en el capítulo 6. Se presentan varios escenarios, cada uno de ellos con unas características determinadas, que prueban el rendimiento del sistema en diferentes situaciones: tratando los patrones extraídos de manera individual, fusionando los patrones y variando la cantidad de texto de las muestras a analizar. Estos experimentos se realizan utilizando la base de datos IAM [10], de libre acceso para la comunidad científica y ampliamente usada en publicaciones de referencia recientes [3] y [6].

Por último, en el capítulo 7 se discuten los resultados extrayendo conclusiones y planteando posibles vías para trabajo futuro.

Capítulo 2

Introducción a la biometría

2 Introducción a la biometría

Desde la antigüedad los seres humanos han utilizado los rasgos biométricos tales como la cara y la voz para reconocerse unos a otros. Actualmente en una sociedad interconectada como la nuestra, establecer de forma unívoca la identidad de un individuo se ha convertido en un aspecto crítico y a la vez cotidiano en una gran variedad de escenarios que se extienden desde el uso de cajeros automáticos hasta el permiso de entrada a un país. La biometría, descrita como la ciencia de reconocimiento de individuos basándose en sus características físicas o de comportamiento, está ganando gran aceptación como método para determinar la identidad de cada persona y ya se está utilizando en varias aplicaciones tanto comerciales, como de seguridad y forenses.

2.1 Características de los rasgos biométricos

Cualquier característica fisiológica o de conducta del ser humano puede ser empleada como rasgo biométrico siempre que reúna las siguientes condiciones [2]:

- **Universalidad:** todo el mundo debe poseer esa característica.
- **Unicidad:** dos personas cualesquiera deben ser suficientemente diferentes en términos de ese rasgo, es decir, un mismo rasgo para dos personas diferentes no puede ser idéntico.
- **Estabilidad:** el rasgo debe permanecer suficientemente invariable en el tiempo durante un periodo de tiempo aceptable.
- **Evaluabilidad:** el rasgo debe poder ser medido cuantitativamente.
- **Rendimiento:** hace referencia a la exactitud y velocidad alcanzable en el reconocimiento y a los recursos empleados, que deben ser razonables y no depender de factores del entorno.
- **Aceptabilidad:** los usuarios deben estar dispuestos a emplear ese rasgo en las actividades de su vida cotidiana.
- **Fraude:** los sistemas que usen ese rasgo deben ser suficientemente seguros de forma que resulte difícil engañarlos.

2.2 Rasgos biométricos

Existen diversos rasgos biométricos empleados en la actualidad en una gran variedad de aplicaciones. Cada uno presenta ciertas ventajas y desventajas, por lo que su elección dependerá de cada aplicación en concreto. Ningún rasgo cumple todas las características descritas en la sección 2.1, por lo tanto, la selección de un rasgo específico para una aplicación particular está condicionada por las características concretas del mismo y los requisitos de la aplicación. A continuación se incluye una lista de los rasgos más empleados (ver Figura 3) con sus propiedades más relevantes. En la Tabla 1, se muestra una clasificación de estos rasgos biométricos en función de las características explicadas en el apartado anterior.

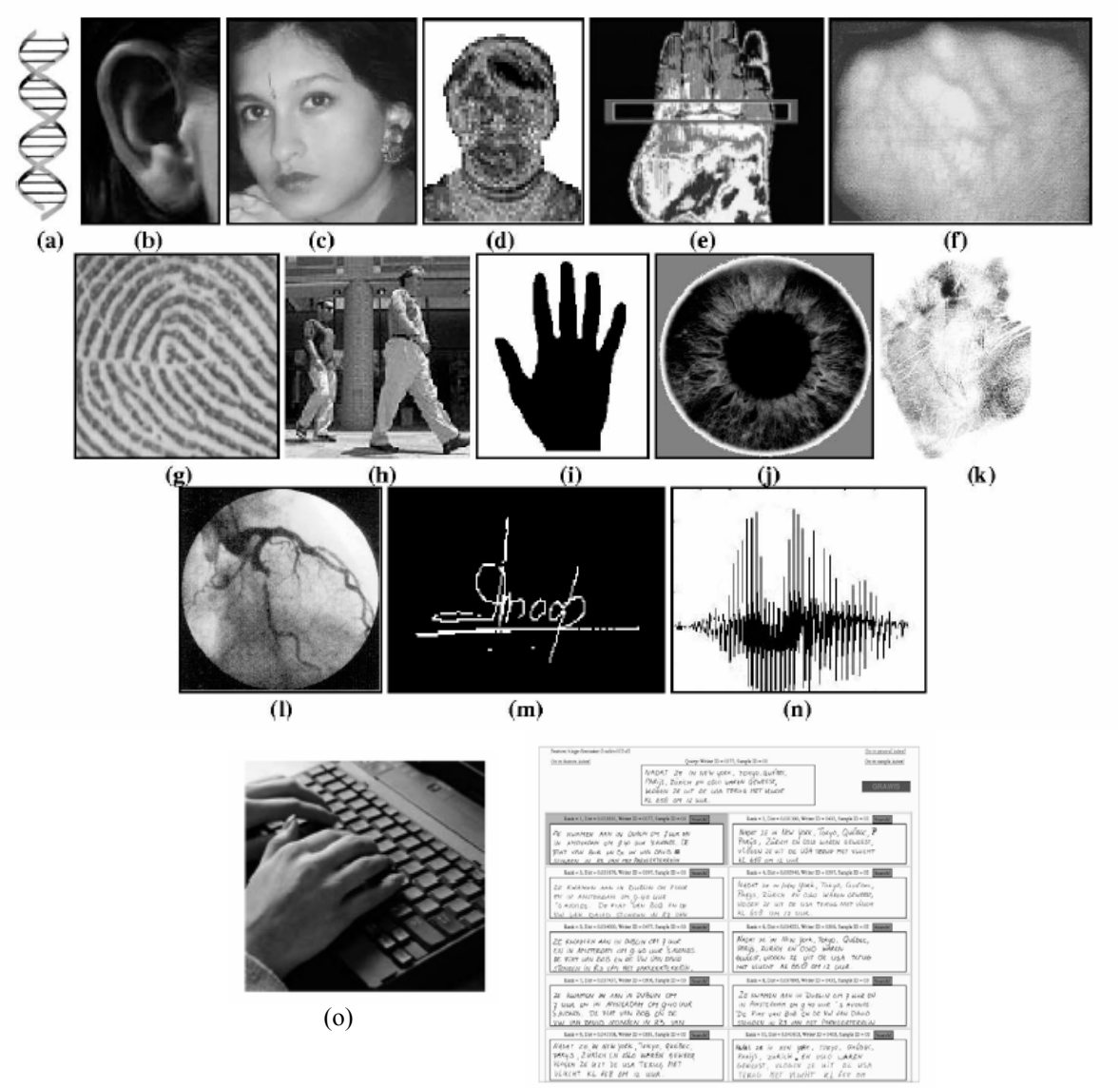


Figura 3. Rasgos biométricos empleados en la actualidad: (a) ADN, (b) Oreja, (c) Cara, (d) Termograma de la cara, (e) Termograma de la mano, (f) Venas de la mano, (g) Huella dactilar, (h) Forma de caminar, (i) Geometría de la mano, (j) Iris, (k) Huella de la palma de la mano, (l) Retina, (m) Firma, (n) Voz, (o) Dinámica de tecleo, (p) Escritura.

ADN

El ADN es un código único para cada individuo, excepto en el caso de los gemelos idénticos (monocigóticos). Actualmente es el método más común en aplicaciones forenses para reconocimiento de personas, pero presenta ciertas limitaciones en aplicaciones de reconocimiento automático. Los factores que limitan su uso en este tipo de aplicaciones son la facilidad para robar este rasgo biométrico, la lentitud del proceso de reconocimiento y la necesidad de que sea asistido por una persona. Además, la información que se puede extraer a partir del ADN de una persona puede revelar discapacidades u otras características que el individuo no desee hacer públicas.

Oreja

La forma del borde de la oreja y la estructura gelatinosa es una característica única en cada persona. Los sistemas propuestos en la actualidad suelen emplear la distancia de los salientes del borde de la oreja con respecto a una referencia común del interior de la oreja.

Rostro

El rostro es probablemente el rasgo biométrico más usado en el reconocimiento humano entre individuos y supone un método de reconocimiento no invasivo. Las aproximaciones para el reconocimiento facial se basan bien en la localización y forma de los atributos faciales como ojos, nariz, labios y barbilla junto con su relación espacial (análisis local), o bien en un análisis global de la imagen de la cara. Las mayores limitaciones consisten en la forma de adquisición de las imágenes, requiriendo a veces un fondo fijo y simple o una iluminación especial, y en los problemas de reconocimiento de imágenes capturadas desde diferentes ángulos y bajo diferentes condiciones de iluminación.

Termogramas

El patrón de calor radiado por el cuerpo humano es característico de cada individuo. Puede ser capturado por una cámara de infrarrojos de forma no intrusiva o incluso oculta. La mayor desventaja de esta clase de sistemas es el coste de los sensores y su vulnerabilidad ante otras fuentes de calor no controlables. Los termogramas también se emplean para captar la estructura de las venas de la mano.

Huella dactilar

La huella dactilar se lleva usando como método de identificación de individuos desde hace ya varios siglos en entornos policiales y forenses. Una huella consiste en un conjunto de valles y crestas que son capturados al presionar el dedo contra un sensor. Es única para cada persona y cada dedo. Actualmente la exactitud de los sistemas de reconocimiento de huella disponibles es muy elevada. Los sensores son baratos y una gran cantidad de dispositivos portátiles comienzan a incluirlos (PDAs, móviles, etc).

Forma de caminar

La forma de caminar de cada individuo es un rasgo biométrico complejo a nivel espacio-temporal. No es un rasgo muy distintivo, pero puede ser suficientemente discriminatorio en aplicaciones que requieran un bajo nivel de seguridad. Forma parte de los rasgos biométricos de comportamiento y varía a lo largo de tiempo, pero su adquisición es no invasiva y para su captura es suficiente una cámara de vídeo.

Geometría de la mano

Los sistemas de reconocimiento para este rasgo se basan en un conjunto de medidas físicas como la forma de la mano, el tamaño de la palma y la longitud y el ancho de los dedos. Los factores ambientales no suponen un problema pero la geometría de la mano es un rasgo de baja distintividad de cada individuo y está sujeto a cambios a lo largo de la vida de una persona.

Iris

El iris es altamente distintivo para cada uno de los dos ojos de cada individuo. Aunque su captura requiere participación por parte del usuario, ya que debe situarse a una distancia predeterminada del sensor, y la tecnología es cara, han aparecido nuevos sistemas menos intrusivos y con mejor relación precio-efectividad.

Dinámica de tecleo

Hipotéticamente cada persona tiene una dinámica de tecleo característica. Este rasgo es de conducta por lo que varía a lo largo del tiempo y es poco distintivo, pero proporciona información suficientemente discriminatoria para identificación en casos sencillos. Para su captura basta con emplear secuencias de tecleo del usuario, por lo que no es intrusivo.

Olor

Cada objeto produce un olor que es característico de su composición química que lo distingue del resto de objetos y que puede ser capturado por sensores químicos, cada uno sensible a una sustancia química. Una parte del olor emitido por los seres humanos es distintiva para cada uno de ellos, pero resulta complicado descartarla de sustancias artificiales como perfumes y desodorantes.

Huella de la palma de la mano

La palma de la mano, al igual que la huella dactilar, consiste en una estructura de valles y crestas. Al tener un área mayor que la de un dedo, este rasgo es más distintivo que la huella dactilar y proporciona información adicional que permite una mayor exactitud.

Escáner de retina

La estructura vascular de la retina se supone diferente para cada individuo y cada ojo. Es el rasgo biométrico más seguro por su dificultad para duplicarlo. Pero su captura requiere la cooperación del usuario y contacto con el sensor, por lo que su aceptabilidad por parte del usuario se ve seriamente afectada. Además, puede revelar ciertas afecciones médicas.

Firma

La forma de firmar de cada persona es característica de ella misma. Aunque requiere contacto con una superficie y la cooperación del usuario, es un rasgo muy aceptado como método de autenticación ya que se usa ampliamente en cantidad de transacciones. La firma varía a lo largo del tiempo para un mismo individuo y está influenciado por su estado físico y emocional. Además existen sujetos cuya firma varía muy significativamente en cada realización, por lo que su identificación es compleja.

Voz

La voz es una combinación de características físicas y de conducta. Las características físicas del habla de cada individuo permanecen invariantes, pero las características de conducta cambian a lo largo del tiempo y se ven influenciadas por la edad, las afecciones médicas o el estado de ánimo de la persona. Las principales desventajas de este rasgo son su baja distintividad y la facilidad con la que puede ser imitado. Por el contrario, la voz es un rasgo biométrico muy aceptado y fácil de obtener.

Escritura

La escritura está dentro de los rasgos biométricos de comportamiento, por lo que es variable a lo largo del tiempo. Su captura es poco invasiva pero no constituye un rasgo tan discriminatorio como el ADN, por ejemplo.

Identificador biométrico	Universalidad	Unicidad	Estabilidad	Evaluabilidad	Rendimiento	Aceptabilidad	Fraude
ADN	A	A	A	B	A	B	B
Dinámica de tecleo	B	B	B	M	B	M	M
Escáner de retina	A	A	M	B	A	B	B
Escritura	B	B	B	A	B	A	A
Firma	B	B	B	A	B	A	A
Forma de caminar	M	B	B	A	B	A	M
Geometría de la mano	M	M	M	A	M	M	M
Huella dactilar	M	A	A	M	A	M	M
Iris	A	A	A	M	A	B	B
Olor	A	A	A	B	B	M	B
Oreja	M	M	A	M	M	A	M
Rostro	A	A	M	A	B	A	A
Termograma facial	A	A	B	A	M	B	A
Venas de la mano	M	M	M	M	M	M	B
Voz	M	B	B	M	B	A	A

Tabla 1. Comparación de tecnologías biométricas. Niveles Alto, Medio y Bajo son denotados por A, M y B respectivamente. Tabla extraída de [1] y [2].

2.3 Sistemas biométricos

Un sistema biométrico consiste en un sistema reconocedor de patrones cuyo modo de operación es el siguiente: captura un rasgo biométrico, extrae un conjunto de características y las compara con varios patrones almacenados en una base de datos para decidir si los datos de entrada pertenecen a un individuo determinado o a un impostor.

2.3.1 Aplicaciones de los sistemas biométricos

Las aplicaciones de los sistemas biométricos se dividen en los siguientes tres grandes grupos [2]:

- **Aplicaciones comerciales:** protección de datos electrónicos, protección en red, e-comercio, cajeros automáticos, control de acceso físico, etc.

- **Aplicaciones gubernamentales:** DNI, carné de conducir, pasaporte, control en fronteras, etc.
- **Aplicaciones forenses:** identificación de cadáveres, investigación criminal, identificación de terroristas, determinación de parentesco, etc.

2.3.2 Problemas y limitaciones de los sistemas biométricos

Los rasgos biométricos de una persona y su representación varían considerablemente según el método de adquisición, el entorno en el que se realiza la captura y la interacción del usuario con el sistema de adquisición. Las razones más comunes por las que se producen variaciones son [1]:

- **Presentación inconsistente** (Figura 4a): la señal capturada por el sensor depende tanto de las características intrínsecas del rasgo biométrico como de la forma en la que se presenta dicho rasgo. Por ejemplo, la forma tridimensional de un dedo se mapea en una superficie bidimensional del sensor por lo que se pueden tener diferentes impresiones de un mismo dedo.
- **Presentación irreproducible** (Figura 4b): los rasgos biométricos representan medidas de una característica biológica o de comportamiento y están expuestos a accidentes y heridas que pueden cambiar su estructura de forma permanente, a cambios en su aspecto externo debido a adornos como joyas o al maquillaje, etc. Todos estos fenómenos contribuyen a la variación de la señal capturada en diferentes adquisiciones.

Figura 4. Variaciones en una señal biométrica. (a) Presentación inconsistente: cambio en la posición facial respecto a la cámara. (b) Presentación irreproducible: cambio de una huella dactilar en el tiempo.

- **Captura imperfecta** (Figura 5): las condiciones de captura de una señal en situaciones prácticas no son perfectas y causan variaciones en la señal capturada. Estas condiciones pueden ser la iluminación para la captura de imágenes faciales, las características del canal para señales de voz, un contacto no uniforme en la toma de huellas dactilares, etc.

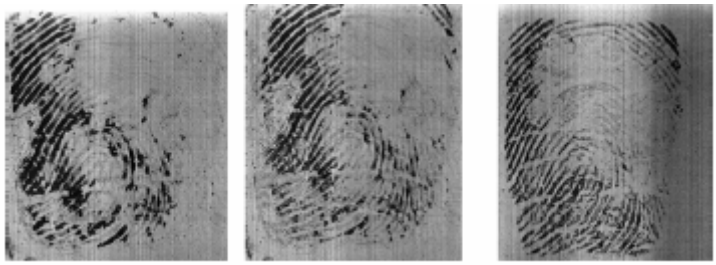


Figura 5. Captura imperfecta: tres impresiones diferentes de la huella de un usuario.

Asimismo, los sistemas biométricos que operan usando una única característica biométrica tienen una serie de limitaciones:

- **Ruido en los datos adquiridos** (Figura 6): los datos adquiridos pueden tener una componente ruidosa o estar distorsionados. El ruido puede estar producido por un sensor sucio o en mal estado o por condiciones ambientales desfavorables. Los datos adquiridos con ruido pueden dar lugar a que un usuario sea rechazado erróneamente.

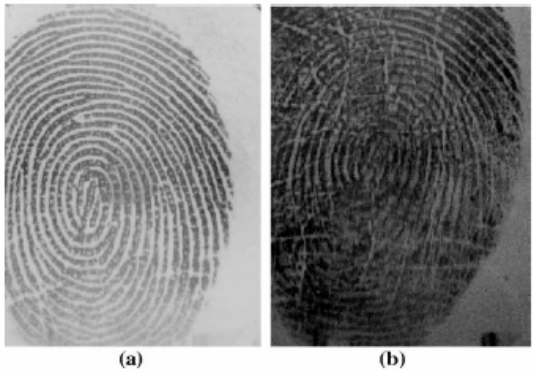


Figura 6. Efecto de imágenes afectadas por ruido en un sistema biométrico. (a) Huella dactilar obtenida en un proceso de captura. (b) Huella dactilar del mismo usuario durante un proceso de verificación después de tres meses.

- **Variaciones intra-clase:** los datos biométricos adquiridos durante un proceso de autenticación suelen ser diferentes de los datos que fueron usados para generar el patrón durante el proceso de registro, afectando por tanto al proceso de verificación. Estas variaciones pueden producirse porque el usuario interactúa de forma diferente con el sensor (ejemplo: cambia la forma de poner el dedo) o porque hay cambios en el rasgo (ejemplo: el usuario lleva distinta indumentaria en una foto o vídeo).

- **Unicidad:** aunque se espera que un rasgo biométrico varíe entre individuos, pueden existir similitudes entre diferentes usuarios en el conjunto de características usadas para representar ese rasgo. Esta limitación restringe la capacidad de discriminar usando ese rasgo biométrico.
- **No universalidad:** aunque se espera que todos los individuos posean un cierto rasgo biométrico, es posible que exista un subconjunto de individuos que carecen de él, por ejemplo: sufrir daños irreparables en un dedo, no saber escribir, etc.
- **Ataques:** un impostor puede intentar imitar el rasgo biométrico de un usuario legítimo para sortear el sistema. Los rasgos biométricos de comportamiento son más susceptibles a este tipo de ataques que los fisiológicos (imitadores de firma o voz, etc).

2.4 Aceptación en la sociedad y privacidad

La sociedad es la que determina el éxito de los sistemas de identificación basados en rasgos biométricos [2]. La facilidad y comodidad en la interacción con el sistema contribuye a su aceptación. Si un sistema biométrico permite medir una característica de un individuo sin necesidad de contacto directo, se percibe como mejor. Además, las tecnologías que requieren muy poca cooperación o participación de los usuarios suelen ser percibidas como más convenientes. Por otro lado, los rasgos biométricos que no requieren la participación del usuario en su adquisición pueden ser capturados sin que el individuo se dé cuenta y esto es percibido como una amenaza a la privacidad por parte de muchos usuarios. El tema de la privacidad adquiere gran relevancia con los sistemas de reconocimiento biométrico porque los rasgos biométricos pueden proporcionar información muy personal de un individuo, como afecciones médicas, y esta información puede ser utilizada de forma poco ética.

Por otro lado, los sistemas biométricos pueden ser empleados como uno de los medios más efectivos para la protección de la privacidad individual. Si un individuo extravía su tarjeta de crédito y otra persona la encuentra podría hacer un uso fraudulento de ella. Pero si la tarjeta de crédito únicamente pudiese ser utilizada si el impostor suplantase los rasgos biométricos del usuario, éste estaría protegido. Otra ventaja del uso de los rasgos biométricos consiste en limitar el acceso a información personal.

La mayoría de los sistemas biométricos comerciales disponibles hoy en día no almacenan las características físicas capturadas en su forma original, sino que almacenan una representación digital en un formato encriptado. Esto tiene dos propósitos: el primero consiste en que la característica física real no pueda ser recuperada a partir de su representación digital, lo que asegura privacidad, y el segundo se basa en que el encriptado asegura que sólo la aplicación designada puede usar dicha representación digital.

Capítulo 3

Sistemas automáticos de reconocimiento

3 Sistemas automáticos de reconocimiento

3.1 Estructura general de un sistema automático de reconocimiento

Todos los sistemas de reconocimiento automático de patrones poseen una estructura funcional común formada por varias fases cuya forma de proceder depende de la naturaleza del patrón o señal a reconocer. La siguiente figura muestra esta estructura. En general el usuario únicamente tiene acceso al sensor, el cual captura el rasgo biométrico. Los módulos marcados con línea continua son las entidades hardware o software básicas del sistema, y las etapas de procesado opcionales son las marcadas con línea discontinua.

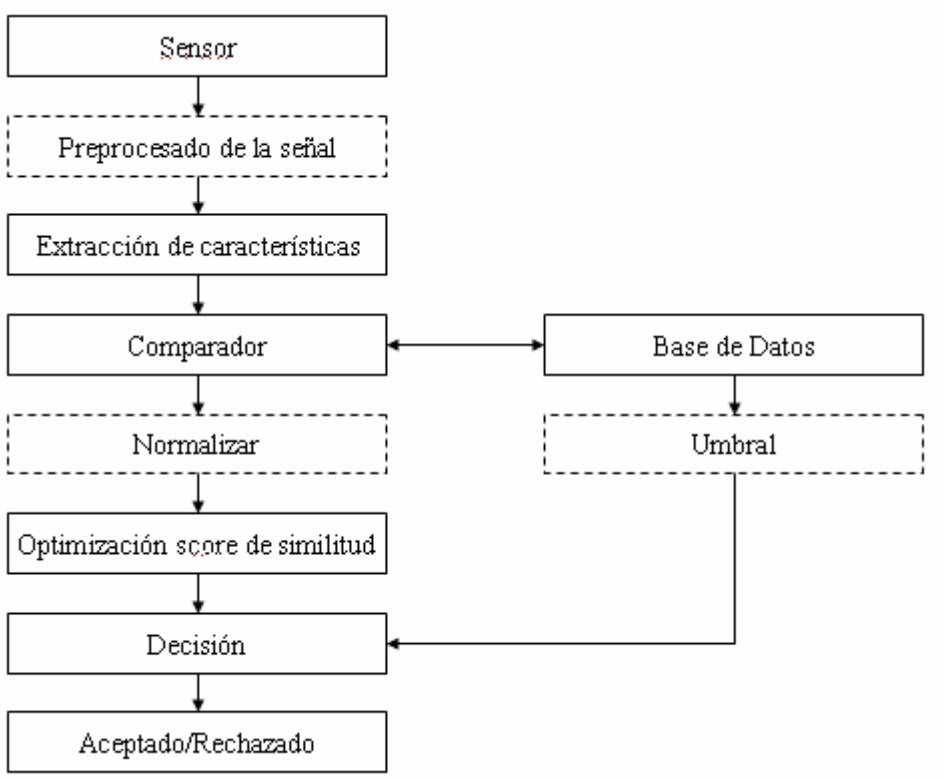


Figura 7. Arquitectura de un sistema de reconocimiento biométrico.

Adquisición de datos

En esta fase se recogen los datos analógicos de partida a través de un transductor o sensor y se convierten en un formato digital. Este proceso es determinante ya que de él depende la cantidad y la calidad de la información adquirida, la implementación de las siguientes fases, y, por tanto, el resultado final que se obtiene.

Preprocesado

En algunos casos es necesario acondicionar la información capturada para eliminar posibles ruidos o distorsiones producidas en la etapa de adquisición, o para normalizar la información a unos rasgos específicos para tener una mayor efectividad en el reconocimiento posterior.

Extracción de características

En esta etapa se elimina la información que no resulte útil en el proceso de reconocimiento, ya sea por no ser específica de cada individuo o por ser redundante. De este modo, se extraen únicamente aquellas características que sean discriminantes entre distintos individuos y que al mismo tiempo permanezcan invariantes para un mismo usuario, reduciéndose así mismo la duración de todo el proceso de reconocimiento y su coste computacional.

Generación de un modelo y comparación de patrones

Una vez extraídas las características más significativas, es necesario elaborar un modelo que represente a cada individuo y que permita la evaluación de la correspondencia entre los patrones de entrada y el modelo de un individuo en particular.

3.2 Modos de operación de un sistema biométrico

Modo Registro

En el modo registro se genera la base de datos con la que se compararán los datos de entrada. Los usuarios son dados de alta en el sistema y para ello se realiza la adquisición de sus rasgos biométricos, se extraen sus características y se genera un modelo o patrón representativo del individuo correspondiente, que queda almacenado en la base de datos de usuarios del sistema. En la base de datos se podrán almacenar además otros datos personales de los usuarios. (Ver Figura 8, parte superior).

Modo Verificación

En el modo verificación, el sistema valida la identidad de una persona comparando el rasgo biométrico capturado en la entrada con su propia plantilla biométrica previamente almacenada en la base de datos. En general, el usuario indicará su identidad mediante un número de identificación personal, un nombre de usuario o algún tipo de código. Posteriormente el sistema realizará una comparación *uno a uno* para determinar si el individuo es quien dice ser. (Ver Figura 8, parte central).

Las dos posibles salidas en este modo de funcionamiento dan lugar a la aparición de dos errores distintos:

- Falso Rechazo: se produce cuando el sistema indica que la información adquirida del usuario en la entrada no se corresponde con la plantilla almacenada, cuando realmente sí se corresponde.
- Falsa Aceptación: es complementario al falso rechazo y se produce cuando el sistema indica que la información adquirida del usuario en la entrada sí se corresponde con la plantilla almacenada, cuando realmente no se corresponde.

Modo Identificación

En el modo identificación, el sistema reconoce a un usuario comparando sus rasgos biométricos con los patrones de todos los usuarios almacenados en la base de datos. El usuario no introduce una identificación, sino que es el sistema el que determina su identidad. Para ello realiza una comparación *uno a varios* devolviendo el patrón de la base de datos que más se parece a los datos de entrada o una indicación de

que el individuo no se encuentra en la base de datos si el parecido no es suficiente. (Ver Figura 8, parte inferior).

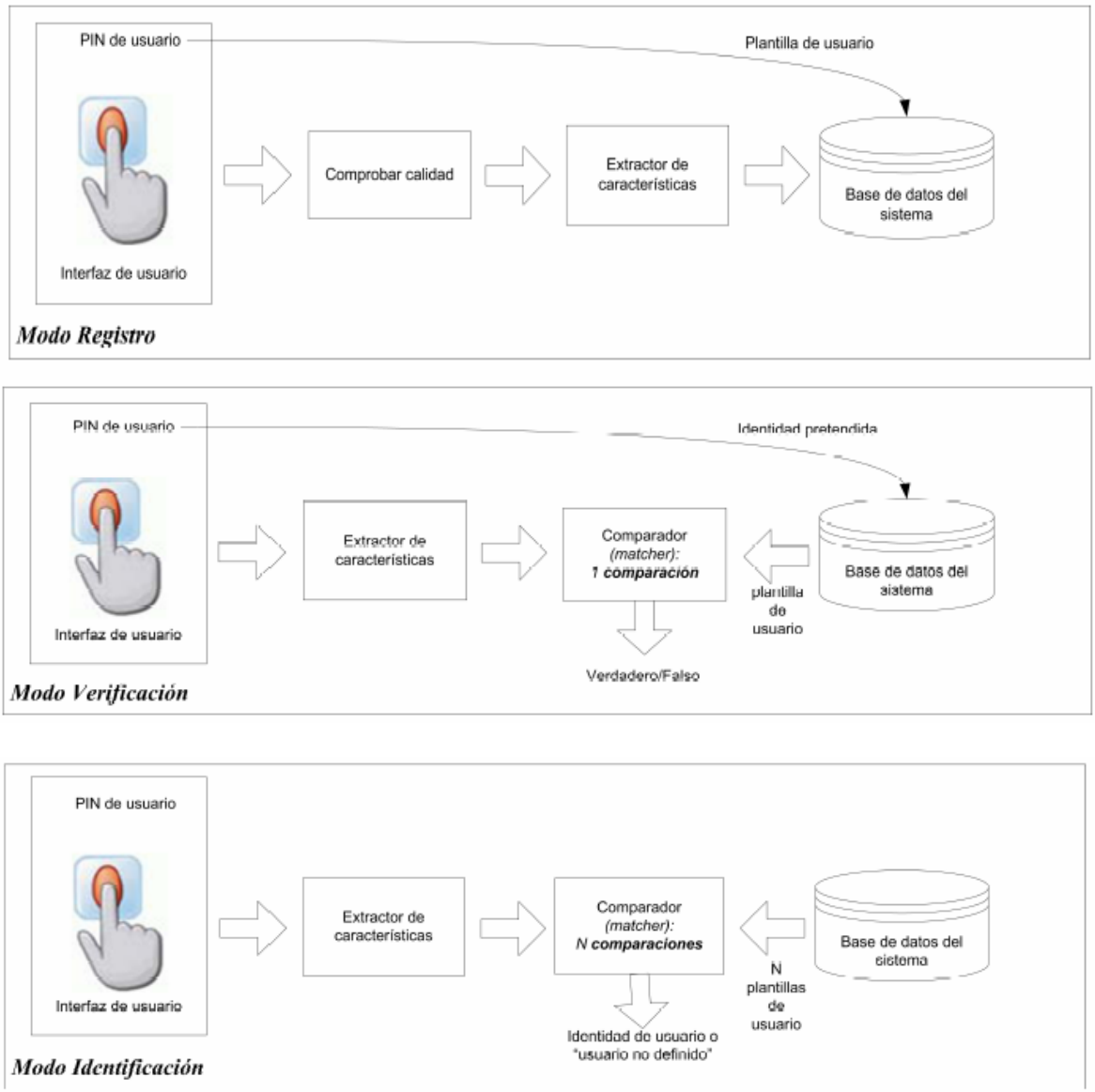


Figura 8. Modos de funcionamiento de un sistema automático de reconocimiento.

Modo Screening

Este modo se usa en aplicaciones para determinar si una persona pertenece a una lista de identidades buscadas. Por ejemplo, en aplicaciones de seguridad en aeropuertos, en seguridad en eventos públicos, en la búsqueda de terroristas, etc. Es un caso particular de identificación donde no tenemos garantía acerca de la calidad de la base de datos, obtenida en escenas de crimen por ejemplo, y donde seguramente los individuos no hayan decidido o colaborado para estar.

La finalidad de los sistemas de reconocimiento puede ser el reconocimiento positivo o el reconocimiento negativo. El reconocimiento positivo consiste en comprobar que un usuario es quien dice ser y suele hacerse en modo verificación,

aunque puede hacerse en modo identificación por conveniencia (evitar que el usuario tenga que indicar su identidad recordando un código o clave). Por el contrario, el reconocimiento negativo consiste en comprobar que un usuario es quien niega ser y sólo puede hacerse en modo identificación (por ejemplo modo screening).

3.3 Rendimiento de los sistemas automáticos de reconocimiento

Dos muestras de un mismo rasgo biométrico no son exactamente iguales debido a imperfecciones en las condiciones en las que se captura la imagen, cambios en los rasgos fisiológicos o de comportamiento del usuario, factores ambientales y a la interacción del usuario con el sensor entre otros. Por tanto, la respuesta del comparador de un sistema biométrico consiste en una puntuación o score que cuantifica la similitud entre la entrada y el patrón de la base de datos con el que se está comparando. Cuanto mayor sea el parecido entre las muestras, mayor será la puntuación devuelta por el comparador y más seguro estará el sistema de que las dos medidas biométricas pertenecen a la misma persona.

La decisión del sistema está regulada por un umbral: los pares de muestras que generen puntuaciones mayores o iguales que el umbral se supondrán correspondientes a la misma persona mientras que los pares de muestras cuya puntuación sea menor que el umbral se considerarán de personas diferentes.

Como ya se ha comentado anteriormente, un sistema biométrico de verificación puede cometer dos errores: determinar que dos muestras de diferentes usuarios corresponden a la misma persona (falsa aceptación, FA) y determinar que dos muestras de la misma persona pertenecen a diferentes usuarios (falso rechazo, FR).

Criterios de evaluación de sistemas de verificación

- **Representación mediante curvas FA y FR.** En un sistema ideal, los rangos de variación de las puntuaciones obtenidas para usuarios impostores y auténticos están separados, de manera que no hay solapamiento entre sus distribuciones, pudiéndose establecer un umbral de decisión que discrimine perfectamente ambas clases. Sin embargo, en un sistema real existe una región en la que se solapan ambas distribuciones, como se muestra en la Figura 9. Si se fija un umbral, todas las puntuaciones, tanto de usuarios como de impostores, cuyo valor sea superior a ese umbral serán interpretadas por el sistema como usuarios registrados. Como consecuencia, el área bajo la curva de impostores que queda por encima del umbral es la probabilidad de que un impostor sea aceptado y se conoce como la tasa de falsa aceptación (FAR). De igual modo, el área bajo la curva de usuarios válidos que queda por debajo del umbral es la probabilidad de que un usuario registrado no sea aceptado por el sistema y se denomina tasa de falso rechazo (FRR).

  Según se sitúe el umbral, la FAR y la FRR varían. Si el umbral es bajo, el sistema será muy permisivo y dará como válidas informaciones impostoras, mientras que si el umbral es alto, se producirá el efecto contrario.

Como medida conjunta de ambos tipos de error, los sistemas se suelen caracterizar mediante la EER (Equal Error Rate), que es el punto en el que la FAR y la FRR son iguales. Es fácil deducir de la Figura 9 que cuanto menor sea el EER, menor es el solape entre las curvas de usuario e impostor.

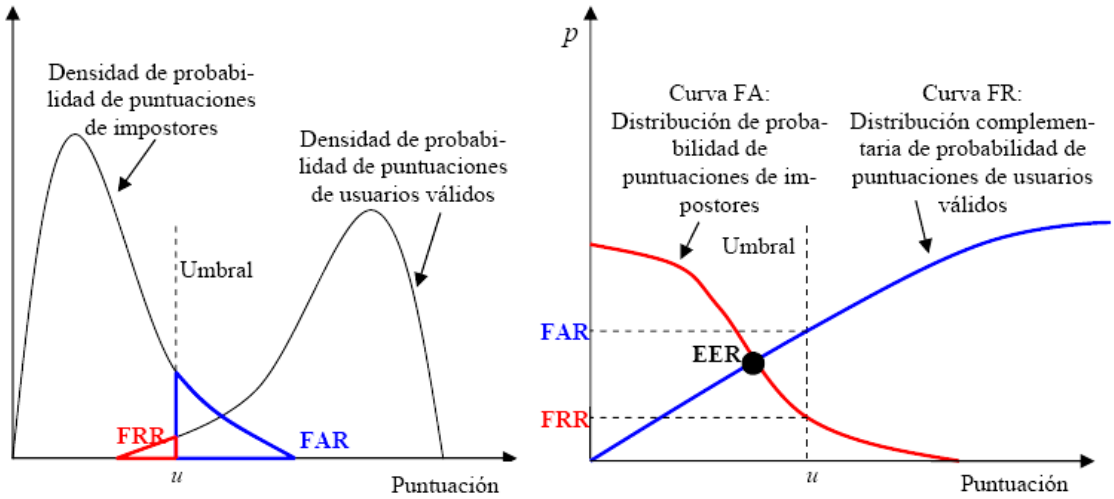


Figura 9. Densidades y distribuciones de probabilidad de usuarios e impostores.

- **Representación mediante curvas DET (Detection Error Tradeoff).** Aunque el punto de EER corresponde al umbral donde se igualan FA y FR, esto no implica que el sistema deba trabajar en ese punto. Para establecer el punto de trabajo del sistema se suele emplear la representación en forma de curvas DET, que consiste en la presentación de un error frente al otro en un eje normalizado, obteniéndose así una única curva para ambos tipos de error definida por todos los posibles puntos de trabajo del sistema. En esta curva (ver Figura 10), cualquier punto está dado por un valor de FA y otro de FR, de modo que no es necesario estar manejando varias curvas para determinar el punto de trabajo. Por contra, perdemos la información del umbral

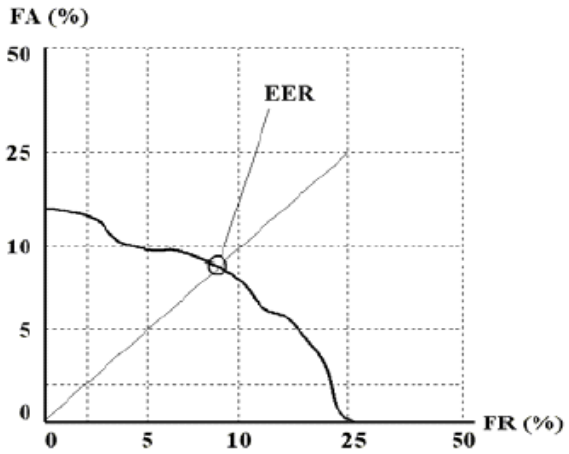


Figura 10. Curva DET

En aplicaciones de alta seguridad (control de accesos), el punto de trabajo suele situarse en valores bajos de FA, para evitar que accedan impostores, a costa de tener alta FR. Por el contrario, en aplicaciones forenses se trabaja en baja FR, para no perder individuos buscados, a costa de una alta FA. Las aplicaciones civiles suelen trabajar en un punto intermedio.

Criterios de evaluación de sistemas de identificación

En modo identificación, el sistema tiene que comparar los datos de entrada con todos los modelos de identidad almacenados en la base de datos, devolviendo el modelo con mayor parecido. Llamemos a las tasas de error en modos identificación FARN y FRRN, donde N representa el número de modelos que hay almacenados en la base de datos. Bajo ciertas simplificaciones, puede considerarse que FRRN = FRR y FARN = (N-1)FAR, donde FRR y FAR son las tasas de error del sistema si estuviera funcionando en modo verificación. Tanto si el sistema funciona en identificación como en verificación, solamente un modelo de la base de datos se corresponde con los datos de entrada, por lo que la probabilidad de falso rechazo es la misma. Por el contrario, en identificación se realizan N-1 comparaciones contra modelos que no se corresponden con los datos de entrada (frente a una que se realizaría funcionando en modo verificación). De esta manera, la probabilidad de una falsa aceptación se multiplica por N-1.

3.4 Sistemas biométricos multimodales

Algunas de las limitaciones impuestas por los sistemas biométricos unimodales, aquellos que emplean un único rasgo biométrico, pueden solventarse utilizando más de un rasgo biométrico para el reconocimiento, lo que da lugar a sistemas multimodales. Estos sistemas biométricos son más fiables al combinar varios frentes de información, son más difíciles de suplantar y permiten cubrir mayor población que un sistema unimodal.

Un sistema de este tipo puede operar de tres modos diferentes:

- **Modo serie**: las salidas del análisis de un rasgo biométrico se usan como entrada para análisis del siguiente rasgo, reduciendo así en cada paso el número de identidades posibles antes de emplear la siguiente característica. Este modo se usa, por ejemplo, poniendo en primer lugar un sistema poco preciso pero de rápido procesado para después, una vez reducidas rápidamente las posibles identidades, emplear un sistema más preciso.
- **Modo paralelo**: la información de múltiples rasgos biométricos se emplea simultáneamente en el proceso de reconocimiento. En contraste al caso anterior, siempre se utilizan todos los sistemas fusionados lo cual a su vez requiere capturar todos los rasgos antes de decidir.
- **Modo jerárquico**: los clasificadores individuales se combinan en una estructura de árbol.

A su vez, existen varios niveles donde se puede combinar la información de múltiples sistemas:

- **A nivel de extracción de características:** combinando las diferentes características extraídas (ver Figura 11).

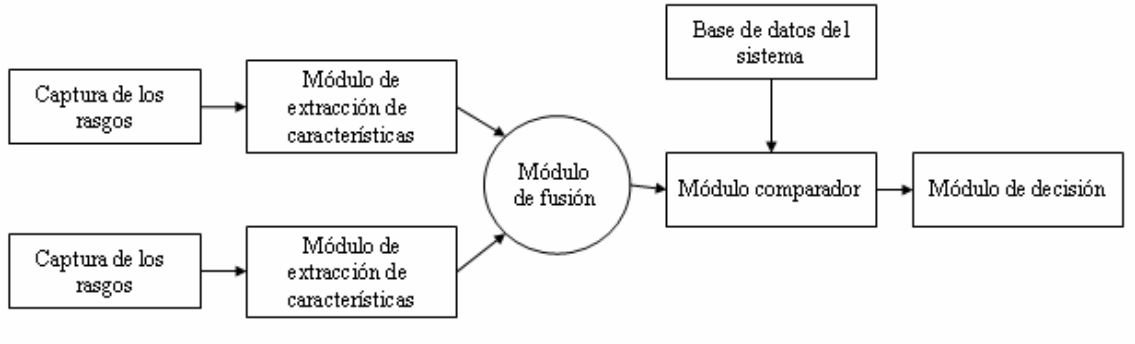


Figura 11. Fusión a nivel de extracción de características.

- **A nivel de score:** combinando los diferentes scores de similitud, por ejemplo un promedio (ver Figura 12).

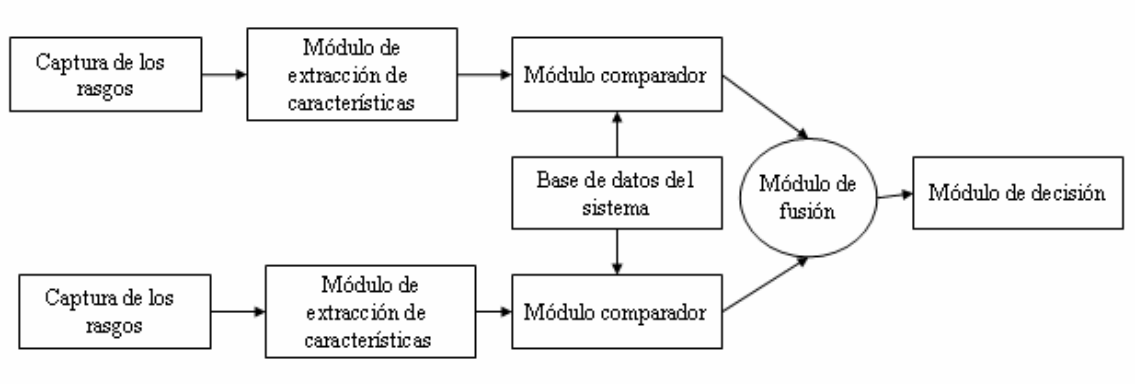


Figura 12. Fusión a nivel de score.

- **A nivel de decisión:** a partir de las distintas decisiones de aceptado/rechazado, por ejemplo por mayoría (ver Figura 13).

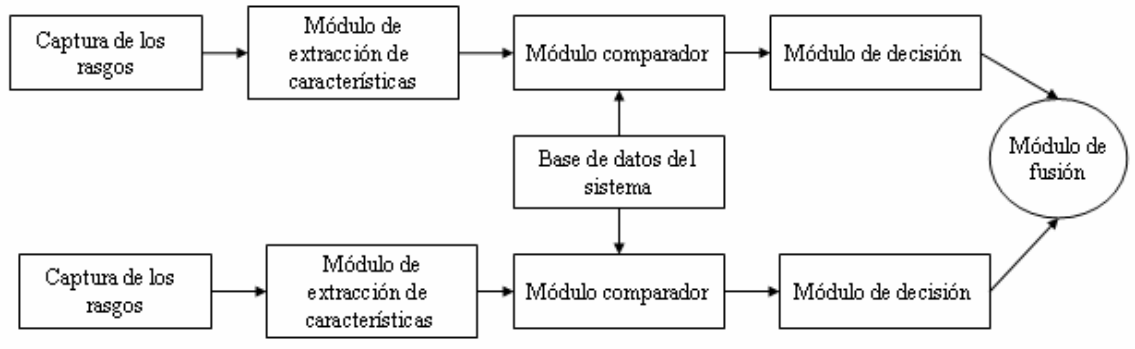


Figura 13. Fusión a nivel de decisión.

En este proyecto se estudian diferentes características extraídas a partir de un texto escrito y, como se verá después, también se estudia la fusión a nivel de score a partir de las puntuaciones individuales de cada característica.

Por último, un sistema multimodal opera en alguno de los siguientes escenarios:

- **Múltiples sensores:** se combina la información obtenida de diferentes sensores para el mismo rasgo biométrico. Por ejemplo, para capturar huellas dactilares se pueden emplear sensores ópticos, sensores basados en ultrasonidos y sensores de estado sólido.
- **Múltiples rasgos:** se combinan diferentes rasgos biométricos como pueden ser la cara y la huella dactilar. Estos sistemas contendrán necesariamente más de un sensor, cada uno para un rasgo biométrico distinto. En un sistema de verificación, un escenario de múltiples rasgos se emplea para mejorar la exactitud del sistema, mientras que en un sistema de identificación se utiliza para mejorar la velocidad de comparación.
- **Múltiples instancias de un mismo rasgo:** permite combinar las huellas dactilares de dos o más dedos de una persona, o una imagen de cada uno de los dos iris de un sujeto.
- **Múltiples capturas de un mismo rasgo:** se emplea más de una captura del mismo rasgo biométrico. Por ejemplo, se combinan múltiples impresiones del mismo dedo, múltiples muestras de voz o múltiples imágenes de la cara.
- **Múltiples representaciones/comparaciones para un mismo rasgo:** implica combinar diferentes enfoques para la extracción y comparación de las características biométricas.

En este proyecto se hará uso del último escenario (múltiples representaciones/comparaciones para un mismo rasgo) cuando se fusionen diferentes características extraídas a partir de un mismo texto.

Capítulo 4

Reconocimiento de escritor
Estado del arte

4 Reconocimiento de escritor. Estado del arte

4.1 Introducción

Como ya se ha comentado anteriormente, los rasgos biométricos se clasifican en dos categorías: rasgos biométricos fisiológicos que identifican a un usuario basándose en la medida de una característica física del cuerpo humano, y rasgos biométricos de comportamiento o conducta que emplean características individuales del comportamiento de un individuo para su identificación. La identificación de escritor pertenece a esta segunda categoría.

Los rasgos biométricos fisiológicos, como la huella dactilar, el iris o el ADN, suponen un modo robusto de identificación de individuos debido a la reducida variabilidad a lo largo del tiempo. Sin embargo, en general su adquisición es más invasiva y requiere la cooperación de los usuarios. Por el contrario, los rasgos biométricos de comportamiento o conducta son menos invasivos, pero presentan una exactitud menor en la identificación debido a la variabilidad de los patrones de comportamiento.

4.2 Reconocimiento de escritor vs. Reconocimiento de escritura

En el reconocimiento de escritura se buscan representaciones capaces de eliminar variaciones entre diferentes escrituras con el objetivo de clasificar la forma de los caracteres y de las palabras de manera robusta. Su objetivo es averiguar el texto escrito independientemente de la fuente. Por el contrario, el reconocimiento de escritor requiere representaciones realzadas de estas variaciones ya que son características de cada escritor, siendo su objetivo distinguir o averiguar la fuente que ha producido el texto.

Debido a sus múltiples aplicaciones, el reconocimiento de escritura siempre ha tenido más peso en las investigaciones del área de análisis de la escritura [5]. Pero en los últimos años, el reconocimiento de escritor ha empezado a cobrar importancia como consecuencia de sus aplicaciones en el campo forense y en el análisis de documentos históricos. La meta del reconocimiento de escritura consiste en obtener generalizaciones y eliminar las variaciones, mientras que en el reconocimiento de escritor lo que se pretende es maximizar las características específicas del estilo de escritura individual para poder discriminar entre escritores.

Es importante destacar que el reconocimiento de escritor podría reducir ciertas ambigüedades en el proceso de reconocimiento de patrones si la información de los hábitos de escritura generales del escritor estuviese disponible en el sistema de reconocimiento de escritura. Así pues, se bien su objetivo es contrapuesto, ambas podrían complementarse para un único fin.

4.3 Identificación de escritor vs. Verificación de escritor

Tal como hemos visto, un sistema de identificación de escritor realiza una búsqueda de uno a muchos en una base de datos con multitud de muestras de escritura de autores conocidos y devuelve una lista de posibles candidatos. Un sistema de verificación implica una comparación de uno en uno y decide si dos muestras han sido escritas o no por la misma persona.

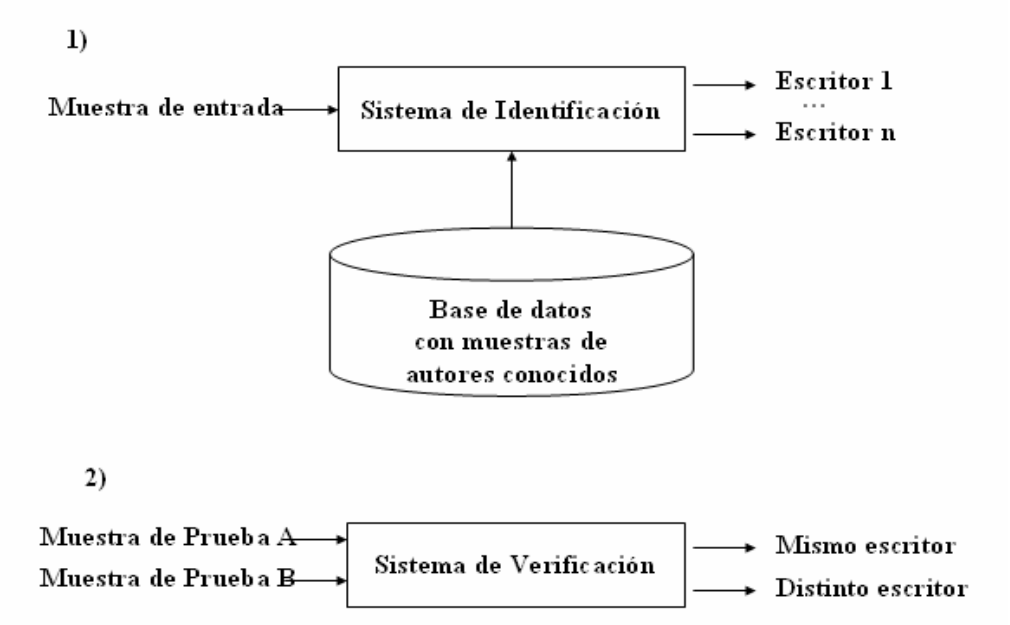


Figura 14. (1) Sistema de identificación de escritor. (2) Sistema de verificación de escritor.

En las búsquedas del proceso de identificación de escritor, todas las muestras de la base de datos son ordenadas de forma que su parecido con la muestra de entrada va disminuyendo. En la verificación de escritor, si el parecido entre dos muestras elegidas es mayor que el valor de un umbral predefinido, se estima que ambas han sido escritas por la misma persona. Por el contrario, si el parecido es menor, se considera que pertenecen a diferentes escritores.

4.4 Métodos dependientes de texto vs. Métodos independientes de texto

La identificación y verificación de escritor pueden a su vez estar dentro de dos categorías: métodos dependientes de texto y métodos independientes de texto [5].

Los métodos dependientes de texto se basan en comparaciones entre caracteres o palabras individuales de contenido semántico conocido. Por tanto, estos métodos requieren en primer lugar localizar y segmentar la información relevante, lo que los hace más complejos. Su ventaja consiste en que permiten alcanzar un alto rendimiento incluso con pequeñas cantidades de material escrito disponible, pero tienen limitada su aplicabilidad debido a que suponen un texto fijo o a la necesidad de intervención humana para localizar los objetos de interés.

Los métodos independientes de texto para identificación y verificación de escritor utilizan características estadísticas extraídas de la imagen entera de un bloque de texto. Las características proporcionan una descripción global de la región escrita mediante la eliminación de la información de localización. Se necesita una cantidad suficiente de escritura, como por ejemplo un párrafo o unas cuantas líneas, para poder obtener características estables insensibles al contenido del texto de las muestras. Desde el punto de vista de la aplicación, el gran avance consiste en que la intervención humana y la complejidad se reducen.

4.5 Variabilidad en la escritura

Existen cuatro factores que producen variabilidad en la escritura [5]:

- **Transformaciones afines:** (Figura 15a) pueden ser controladas voluntariamente por el escritor. Entre ellas se encuentran los cambios en el tamaño y la inclinación de la escritura, las translaciones y las rotaciones. Son una molestia tanto en la identificación de escritor como en el reconocimiento de escritura, pero no suponen un obstáculo importante. En concreto, la inclinación, que está determinada por el bolígrafo con el que se ha escrito y por la orientación de la muñeca respecto a los dedos, es un parámetro empleado habitualmente en la identificación de escritor.
- **Variabilidad neuro-biomecánica:** (Figura 15b) hace referencia al contexto local y el estado fisiológico que determinan la legibilidad de una muestra escrita y la cantidad de esfuerzo que supone hacer la forma de un carácter. También tiene en cuenta los temblores y los efectos de sustancias psicotrópicas en los procesos motor y de control durante la escritura. Este factor está más relacionado con el estado del escritor que con la identidad del mismo.
- **Variabilidad de la secuencia u orden de los trazos:** (Figura 15c) este factor tiene una gran dependencia con el estado del escritor durante el proceso de escritura. El orden de los trazos puede variar estocásticamente. Por ejemplo, al escribir una E mayúscula los cuatro trazos que la forman pueden ser escritos de $4!*2^4=384$ formas diferentes. Generalmente, este problema afecta más la proceso de reconocimiento de escritura que al de identificación de escritor
- **Variación alográfica:** (Figura 15d) hace referencia a la forma específica con la que cada individuo escribe un carácter. Proporciona información esencial para el reconocimiento automático de escritor.

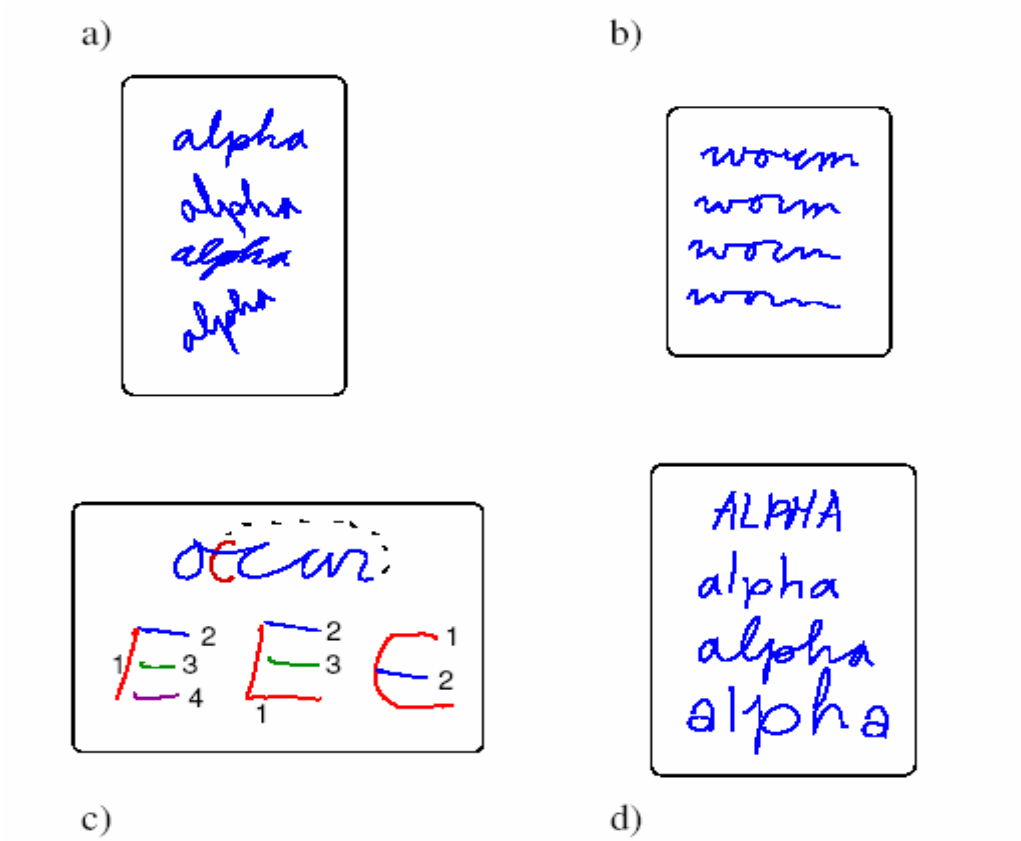


Figura 15. Factores que producen variabilidad en la escritura. (a) Transformaciones afines. (b) Variabilidad neuro-biomecánica. (c) Variabilidad de la secuencia de trazos. (d) Variación alográfica.

Las transformaciones afines y la variación alográfica son las fuentes de información más útiles en la identificación y verificación de escritor.

La escritura de una persona también cambia con la edad y constituye un factor de variabilidad importante que hay que considerar. Al crecer, la escritura de una persona se vuelve más rápida, continua, rítmica y suave, y en las personas mayores puede verse afectada por las condiciones médicas que influyen en la fuerza y la destreza de la mano.

4.6 Individualidad de la escritura

A medida que una persona va madurando, su estilo de escritura se va desviando del aprendido en el colegio y progresivamente va incorporando características suyas propias. Existen dos factores fundamentales que contribuyen a la individualidad de la escritura:

- **Factores genéticos:** también llamados factores biológicos. Son los siguientes:
  - Estructura biomecánica de la mano, es decir, tamaño relativo de los huesos de la muñeca y su influencia sobre el bolígrafo.
  - Ser zurdo o diestro.
  - Fuerza muscular.
  - Características del sistema nervioso central.

- **Factores miméticos:** también conocidos como factores culturales, influyen en la manera de coger el bolígrafo y en la forma de los caracteres. La idoneidad de la forma de una letra para su reconocimiento está influenciada por la legibilidad y la facilidad de escribir con las herramientas de escritura disponibles. Las características alográficas de una población de escritores se ve condicionada por los métodos de escritura enseñados en el colegio, que a su vez dependen de factores como la distribución geográfica, la religión y el tipo de colegio.

En conjunto, los factores genéticos y los miméticos determinan el proceso habitual de escritura de un individuo.

Escribir consiste en movimientos rápidos de los dedos y la mano, y superpuesto a ellos, un movimiento horizontal progresivo y lento del extremo inferior del brazo [3]. Por último, cabe mencionar que el proceso de escritura no es un proceso realimentado controlado por factores del entorno ya que sería demasiado lento. Al contrario se puede describir como un proceso psicomotor jerárquico cuyo nivel más alto (un programa motor abstracto) procede de la memoria a largo plazo, después se especifican los parámetros para este programa motor, tales como el tamaño y la forma, y finalmente se generan comandos para el movimiento de los músculos.

4.7 Algoritmos existentes para reconocimiento de escritor

Un resumen de los primeros trabajos sobre reconocimiento automático de escritor se hace en [11]. En los últimos años, no obstante, han aparecido un número de algoritmos dado el renovado interés por este ámbito. A continuación se explican brevemente alguno de ellos.

- El algoritmo propuesto en [12] se basa en características de textura, filtros de Gabor multicanal y matrices de co-ocurrencia en escala de gris. Para clasificar los resultados utiliza la distancia Euclídea ponderada obteniendo una exactitud del 96%.
- Otro algoritmo propuesto en la literatura [13] emplea perfiles de proyección horizontal y operadores morfológicos. Para la clasificación usa un clasificador Bayesiano o perceptrón multicapa, consiguiendo una exactitud del 95% tanto para palabras del alfabeto inglés como griego.
- El siguiente algoritmo [14] divide las características en dos categorías: macrocaracterísticas y microcaracterísticas. Las macrocaracterísticas operan en el nivel de documento/párrafo/palabra: umbral, número de píxeles de tinta, número de contornos externos/internos, inclinación media y longitud de las palabras. Por su parte, las microcaracterísticas operan en el nivel de

palabra/letra: gradiente, características estructurales y de concavidad. En las pruebas de identificación realizadas en [14] las microcaracterísticas presentan mejor funcionamiento que las macrocaracterísticas, alcanzando un rendimiento que excede el 80%. Para verificación de escritor este algoritmo emplea perceptrón multicapa o distribuciones paramétricas, obteniendo una exactitud del 96%.

- También se pueden utilizar grafemas generados mediante la segmentación de la escritura para codificar las características individuales de las letras. Algoritmos de este tipo se proponen en [15], consiguiendo tasas de identificación de escritor alrededor del 90%.
- En [16] las características independientes de texto se calculan utilizando el peso de las tres principales zonas de escritura, la inclinación y el ancho de las letras, las distancias entre componentes conectados, los contornos superior e inferior, y el grosor del trazo procesado mediante operaciones de dilatación. Empleando un clasificador k-vecino más cercano, las tasas alcanzadas exceden el 92%.
- Otro algoritmo que se utiliza en identificación y verificación de escritor consiste en un reconocedor de escritura basado en HMM [3], cuyos resultados alcanzan en identificación una tasa de acierto del 96% y en verificación una tasa de error de 2,5%.
- Por último, en este proyecto se implementan y evalúan algoritmos basados en características del nivel de textura [6] para capturar la individualidad de la escritura. A partir de ellos vamos a obtener funciones de distribución de probabilidad, autocorrelaciones y fusiones de características que clasificaremos mediante la distancia χ^2, la distancia Euclídea y la distancia de Hamming, respectivamente.

Capítulo 5

Sistema de identificación y verificación automática de escritor desarrollado

5 Sistema de identificación y verificación automática de escritor desarrollado

5.1 Descripción general del sistema

El sistema automático de identificación y verificación de escritor desarrollado [6] y evaluado en este proyecto se basa en imágenes escaneadas de escritura, es decir, escritura off-line. Se caracteriza por minimizar la intervención humana en el proceso de identificación de escritor y por codificar el estilo de escritura individual de cada usuario mediante características independientes del contenido del texto de la muestra a analizar. La individualidad de cada escritor se codifica empleando funciones de distribución de probabilidad extraídas de bloques de texto manuscritos y el sistema es ajeno a lo que ha sido escrito en las muestras que procesa. Estas funciones no son un único valor, sino un vector de probabilidades que captura la unicidad de la escritura. Desde este punto y a lo largo de todo el proyecto se hará referencia a estas funciones de distribución de probabilidad como *características*.

Para la ***identificación*** automática de escritor se realiza, como ya se ha comentado anteriormente, una búsqueda de uno a muchos en una base de datos con multitud de muestras de escritura de autores conocidos y se devuelve una lista de posibles candidatos. En el caso de la ***verificación*** automática de escritor se realiza una comparación uno a uno y el sistema decide si dos muestras han sido escritas o no por la misma persona. La base de datos empleada es la IAM, de libre acceso a la comunidad científica y ampliamente utilizada en estudios previos en el campo de escritor.

Para llevar a cabo los experimentos se han seleccionado de la base de datos IAM dos muestras por escritor y se ha asumido que las muestras de escritura de ésta han sido tomadas de forma natural, es decir, sin alterar el estilo en el que el individuo suele escribir y manteniendo la curvatura y forma de las letras junto con su separación original, tal como se especifica en la descripción de dicha base de datos.

Por último hay que mencionar que el sistema opera en el nivel de análisis de textura. Las características de este nivel proporcionan información referente a la forma habitual de cada individuo de coger el bolígrafo y la inclinación preferente a la hora de escribir, junto con la curvatura del trazo. El escritor tiende a mantener estas características a lo largo del texto manuscrito independientemente del movimiento progresivo horizontal.

Las tres fases principales del sistema desarrollado, descritas más adelante, son:

- Preprocesado de la imagen
- Extracción de características
- Comparación de las características extraídas con el modelo (verificación) o modelos (identificación) pretendidos

5.2 Preprocesado

Para la extracción de las características de las imágenes, primero es necesario preprocesarlas siguiendo una sucesión de pasos que proporcionan varias representaciones base alternativas con el objetivo de facilitar la extracción de características [6].

En primer lugar, las imágenes iniciales en escala de gris, es decir, las imágenes escaneadas, se han binarizado utilizando el método de Otsu [8]. Este método consiste en una técnica de umbralización que se emplea cuando hay una clara diferencia entre los objetos a extraer y el fondo de la escena (la escena debe caracterizarse por un fondo uniforme y por objetos parecidos), y es un método que no necesita supervisión humana ni información previa de la imagen antes de su procesamiento. En nuestro caso, hay una clara diferencia entre los trazos de escritura y el fondo blanco de la imagen.

Después, se ha eliminado el ruido de las imágenes binarias resultantes mediante un proceso de apertura y otro de cierre, y se ha realizado una detección de componentes conectados en las imágenes binarias ya sin ruido utilizando conectividad ocho. Los procesos de apertura y cierre combinan las operaciones de dilatación y erosión, que se utilizan para hacer más gruesos y más finos los objetos en una imagen binaria respectivamente. En el proceso de apertura primero se ha realizado una erosión y después una dilatación y en el de cierre se ha realizado una dilatación seguida de una erosión. Puede observarse en la Figura 16 cómo estos procesos contribuyen a eliminar el ruido presente en la imagen, así como a rellenar los pequeños huecos a lo largo de los trazos.

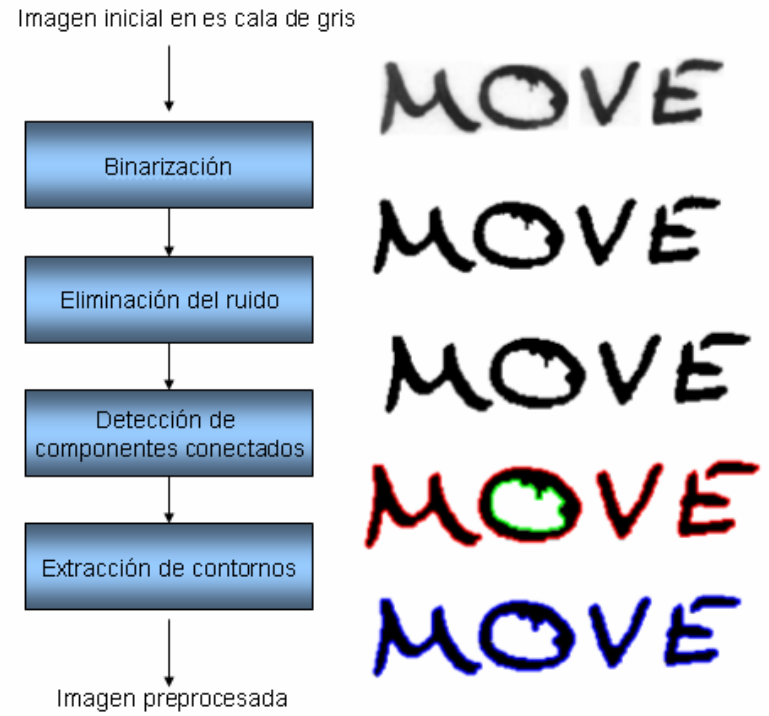


Figura 16. Aspecto de la imagen original y de las imágenes resultantes en cada paso del preprocesado. En la imagen que representa los componentes conectados detectados, los componentes internos se representan en color verde y los externos en color rojo.

En todos estos pasos se han empleado funciones disponibles del programa Matlab, programa en el cual se ha desarrollado el proyecto. Estas funciones son las siguientes:

- `Graythresh`: para la binarización de la imagen según el método Otsu.
- `Imdilate` e `Imerode`: para los procesos de apertura y cierre.
- `Bwboundaries`: para la detección de componentes conectados con conectividad ocho.

Por último, para todos los componentes conectados se han extraído tanto los posibles contornos internos como el externo mediante el algoritmo de seguimiento de contornos de Moore. En este algoritmo se va recorriendo la imagen que contiene cada componente conectado de arriba abajo y de izquierda a derecha hasta encontrar un primer píxel del contorno, que se toma como punto de partida. Una vez encontrado este píxel, se busca un píxel de contorno a su alrededor siguiendo el sentido de las agujas del reloj (ver Figura 17) y se repite este proceso hasta llegar de nuevo al píxel de partida por la misma posición desde la que se accedió a él al comenzar el algoritmo. El resultado es una secuencia con las coordenadas de todos los píxeles situados en el borde de la tinta del componente. Esta representación vectorial es muy efectiva ya que permite un rápido procesamiento computacional de las características de dirección que se usarán después.

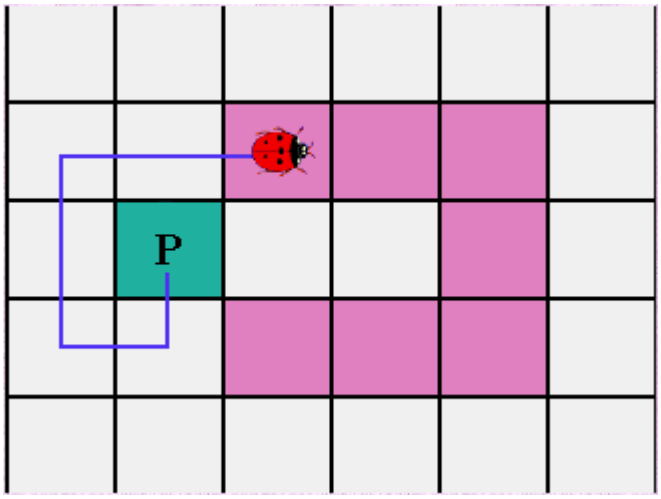


Figura 17. Funcionamiento del algoritmo de Moore.

Por tanto, a partir del preprocesado se obtienen cuatro representaciones principales de los documentos manuscritos para el cálculo computacional de las características: las imágenes en escala de gris, las imágenes binarias sin ruido, los componentes conectados y los contornos. La siguiente tabla muestra las características que se van a hallar y qué representación ha sido utilizada para cada una.

	Característica	Nombre	Dimensiones	Calculada a partir de
f1	$p(\phi)$	Contour-direction PDF	12	Contornos
f2	$p(\phi1,\phi2)$	Contour-hinge PDF	300	Contornos
f3h *f3v*	$p(\phi1,\phi2)$ *h* $p(\phi1,\phi2)$ *v*	Direction co-occurence PDFs	144 144	Contornos
f5h *f5v*	*p(rl) h* *p(rl) v*	Run-length on background PDFs	60 60	Imagen binaria sin ruido
f6	ACF	Autocorrelación horizontal	60	Imagen en escala de gris

Tabla 2. Características de textura. Adaptación de la tabla de [6].

5.3 Extracción de características

En este proyecto se usan características del nivel de textura. En este tipo de características, la escritura se modela como una textura que se puede describir mediante distribuciones de probabilidad calculadas a partir de la imagen y que capturan la apariencia visual única de las muestras escritas.

Se pueden agrupar en tres categorías distintas: funciones de distribución de probabilidad (PDFs) de dirección (características *f1, f2, f3h, f3v*), PDFs de longitud (características *f5h, f5v*) y autocorrelación (característica *f6*).

5.3.1 Contour-Direction PDF (*f1*)

La característica visual más importante de los textos escritos que revela el estilo individual de escribir es la inclinación de los trazos. Ésta constituye una característica personal muy estable. La distribución de las direcciones en la escritura proporciona información muy útil para el reconocimiento de escritor y puede ser calculada rápidamente utilizando la representación del contorno, con la ventaja adicional de que la influencia del grosor del trazo de tinta es eliminada.

Para extraer la distribución de esta característica hay que considerar la orientación de fragmentos locales del contorno. Cada fragmento se determina mediante dos píxeles del contorno situados a una cierta distancia ε uno del otro. El ángulo que el fragmento forma con la horizontal (ver Figura 18) se calcula mediante la siguiente expresión:

$$\phi = \arctan\left(\frac{y_{k+\varepsilon} - y_k}{x_{k+\varepsilon} - x_k}\right)$$

Mientras el algoritmo recorre el contorno, se va calculando la orientación de los fragmentos locales de contorno y simultáneamente se construye un histograma de ángulos. Más tarde dicho histograma se normaliza obteniendo así la distribución de probabilidad que indica la probabilidad de encontrar en un texto un fragmento del contorno orientado con cada ángulo medido respecto la horizontal.

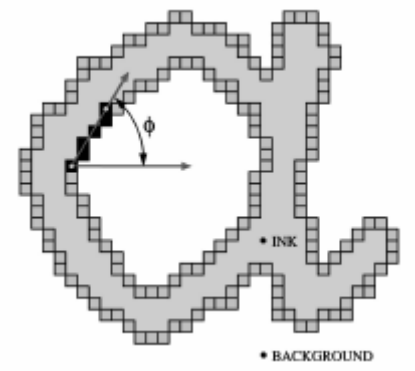


Figura 18. Descripción esquemática del método de extracción de la función de distribución de probabilidad de la característica Contour-Direction (*f1*).

Con el fin de controlar la longitud de los fragmentos de contorno analizados, se establece el parámetro ε, cuyo valor se elige para que sea comparable al grosor del trazo de tinta (ε=5 en este proyecto) [6]. Se considera que el ángulo reside en los dos primeros cuadrantes porque al no disponer de información online, no se puede conocer la forma en la que el escritor trazó el contorno que se está analizando. Como consecuencia, el histograma se extiende dentro del intervalo de 0º a 180º, que se divide en n=12 secciones, con lo que cada una de ellas abarca un rango de 15º. Este rango proporciona una descripción suficientemente detallada y al mismo tiempo robusta de la escritura para los procesos de identificación y verificación de escritor [6]. Estos datos también serán utilizados en todas las características de dirección explicadas más adelante.

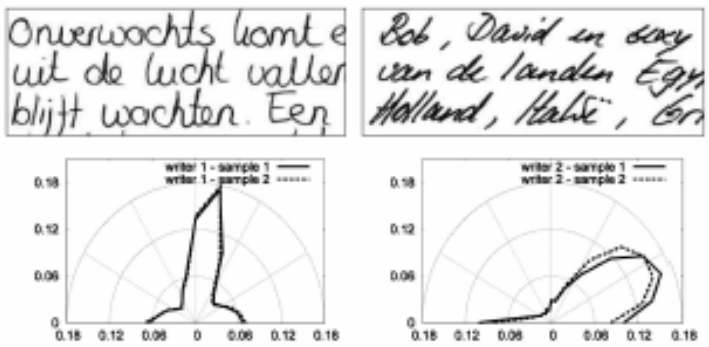


Figura 19. Ejemplo de escritura de dos sujetos diferentes y sus diagramas en coordenadas polares de la distribución de la dirección (*f1*) de las muestras escritas.

Como se puede ver en el ejemplo de la Figura 19, la dirección que predomina en la función de distribución de probabilidad calculada se corresponde, como era de esperar, con la inclinación de la escritura. Pero no sólo la inclinación de la distribución,

sino la distribución completa aporta información para la identificación de escritor, ya que por ejemplo, incluso para dos inclinaciones con el mismo ángulo, una letra más redondeada tendrá una PDF diferente (más diseminada) que una letra más puntiaguda.

5.3.2 Contour-Hinge PDF (*f2*)

La distribución calculada anteriormente representa el punto de partida en el diseño de características más complejas que dan lugar a una caracterización más profunda del estilo de escritura individual y proporcionan mejoras importantes en el funcionamiento de los sistemas de identificación y verificación de escritor.

Para capturar, además de la orientación, la curvatura del trazo de tinta, que constituye un rasgo muy discriminatorio entre escritores diferentes, se diseñan las características "hinge" (bisagra). La idea principal consiste en considerar no uno sino dos fragmentos de contorno sujetos al mismo píxel y calcular la distribución de probabilidad conjunta de las orientaciones de las dos ramas del contorno obtenido, el "contour-hinge". Por tanto, por cada píxel del contorno se miden dos ángulos, que se indican en la figura 20, respecto a la horizontal y los resultados se van almacenando en un histograma bidimensional. Después se normaliza el histograma y se obtiene la función de distribución de probabilidad conjunta que cuantifica la posibilidad de encontrar en la imagen del texto dos fragmentos orientados según los ángulos medidos y sujetos al mismo píxel.

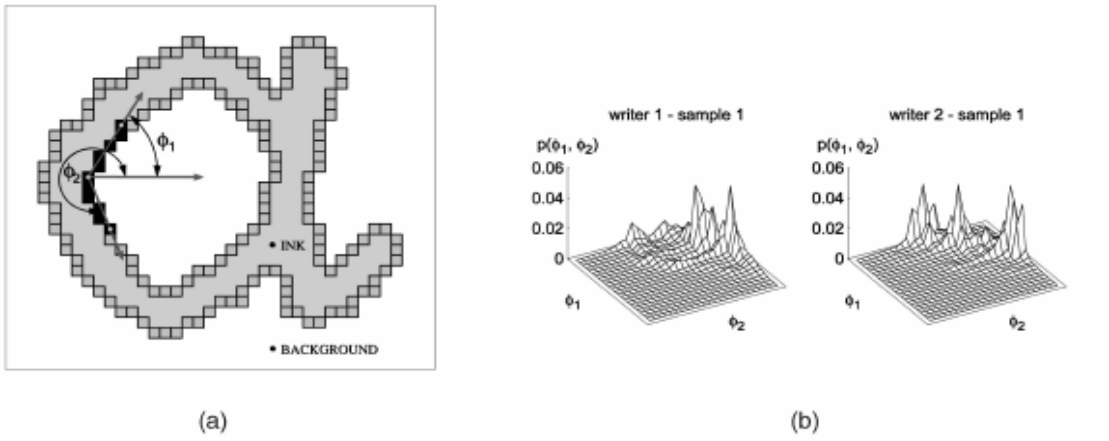


Figura 20. (a) Descripción esquemática para el método de extracción de la PDF "contour-hinge" (*f2*). (b) Funciones de distribución de probabilidad conjunta "contour-hinge" para dos escritores diferentes. La mitad de la representación tridimensional de las PDFs es plana porque únicamente se consideran combinaciones de ángulos en los que $\phi 1 \leq \phi 2$.

En comparación con la característica anterior en la que es suficiente con abarcar los dos primeros cuadrantes (180º), ahora es necesario extenderse por los cuatro cuadrantes (360º) alrededor del píxel central de unión para calcular los ángulos de los dos fragmentos considerados. Como consecuencia, la orientación ahora es cuantificada en *2n* direcciones por cada rama del "contour-hinge". Del total de combinaciones de los dos ángulos sólo se consideran aquellas no redundantes, es decir, las que cumplen la condición $\phi 1 \leq \phi 2$. El resultado es un vector de características de 300 dimensiones.

La característica obtenida es una función de distribución de probabilidad bivariable que captura tanto la orientación como la curvatura de los contornos.

Por último hay que mencionar que esta característica es altamente discriminatoria y proporciona resultados muy satisfactorios en la identificación y verificación de escritor, como veremos más adelante.

5.3.3 Direction Co-Occurrence PDFs (*f3h, f3v*)

Esta característica se basa en la misma idea de combinar fragmentos de contorno orientados. Se consideran las combinaciones de ángulos que ocurren al final de segmentos contenidos en el interior de los contornos, es decir, en los puntos donde comienza el contorno en su parte interna. En la siguiente figura se pueden observar estos puntos:

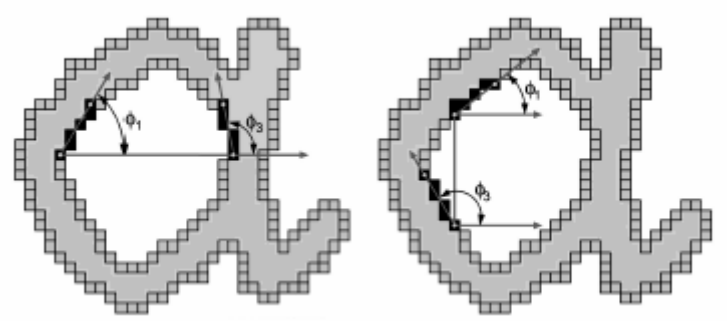


Figura 21. Descripción esquemática de los métodos de extracción de las PDFs de "Direction Co-Occurrence (*f3v, f3h*)" (en la izquierda la exploración horizontal para la característica horizontal y en la derecha la exploración vertical para la característica vertical).

La función de distribución de probabilidad conjunta de estos dos ángulos captura un mayor rango de correlaciones entre las direcciones del contorno y proporciona una medida de la redondez de los caracteres escritos. Los recorridos horizontales a lo largo de las filas de la imagen generan la característica horizontal f3h y los recorridos por las columnas de la imagen generan la característica vertical f3v. Las funciones de distribución de probabilidad de ambas se extienden en un rango comprendido entre los 0º a los 180º, por lo que su implementación consta de 144 dimensiones (n^2). Estas características se derivan conceptualmente de la distribución de la primera característica del nivel de textura explicada (*f1*) y de las distribuciones de las características que se explicará a continuación (*f5h, f5v*).

5.3.4 Run-Length PDFs (*f5h, f5v*)

Estas características se determinan tomando como base la imagen binaria sin ruido y considerando tanto los píxeles negros correspondientes a los trazos de tinta como los píxeles blancos que constituyen el fondo de la imagen. Las propiedades estadísticas de los píxeles negros están muy influenciadas por el grosor de la tinta y por tanto, por el tipo de bolígrafo empleado para escribir. Los recorridos de los píxeles

blancos sin embargo capturan las regiones encerradas dentro de las letras y también los espacios vacíos entre ellas, y son menos sensibles al grosor de tinta. Por tanto, para estas características, se considera la longitud de los segmentos contenidos en el interior de los contornos (píxeles blancos).

Existen dos métodos básicos de exploración: horizontal a lo largo de las filas de la imagen (*f5h*) y vertical a lo largo de las columnas de la misma (*f5v*). Al igual que en los casos anteriores, se obtendrá un histograma, esta vez de tramos recorridos, que se normalizará y será interpretado como una distribución de probabilidad.

Se han considerado solamente tramos que no superen los 60 píxeles de longitud para impedir que las medidas verticales sean entre líneas de texto sucesivas. Este valor se ha elegido así porque se sabe que la altura de una línea escrita en la base de datos utilizada está sobre los 120 píxeles [10].

5.3.5 Autocorrelación (*f6*)

Para calcular la autocorrelación, todas las filas de la imagen se desplazan sobre sí mismas por un valor de offset o desplazamiento dado para después calcular el producto entre la fila original y la desplazada y normalizarlo. La imagen inicial en escala de gris se utiliza como representación base para los cálculos y se fija como valor máximo de offset 60 píxeles. Para cada valor de offset, los coeficientes de autocorrelación son promediados a través de todas las filas de la imagen.

La función de autocorrelación detecta la presencia de regularidades en la escritura: los trazos verticales regulares se solapan en la fila original y en su copia desplazada horizontalmente para valores de offset iguales a múltiplos enteros de la longitud de onda espacial de las letras.

Hay que mencionar que la autocorrelación y el espectro de potencia son pares de transformadas de Fourier. Por esta razón, la función de autocorrelación efectúa un análisis de Fourier directamente en la imagen a lo largo de los píxeles de las filas. Guarda la información de amplitud y la promedia a través de todas las filas de la imagen a la vez que descarta la información de fase. Las características de dirección (*f1, f2 y f3*) se construyen esencialmente a partir de la información de fase mientras que la autocorrelación codifica solamente la información de amplitud.

5.3.6 En conclusión

Las características del nivel de textura son descriptores genéricos que aplicados a los caracteres escritos capturan la individualidad de cada escritor y proporcionan una base para la identificación de escritor. Su principal ventaja consiste en el procesamiento local de la imagen por lo que en general son aplicables y no imponen restricciones adicionales. El uso de la representación del contorno para extraer las distribuciones de dirección aporta ventajas respecto a la velocidad de procesamiento y al control de las dimensiones de las características. Las funciones de distribución de probabilidad pueden

ser estimadas incluso a partir de muestras con una cantidad muy reducida de texto escrito.

5.4 Marco de referencia para identificación y verificación de escritor usando las características descritas

En este apartado se describe el marco de referencia experimental usado en este proyecto. Dicho marco de referencia es el propuesto por los autores de las características descritas y se usará como punto de partida para los experimentos posteriores.

Una vez que las muestras de escritura han sido convertidas en características que capturan la individualidad del escritor, es necesario calcular con una medida de distancia apropiada, la similitud entre los vectores de características de dos muestras dadas.

Para las características *f1, f2, f3* y *f5*, se usa la distancia χ^2 para combinar una muestra incógnita *q* con cualquier otra muestra *i* almacenada en la base de datos:

$$x_{qi}^2 = \sum_{n=1}^{N\dim s} \frac{(p_{qn} - p_{in})^2}{p_{qn} + p_{in}}$$

donde *p* son las entradas de la función de distribución de probabilidad, *n* es el índice de la sección de la PDF que se está analizando y *Ndims* es el número de secciones (dimensiones de la característica). Para la autocorrelación (*f6*) se utiliza la distancia Euclídea, ya que es la única característica del análisis que no es una función de distribución de probabilidad y requiere una medida de distancia diferente al resto.

La *identificación* de escritor se efectúa empleando una clasificación del vecino más cercano. Para una muestra incógnita *q*, las distancias al resto de las muestras $i \neq q$ se calculan utilizando una característica previamente seleccionada. Después todas las muestras *i* se colocan en una lista ordenada de manera que la distancia de la muestra *i* a la muestra incógnita *q* va aumentando. Lo ideal debería ser que la muestra situada en la primera posición de la lista hubiese sido escrita por el mismo usuario que la muestra incógnita. Si se considera no sólo la muestra más cercana (Top 1) sino una serie de muestras comenzando desde la primera de la lista hasta llegar a un rango establecido, por ejemplo las diez primeras muestras de la lista ordenada (Top 10), la posibilidad de encontrar al escritor correcto se incrementa proporcionalmente al tamaño de la lista seleccionada. Para la realización de los experimentos de identificación de escritor no se realiza una separación entre conjunto de muestras de entrenamiento y conjunto de muestras de prueba, sino que se tienen todos los datos en un único conjunto. Esto supone una condición de prueba más difícil y realista porque implica más distracciones ya que incluye no una sino dos muestras por cada falso escritor y solamente una muestra del usuario correcto.

Para la *verificación* de escritor, la distancia entre dos muestras manuscritas dadas se calcula utilizando una característica previamente seleccionada, al igual que en el proceso de identificación de escritor. Las distancias situadas por debajo de un cierto umbral de decisión predefinido T se estiman suficientemente bajas como para considerar que las dos muestras han sido escritas por la misma persona. Para distancias cuyo valor supera el valor de T se considera que las muestras pertenecen a diferentes escritores. Existen dos posibles tipos de error, como hemos visto antes: el de falsa aceptación (FA) en el se considera que dos muestras han sido escritas por la misma persona cuando realmente no es así, y el de falso rechazo (FR) cuando dos muestras del mismo escritor son consideradas de escritores diferentes. Las tasas de error asociadas son la tasa de falsa aceptación (FAR) y la tasa de falso rechazo (FRR).

Variando el umbral de decisión T se puede obtener la curva DET (ver sección 3.3) que ilustra la relación entre las dos tasas de error. La Tasa de Igual Error (EER - Equal Error Rate) se corresponde con el punto de la curva DET en el que la tasa de falsa aceptación es igual a la tasa de falso rechazo (FAR = FRR) y cuantifica en un único número el rendimiento del proceso de verificación de escritor.

Las características consideradas en el proyecto no son totalmente ortogonales, pero, sin embargo, ofrecen diferentes puntos de vista de una muestra escrita. Por tanto, es natural intentar combinarlas para mejorar el rendimiento del sistema. Para ello se ha calculado una única distancia final para cualquier combinación de muestras como la media de las distancias de cada una de las características individuales que se van a combinar.

Para las combinaciones de características, la distancia de Hamming es la que proporciona el mejor funcionamiento según sus autores [6]:

$$H_{qi} = \sum_{n=1}^{N\dim s} \left| p_{qn} - p_{in} \right|$$

La distancia χ^2, debido al denominador, da mayor peso a las regiones de las funciones de distribución de probabilidad con baja probabilidad y maximiza el rendimiento para cada característica individual. Por otro lado, la distancia de Hamming genera valores de distancia comparables entre diferentes características y ofrece un terreno común con ligeras ventajas en la combinación de características.

Capítulo 6

Experimentos y resultados

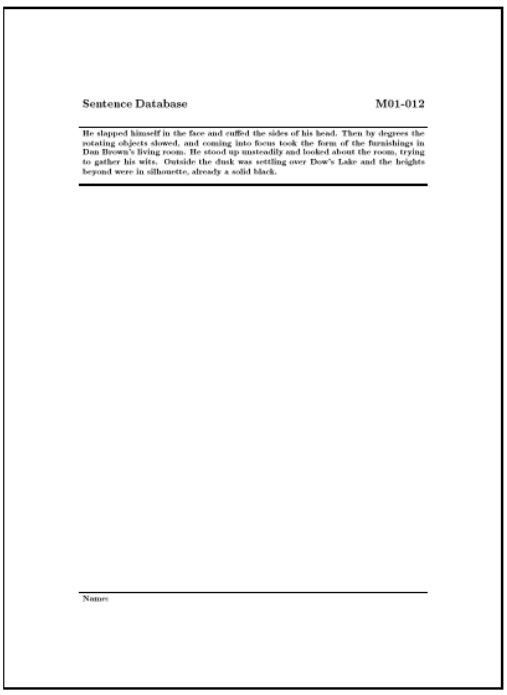
6 Experimentos y resultados

6.1 Base de datos utilizada (IAM)

Un prerrequisito fundamental para desarrollar un sistema de identificación y verificación de escritor es la disponibilidad de grandes cantidades de datos para el entrenamiento y las pruebas. En el presente proyecto se ha utilizado la base de datos IAM [10], en concreto su versión de Octubre del 2000, para cubrir este requerimiento.

La base de datos IAM presenta dos ventajas: es de libre acceso para la comunidad científica y está disponible en formato electrónico. Incluye un total de 1066 formularios escritos aproximadamente por 400 escritores diferentes y consiste en oraciones en inglés con 82227 ejemplos de palabras manuscritas de un vocabulario formado por 10841 palabras diferentes y distribuidas en 9285 líneas de texto. Además incluye procedimientos de procesado de imagen para la extracción del texto escrito de los formularios y la segmentación del mismo en líneas y palabras, también incluidas en la base de datos.

Los formularios constan de cuatro partes. La primera de ellas está formada por el título "Sentence Database" y un número asignado al texto para su identificación. La segunda parte contiene el texto impreso con una extensión de tres a seis líneas con 50 palabras como mínimo, que los individuos deben escribir. La tercera parte es una zona en blanco en la que los escritores deben escribir el texto superior y la última parte consiste en un espacio para el nombre, que el sujeto puede rellenar voluntariamente. Cada una de estas cuatro partes está separada del resto por una línea horizontal. Un ejemplo se muestra en la Figura 22:


Sentence Database M01-012

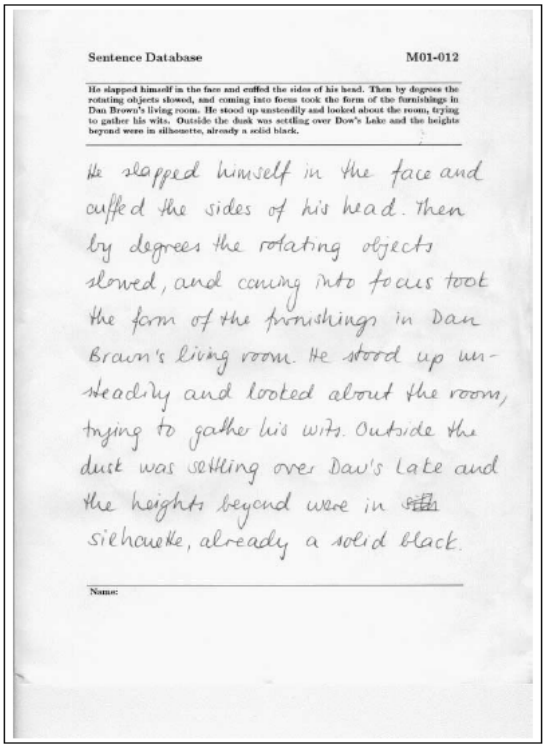
He slapped himself in the face and cuffed the sides of his head. Then by degrees the rotating objects slowed, and coming into focus took the form of the furnishings in Dan Brown's living room. He stood up unsteadily and looked about the room, trying to gather his wits. Outside the dusk was settling over Dow's Lake and the heights beyond were in silhouette, already a solid black.

Name:

Figura 22. Formulario vacío.

En el proceso de adquisición de la base de datos se imprimió una hoja con líneas horizontales espaciadas entre sí 1,5 cm que se colocaba debajo del formulario para que los sujetos escribiesen guiándose por ella. Se les solicitaba además, que utilizasen su letra habitual y que parasen de escribir si no había espacio suficiente en el formulario para transcribir todo el texto impreso en la segunda parte, para evitar que las palabras se comprimiesen y deformasen. No se impuso ninguna restricción sobre el instrumento empleado para escribir, por lo que la base de datos incluye formularios escritos con bolígrafos, plumas, lapiceros, etc.

Los formularios rellenos están escaneados a 300 dpi con una resolución en nivel de gris de 8 bits y las imágenes resultantes están guardadas en un formato TIFF con compresión LZW.


Sentence Database M01-012

He slapped himself in the face and cuffed the sides of his head. Then by degrees the rotating objects slowed, and coming into focus took the form of the furnishings in Dan Brown's living room. He stood up unsteadily and looked about the room, trying to gather his wits. Outside the dusk was settling over Dow's Lake and the heights beyond were in silhouette, already a solid black.

He slapped himself in the face and cuffed the sides of his head. Then by degrees the rotating objects slowed, and coming into focus took the form of the furnishings in Dan Brain's living room. He stood up unsteadily and looked about the room, trying to gather his wits. Outside the dusk was settling over Dan's Lake and the heights beyond were in silhouette, already a solid black.

Name:

Figura 23. Formulario relleno.

6.1.1 Uso de la base de datos en el proyecto

Al comienzo del proyecto se pensó en utilizar los formularios para implementar el sistema de identificación y verificación de escritor. Pero una vez desarrollados los códigos, se comprobó que el tiempo de procesamiento y ejecución para cada formulario era excesivo. Además surgió el problema de la extracción automática de la tercera parte del formulario, es decir, de la parte escrita por cada individuo. Al procesar las imágenes de los formularios, éstas perdían definición en las líneas horizontales que separan cada parte llegando incluso en muchos casos a desparecer, por lo que se hacía muy difícil la extracción o separación de las partes siguiendo estas guías.

Estos dos problemas, en especial el primero, impulsaron la búsqueda de alternativas al uso de los formularios completos. Como la base de datos disponía de los textos segmentados en líneas, se pensó en utilizarlas. Este cambio no afectaba al sistema desarrollado ya que la única diferencia consistía en que ahora en lugar de pasar a los códigos el texto completo se les pasaba línea a línea. El resultado fue un tiempo de ejecución mucho menor y por lo tanto un mejor rendimiento del sistema, dejando fuera de los objetivos de este proyecto la segmentación y extracción automática de líneas.

La base de datos también contiene los textos segmentados en oraciones y en palabras, pero se consideró que las palabras constituían una segmentación excesiva y que con las oraciones no se podía controlar con la misma facilidad que con las líneas lo que ocupaba el texto y las partes en las que se podía dividir éste de manera equitativa o no, interés muy importante para los experimentos realizados posteriormente.

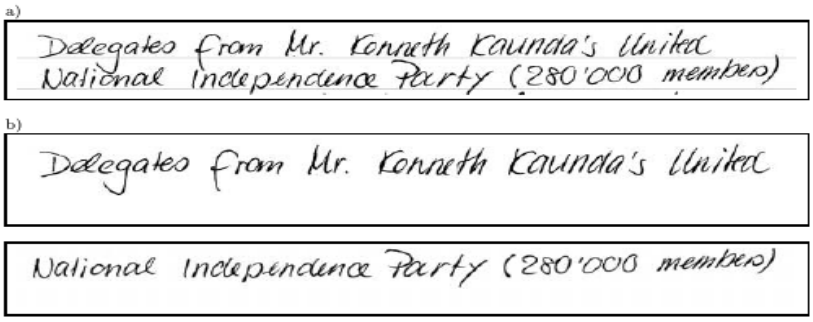


Figura 24. (a) Dos líneas de texto de un formulario. (b) Las mismas líneas separadas.

Para los experimentos se creó una base de datos que consistía en un subconjunto de la IAM. En la base de datos original IAM cada escritor tiene entre uno y tres formularios diferentes escritos y algunos además tienen varios formularios del mismo fragmento de texto. En el presente proyecto se necesitaban únicamente dos muestras por escritor, es decir, dos formularios por escritor. Para ello de cada escritor se seleccionaron los dos primeros formularios que aparecían en la base de datos y para los escritores con un solo formulario se decidió dividir éste en dos. El resultado fue una nueva base de datos modificada con 650 escritores diferentes, cada uno de ellos con dos muestras de escritura en letras minúsculas. La cantidad de tinta es prácticamente igual en las dos muestras de cada escritor, pero varía entre las tres líneas hasta una página entera entre escritores. Se conserva la identificación original de las muestras de la base de datos.

6.2 Escenarios de pruebas

En esta sección se presentan los diferentes escenarios que se han preparado para probar el sistema de identificación y verificación de escritor que se ha implementado. El objetivo de considerar varios escenarios es el de poder comparar los resultados y el rendimiento del sistema con los propuestos por la literatura (marco de referencia) [6], así como realizar un estudio diferente y novedoso. Se ha comprobado la influencia que tiene en el rendimiento del sistema la utilización características tratadas de forma individual y también combinaciones de las mismas. Asimismo se ha

estudiado la influencia que tiene la cantidad de escritura que se posee de cada escritor con el funcionamiento del sistema y sus resultados. Para ello se han realizado diversas pruebas, cada una de ellas con un número diferente de líneas de texto de cada escritor, es decir, comenzando desde una línea de texto escrito por muestra hasta llegar a tener todas las líneas del texto completo almacenado de cada escritor en cada muestra. Por último, se ha comprobado cómo evoluciona la tasa de error en modo identificación a medida que se aumenta el número de escritores que se tienen en cuenta en la lista de los N mejores (Top N) y al variar el número de escritores de test.

6.2.1 Pruebas marco de referencia: características individuales

En este primer escenario se estudia el rendimiento del sistema cuando se analizan las características de manera individual y se trabaja con las muestras completas de cada escritor, es decir, con todas las líneas de texto disponibles en la muestra. Como se ha dicho, en este caso la distancia utilizada es la de χ^2 para las funciones de distribución de probabilidad y la distancia Euclídea en el caso de la autocorrelación.

Las medidas de rendimiento empleadas son las de Top 1 y Top 10 para identificación y la de la Tasa de Igual Error (EER) para verificación.

El objetivo que se persigue en este caso, consiste en obtener resultados comparables con los del sistema propuesto en la literatura [6] que se ha empleado como referencia y comprobar qué característica proporciona mejor rendimiento. En la siguiente tabla se presentan los resultados del sistema de referencia publicados por sus autores:

	f1	***f2***	***f3h***	***f3v***	***f5h***	***f5v***	***f6***
EER	7,1	5,0	5,5	9,6	17,0	15,5	16,1
Top 1	46	81	68	65	10	8	13
Top 10	76	92	87	84	32	31	38

Tabla 3. Rendimiento del sistema de identificación y verificación de escritor de referencia al utilizar características individuales. En identificación se emplea como medida de rendimiento la tasa de acierto, mientras que en verificación se utiliza la tasa de error.

De la observación de la tabla anterior se desprende que la característica contour-direction (*f1*), que codifica la información de fase local, funciona mucho mejor que la característica de autocorrelación *f6*, que codifica la información de amplitud. Por otro lado, las características de combinación de ángulos *f2, f3h* y *f3v*, basadas en correlaciones de fase local, producen mejoras importantes en el rendimiento del sistema por encima de la función de distribución de probabilidad de la característica de dirección *f1*. Esto confirma el hecho de que las distribuciones de probabilidad conjunta capturan más información de la señal de entrada. Por tanto, se puede concluir que las

características de contorno basadas en combinaciones de ángulos (*f2, f3h* y *f3v*) producen un mejor rendimiento que el resto de las características extraídas.

Comparando los resultados del sistema de referencia con los obtenidos en el presente proyecto, que aparecen en la Tabla 4, se puede observar que las conclusiones a las que se llegaban a partir de los resultados del sistema de referencia son también válidas para la implementación hacha en este proyecto.

	f1	***f2***	***f3h***	***f3v***	***f5h***	***f5v***	***f6***
EER	6,38	3,95	4,41	4,86	17,90	19,80	26,33
Top 1	44,75	83,56	77,02	77,02	9,59	8,68	3,35
Top 10	75,19	92,54	90,56	90,72	32,27	29,53	10,2

Tabla 4. Resultados experimentales del rendimiento del sistema implementado en este proyecto para identificación y verificación de escritor empleando características individuales.

Se aprecia que el rendimiento en ambos casos es muy similar, pudiéndose deber las pequeñas diferencias de los resultados a la forma de extraer las características, ya que aunque el procedimiento es el mismo, el cálculo de éstas se puede implementar de maneras diferentes. La única diferencia relevante se produce en el caso de la autocorrelación (*f6*).

En la siguiente figura se muestran los resultados de verificación en forma de curvas DET:

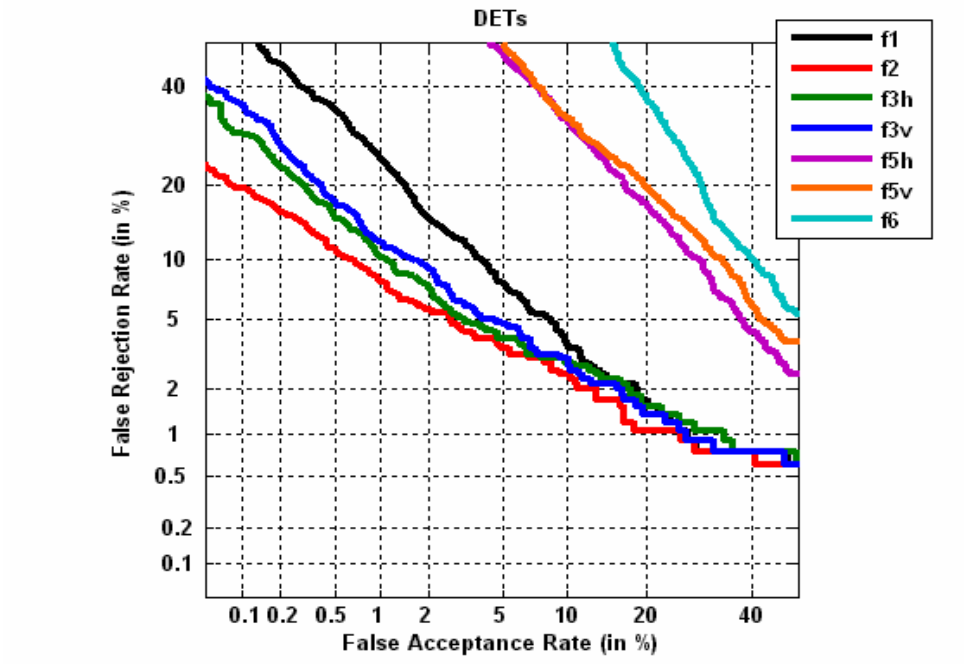


Figura 25. Curvas DET de las características *f1, f2, f3h, f3v, f5h, f5v* y *f6*.

6.2.2 Pruebas marco de referencia: combinación de características

En este escenario se estudia el rendimiento del sistema cuando se emplean combinaciones de características y, al igual que en el caso anterior, se trabaja con todas las líneas de texto disponibles de cada escritor. En este caso la distancia utilizada es la de Hamming. Primero se calcula la distancia de Hamming de cada característica en particular y después se promedian. Las medidas de rendimiento empleadas siguen siendo las de Top 1 y Top 10 para identificación y la Tasa de Igual Error (EER) para verificación.

El objetivo que se persigue en este caso consiste en, además de obtener resultados comparables con los del sistema de referencia [6], estudiar qué combinaciones de características producen mejores resultados. Para ello no sólo se han implementado las combinaciones propuestas en el marco de referencia [6], sino que se han realizado todas fusiones posibles de características.

Las Tablas 5 y 6 muestran los resultados del sistema de referencia publicados por sus autores (Tabla 5) y los obtenidos en el presente proyecto (Tabla 6). Como se puede observar, el rendimiento en ambos casos es similar pudiéndose deber las pequeñas diferencias a las mismas razones comentadas en el escenario anterior. El hecho de probar todas las combinaciones de características posibles tiene como objeto estudiar qué fusiones proporcionan una mayor tasa de rendimiento.

Las características estudiadas en este proyecto (*f1, f2, f3h, f3v, f5h, f5v* y *f6*) se pueden agrupar en tres categorías: PDFs de dirección (*f1, f2, f3h* y *f3v*), PDFs de longitud (*f5h* y *f5v*) y autocorrelación (*f6*). Las fusiones de este escenario se han realizado tanto entre características del mismo grupo como entre características de grupos diferentes.

Analizando en primer lugar las características *f3* y *f5* obtenidas a partir de *f3h* con *f3v* y *f5h* con *f5v,* respectivamente, (combinación de las dos direcciones perpendiculares de la imagen de entrada), lo ideal sería esperar que el rendimiento del sistema implementado mejorase al utilizar las características fusionadas. Sin embargo, se comprueba que mientras que para *f5* sí se mejora el rendimiento, para *f3* se empeora aunque no de manera significativa.

	f3: f3h & f3v	*f5: f5h & f5v*	*f1 & f5*	*f3 & f5*
EER	5,3	9,0	4,0	3,9
Top 1	77	31	68	82
Top 10	91	60	91	94

Tabla 5. Rendimiento del sistema de referencia de identificación y verificación de escritor al emplear combinaciones de características.

	f3: f3h+f3v	*f5: f5h+f5v*	*f1f2*	*f1f3*	*f1f5*	*f2f3*	*f2f5*	*f3f5*	*f1f2f3*	*f1f2f5*	*f2f3f5*	*f1f2f3f5*
EER	5,02	**11,92**	4,72	5,02	**4,74**	4,41	3,96	**3,85**	4,72	**3,81**	**3,56**	**3,80**
Top 1	71,99	**29,53**	68,65	67,12	**64,99**	77,47	77,17	**79,15**	72,15	77,47	82,04	79,76
Top 10	88,59	**58,90**	87,82	86,45	**89,19**	90,72	92,39	**93,46**	88,89	92,09	**93,61**	92,54

Tabla 6. Resultados experimentales de rendimiento del sistema implementado en este proyecto empleando combinaciones de características. Se marcan en negrita los casos donde se mejora el rendimiento respecto a las características individuales.

Es importante destacar que las fusiones de características de dirección entre sí (*f1&f2, f1&f3, f2&f3* y *f1&f2&f3*) no proporcionan mejoras extra sobre el rendimiento de la mejor característica de la combinación, sino que las mejoras se obtienen al combinar características de diferentes categorías (*f1&f5, f3&f5*). La razón de que las fusiones de características de diferentes categorías presenten un mejor rendimiento que las combinaciones dentro de la misma categoría, se debe a que las características de una misma categoría son menos independientes unas de otras. Por ejemplo, todas las PDFs de dirección están formadas a partir de medidas de ángulos y entre ellas existe mayor dependencia que entre cualquier PDF de dirección y una PDF de longitud

Las siguientes figuras (26 a 29) muestran las curvas DET obtenidas para verificación de las combinaciones de características de diferentes categorías:

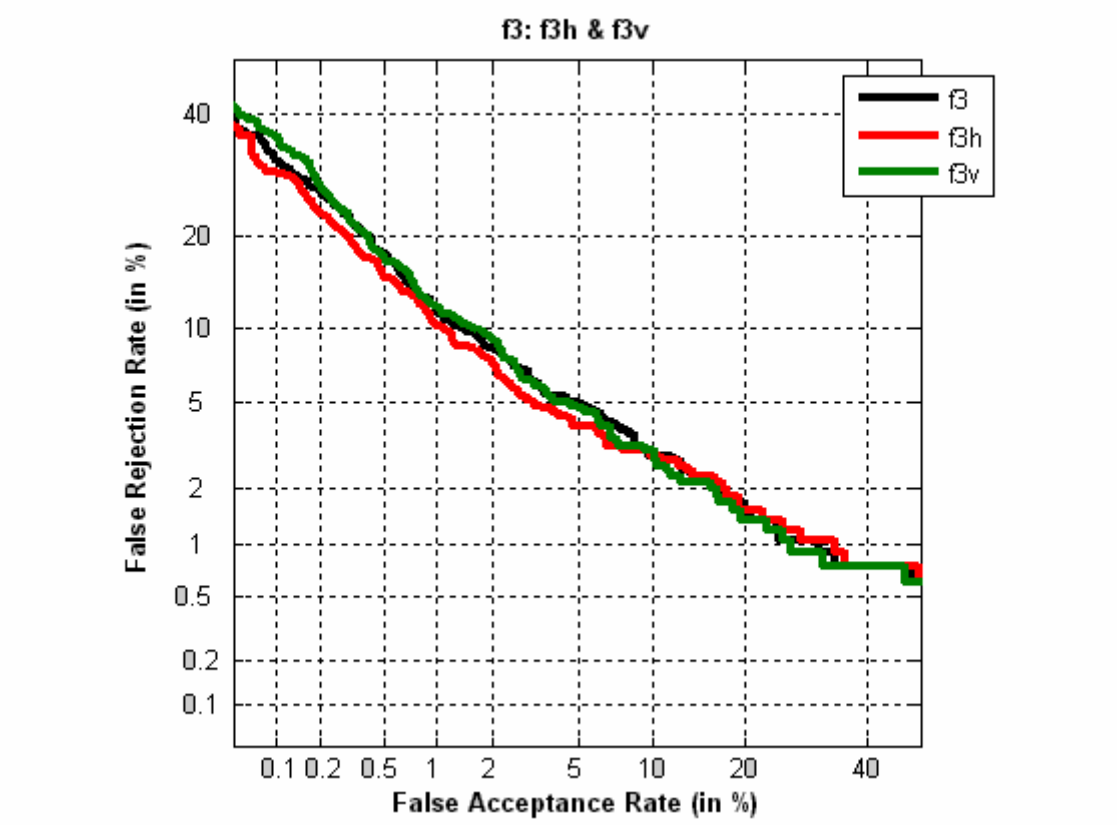


Figura 26. Curva DET de la característica *f3* con las curvas DET de las características individuales que forman la combinación (*f3h* y *f3v*).

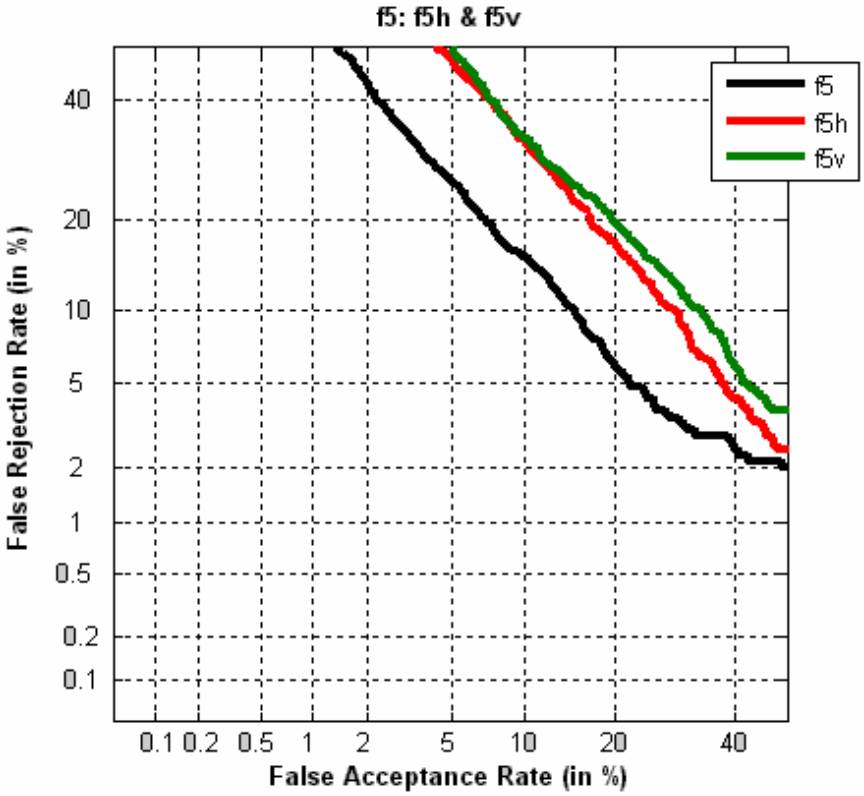


Figura 27. Curva DET de la característica *f5* con las curvas DET de las características individuales que forman la combinación (*f5h* y *f5v*).

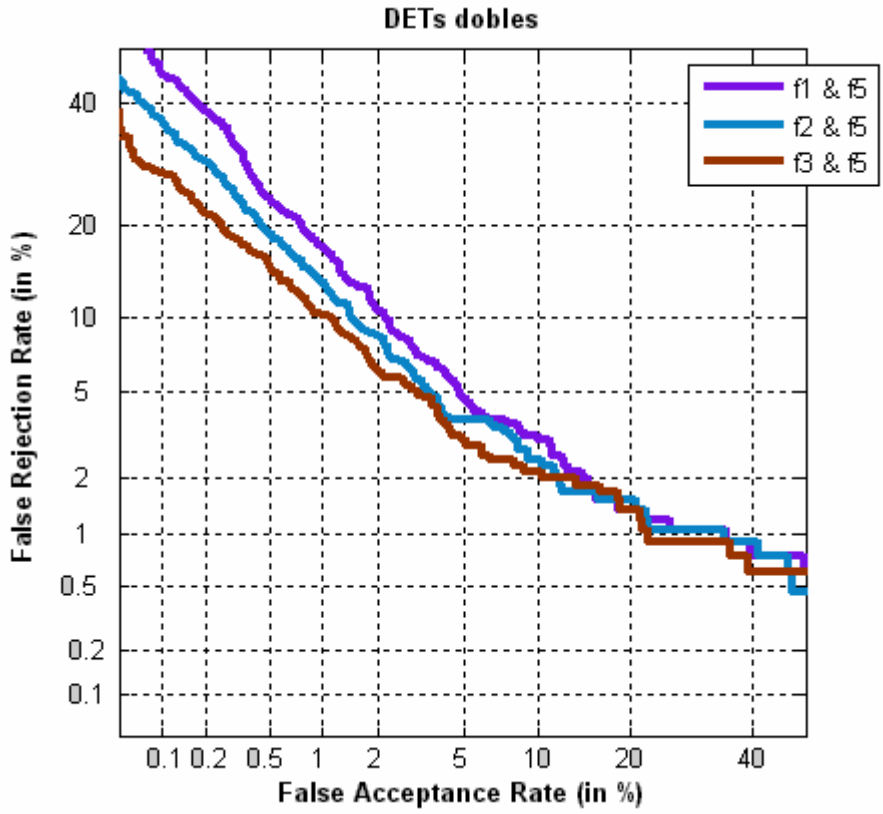


Figura 28. Curvas DET de las combinaciones compuestas por dos características.

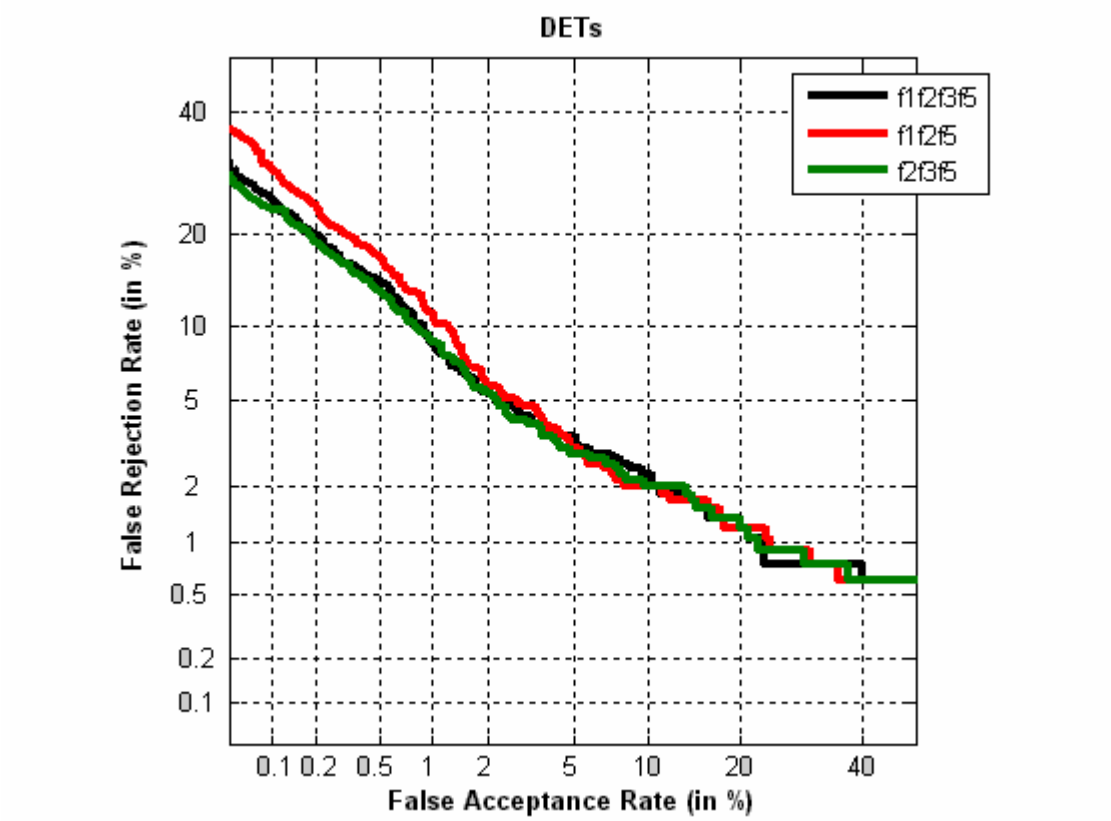


Figura 29. Curvas DET de las combinaciones compuestas por tres y cuatro características.

6.2.3 Pruebas identificación de escritor: Top N

En este escenario de pruebas se estudia cómo influye, en el rendimiento del modo identificación de escritor del sistema desarrollado, el número de candidatos seleccionados para la clasificación Top N de entre todos los posibles escritores. Hasta ahora se habían utilizado dos opciones: en la primera (Top 1) se escogía solamente el primer candidato que aparecía en la lista ordenada una vez calculadas las distancias entre la muestra de entrada y todas las muestras almacenadas en la base de datos, y en la segunda se seleccionaban los diez primeros candidatos (Top 10) de dicha lista.

Los experimentos realizados para implementar este estudio consisten en modificar el número de componentes de la lista del Top desde uno hasta cien escritores. Se ha decidido no experimentar con listas superiores a cien posibles candidatos debido al tamaño de la base de datos, que como ya se comentó en la sección 6.1.1 es de 650 escritores con dos muestras por escritor. Se ha considerado excesivo seleccionar más de 100 usuarios en una base de datos de este tamaño porque la proporción resultaría demasiado grande [3].

A continuación (ver Figuras 30 a 32) se muestran los resultados obtenidos. Se puede comprobar que los experimentos se han realizado tanto para las características empleadas de manera individual (sistema monomodal) (Figura 30) como para las combinaciones de características (sistema multimodal) (Figuras 31 y 32).

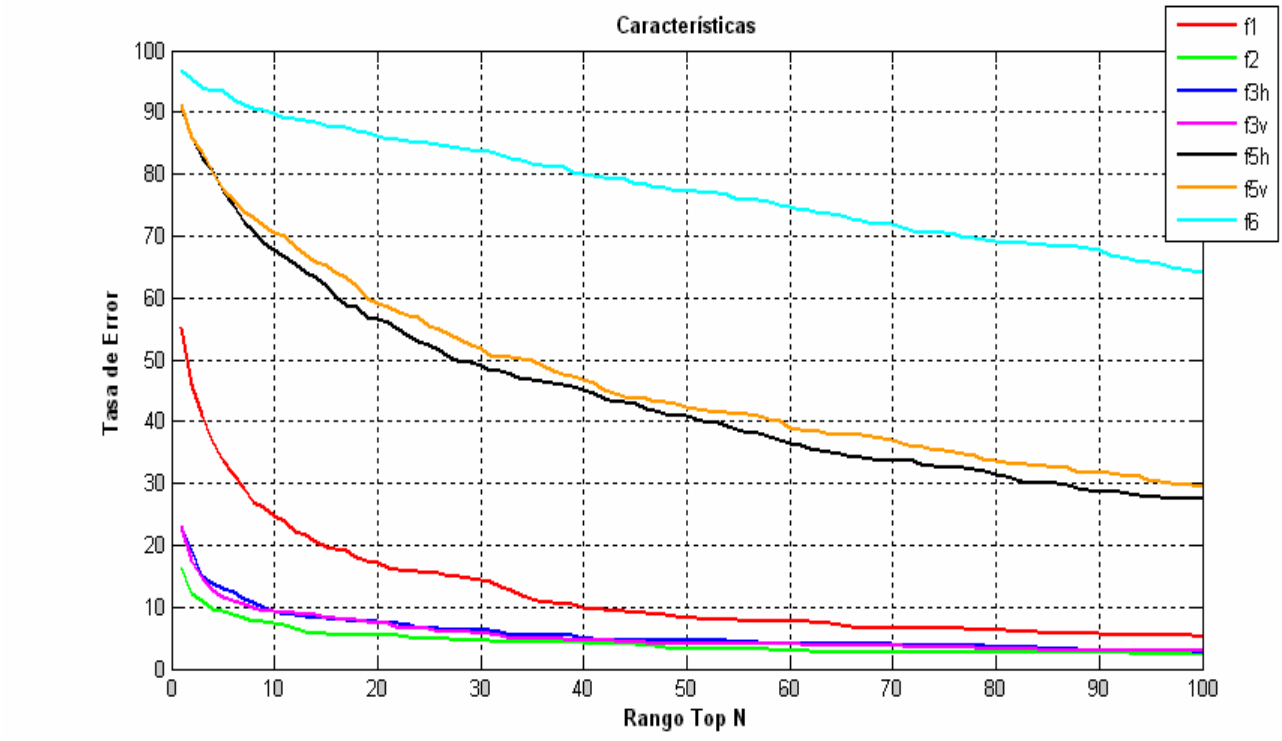


Figura 30a. Comparativa de las tasas de error de las características individuales (*f1*, *f2*, *f3h*, *f3v*, *f5h*, *f5v* y *f6*) en función del número de candidatos en identificación.

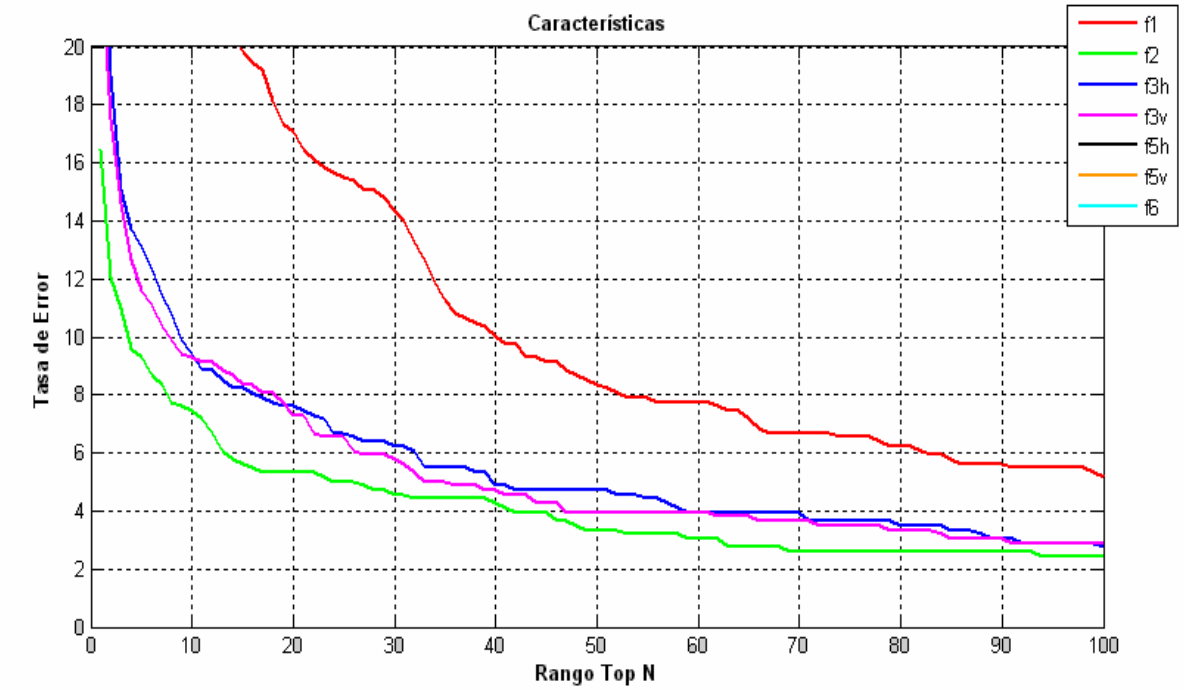


Figura 30b. Ampliación de la gráfica comparativa anterior.

De la observación de la Figura 30a se puede comprobar que a medida que se aumenta el número de posibles candidatos de la lista Top N para las características utilizadas de forma individual, se obtienen mejores resultados en identificación ya que la tasa de error va disminuyendo. Excepto para la autocorrelación (*f6*) en la que la tasa de error decrece a un ritmo constante, en el resto de características y especialmente en las de dirección (*f1*, *f2*, *f3h, f3v*) la tasa de error experimenta una caída significativa al principio al aumentar el número de posibles escritores desde el Top 1 hasta aproximadamente el Top 10 (*f3h* y *f3v*), Top 15 (*f2*) o Top 20 (*f1*) y después se estabiliza decreciendo de manera más lenta. En el caso de la característica *f2* la tasa de error llega a ser prácticamente constante al alcanzar una lista de setenta candidatos (Top 70).

Para *f2*, *f3h* y *f3v* la tasa de error cae por debajo del 3% al emplear listas de 100 posibles escritores, ver Figura 30b. La menor tasa de error es para *f2*, cerca del 2%. Las otras características resultan: *f1* en torno al 5%, *f5h* y *f5v* un 30% y en *f6* un 65%. Por tanto, la característica que mejor funciona en identificación es la de dirección *f2*. Estos resultados son coherentes con lo que comentó en el apartado 6.2.1 en el que se concluía que las características que proporcionan un mejor rendimiento del sistema son las de dirección y dentro de ellas las que combinan ángulos (*f2*, *f3h* y *f3v*), es decir, las que construyen sus patrones con distribuciones de probabilidad conjunta porque capturan más información de la señal de entrada.

Los resultados para los casos el que el sistema se emplea con combinaciones de características se muestran en las siguientes figuras:

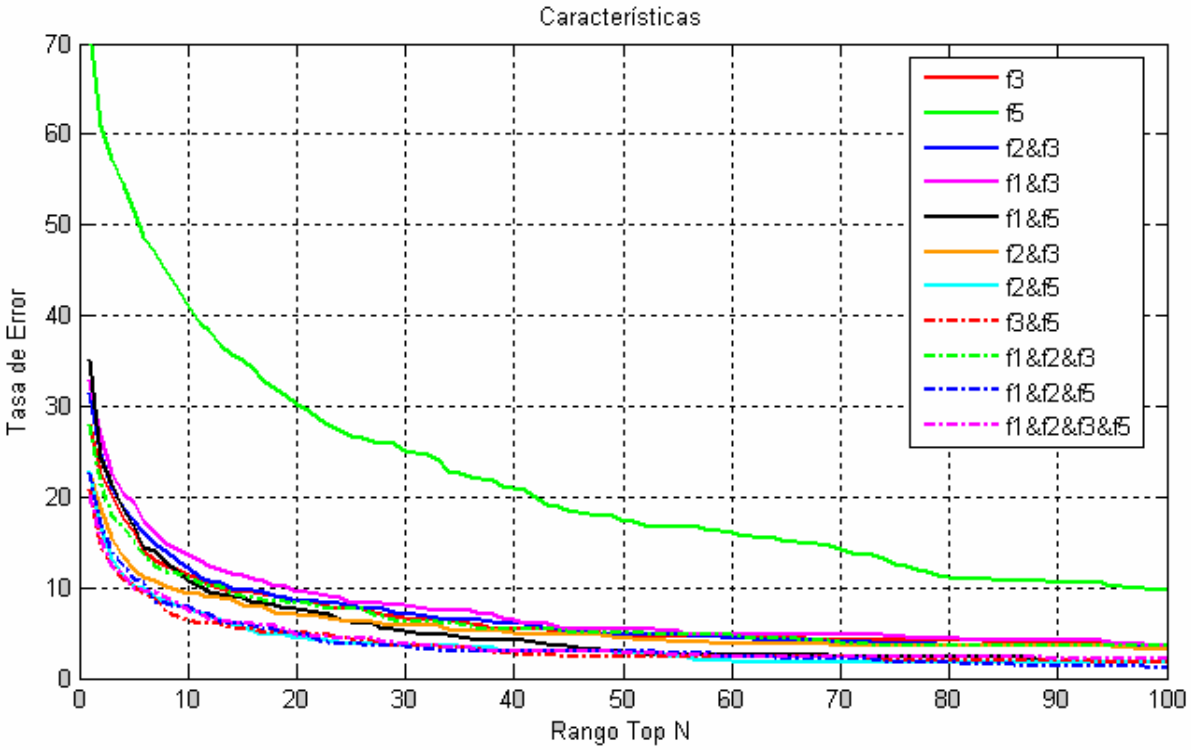


Figura 31. Comparativa de las tasas de error de todas las combinaciones de características en función del número de candidatos en identificación.

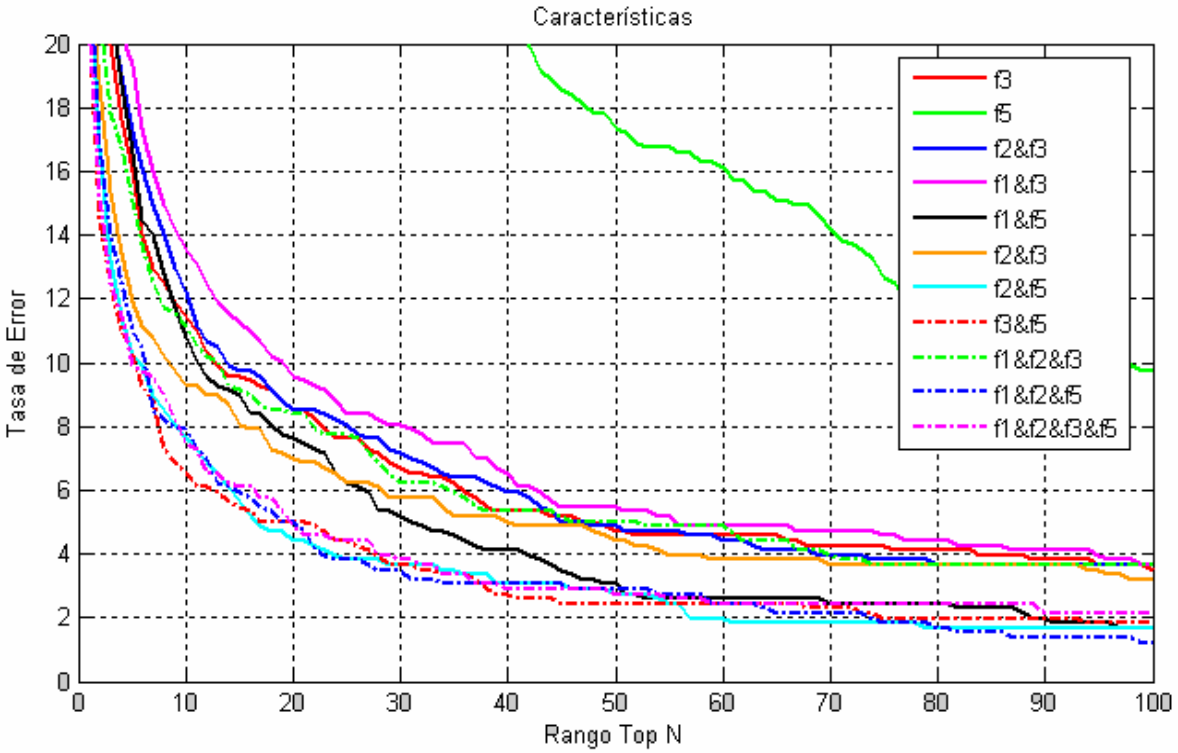


Figura 32. Ampliación de la gráfica comparativa anterior.

Al igual que ocurre con las características individuales, se puede comprobar al observar las figuras anteriores que también en este caso, en el que utilizamos combinaciones de características, al aumentar el número de posibles candidatos de la lista Top N la tasa de error de identificación disminuye, como cabe esperar. Comparando las tasas de error resultantes al emplear las características de forma individual o combinada (Figuras 30-32), se puede apreciar que la fusión de características proporciona una cierta mejora en el rendimiento del sistema. Para una lista de tamaño dado, la fusión de características mejora en torno a un 1% la tasa de acierto (para N=10 tenemos 7.5% frente a 6.5%, por ejemplo). Vemos que para las características de forma individual (Figura 30b), la tasa de error cae aproximadamente a un 2.5% para listas de 100 escritores (Top 100), mientras que para la fusión de características cae hasta un 1.5% aproximadamente.

También se puede observar que la tasa de error decrece significativamente al principio, en general al aumentar el número de posibles escritores de 1 (Top 1) a 10 (Top 10), manteniéndose después prácticamente constante.

Por último, cabe destacar que las fusiones de características de distintas categorías presentan un mejor rendimiento, es decir, una tasa de error menor situada en torno al 2% al llegar al Top 100 (ver Figura 32), mientras que las combinaciones de características de la misma categoría tienen tasas de error mayor, del 5% aproximadamente. El hecho de que las fusiones entre características de categorías diferentes proporcionen mejoras en el rendimiento del sistema respecto a las combinaciones de características de igual categoría también aparecía en los experimentos del escenario de la sección 6.2.2 y se debe a que las características de grupos distintos son más independientes entre ellas que las de un mismo grupo.

6.2.4 Pruebas identificación de escritor: variación del número de escritores de test

El objetivo de este escenario consiste en estudiar cómo influye en el rendimiento del modo identificación de escritor del sistema desarrollado, tanto para Top 1 como para Top 10, el número de escritores de test disponibles. En los experimentos anteriores se ha empleado un conjunto fijo de 650 escritores diferentes para realizar las pruebas. Para este escenario, en cambio, se ha variado el número de escritores del conjunto de test. Para cada grupo de escritores se han realizado veinte iteraciones, en cada una de las cuales se han elegido aleatoriamente a los escritores y se ha calculado la tasa de acierto de identificación. La tasa de acierto final se ha hallado promediando los resultados obtenidos en cada iteración.

Se han efectuado cuatro pruebas, dos de ellas para el caso en el que el sistema utiliza las características de manera individual y las otras dos para cuando usa las combinaciones de características. En ambos casos se ha probado tanto para Top 1 como para Top 10. Los resultados obtenidos se muestran a continuación (ver Figuras 33 a 36):

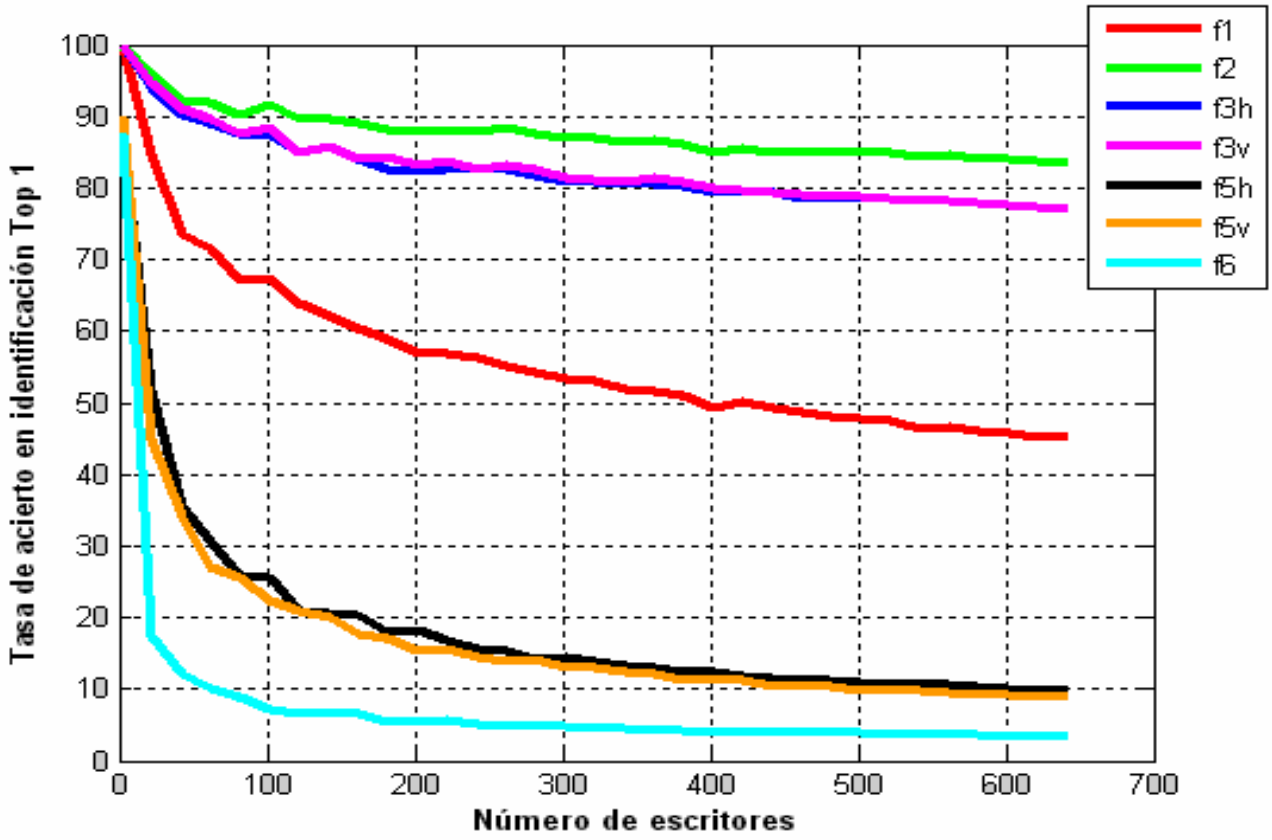


Figura 33. Tasa de acierto en identificación con Top 1 cuando el sistema emplea las características de forma individual.

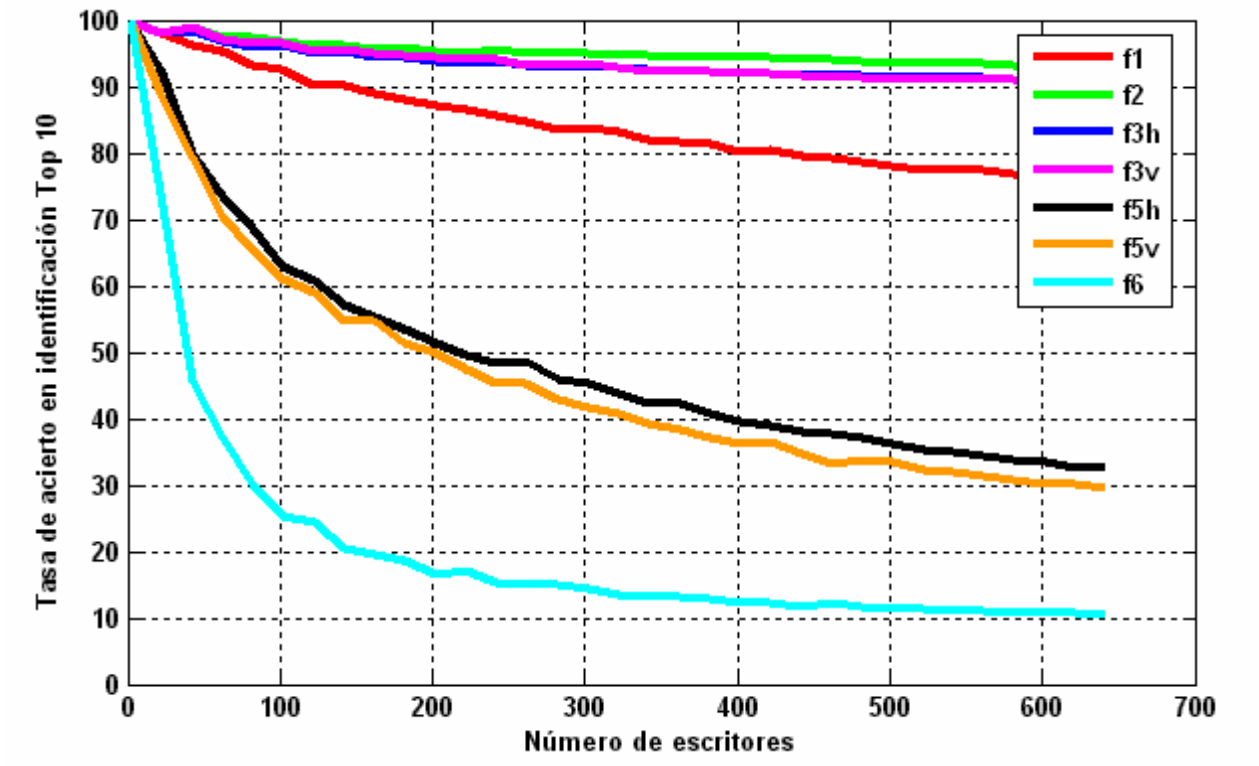


Figura 34. Tasa de acierto en identificación con Top 10 cuando el sistema emplea las características de forma individual.

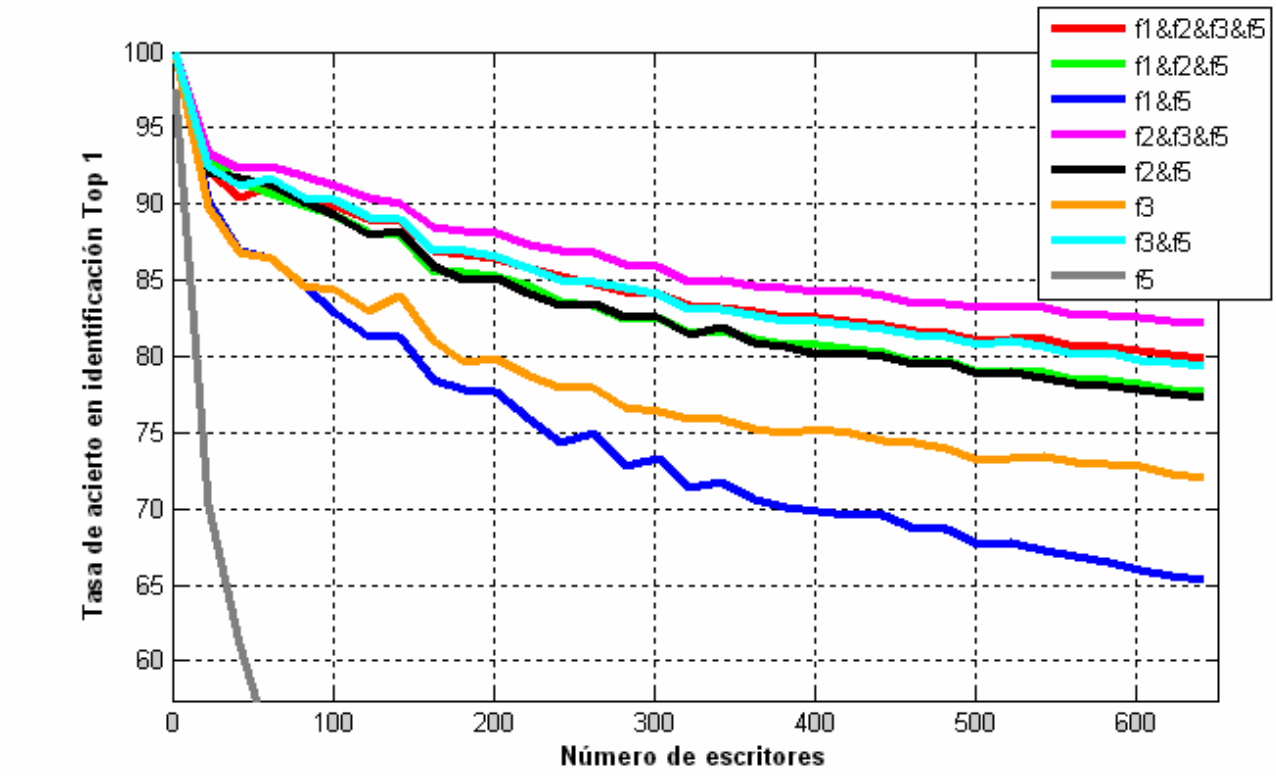


Figura 35. Tasa de acierto en identificación con Top 1 cuando el sistema emplea las combinaciones de características.

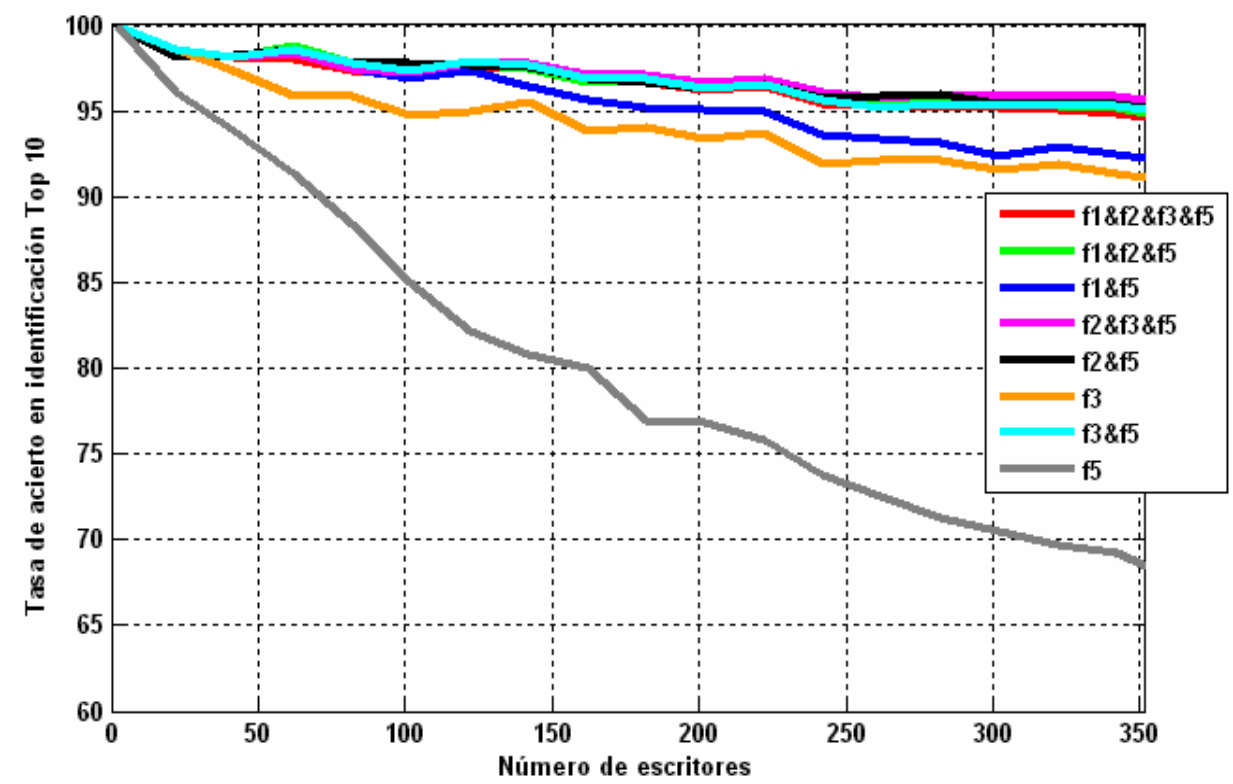


Figura 36. Tasa de acierto en identificación con Top 10 cuando el sistema emplea las combinaciones de características.

Como era de esperar, se puede observar que a medida que aumenta el número de escritores del conjunto de test, la tasa de acierto de identificación disminuye tanto para la clasificación Top 1 como para la clasificación Top 10 y tanto para el sistema en modo monomodal como en modo multimodal. Este resultado es coherente ya que cuantos más candidatos haya entre los que elegir, mayor es la probabilidad de fallar en la elección (seleccionar un impostor). Se puede apreciar que la caída de la tasa de acierto es más significativa al principio cuando el número de componentes del conjunto de test aumenta hasta 50 escritores aproximadamente, y se va haciendo más constante de ahí en adelante (hasta 100 escritores en algunos casos).

Comparando los resultados de la clasificación Top 1 y Top 10 para el sistema monomodal se observa que al igual que en los escenarios anteriores, la característica que mejor funciona, es decir, la de mayor tasa de acierto, es la *f2* seguida de las características *f1, f3h* y *f3v,* y por el contrario, la que peor funciona es la *f6.* Este hecho sigue demostrando que las características de combinación de ángulos (*f1, f2, f3h, f3v*), que realizan un análisis de fase local, producen mejores resultados que las que capturan información de amplitud (*f6)*. También se aprecia que al emplear la lista Top 10 se obtiene un mejor rendimiento del sistema, con tasas de acierto mayores, que al utilizar el Top 1, ya que la probabilidad de que el escritor se encuentre en la clasificación Top 10, que contiene diez posibles candidatos, es mayor que la probabilidad de que se encuentre en la Top 1 que sólo contiene un candidato.

Para el sistema en modo multimodal los resultados obtenidos también son mejores al utilizar la lista Top 10 que la lista Top 1. Con la clasificación Top 10 la tasa de acierto mejora en torno a un 10% respecto a la Top 1 en todas las combinaciones de características excepto en la de *f5* (*f5h* & *f5v*). Esta fusión presenta un comportamiento completamente diferente al resto ya que su tasa de acierto disminuye rápidamente y de forma brusca al aumentar el número de escritores de test mientras que para el resto de combinaciones de características la tasa de acierto decae al principio y después se mantiene más o menos estable.

Por último se puede apreciar que la fusión de características proporciona un mejor rendimiento del sistema, con tasas de acierto mayores (en general por encima del 70% excepto para *f5*), que las características individuales, cuya tasa de acierto disminuye hasta llegar al 10% aproximadamente en algunas características. Esto es debido, como ya se ha comentado en secciones anteriores, a que la combinación de información que se produce con la fusión de características, resulta en mejoras del funcionamiento del sistema.

6.2.5 Pruebas identificación y verificación: variación del número de líneas

En este último escenario de pruebas el objetivo consiste en estudiar la influencia que tiene la cantidad de texto disponible de cada escritor en el rendimiento del sistema, tanto en identificación como en verificación. Para ello se ha variado el número de líneas de cada modelo de la base de datos desde una línea hasta nueve (máximo número de líneas encontrado para un escritor) correspondiente a una página entera de texto.

Es importante destacar que no todas las muestras tienen la misma extensión, por lo que llegados a cierto número de líneas el número de escritores disponibles para la experimentación disminuye. Este hecho no afecta tanto al rendimiento del sistema en verificación como al rendimiento del mismo en identificación. La razón de que la tasa de error de verificación no se vea afectada por el número de usuarios contenidos en la base de datos se debe a que en verificación se realiza una comparación uno a uno y para la de tasa de error es indiferente hacer más o menos comparaciones, solamente se ve afectada la significancia estadística de los resultados obtenidos. En cambio, en identificación sí es muy importante el número de componentes de la base de datos, ya que para identificar a un usuario es necesario realizar una comparación de uno a muchos y cuantos más candidatos haya más probabilidad existe de cometer un error. Por tanto, al interpretar los resultados obtenidos en identificación hay que tener en cuenta dos factores: el número de líneas que contienen las muestras y el número de usuarios con los que se ha desarrollado el experimento. A continuación se muestra una tabla (Tabla 7) con el número de escritores de la base de datos no disponibles en cada prueba por no contener todas las líneas necesarias:

Número de líneas	Todas	1	2	3	4	5	6	7	8	9
Escritores no disponibles	0	0	1	7	90	244	356	406	481	564

Tabla 7. Número de escritores no disponibles según el número de líneas necesario por muestra.

Tras observar la Tabla 7 vamos a suponer que para interpretar los resultados obtenidos en identificación, no es necesario tener en cuenta el número de escritores disponibles de la base de datos hasta llegar a muestras de cuatro/cinco líneas. Esta suposición se basa en que hasta ese punto se está utilizando la mayoría de los escritores que forman la base de datos (aproximadamente un 86% del total en el caso de cuatro líneas por muestra) por lo que el resultado no se verá significativamente afectado. Otra razón que justifica esta suposición es que el número medio de líneas por escritor de toda la base de datos es de 4,133 líneas, es decir, cuatro líneas por muestra.

Las pruebas realizadas en este escenario tienen en cuenta el funcionamiento del sistema tanto en modo monomodal (características individuales) como multimodal (fusión de características). Para el rendimiento en identificación se incluye la clasificación Top 1 y la Top 10. A continuación se muestran los resultados obtenidos (ver Figuras 37 a 42).

Las Figuras 37 y 38 representan el rendimiento del sistema obtenido en verificación (EER) frente al número de líneas contenidas en las muestras:

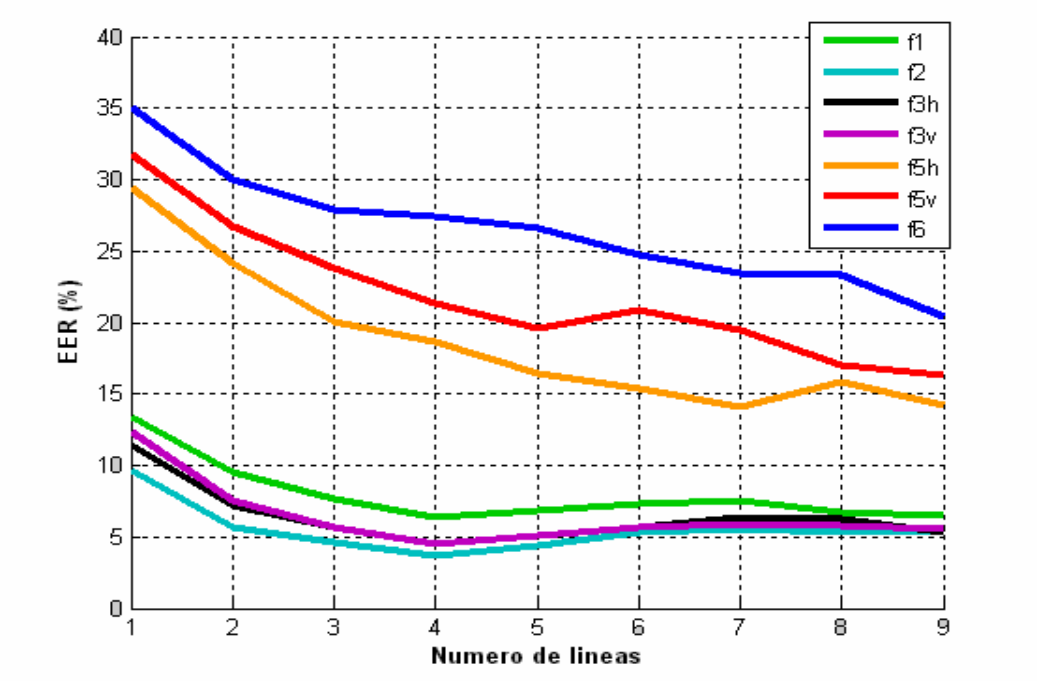


Figura 37. Rendimiento del sistema en verificación al utilizar las características de manera individual.

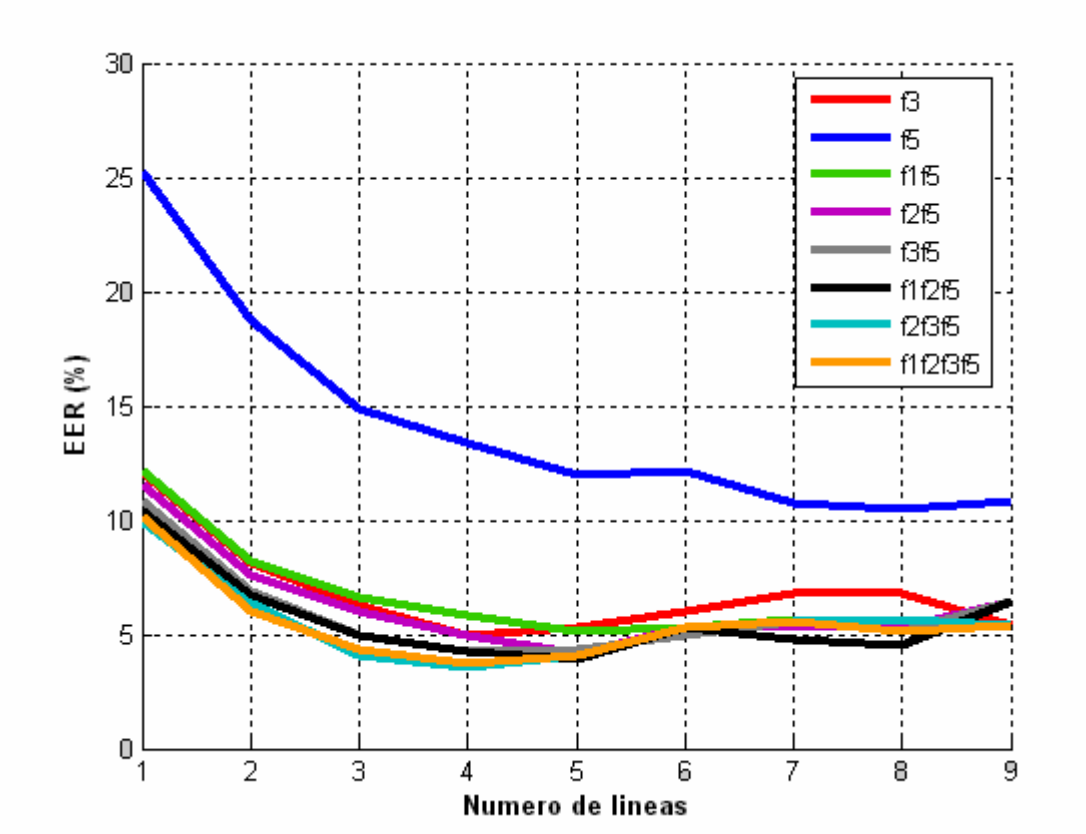


Figura 38. Rendimiento del sistema en verificación al utilizar combinaciones de características.

Se puede observar que al aumentar el número de líneas de las muestras la tasa de error disminuye tanto para el caso en el que el sistema utiliza las características individuales como cuando emplea combinaciones de ellas, por lo que cuánto más cantidad de texto esté disponible en los modelos mejor rendimiento se obtendrá en verificación.

También se puede apreciar que en torno a las cuatro líneas, coincidiendo con el número medio de líneas calculado anteriormente, se produce el mínimo error de verificación en ambos casos.

En general, aunque el sistema multimodal presenta mejor rendimiento que el sistema monomodal, no se observa una diferencia significativa entre emplear fusión de características o utilizarlas individualmente, en este caso.

Por último, se comprueba que las características individuales que mejor rendimiento producen son las de combinación de ángulos (*f1, f2, f3h* y *f3v*), en concreto *f2*, y la que mayor tasa de error presenta es *f6*, como ya ocurría en los escenarios anteriores. Por tanto, se puede decir que la cantidad de texto de las muestras no afecta al comportamiento de las características en términos de cuál proporciona menor tasa de error y cuál mayor. Este comportamiento también se puede observar para las combinaciones de características.

En las siguientes figuras (39 a 42) se muestran los resultados obtenidos por el sistema implementado para identificación al variar el número de líneas de las muestras:

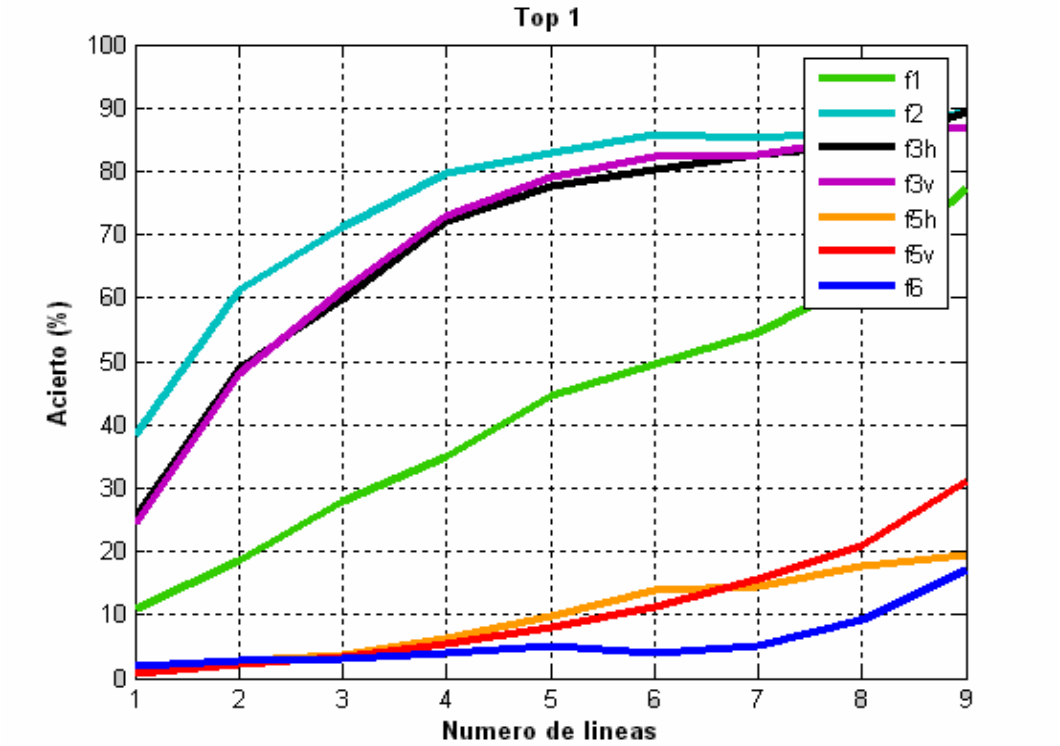


Figura 39. Rendimiento del sistema en identificación con Top 1 en función del número de líneas contenidas en las muestras para las características individuales.

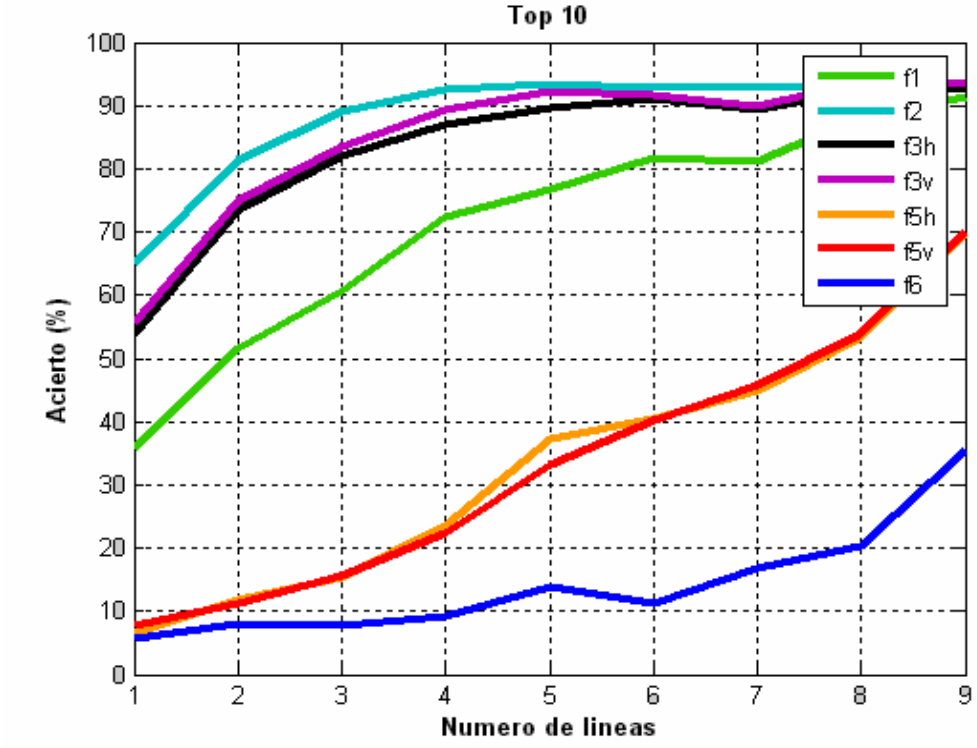


Figura 40. Rendimiento del sistema en identificación con Top 10 en función del número de líneas contenidas en las muestras para las características individuales.

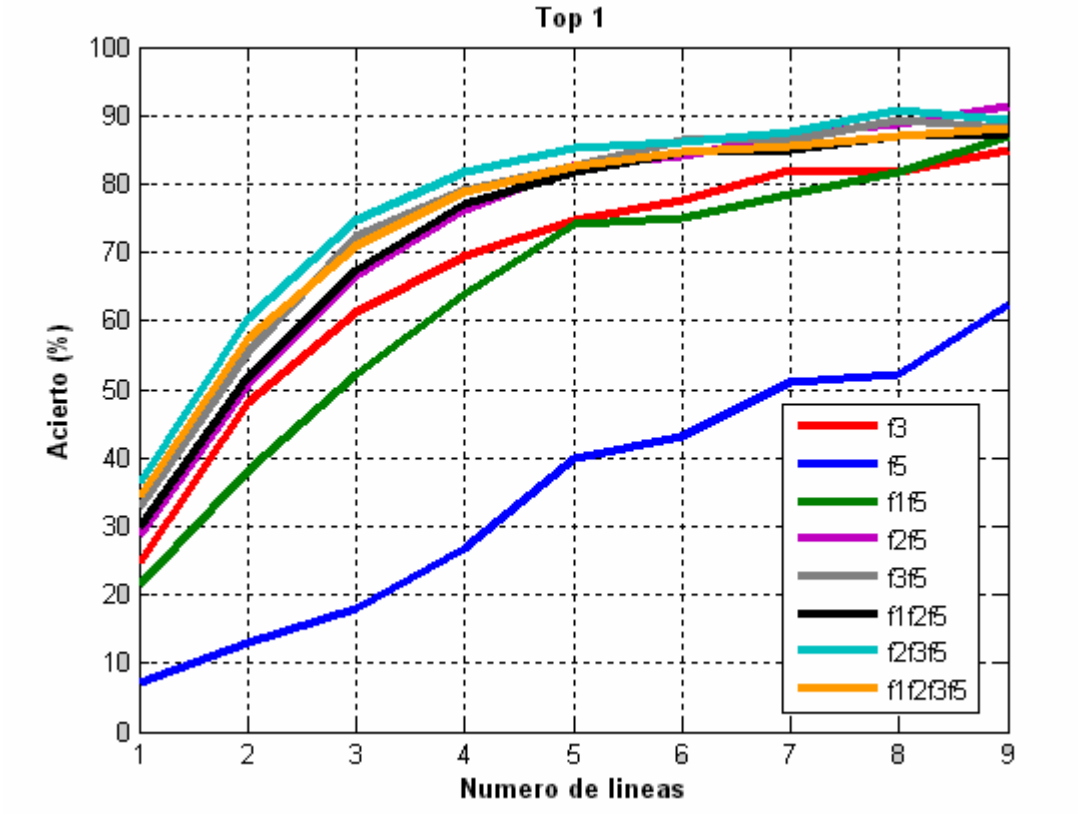


Figura 41. Rendimiento del sistema en identificación con Top 1 en función del número de líneas contenidas en las muestras para las combinaciones de características.

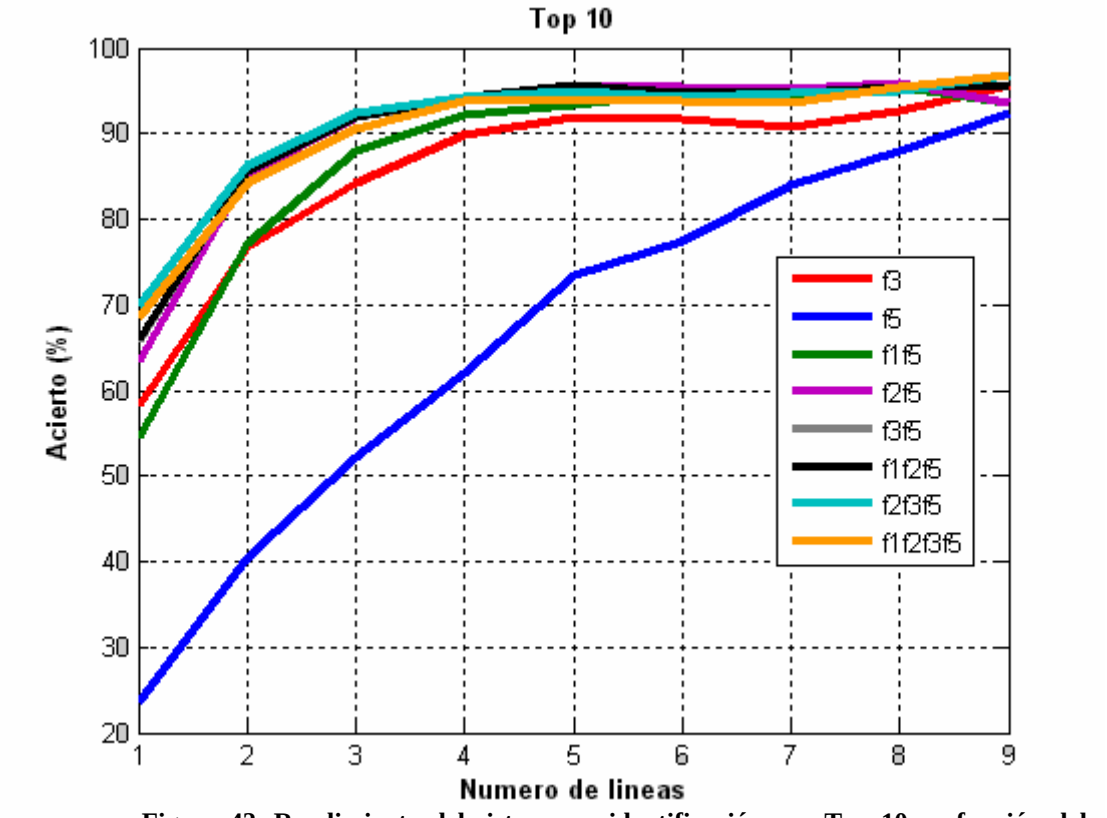


Figura 42. Rendimiento del sistema en identificación con Top 10 en función del número de líneas contenidas en las muestras para las combinaciones de características.

Hasta muestras de cuatro líneas

Considerando los resultados que se obtienen hasta alcanzar muestras de cuatro líneas, éstas incluidas, se observa una mejora progresiva de la tasa de acierto a medida que aumenta la cantidad de texto disponible. Este hecho confirma que, al igual que ocurría en verificación, cuántas más líneas de texto contengan las muestras mejor rendimiento produce el sistema en identificación.

El uso de la clasificación Top 10 proporciona un mejor rendimiento que el uso de la lista Top 1 tanto para las combinaciones de características como para las características individuales, con tasas de acierto alrededor del 90% (para muestras de cuatro líneas) frente al 80% del Top 1. Este comportamiento se corresponde con el observado a lo largo de los experimentos realizados en el presente proyecto.

Respecto a las combinaciones de características frente a las características individuales, se puede observar que las primeras presentan un mejor rendimiento que las últimas. Este hecho coincide con lo que se ha observado en todos los escenarios de pruebas y se debe a que la fusión de características combina la información capturada independientemente por cada característica que forma parte del grupo.

Por tanto se puede concluir que disponer de más texto mejora el rendimiento del sistema en identificación, siempre que se mantenga el número de usuarios, pero no influye en el comportamiento de las combinaciones de características frente a las características individuales y en la lista Top 1 frente a la lista Top 10.

A partir de muestras de cuatro líneas

Para interpretar los resultados obtenidos a partir de los experimentos de muestras con cinco líneas en adelante hay tener en cuenta, además del número de líneas de las muestras el número de escritores con los que se experimenta. Como éstos disminuyen a medida que se aumentan las líneas, es lógico que la tasa de acierto aumente porque hay menos posibles usuarios.

En las Figuras 39 a 42 se observa que como era de esperar la tasa de acierto aumenta. También se puede apreciar que el comportamiento de los resultados sigue las mismas pautas que los resultados obtenidos para las cuatro primeras líneas. Al no ser, en general, muy significativa la mejora del rendimiento al disponer de más líneas de texto se puede deducir que a partir de cierto punto el tener más o menos cantidad de texto no influye de manera importante en el funcionamiento del sistema.

Capítulo 7

Conclusiones y trabajo futuro

En el presente proyecto se ha estudiado, desarrollado, implementado y documentado un sistema de identificación y verificación de escritor basado en características de textura como fuente de información de la individualidad de la escritura. La forma habitual de coger el bolígrafo al escribir junto con la inclinación preferente de los trazos y la curvatura de las letras, se reflejan en estas características que operan en la escala del grosor del trazo de tinta. Las características se modelan como distribuciones de probabilidad extraídas de imágenes de textos y caracterizan de forma robusta e independiente del texto el estilo individual de escritura.

A través de los diferentes escenarios de experimentación se ha comprobado el rendimiento del sistema desarrollado, tanto en identificación como en verificación, cuando éste opera con las características individuales y cuando opera con combinaciones de características. También se ha estudiado la influencia que tiene en la identificación de escritor el número de posibles candidatos que forman la lista Top N y el número de escritores del conjunto de test. Por último se ha estudiado cómo afecta al funcionamiento del sistema la cantidad de texto disponible.

Comparando los resultados obtenidos en los primeros experimentos, en los que se comprobaba el funcionamiento del sistema monomodal y el funcionamiento del sistema multimodal, se puede concluir que la fusión de características es beneficiosa para el rendimiento del sistema. Esta afirmación se ha ido corroborando en los escenarios restantes aunque no fuese su objetivo directo, ya que en todos ellos se podía observar que al combinar las características las tasas de error en verificación eran menores y las tasas de acierto en identificación mayores.

En los experimentos centrados en el modo de identificación de escritor se ha comprobado que cuantos más usuarios contenga la lista Top N mayor es la tasa de acierto. Este resultado se basa en que cuantos más posibles candidatos se seleccionen más probable es que el escritor buscado se encuentre entre ellos. Por otro lado, al variar el número de escritores que forman parte del conjunto de test se ha observado que cuando éstos aumentan también lo hace la tasa de error. Esto es debido a que cuantos más usuarios existan entre los que elegir mayor es la probabilidad de seleccionar un impostor, puesto que más escritores en el conjunto de test implican más impostores.

Respecto a la influencia de la cantidad de texto disponible en las muestras de la base de datos en el rendimiento del sistema implementado, ha sido necesario diferenciar entre modo verificación y modo identificación para interpretar los resultados. La razón de esta diferenciación consiste en que el número de componentes de la base de datos que utilizamos disminuye al requerir, en cada prueba, muestras con más texto. En verificación tener una base de datos más o menos amplia no afecta significativamente a los resultados porque las comparaciones son uno a uno. Sin embargo, en identificación el tamaño de la base de datos es un factor muy importante con el que hay que contar porque las comparaciones son de uno a muchos y cuanto menor sea la base de datos menor es la probabilidad de cometer un error.

Analizando los resultados de este último escenario, se ha comprobado que el rendimiento del sistema en verificación mejora al disponer de muestras con más texto. Para identificación, considerando los resultados obtenidos mientras no varía el tamaño

de la base de datos, se observa el mismo comportamiento que en verificación, es decir, cuanto más texto esté disponible mejor funciona el sistema.

Como trabajo futuro se propone en primer lugar la utilización de este sistema de identificación y verificación de escritor con otras bases de datos y con otros idiomas. En este proyecto se ha utilizado la base de datos IAM compuesta por formularios en inglés que combinan tanto letras mayúsculas como minúsculas. Una opción interesante consistiría en ejecutar este sistema con idiomas cuyos caracteres distasen de los ingleses y comparar los rendimientos. También sería interesante observar el comportamiento del sistema con textos íntegramente en mayúsculas y textos íntegramente en minúsculas.

Continuando con el trabajo realizado en este proyecto una línea a seguir está en incluir un análisis de otro tipo de características que proporcionen información diferente sobre los textos escritos y así hacer más robusto y fiable el sistema. Además se propone comparar el sistema desarrollado con otros sistemas de identificación y verificación de escritor distintos, con el objetivo de mejorar los puntos en los que resulte más vulnerable.

Por otro lado, el proyecto abre una amplia gama de posibilidades dentro del campo de escritor y supone un punto de partida a partir del cual se puede continuar la labor de investigación.

Capítulo 8

Referencias

Referencias

[1] A. K. Jain, A. Ross and S. Pankanti, "Biometrics: A Tool for Information Security", IEEE Transactions on Information Forensics and Security Vol. 1, No. 2, pp. 125-143, June 2006.

[2] A. K. Jain, A. Ross and S. Prabhakar, "An Introduction to Biometric Recognition", IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image-and-Video-Based Biometrics, Vol. 14, No. 1, pp. 4-20, January 2004.

[3] A. Schlapbach and H. Bunke, "A Writer Identification and Verification System Using HMM Based Recognizers", Pattern Analysis and Applications, Vol 10, No. 1, pp. 33-43, 2007.

[4] B. Bidyut Chaudhuri, "Digital Document Processing : Major Directions and Recent Advances", Springer, 2006.

[5] M. Bulacu, "Statistical Pattern Recognition for Automatic Writer Identification and Verification", PhD Thesis, University of Groningen, The Netherlands, 2006.

[6] M. Bulacu, L. Schomaker, "Text-Independent Writer Identification and Verification Using Textural and Allographic Features", IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Special Issue – Biometrics: Progress and Directions, Vol. 29, No. 4, pp. 701-717, April 2007.

[7] L. Schomaker, "Advances in Writer Identification and Verification", International Conference on Document Analysis and Recognition (ICDAR), Keynote Speech, 2007.

[8] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms", IEEE Trans. on Systems, Man and Cybernetics 9, pp. 62–66, 1979.

[9] R. Niels, L. Vuurpijl and L. Schomaker (2007),"Automatic Allograph Matching in Forensic Writer Identification", International Journal of Pattern Recognition and Artificial Intelligence, Vol. 21, No. 1, pp. 61-81, 2007.

[10] U. Marti and H. Bunke, "The IAM-database: an English Sentence Database for Off-line Handwriting Recognition", Int. Journal on Document Analysis and Recognition, Vol. 5, pp. 39 - 46, 2002.

[11] R. Plamodon and G. Lorette, "Automatic Signature Verification and Writer Identification – The State of the Art", Pattern Recognition, Vol. 22, No. 2, pp. 107 – 131, 1989.

[12] H. Said, T. Tan and K. Baker, "Personal Identification Based on Handwriting", Pattern Recognition, Vol. 33, No. 1, pp. 149 – 160, 2000.

[13] E. Zois and V. Anastassopoulos, "Morphological Waveform Coding for Writer Identification", Pattern Recognition, Vol. 33, No. 3, pp. 385 – 398, Mar. 2000.

[14] S. Srihari, M. Beal, K. Bandi, V. Shah and P. Krishnamurthy, "A Statistical Model for Writer Veritification", Proc. Eigth Int'l Conf. Document Analysis and Recognition (ICDAR), pp. 1105 – 1109, 2005.

[15] A. Bensefia, T. Paquet and L. Heutte, "A Writer Identification and Verification System", Pattern Recognition Letters, Vol. 26, No. 10, pp. 2080 - 2092, Oct. 2005.

[16] C. Herel and H. Bunke, "A set of Novel Features for Writer Identification", Proc. Fourth Int'l Conf. Audio and Video – Based Biometric Person Authentication, pp. 679 – 687, 2003.

Anexo I

Pliego de condiciones

Medios materiales

- Ordenador personal tipo PC con sistema operativo Windows para el desarrollo y ejecución del *software* así como para a documentación de los trabajos realizados.
- Impresora a color.

Software

- Matlab versión 7.0 para el desarrollo y ejecución del software así como para la obtención de las gráficas e imágenes resultantes.
- Microsoft Office para la documentación de los trabajos realizados.

Presupuesto

El cálculo del presupuesto está realizado en base al coste del material y a la mano de obra utilizada.

1. Material empleado

	COSTE
Ordenador e impresora	*3.125 euros*
Software	*1.250 euros*
Material de oficina	*187 euros*
TOTAL	***4.562 euros***

2. Mano de obra

Los costes de la mano de obra para la realización del proyecto se calculan a partir del salario base diario por persona, de las cargas sociales aplicables y del total de jornadas empleadas.

La ejecución del proyecto ha requerido de la participación de:

- Un ingeniero de Telecomunicación, responsable de la dirección y realización del Proyecto, junto con su mecanografiado y edición.

La cuantía de cada uno de los conceptos de la mano de obra es la siguiente:

- Salario base

	Salario base diario
Ingeniero de Telecomunicación	*84 euros*
Mecanógrafo	*46 euros*

- Cargas sociales

CARGA SOCIAL	**IMPORTE (%)**
Vacaciones Anuales	*4,86*
Seguro de accidentes	*6,80*
Subsidio familiar	*2,70*
Indemnización de despido	*0,70*
Subsidio de vejez	*2,00*
Abono días festivos	*15,33*
Días de baja por enfermedad	*3,80*
Cuota sindical	*2,70*
Seguro de enfermedad	*1,00*
Cargas familiares	*4,20*

Pagas extraordinarias	*24,00*
Plus por carestía de vida	*9,40*
Paro tecnológico	*17,20*
TOTAL (% sobre el sueldo base)	***94,69***

- Total de jornadas empleadas

	JORNADAS (*8 horas*)
Ingeniero de Telecomunicación	*100*
Mecanógrafo	*45*

El total de la mano de obra es:

	JORNADAS	**SALARIO EFECTIVO**	**COSTE TOTAL**
Ingeniero de Telecomunicación	*100*	*163,53 euros*	*16.353 euros*
Mecanógrafo	*45*	*89,55 euros*	*4.030 euros*
TOTAL MANO DE OBRA			***20.383 euros***

3. Coste Total

	COSTE
Total material empleado	*4.562,87 euros*
Total mano de obra	*20.383 euros*
Suma total	*20.915,87 euros*
I.V.A (16%)	*3.346,53 euros*
IMPORTE TOTAL	***49.208,27 euros***

Por tanto, el importe total del presente proyecto asciende a la cantidad de:

Cuarenta y nueve mil doscientos ocho euros con veintisiete céntimos de euro

Anexo II

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Bnueva=algoritmoMoore(dim,matriz)
%   Bnueva: vector que contiene la posición (x,y)
%   de los píxeles del contorno
%   dim: dimensión del cuadrado que contiene el componente
%   matriz: imagen con el componente
%   Descripción: función que calcula el contorno externo
%   del componente
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Bnueva=algoritmoMoore(dim,matriz)

%x representa las filas
%y representa las columnas

Bnueva=[];
preant_x=1;
preant_y=dim;
anterior_x=1;
anterior_y=dim;
h=0;    %flag para el recorrido en el sentido de las agujas del reloj
%Comienzo del algoritmo
%Se recorre la matriz de abajo a arriba y de izqda a dcha para
%encontrar el primer pixel
for i=1:dim      %columnas
    for j=1:dim  %filas
        j=dim+1-j;
        if matriz(j,i)==1
            s=[j;i];
            Bnueva=[Bnueva s];
            comienzo_x=j;  %pixel de comienzo del componente conectado
            comienzo_y=i;
            p_x=comienzo_x; %pixel de partida
            p_y=comienzo_y;
            ant_x=anterior_x;   %se retocede al pixel anterior
            ant_y=anterior_y;
            preant_x=p_x;
            preant_y=p_y;
            [c_x c_y]=siguientepartida(p_x,p_y,ant_x,ant_y);
            %siguente pixel segun las agujas del reloj
            c=matriz(c_x,c_y);
            h=2;
            while c_x~=comienzo_x | c_y~=comienzo_y |
ant_x~=anterior_x | ant_y~=anterior_y
                if c==1
                    b=[c_x;c_y];
                    Bnueva=[Bnueva b];
                    p_x=c_x;    %pixel de partida
                    p_y=c_y;
                    c_x=ant_x;  %se retocede al pixel anterior
                    c_y=ant_y;
                    c=matriz(c_x,c_y);
                    preant_x=ant_x;
                    preant_y=ant_y;
                    ant_x=p_x;
                    ant_y=p_y;
                    h=1;
                else
```

```
                    [n_x
n_y]=siguientenormal(c_x,c_y,ant_x,ant_y,preant_x,preant_y,h);
%siguente pixel segun las agujas del reloj
                     c=matriz(n_x,n_y);
                     preant_x=ant_x;
                     preant_y=ant_y;
                     ant_x=c_x;
                     ant_y=c_y;
                     c_x=n_x;
                     c_y=n_y;
                     if h==1
                         h=2;
                     else
                         h=0;
                     end
                 end
             end
             return;
         end
         preant_x=anterior_x;
         preant_y=anterior_y;
         anterior_x=j;
         anterior_y=i;
     end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   autocor=autocorrelacion(imagen)
%   autocor: estructura con la autocorrelación
%   y el número de filas
%   imagen: imagen en escala de gris con la muestra
%   de escritura
%   Descripción: función que calcula la autocorrelación
%   de cada fila de la imagen
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function autocor=autocorrelacion(imagen)

[fils,cols]=size(imagen);
desp=60;

for i=0:desp    %para cada desplazamiento
    for k=1:fils,   %para cada fila
        aux1=double(imagen(k,:));
        aux2=double(imagen(k,i+1:end));
        aux2=[aux2 double(imagen(k,1:i))];
        autocorrelacionfila(k)=sum(aux1.*aux2)/cols;
    end
    autocorrelacion(i+1)=sum(autocorrelacionfila);
end

autocor=struct('autocorrelacion',autocorrelacion,'nfils',fils);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script que busca las carpetas de usuarios que
%   están vacías porque no poseen muestras con tantas
%   líneas como con las que se está experimentando y
%   almacena en un vector la identificación del escritor
%   correspondiente
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all;
clc;
clear all;

vacias_entrenamiento=[];
vacias_test=[];

for w=1:2
    if w==1
        carpeta=dir('./UsuariosEntrenamiento2/');
        nombre_carpeta=['./UsuariosEntrenamiento2/'];
    else
        carpeta=dir('./UsuariosTest2');
        nombre_carpeta=['./UsuariosTest2/'];
    end

    %Se hallan las carpetas que estan vacias
    for n=3:length(carpeta)
        usuario=carpeta(n).name;
        contenido=dir([nombre_carpeta usuario]);
        longitud=length(contenido);
        if longitud<3
            for i=1:length(usuario)
                if usuario(i)=='o'
                    numero_usuario=usuario(i+1:end);
                end
            end
            if
length(nombre_carpeta)==length(['./UsuariosEntrenamiento2/'])
                vacias_entrenamiento=[vacias_entrenamiento '-'
numero_usuario];
            else
                vacias_test=[vacias_test '-' numero_usuario];
            end
        end
    end
end

vacias_entrenamiento=vacias_entrenamiento(2:end);
vacias_test=vacias_test(2:end);
for i=1:length(vacias_entrenamiento)
    if vacias_entrenamiento(i)=='-'
        vacias_entrenamiento(i)=' ';
    end
end

for i=1:length(vacias_test)
    if vacias_test(i)=='-'
        vacias_test(i)=' ';
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  [funcionPDFtotal,anguloradianvector,dim]=contourdirection(contorno)
%   funcionPDFtotal: vector con los resultados de la función de
%   distribución de probabilidad calculada
%   anguloradianvector: vector con los resultados de la función de
%   distribución de probabilidad calculada en coordenadas polares
%   dim: dimensión del vector anguloradianvector
%   contorno: contorno del componente
%   Descripción: función que calcula característica f1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function
[funcionPDFtotal,anguloradianvector,dim]=contourdirection(contorno)

e=5;    %controla la longitud del fragmento de contorno analizado
grados=15;  %cada intervalo del histograma abarca 15º
n=12;   %numero de intervalos del histograma
centro=[7.5:15:172.5];  %centro de las cajas del histograma, entre
0+(15/2) y 180-(15/2)
angulos=[];

for k=1:length(contorno),  %se calcula tanto para contornos internos
como externos
    componente = contorno{k};

    [fils,cols]=size(componente);   %cada columna es una
coordenada(fila;columna)
    nivel=zeros(1,181);

    %Se da la vuelta a los ejes para que el origen de coordenadas de
Matlab y el del sistema cartesiano coincidan para calcular el
arcotangente
    componente(1,:)=max(componente(1,:))-componente(1,:);

    %Se alarga el componente para simular continuidad
    comp=componente;
    comp=[comp componente(:,1:5)];

    for j=1:cols
        xk=componente(2,j);    %columna
        yk=componente(1,j);    %fila

        if j>(cols-5)
            xke=comp(2,j+e);
            yke=comp(1,j+e);
        else
            xke=componente(2,j+e);
            yke=componente(1,j+e);
        end

        % calculo el angulo
        if (xke-xk)==0
            angulo(j)=90; %resultado en grados
        elseif (yke-yk)==0
            angulo(j)=0; %resultado en grados
        else
            if (xke-xk)>0
                if (yke-yk)>0   %esta en el primer cuadrante
```

```
                    angulo(j)=atand((yke-yk)/(xke-xk));  %resultado en
grados
                elseif (yke-yk)<0   %esta en el cuarto cuadrante
                    angulo(j)=atand((yke-yk)/(xke-xk))+180;
%resultado en grados
                end
            elseif (xke-xk)<0
                if (yke-yk)>0   %esta en el segundo cuadrante
                    angulo(j)=atand((yke-yk)/(xke-xk))+180;
%resultado en grados
                elseif (yke-yk)<0   %esta en el tercer cuadrante
                    angulo(j)=atand((yke-yk)/(xke-xk));  %resultado en
grados
                end
            end
        end

    end

        angulos=[angulos angulo];
        angulo=[];

end

%funcion PDF de todo el contorno
val=hist(angulos,centro);

figure
%funcionPDFtotal=val./sum(val);
funcionPDFtotal=val;
plot(centro,funcionPDFtotal,'bo-');
title('\bf Funcion PDF contour-direction');

figure
anguloradian=(angulos.*pi)./180;
rose(anguloradian,2*n); %la representacion va de 0 a 2*pi
title('\bf Representacion angular contour-direction');

anguloradianvector=anguloradian;
dim=2*n;

%anguloradian=(angulos.*pi)./180;
%rangoradian=(0+(15/2):15:360-(15/2))*pi/180;
%figure
%rose(anguloradian,rangoradian)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   [amplitudtotal]=contourhinge(contorno)
%   amplitudtotal: funcion de distribucion de probabilidad conjunta
%   contorno: contorno del componente
%   Descripcion: función que calcula característica f2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [amplitudtotal]=contourhinge(contorno)

e=5;    %controla la longitud del fragmento de contorno analizado
grados=15;  %cada intervalo del histograma abarca 15º
n=12;   %numero de intervalos del histograma
centro=[7.5:15:352.5];   %centro de las cajas del histograma, entre
0+(15/2) y 360-(15/2)

suma=zeros(360,360,length(contorno));

for k=1:length(contorno),    %se calcula tanto para contornos internos
como externos
    componente = contorno{k};

    [fils,cols]=size(componente);             %cada columna es una
coordenada(fila;columna)


%Se da la vuelta a los ejes para que el origen de coordenadas de
Matlab y el del sistema cartesiano coincidan para calcular el
arcotangente
    componente(1,:)=max(componente(1,:))-componente(1,:);

    %Se alarga el componente para simular continuidad
    comp=componente;
    comp=[componente(:,cols-4:cols) comp componente(:,1:5)];

    for j=6:cols+5
        xk=comp(2,j);    %columna
        yk=comp(1,j);    %fila

        xkemenos=comp(2,j-e);
        ykemenos=comp(1,j-e);

        xkemas=comp(2,j+e);
        ykemas=comp(1,j+e);

        %calculo un angulo
        if (xkemenos-xk)==0
            if (ykemenos-yk)>0
                angulo2(j-5)=90;    %resultado en grados
            else
                angulo2(j-5)=270;    %resultado en grados
            end
        elseif (ykemenos-yk)==0
            angulo2(j-5)=1; %resultado en grados
        else
            if (xkemenos-xk)>0
                if (ykemenos-yk)>0   %esta en el primer cuadrante
                    angulo2(j-5)=atand((ykemenos-yk)/(xkemenos-xk));
%resultado en grados
                elseif (ykemenos-yk)<0   %esta en el cuarto cuadrante
```

```
                    angulo2(j-5)=atand((ykemenos-yk)/(xkemenos-
xk))+360;  %resultado en grados
                end
            elseif (xkemenos-xk)<0
                if (ykemenos-yk)>0   %esta en el segundo cuadrante
                    angulo2(j-5)=atand((ykemenos-yk)/(xkemenos-
xk))+180;  %resultado en grados
                elseif (ykemenos-yk)<0   %esta en el tercer cuadrante
                    angulo2(j-5)=atand((ykemenos-yk)/(xkemenos-
xk))+180;  %resultado en grados
                end
            end
        end


        % calculo el otro angulo
        if (xkemas-xk)==0
            if (ykemas-yk)>0
                angulo1(j-5)=90;    %resultado en grados
            else
                angulo1(j-5)=270;    %resultado en grados
            end
        elseif (ykemas-yk)==0
            angulo1(j-5)=1; %resultado en grados
        else
            if (xkemas-xk)>0
                if (ykemas-yk)>0   %esta en el primer cuadrante
                    angulo1(j-5)=atand((ykemas-yk)/(xkemas-xk));
%resultado en grados
                elseif (ykemas-yk)<0   %esta en el cuarto cuadrante
                    angulo1(j-5)=atand((ykemas-yk)/(xkemas-xk))+360;
%resultado en grados
                end
            elseif (xkemas-xk)<0
                if (ykemas-yk)>0   %esta en el segundo cuadrante
                    angulo1(j-5)=atand((ykemas-yk)/(xkemas-xk))+180;
%resultado en grados
                elseif (ykemas-yk)<0   %esta en el tercer cuadrante
                    angulo1(j-5)=atand((ykemas-yk)/(xkemas-xk))+180;
%resultado en grados
                end
            end
        end


        %se establece como fi2 el mayor de los dos angulos
        if angulo2(j-5)<angulo1(j-5)
            fi2=angulo1(j-5);
            fi1=angulo2(j-5);
        else
            fi2=angulo2(j-5);
            fi1=angulo1(j-5);
        end

        %se calcula el histograma
        suma(round(fi1),round(fi2),k)=suma(round(fi1),round(fi2),k)+1;

    end
```

```
end

%Funcion de densidad de probabilidad del contorno
figure
for k=1:360
    for m=1:360
        vector=suma(k,m,:);
        sumadim(k,m)=sum(vector(:));
    end
end

cont1=1;
for k=1:grados:360
    cont2=1;
    for m=1:grados:360
        sumatotal(cont1,cont2)=sum(sum(sumadim(k:k+grados-
1,m:m+grados-1)));
        cont2=cont2+1;
    end
    cont1=cont1+1;
end

%amplitudtotal=sumatotal./sum(sum(sumatotal));
amplitudtotal=sumatotal;
mesh(centro,centro,amplitudtotal);
title('\bf Funcion PDF conjunta contour-hinge');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   im_sin_ruido=elim_ruido(imagenbin)
%   im_sin_ruido: imagen binaria sin ruido
%   imagenbin: imagen binaria con ruido
%   Descripcion: Elimina el ruido de la imagen por el método
%   del filtrado morfológico (apertura+cierre)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function im_sin_ruido=elim_ruido(imagenbin)

%Elemento estructurante
EE=ones(2);

%Invierte los valores de la imagen
im_inv=1-double(imagenbin);

%Apertura
a=imdilate(imerode(im_inv,EE),EE);

%Cierre
c=imerode(imdilate(a,EE),EE);

%Invierte los valores de c
im_sin_ruido=uint8(1-double(c));
figure;
imshow(255*im_sin_ruido);
title('\bf Imagen binarizada sin ruido');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   angulo=hallarangulo(fil1,col1,fil2,col2)
%   fil1: coordenada de la fila del primer pixel
%   col1: coordenada de la columna del primer pixel
%   fil2: coordenada de la fila del segundo pixel
%   col2: coordenada de la columna del segundo pixel
%   Descripcion: calcula los dos angulos correspondientes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function angulo=hallarangulo(fil1,col1,fil2,col2)

if (col2-col1)==0
    if (fil2-fil1)>0
        angulo=90;    %resultado en grados
    else
        angulo=270;    %resultado en grados
    end
elseif (fil2-fil1)==0
    angulo=1; %resultado en grados
else
    if (col2-col1)>0
        if (fil2-fil1)>0   %esta en el primer cuadrante
            angulo=atand((fil2-fil1)/(col2-col1));  %resultado en
grados
        elseif (fil2-fil1)<0   %esta en el cuarto cuadrante
            angulo=atand((fil2-fil1)/(col2-col1))+360;  %resultado en
grados
        end
    elseif (col2-col1)<0
        if (fil2-fil1)>0   %esta en el segundo cuadrante
            angulo=atand((fil2-fil1)/(col2-col1))+180;  %resultado en
grados
        elseif (fil2-fil1)<0   %esta en el tercer cuadrante
            angulo=atand((fil2-fil1)/(col2-col1))+180;  %resultado en
grados
        end
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script que calcula el numero medio de lineas del total
%   de las muestras de la base de datos
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc
close all
clear all

%Se halla el numero de escritores diferentes que hay
[texto, escritor, a, b, c, d, e, f]=textread('formstxt.txt','%s %d %d
%s %d %d %d %d');

carpeta=['CarpetaModelos'];
carpeta2=['./CarpetaModelos/'];
carpeta3=['Modelo'];
contenido=dir('./CarpetaModelos/');

%Se agrupan y normalizan todas las partes de una caracteristica de un
mismo usuario
nombre=[];
nombre_usuario=[];
lineas=[];
total_usuarios=0;

i=3;
while i<=length(contenido)
    saltar=0;
    nombre=contenido(i).name;
    guion=0;
    %Se obtiene el nombre del usuario
    for m=1:length(nombre)
        if guion==0 & nombre(m)=='-'
            guion=1;
        elseif guion==1 & nombre(m)=='-'
            nombre_usuario=nombre(1:m-1);
        end
    end

    %Se buscan todas las lineas del texto del usuario
    nom=[];
    numero=0;
    guion_c=0;
    for m=i:length(contenido)
        nom=contenido(m).name;
        %Se obtiene el nombre del usuario para comparar
        for z=1:length(nom)
            if guion_c==0 & nom(z)=='-'
                guion_c=1;
            elseif guion_c==1 & nom(z)=='-'
                nombre_comparar=nom(1:z-1);
            end
        end
        if length(nombre_usuario)==length(nombre_comparar)
            if nombre_usuario==nombre_comparar
                numero=numero+1;
            end
        end
    end
```

```
    nom=[];
    nombre_comparar=[];

    %Se busca el numero que identifica al escritor
    longitud=length(carpeta3);
    nombre_us=nombre_usuario(longitud+1:end);
    for p=1:length(texto)
        nombre_texto=texto{p};
        if length(nombre_texto)==length(nombre_us)
            if nombre_texto==nombre_us
                usuario=escritor(p);
            end
        end
    end

    %Se almacena en un vector el numero de lineas de cada usuario
    lineas=[lineas numero];
    total_usuarios=total_usuarios+1;
    nombre=[];
    usuario=[];
    nombre_usuario=[];
    i=i+saltar+1;
end %while

media=sum(lineas)/total_usuarios;
%RESULTADO-->4.133
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script principal que llama a las funciones que llevan a
%   a cabo el preprocesado y calculan las PDFs y autocorrelación.
%   También almacena los resultados obtenidos.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc;
close all;
clear all;

for w=1:2

    if w==1
        imagenesbd=dir('CarpetaModelos');
        carpeta=['./CarpetaModelos/'];
    else
        imagenesbd=[];
        imagenesbd=dir('CarpetaTest');
        carpeta=['./CarpetaTest/'];
    end
    longitud_bucle=length(imagenesbd);

    for ind=3:longitud_bucle    %se lee a partir del 3 pq lo anterior
son otras cosas

        nombre=[carpeta imagenesbd(ind).name];
        imagen=imread(nombre);
        imshow(imagen);
        title('\bf Imagen de la linea para procesar');

        %Binarizacion de la imagen segun el metodo de Otsu
        level=graythresh(imagen);
        imagenbin=im2bw(imagen,level);
        figure;
        imshow(imagenbin);
        title('\bf Imagen binarizada segun el metodo de Otsu');

        %Eliminacion del ruido de la imagen mediante la apertura y el
cierre
        imagenbin_sin_ruido=elim_ruido(imagenbin);

        %Deteccion de componentes 8-connectivity (labeling connected
components)
        [cellarraybordes,L,N]                                          =
bwboundaries(not(imagenbin_sin_ruido),8);
        figure;
         imshow(255*imagenbin_sin_ruido);
         title('\bf Imagen 8-connectivity');
         hold on;
         for k=1:length(cellarraybordes),
            boundary = cellarraybordes{k};
            if(k > N)
              plot(boundary(:,2), boundary(:,1), 'g');    %por defecto
las lineas son de 0.5 puntos
            else
              plot(boundary(:,2), boundary(:,1), 'r');    %por defecto
las lineas son de 0.5 puntos
            end
         end
```

```
        %Extraccion de contornos mediante Moore's contour-following
algorithm
        cont=1;
        contorno=cell(size(cellarraybordes));
        for k=1:length(cellarraybordes),
            B=[];   %Guarda los pixeles que forman el contorno de cada
componente
            componente=cellarraybordes{k};
            [fils,cols]=size(componente);

            %construccion de la matriz cuadrada con el componente
            %y representa las filas
            %x representa las columnas
            minimoy=min(componente(:,1));
            maximoy=max(componente(:,1));
            dify=maximoy-minimoy;
            minimox=min(componente(:,2));
            maximox=max(componente(:,2));
            difx=maximox-minimox;
            if dify==difx
                dim=dify+1+4;
                matriz=zeros(dim,dim);
                for n=1:fils
                    y=componente(n,1)-minimoy+3;
                    x=componente(n,2)-minimox+3;
                    matriz(y,x)=1;
                end
            elseif dify<difx
                dim=difx+1+4;
                matriz=zeros(dim,dim);
                for n=1:fils
                    y=componente(n,1)-minimoy+3;
                    x=componente(n,2)-minimox+3;
                    matriz(y,x)=1;
                end
            elseif dify>difx
                dim=dify+1+4;
                matriz=zeros(dim,dim);
                for n=1:fils
                    y=componente(n,1)-minimoy+3;
                    x=componente(n,2)-minimox+3;
                    matriz(y,x)=1;
                end
            end

            matrizsuma=sum(sum(matriz));
            if matrizsuma > 35 & difx>0 & dify>0     %para evitar
componentes muy pequeños cuyos contornos no interesan
                %Algoritmo de Moore
                Bnueva=algoritmoMoore(dim,matriz);
                longitud=length(Bnueva);

                for i=1:longitud
                    comp_y=Bnueva(1,i)+minimoy-3;    %coordenada de la
fila
                    comp_x=Bnueva(2,i)+minimox-3;    %coordenada de la
columna
                    b=[comp_y;comp_x];
```

```
                    B=[B b];    %array de dos filas tantas columnas
como puntos el contorno
                end

                %Cell array con el contorno de cada componente
                contorno(cont)={B};
                cont=cont+1;
            end
        end

        contador=cont-1;
        contornofin=cell(contador,1);
        for k=1:contador
            aux=contorno{k};
            contornofin(k)={aux};
        end

        %Se almacena el contorno de la imagen
        save(['./CarpetaContorno/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Contorno.mat'],'contornofin');

        %Imagen con los contornos
         figure
         imshow(255*imagenbin_sin_ruido);
         title('\bf Imagen con los contornos');
         hold on;
         for k=1:length(contornofin),
             region = contornofin{k};
             plot(region(2,:), region(1,:), 'b');
         end

        %Calculo de la funcion PDF y la inclinacion (orientacion) de
la escritura

[funcionPDFtotal,anguloradianvector,dim]=contourdirection(contornofin)
;
        save(['./CarpetaFuncionPDF/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'FuncionPDF.mat'],'funcionPDFtotal');
        save(['./CarpetaInclinacion/' carpeta(3:end)
imagenesbd(ind).name(1:end-4)
'Inclinacion.mat'],'anguloradianvector');

        %Calculo de la funcion de PDF conjunta y la curvatura de la
escritura
        [amplitudtotal]=contourhinge(contornofin);
        save(['./CarpetaCurvatura/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Curvatura.mat'],'amplitudtotal');

        %Calculo de la redondez
        [rverticaltotal,rhorizontaltotal]=redondez(contornofin);
        save(['./CarpetaRedondez/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Rvertical.mat'],'rverticaltotal');
        save(['./CarpetaRedondez/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Rhorizontal.mat'],'rhorizontaltotal');

        %Calculo del espacio vertical y horizontal
        [lverticalPDF,lhorizontalPDF]=runlength(imagenbin_sin_ruido);
        save(['./CarpetaRunlength/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Lvertical.mat'],'lverticalPDF');
```

```
        save(['./CarpetaRunlength/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Lhorizontal.mat'],'lhorizontalPDF');

        %Calculo de la autocorrelacion
        autocor=autocorrelacion(imagen);
        save(['./CarpetaAutocorrelacion/' carpeta(3:end)
imagenesbd(ind).name(1:end-4) 'Autocorrelacion.mat'],'autocor');


    end
end

close all
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   [rverticaltotal,rhorizontaltotal]=redondez(contorno)
%   contorno: contorno del componente
%   rverticaltotal: valores de la funcion de distribucion de
%   probabilidad para la exploracion vertival
%   rhorizontaltotal: valores de la funcion de distribucion de
%   probabilidad para la exploracion horizontal
%   Descripcion: funcion que calcula las caracteristicas f3h y f3v
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [rverticaltotal,rhorizontaltotal]=redondez(contorno)

e=5;    %controla la longitud del fragmento de contorno analizado
grados=15;  %cada intervalo del histograma abarca 15º
n=12;   %numero de intervalos del histograma
centro=[7.5:15:172.5];  %centro de las cajas del histograma, entre
0+(15/2) y 180-(15/2)

for k=1:length(contorno),
    componente = contorno{k};
    [fils,cols]=size(componente);     %cada columna es una
coordenada(fila;columna)

    %Construccion de la matriz-imagen con el componente
    maximoy=max(componente(1,:));   %fila
    maximox=max(componente(2,:));   %columna
    if maximoy<maximox
        maxim=maximox;
    else
        maxim=maximoy;
    end
    matriz=zeros(maxim,maxim);
    for n=1:cols
        matriz(componente(1,n),componente(2,n))=1;
    end


    %Se alarga el componente para simular continuidad
    comp=componente;
    comp=[componente(:,cols-4:cols) comp componente(:,1:5)];


   %Para hallar la caracteristica vertical
   angulov=[];
   nov=0;   %variable que indica si se descartan o no los angulos
   sumav=zeros(180,180);
    for n=1:maxim    %se recorren las columnas de la matriz-imagen
        primero=1;
        for m=1:maxim    %se recorren las filas de la matriz-imagen
            if matriz(m,n)==1
                if primero==1
                    coord1_fils=m;
                    coord1_cols=n;
                    primero=2;
                    num1=find(componente(1,:)==m &
componente(2,:)==n);
                    num1=num1(1)+5;    %porque comp empieza 5 numeros
antes que componente
                elseif primero==2
                    coord2_fils=m;
```

```
                    coord2_cols=n;
                    primero=1;
                    num2=find(componente(1,:)==m &
componente(2,:)==n);
                    num2=num2(1)+5;    %porque comp empieza 5 numeros
antes que componente
                    %Se calcula el angulo de las primeras coordenadas
                    xkemenos1=comp(2,num1-e);
                    ykemenos1=comp(1,num1-e);
                    xkemas1=comp(2,num1+e);
                    ykemas1=comp(1,num1+e);
                    %Se da la vuelta a los ejes para que el origen de
coordenadas de Matlab y el del sistema cartesiano coincidan para
calcular el arcotangente
                    coord1_fils=max(componente(1,:))-coord1_fils;
                    ykemenos1=max(componente(1,:))-ykemenos1;
                    ykemas1=max(componente(1,:))-ykemas1;

angulomenos1=hallarangulo(coord1_fils,coord1_cols,ykemenos1,xkemenos1)
;

angulomas1=hallarangulo(coord1_fils,coord1_cols,ykemas1,xkemas1);
                    if angulomenos1>180 & angulomas1>180
                        nov=1;   %se descarta
                    elseif angulomenos1<180 & angulomas1<180
                        nov=1;   %se descarta
                    elseif angulomenos1<180
                        angulo1=angulomenos1;
                    else
                        angulo1=angulomas1;
                    end
                    %Se calcula el angulo de las segundas coordenadas
                    xkemenos2=comp(2,num2-e);
                    ykemenos2=comp(1,num2-e);
                    xkemas2=comp(2,num2+e);
                    ykemas2=comp(1,num2+e);
                    %Se da la vuelta a los ejes para que el origen de
coordenadas de Matlab y el del sistema cartesiano coincidan para
calcular el arcotangente
                    coord2_fils=max(componente(1,:))-coord2_fils;
                    ykemenos2=max(componente(1,:))-ykemenos2;
                    ykemas2=max(componente(1,:))-ykemas2;

angulomenos2=hallarangulo(coord2_fils,coord2_cols,ykemenos2,xkemenos2)
;

angulomas2=hallarangulo(coord2_fils,coord2_cols,ykemas2,xkemas2);
                    if angulomenos2>180 & angulomas2>180
                        nov=1;   %se descarta
                    elseif angulomenos2<180 & angulomas2<180
                        nov=1;   %se descarta
                    elseif angulomenos2<180
                        angulo2=angulomenos2;
                    else
                        angulo2=angulomas2;
                    end

                    if nov==0
                        ang=[angulo1;angulo2];
```

```
                        angulov=[angulov ang];
                        %Se calcula el histograma

sumav(round(angulo1),round(angulo2))=sumav(round(angulo1),round(angulo
2))+1;
                    end
                    nov=0;
                end
            end
        end
    end
    %PDF vertical
%     norm=sum(sum(sumav));
%     if norm==0
%         rvertical(:,:,k)=0*sumav;
%         'cero rvertical'
%     else
%         rvertical(:,:,k)=sumav./norm;
%     end
    rvertical(:,:,k)=sumav;

    %Para hallar la caracteristica horizontal
   anguloh=[];
   noh=0;   %variable que indica si se descartan o no los angulos
   sumah=zeros(180,180);
    for m=1:maxim    %se recorren las filas de la matriz-imagen
        primero=1;
        for n=1:maxim    %se recorren las columnas de la matriz-imagen
            if matriz(m,n)==1
                if primero==1
                    coord1_fils=m;
                    coord1_cols=n;
                    primero=2;
                    num1=find(componente(1,:)==m &
componente(2,:)==n);
                    num1=num1(1)+5;    %porque comp empieza 5 numeros
antes que componente
                elseif primero==2
                    coord2_fils=m;
                    coord2_cols=n;
                    primero=1;
                    num2=find(componente(1,:)==m &
componente(2,:)==n);
                    num2=num2(1)+5;    %porque comp empieza 5 numeros
antes que componente
                    %Se calcula el angulo de las primeras coordenadas
                    xkemenos1=comp(2,num1-e);
                    ykemenos1=comp(1,num1-e);
                    xkemas1=comp(2,num1+e);
                    ykemas1=comp(1,num1+e);
                    %Se da la vuelta a los ejes para que el origen de
coordenadas de Matlab y el del sistema cartesiano coincidan para
calcular el arcotangente
                    coord1_fils=max(componente(1,:))-coord1_fils;
                    ykemenos1=max(componente(1,:))-ykemenos1;
                    ykemas1=max(componente(1,:))-ykemas1;

angulomenos1=hallarangulo(coord1_fils,coord1_cols,ykemenos1,xkemenos1)
;
```

```
angulomas1=hallarangulo(coord1_fils,coord1_cols,ykemas1,xkemas1);
                    if angulomenos1>180 & angulomas1>180
                        noh=1;   %se descarta
                    elseif angulomenos1<180 & angulomas1<180
                        noh=1;   %se descarta
                    elseif angulomenos1<180
                        angulo1=angulomenos1;
                    else
                        angulo1=angulomas1;
                    end
                    %Se calcula el angulo de las segundas coordenadas
                    xkemenos2=comp(2,num2-e);
                    ykemenos2=comp(1,num2-e);
                    xkemas2=comp(2,num2+e);
                    ykemas2=comp(1,num2+e);
                    %Se da la vuelta a los ejes para que el origen de
coordenadas de Matlab y el del sistema cartesiano coincidan para
calcular el arcotangente
                    coord2_fils=max(componente(1,:))-coord2_fils;
                    ykemenos2=max(componente(1,:))-ykemenos2;
                    ykemas2=max(componente(1,:))-ykemas2;

angulomenos2=hallarangulo(coord2_fils,coord2_cols,ykemenos2,xkemenos2)
;

angulomas2=hallarangulo(coord2_fils,coord2_cols,ykemas2,xkemas2);
                    if angulomenos2>180 & angulomas2>180
                        noh=1;   %se descarta
                    elseif angulomenos2<180 & angulomas2<180
                        noh=1;   %se descarta
                    elseif angulomenos2<180
                        angulo2=angulomenos2;
                    else
                        angulo2=angulomas2;
                    end

                    if noh==0
                        ang=[angulo1;angulo2];
                        anguloh=[anguloh ang];
                        %Se calcula el histograma
sumah(round(angulo1),round(angulo2))=sumah(round(angulo1),round(angulo
2))+1;
                    end
                    noh=0;
                end
            end
        end
    end
    %PDF horizontal
%     norm=sum(sum(sumah));
%     if norm==0
%         rhorizontal(:,:,k)=0*sumah;
%         'cero rhorizontal'
%     else
%         rhorizontal(:,:,k)=sumah./norm;
%     end
    rhorizontal(:,:,k)=sumah;

end
```

```
%PDFs vertical y horizontal de todo el contorno
for k1=1:180,
    for k2=1:180
         rverticaltot(k1,k2)=sum(rvertical(k1,k2,:));
         rhorizontaltot(k1,k2)=sum(rhorizontal(k1,k2,:));
    end
end

cont1=1;
for k=1:grados:180
    cont2=1;
    for m=1:grados:180
        rverticaltotal(cont1,cont2)=sum(sum(rverticaltot(k:k+grados-
1,m:m+grados-1)));

rhorizontaltotal(cont1,cont2)=sum(sum(rhorizontaltot(k:k+grados-
1,m:m+grados-1)));
        cont2=cont2+1;
    end
    cont1=cont1+1;
end

figure
mesh(centro,centro,rverticaltotal);
title('Caracteristica de redondez vertical');
figure
mesh(centro,centro,rhorizontaltotal);
title('Caracteristica de redondez horizontal');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   imagenbin: imagen binaria sin ruido
%   lverticalPDF: funcion de distribucion de probabilidad de los
%   recorridos verticales
%   lhorizontalPDF: funcion de distribucion de probabilidad de los
%   recorridos horizontales
%   Descripcion: funcion que calcula las caracteristicas f5h y f5v
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [lverticalPDF,lhorizontalPDF]=runlength(imagenbin)

[filas,columnas]=size(imagenbin);%Las letras de imagenbin son negras
(0) y el fondo es blanco (1)

%Se recorre la imagen binaria por filas para hallar la longitud
horizontal
anterior=2;
longitudh=0;
lhorizontal=[];
for n=1:filas
    for m=1:columnas
        if imagenbin(n,m)==1 & anterior==0
            longitudh=longitudh+1;
        elseif imagenbin(n,m)==0 & anterior==1 & longitudh~=0
            if longitudh<=60
                lhorizontal=[lhorizontal longitudh];
            end
            longitudh=0;
        end

        if longitudh~=0
            longitudh=longitudh+1;
        end

        anterior=imagenbin(n,m);
    end
end


%Se recorre la imagen binaria por columnas para hallar la longitud
vertical
anterior=2;
longitudv=0;
lvertical=[];
for m=1:columnas
    for n=1:filas
        if imagenbin(n,m)==1 & anterior==0
            longitudv=longitudv+1;
        elseif imagenbin(n,m)==0 & anterior==1 & longitudv~=0
            if longitudv<=60
                lvertical=[lvertical longitudv];
            end
            longitudv=0;
        end

        if longitudv~=0
            longitudv=longitudv+1;
        end
```

```
        anterior=imagenbin(n,m);
    end
end

%PDFs horizontal y vertical
numelemh=numel(lhorizontal);
numelemv=numel(lvertical);

ordenh=sort(lhorizontal);
ordenv=sort(lvertical);

nivelh=zeros(1,59);
contador=1;
for i=2:numelemh
    if ordenh(i-1)==ordenh(i)
        contador=contador+1;
    else
        nivelh(ordenh(i-1))=contador;
        contador=1;
    end
end
%numh=sum(nivelh);
%lhorizontalPDF=nivelh./numh;
lhorizontalPDF=nivelh;
figure
plot(lhorizontalPDF);
title('PDF horizontal');
xlabel('Separacion horizontal');
ylabel('Probabilidad');

nivelv=zeros(1,59);
contador=1;
for i=2:numelemv
    if ordenv(i-1)==ordenv(i)
        contador=contador+1;
    else
        nivelv(ordenv(i-1))=contador;
        contador=1;
    end
end
%numv=sum(nivelv);
%lverticalPDF=nivelv./numv;
lverticalPDF=nivelv;
figure
plot(lverticalPDF);
title('PDF vertical');
xlabel('Separacion vertical');
ylabel('Probabilidad');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script que selecciona las dos primeras muestras de escritura
%   de cada escritor que aparecen en la base de datos y en caso
%   de escritores con una unica muestra la divide en dos
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc;
close all;
clear all;

%Para cada texto se guarda su escritor (entre otras cosas)
[texto, escritor, a, b, c, d, e, f]=textread('formstxt.txt','%s %d %d
%s %d %d %d %d');

writer=zeros(1,max(escritor)+1);

%Para detectar a los escritores con un unico form
escritores_ord=sort(escritor);
unicos=[];
for k=2:length(escritores_ord)
    if escritores_ord(k-1)~=escritores_ord(k)
        if escritores_ord(k)~=escritores_ord(k+1)
            unicos=[unicos escritores_ord(k)];
        end
    end
end

%Contenido de la base de datos segmentada en filas y ordenada
carpeta=dir('../lines');    %Contiene las carpetas a01,c03,...
for k=3:length(carpeta)
    nombre=carpeta(k).name;
    subcarpeta=dir(['../lines/'  nombre]);     %Contiene  varios  textos
como p.ej. los de a01

    for m=3:length(subcarpeta)
        nombresubcarpeta=subcarpeta(m).name;

        for n=1:length(texto)
            nombre_texto=texto{n};
            %Si no tienen igual longitud no son el mismo texto
            if length(nombre_texto)==length(nombresubcarpeta)
                if nombre_texto==nombresubcarpeta
                    unic=0;
                    %Primer form del escritor
                    if writer(escritor(n)+1)==0
                        frases=dir(['../lines/'       nombre       '/'
nombresubcarpeta '/*-*.png']); %Contiene las imagenes de las lineas de
un texto
                        %Si el escritor solo ha escrito un form
                        for i=1:length(unicos)
                            if escritor(n)==unicos(i)
                                unic=1;
                            end
                        end
                        for w=1:length(frases)
                            nombreimagen=['../lines/'    nombre    '/'
nombresubcarpeta '/' frases(w).name];
                            imagen=imread(nombreimagen);
                            if unic==0
```

```
imwrite(imagen,['./CarpetaModelos/Modelo' frases(w).name]);
                            else    %Para los escritores con un unico
form
                                if w<=round(length(frases)/2)

imwrite(imagen,['./CarpetaModelos/Modelo' frases(w).name]);
                                else

imwrite(imagen,['./CarpetaTest/Test' frases(w).name]);
                                end
                            end
                        end
                        writer(escritor(n)+1)=1;

                        %Segundo form del escritor
                    elseif writer(escritor(n)+1)==1
                        frases=dir(['../lines/' nombre '/'
nombresubcarpeta '/*-*.png']);
                        for w=1:length(frases)
                            nombreimagen=['../lines/' nombre '/'
nombresubcarpeta '/' frases(w).name];
                            imagen=imread(nombreimagen);
                            imwrite(imagen,['./CarpetaTest/Test'
frases(w).name]);
                        end
                        writer(escritor(n)+1)=2;
                    end

                end
            end
        end
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[sig_x,sig_y]=siguientenormal(c_x,c_y,ant_x,ant_y,preant_x,preant_y,h)
%   c_x: coordenada x del pixel actual
%   c_y: coordenada y del pixel actual
%   ant_x: coordenada x del pixel anterior
%   ant_y: coordenada y del pixel anterior
%   preant_x: coordenada x del pixel anterior al anterior
%   preant_y: coordenada y del pixel anterior al anterior
%   sig_x: coordenada x del pixel siguiente
%   sig_y: coordenada y del pixel siguiente
%   Descripcion: calcula la posicion siguiente del algoritmo de Moore
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function
[sig_x,sig_y]=siguientenormal(c_x,c_y,ant_x,ant_y,preant_x,preant_y,h)


%x representa las filas
%y representa las columnas

if h==2 %el pixel preant era un 1
    if c_x==ant_x   %si estan en la misma fila
        if c_y > ant_y
            sig_x=c_x+1;
            sig_y=c_y;
        elseif c_y < ant_y
            sig_x=c_x-1;
            sig_y=c_y;
        end
    elseif c_y==ant_y   %si estan en la misma columna
        if c_x > ant_x
            sig_x=c_x;
            sig_y=c_y-1;
        elseif c_x < ant_x
            sig_x=c_x;
            sig_y=c_y+1;
        end
    end
else    %el pixel preant era un 0
    if ant_y==preant_y
        if c_y==ant_y   %si estan los tres ultimos pixeles recorridos
en la misma columna
            if c_x > ant_x
                sig_x=c_x;
                sig_y=c_y-1;
            elseif c_x < ant_x
                sig_x=c_x;
                sig_y=c_y+1;
            end
        elseif c_y > ant_y
            sig_x=c_x;
            sig_y=c_y+1;
        elseif c_y < ant_y
            sig_x=c_x;
            sig_y=c_y-1;
        end
    elseif ant_x==preant_x
        if c_x==ant_x  %si estan los tres ultimos pixeles recorridos
en la misma fila
```

```
            if c_y > ant_y
                sig_x=c_x+1;
                sig_y=c_y;
            elseif c_y < ant_y
                sig_x=c_x-1;
                sig_y=c_y;
            end
        elseif c_x > ant_x
            sig_x=c_x+1;
            sig_y=c_y;
        elseif c_x < ant_x
            sig_x=c_x-1;
            sig_y=c_y;
        end
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   [sig_x,sig_y]=siguientepartida(p_x,p_y,ant_x,ant_y)
%   p_x: coordenada x del pixel actual
%   p_y: coordenada y del pixel actual
%   ant_x: coordenada x del pixel anterior
%   ant_y: coordenada y del pixel anterior
%   sig_x: coordenada x del pixel siguiente
%   sig_y: coordenada y del pixel siguiente
%   Descripcion: funcion que calcula el siguiente movimiento
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [sig_x,sig_y]=siguientepartida(p_x,p_y,ant_x,ant_y)

%x representa las filas
%y representa las columnas

if p_x==ant_x   %si estan en la misma fila
    if p_y > ant_y  %si se ha retrocedido una columna
        sig_x=ant_x-1;
        sig_y=ant_y;
    elseif p_y < ant_y  %si se ha avanzado una columna
        sig_x=ant_x+1;
        sig_y=ant_y;
    end
elseif p_y==ant_y   %si estan en la misma columna
    if p_x > ant_x  %si se ha retrocedido una fila
        sig_x=ant_x;
        sig_y=ant_y+1;
    elseif p_x < ant_x  %si se ha avanzado una fila
        sig_x=ant_x;
        sig_y=ant_y-1;
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script que crea dos carpetas por cada usuario, una de
%   entrenamiento y otra de test, y almacena en cada carpeta las
%   caracteristicas de cada escritor
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc
close all
clear all

%Se halla el numero de escritores diferentes que hay
[texto, escritor, a, b, c, d, e, f]=textread('formstxt.txt','%s %d %d
%s %d %d %d %d');
num_escritores=max(escritor)+1;

%Para cada escritor se tiene una carpeta con todas sus caracteristicas
 for p=1:num_escritores
     mkdir(['./UsuariosEntrenamiento/usuario' num2str(p-1)]);
     mkdir(['./UsuariosTest/usuario' num2str(p-1)]);
 end

for w=1:2
    if w==1
        carpeta=['CarpetaModelos'];
        carpeta2=['UsuariosEntrenamiento'];
        carpeta3=['Modelo'];
    else
        carpeta=['CarpetaTest'];
        carpeta2=['UsuariosTest'];
        carpeta3=['Test'];
    end

    %Se selecciona la caracteristica
    for n=1:5
        if n==1
            contenido=dir(['./CarpetaFuncionPDF/' carpeta]);
            subcarpeta=['CarpetaFuncionPDF'];
        elseif n==2
            contenido=dir(['./CarpetaCurvatura/' carpeta]);
            subcarpeta=['CarpetaCurvatura'];
        elseif n==3
            contenido=dir(['./CarpetaRedondez/' carpeta]);
            subcarpeta=['CarpetaRedondez'];
        elseif n==4
            contenido=dir(['./CarpetaRunlength/' carpeta]);
            subcarpeta=['CarpetaRunlength'];
        else
            contenido=dir(['./CarpetaAutocorrelacion/' carpeta]);
            subcarpeta=['CarpetaAutocorrelacion'];
        end

        %Se agrupan y normalizan todas las partes de una
caracteristica de un mismo usuario
        nombre=[];
        nombre_usuario=[];
        nombre1=[]; %se usa para n==3 y n==4 pq tienen componentes
horizontales y verticales

        i=3;
```

```
        while i<=length(contenido)
            saltar=0;
            nombre=contenido(i).name;
            guion=0;
            %Se obtiene el nombre del usuario
            for m=1:length(nombre)
                if guion==0 & nombre(m)=='-'
                    guion=1;
                elseif guion==1 & nombre(m)=='-'
                    nombre_usuario=nombre(1:m-1);
                    nombre1=nombre(m+3:end);
                end
            end
            %Se buscan todas las partes de una caracteristica del
usuario
            total=0;
            totalhorizontal=0;
            totalvertical=0;
            nom=[];
            nombre2=[]; %se usa para n==3 y n==4 pq tienen componentes
horizontales y verticales
            numero=0;
            guion_c=0;
            for m=i:length(contenido)
                nom=contenido(m).name;
                %Se obtiene el nombre del usuario para comparar
                for z=1:length(nom)
                    if guion_c==0 & nom(z)=='-'
                        guion_c=1;
                    elseif guion_c==1 & nom(z)=='-'
                        nombre_comparar=nom(1:z-1);
                        nombre2=nom(z+3:end);
                    end
                end
                if length(nombre_usuario)==length(nombre_comparar)
                    if nombre_usuario==nombre_comparar
                        caracteristica=load(['./' subcarpeta '/'
carpeta '/' nom]); %es una estructura
                        caracteristica=struct2cell(caracteristica);
                        if n==3 || n==4
                            if length(nombre1)==length(nombre2)
                                if length(nombre1)==15

totalhorizontal=totalhorizontal+caracteristica{1};
                                    numero=numero+1;
                                else

totalvertical=totalvertical+caracteristica{1};
                                    numero=numero+1;
                                end
                            end
                        elseif n==5

total=total+caracteristica{1}.autocorrelacion;
                            nfils=nfils+caracteristica{1}.nfils;
                            numero=numero+1;
                        else
                            total=total+caracteristica{1};
```

```
                            numero=numero+1;
                        end
                    end
                end
                nom=[];
                nombre_comparar=[];
                nombre2=[];  %se  usa  para  n==3  y  n==4  pq  tienen
componentes horizontales y verticales
            end
            %Se busca el numero que identifica al escritor
            longitud=length(carpeta3);
            nombre_us=nombre_usuario(longitud+1:end);
            for p=1:length(texto)
                nombre_texto=texto{p};
                if length(nombre_texto)==length(nombre_us)
                    if nombre_texto==nombre_us
                        usuario=escritor(p);
                    end
                end
            end
            %Se almacena en la carpeta de usuario correspondiente
            if n==3 || n==4
                if length(nombre1)==15
                    c_totalh=totalhorizontal/numero;
                    save(['./'  carpeta2  '/usuario'  num2str(usuario)
'/' subcarpeta(8:end) 'Horizontal.mat'],'c_totalh');
                else
                    saltar=(2*numero)-2;
                    c_totalv=totalvertical/numero;
                    save(['./'  carpeta2  '/usuario'  num2str(usuario)
'/' subcarpeta(8:end) 'Vertical.mat'],'c_totalv');
                end
            elseif n==5
                saltar=numero-1;
                c_total=total/nfils;
                save(['./'  carpeta2  '/usuario'  num2str(usuario)  '/'
subcarpeta(8:end) '.mat'],'c_total');
            else
                saltar=numero-1;
                c_total=total/numero;
                save(['./'  carpeta2  '/usuario'  num2str(usuario)  '/'
subcarpeta(8:end) '.mat'],'c_total');
            end
            nombre=[];
            usuario=[];
            nombre_usuario=[];
            nombre1=[]; %se usa para n==3 y n==4 pq tienen componentes
horizontales y verticales
            i=i+saltar+1;
        end %while
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Script que varia el numero de lineas que forman las muestras
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc
close all
clear all

%Se selecciona el numero de lineas
for h=1:10
    lineas=h;
    %Se halla el numero de escritores diferentes que hay
    [texto, escritor, a, b, c, d, e, f]=textread('formstxt.txt','%s %d
%d %s %d %d %d %d');
    num_escritores=max(escritor)+1;

    %Para cada escritor se tiene una carpeta con todas sus
caracteristicas
    for p=1:num_escritores
        mkdir(['./UsuariosEntrenamientoLinea'          num2str(lineas)
'/usuario' num2str(p-1)]);
        mkdir(['./UsuariosTestLinea'     num2str(lineas)     '/usuario'
num2str(p-1)]);
    end
    for w=1:2
        if w==1
            carpeta=['CarpetaModelos'];
            carpeta2=['UsuariosEntrenamientoLinea' num2str(lineas)];
            carpeta3=['Modelo'];
        else
            carpeta=['CarpetaTest'];
            carpeta2=['UsuariosTestLinea' num2str(lineas)];
            carpeta3=['Test'];
        end
        %Se selecciona la caracteristica
        for n=1:5
            if n==1
                contenido=dir(['./CarpetaFuncionPDF/' carpeta]);
                subcarpeta=['CarpetaFuncionPDF'];
            elseif n==2
                contenido=dir(['./CarpetaCurvatura/' carpeta]);
                subcarpeta=['CarpetaCurvatura'];
            elseif n==3
                contenido=dir(['./CarpetaRedondez/' carpeta]);
                subcarpeta=['CarpetaRedondez'];
            elseif n==4
                contenido=dir(['./CarpetaRunlength/' carpeta]);
                subcarpeta=['CarpetaRunlength'];
            else
                contenido=dir(['./CarpetaAutocorrelacion/' carpeta]);
                subcarpeta=['CarpetaAutocorrelacion'];
            end
            %Se  agrupan  y  normalizan  todas  las  partes  de  una
caracteristica de un mismo usuario
            nombre=[];
            nombre_usuario=[];
            nombre1=[]; %se usa para n==3 y n==4 pq tienen componentes
horizontales y verticales
            i=3;
            while i<=length(contenido)
```

```
                saltar=0;
                nombre=contenido(i).name;
                guion=0;
                %Se obtiene el nombre del usuario
                for m=1:length(nombre)
                    if guion==0 & nombre(m)=='-'
                        guion=1;
                    elseif guion==1 & nombre(m)=='-'
                        nombre_usuario=nombre(1:m-1);
                        nombre1=nombre(m+3:end);
                    end
                end
                %Se busca el numero que identifica al escritor
                longitud=length(carpeta3);
                nombre_us=nombre_usuario(longitud+1:end);
                for p=1:length(texto)
                    nombre_texto=texto{p};
                    if length(nombre_texto)==length(nombre_us)
                        if nombre_texto==nombre_us
                            usuario=escritor(p);
                        end
                    end
                end

                %Se buscan todas las partes de una caracteristica del
usuario
                total=0;
                totalhorizontal=0;
                totalvertical=0;
                nom=[];
                nombre2=[];  %se  usa  para  n==3  y  n==4  pq  tienen
componentes horizontales y verticales
                numero=0;
                numero_iguales=0;
                guion_c=0;
                nfils=0; %PARA PROMEDIAR LA AUTOCORRELACION

                for m=i:length(contenido)
                    nom=contenido(m).name;
                    %Se obtiene el nombre del usuario para comparar
                    for z=1:length(nom)
                        if guion_c==0 & nom(z)=='-'
                            guion_c=1;
                        elseif guion_c==1 & nom(z)=='-'
                            nombre_comparar=nom(1:z-1);
                            nombre2=nom(z+3:end);
                        end
                    end
                    if length(nombre_usuario)==length(nombre_comparar)
                        if nombre_usuario==nombre_comparar
                            numero_iguales=numero_iguales+1;
                            if numero<lineas    %Controla el numero de
lineas que se cogen
                                caracteristica=load(['./'    subcarpeta
'/' carpeta '/' nom]); %es una estructura

caracteristica=struct2cell(caracteristica);
                                if n==3 || n==4
```

```
                                    if
length(nombre1)==length(nombre2)
                                        if length(nombre1)==15

totalhorizontal=totalhorizontal+caracteristica{1};
                                            numero=numero+1;
                                        else

totalvertical=totalvertical+caracteristica{1};
                                            numero=numero+1;
                                        end
                                    end
                                elseif n==5

total=total+caracteristica{1}.autocorrelacion;

nfils=nfils+caracteristica{1}.nfils;
                                    numero=numero+1;
                                elseif
                                    total=total+caracteristica{1};
                                    numero=numero+1;
                                end
                                guardar=caracteristica{1};
                                save(['./'     carpeta2     '/usuario'
num2str(usuario) '/' nom],'guardar');
                                guardar=[];
                            end %if que controla el numero de lineas
que se cogen
                        end
                    end
                    nom=[];
                    nombre_comparar=[];
                    nombre2=[]; %se usa para n==3 y n==4 pq tienen
componentes horizontales y verticales
                end  %bucle for interior

                %Se comprueba si el numero de lineas iguales es igual
al requerido
                if numero<lineas
                    rmdir(['./'           carpeta2           '/usuario'
num2str(usuario)],'s'); %se elimina la carpeta
                    mkdir(['./'           carpeta2           '/usuario'
num2str(usuario)]); %se crea una nueva carpeta vacia
                else
                    %Se   almacena   en   la   carpeta   de   usuario
correspondiente
                    if n==3 || n==4
                        if length(nombre1)==15
                            c_totalh=totalhorizontal/numero;
                            save(['./'       carpeta2       '/usuario'
num2str(usuario) '/' subcarpeta(8:end) 'Horizontal.mat'],'c_totalh');
                        else
                            saltar=numero_iguales-1;
                            c_totalv=totalvertical/numero;
                            save(['./'       carpeta2       '/usuario'
num2str(usuario) '/' subcarpeta(8:end) 'Vertical.mat'],'c_totalv');
                        end
                    elseif n==5
                        saltar=numero_iguales-1;
```

```
                        c_total=total/nfils;
                        save(['./'          carpeta2          '/usuario'
num2str(usuario) '/' subcarpeta(8:end) '.mat'],'c_total');
                    else
                        saltar=numero_iguales-1;
                        c_total=total/numero;
                        save(['./'          carpeta2          '/usuario'
num2str(usuario) '/' subcarpeta(8:end) '.mat'],'c_total');
                    end
                end
                nombre=[];
                usuario=[];
                nombre_usuario=[];
                nombre1=[];  %se  usa  para  n==3  y  n==4  pq  tienen
componentes horizontales y verticales
                i=i+saltar+1;
            end %while
        end
    end
end
```

Madrid, a 18 de Octubre de 2007

Fdo: Susana Pecharromán Balbás
Ingeniera Superior de Telecomunicación