Universidad de Los Andes
Postgrado en Computación

Tesis para la obtención del grado de
Magister Scientiae en Computación

Director de Tesis : Dr. Wladimir José Rodríguez Graterol

Identificación de Señales Verbales en el Espacio de Fase Reconstruido

AUTOR
Lic. José Alejandro Brito Boadas

Mérida, Junio de 2004

RESUMEN

Este trabajo discurre sobre la representación y caracterización de señales verbales en el Espacio de Fase Reconstruido, con el propósito de clasificarlas. Se parte considerando al aparato fonador como una *caja negra*, donde la única información disponible es su propia salida: la señal verbal. En este sentido, si la reconstrucción del espacio de fase se desarrolla apropiadamente, las estructuras geométricas o atractores delineados en éste equivalen topológicamente a los del sistema original, lo cual ofrece una forma de acceder a la dinámica subyacente, en principio desconocida, de las pronunciaciones. En concreto, se realizan experimentos con vocales y dígitos, en su mayoría extraídos de la base de datos SpeechDat Venezolana. Para el cálculo de los parámetros de reconstrucción se apela al método de Mínima Entropía Diferencial. La caracterización de las vocales recurre a métricas de densidad espacial y compactación de bloques en el espacio, mientras que en el caso de los dígitos, el vector de características conjuga componentes derivados de la energía de Teager de la señal, con análisis de densidades en segmentos de la misma. Adicionalmente, se determinan los efectos en la clasificación del uso de wavelets para el tratamiento del ruido. Por último, el clasificador consiste en un relativamente sencillo arreglo de redes perceptrónicas multicapas. Los resultados obtenidos confirman la capacidad discriminante del análisis de señales verbales en el Espacio de Fase Reconstruido, particularmente, señales de voces venezolanas.

Claves: Clasificación de señales verbales, reconocimiento de patrones, espacio de fase reconstruido, dinámica no lineal, redes neuronales, SpeechDat.

ÍNDICE GENERAL

1.. Problema de Investigación 9
1.1. Motivación 9
1.2. Planteamiento del Problema 10
1.3. Trabajo realizado 12

2.. Marco Teórico 13
2.1. La Señal Verbal 13
2.1.1. Cualidades físicas de la señal verbal 16
2.1.2. La señal verbal como una serie de tiempo 17
2.2. La Identificación de la Señal Verbal 17
2.3. El Espacio de Fase Reconstruido 20
2.3.1. Sistemas Dinámicos 20
2.3.2. El Espacio de Fase 23
2.4. La Identificación en el Espacio de Fase Reconstruido 26
2.5. Tratamiento de la señal 28
2.6. Métricas de reconocimiento 29

3.. Estructura del Sistema de Identificación 31
3.1. Generalidades 31
3.2. Revisión de Redes Neuronales 32
3.3. Funcionamiento del clasificador 34

4.. Conformación de los Corpus de Voces 36
4.1. Conformación de corpus para experimentos dependientes del hablante 36
4.2. Conformación de corpus para experimentos independientes del hablante 37
4.2.1. Revisión del SpeechDat Venezolano 37
4.2.2. Definición de C_E y C_P 37
4.2.3. Codificación de los Corpus de Voces 39

5.. Reconstrucción del Espacio de Fase 40
5.1. Método basado en Entropía Diferencial para el cálculo de parámetros del Espacio de Fase Reconstruido 40

5.2. Algoritmos para el cálculo de parámetros 42
5.3. Parámetros para las vocales 42

6.. Análisis de Señales en el Espacio de Fase Reconstruido 46
6.1. Primera Normalización 46
6.2. Análisis de Vocales 46
6.3. Análisis de Dígitos 51

7.. Evaluación de Resultados 55
7.1. Vocales 55
7.1.1. Reconocimiento dependiente del hablante 55
7.1.2. Reconocimiento independiente del hablante 61
7.2. Dígitos 64
7.2.1. Reconocimiento dependiente del hablante 64
7.2.2. Reconocimiento independiente del hablante 65
7.3. Efecto del ruido 67

8.. Conclusiones y Recomendaciones 70

9.. Rutinas en MATLAB del Sistema de Identificación 73

ÍNDICE DE FIGURAS

1.1. Modelo Fuente-Filtro de la producción de voz. 11

2.1. Sistema fonatorio. 14
2.2. Gráfica de una señal verbal correspondiente a la vocal "a", con velocidad de muestreo igual a 16KHz, y factor de cuantización de 16 bits. 15
2.3. Proceso de codificación de una señal acústica. 16
2.4. Estructura general de un sistema de identificación. 17
2.5. Series de tiempo caóticas generadas por las ecuaciones de Lorenz. . . 22
2.6. Espacio de Fase tridimensional con el atractor de Lorenz. 23
2.7. Espacio de Fase Reconstruido a partir de una señal verbal arbitraria, con $m = 2$ y $\tau = 5$. 25
2.8. Espacio de Fase Reconstruido a partir de $x(t)$ en las ecuaciones de Lorenz, con $m = 3$ y $\tau = 6$. 26
2.9. Tratamiento del ruido. 29
2.10. Matriz de confusión bidimensional. 30

3.1. Estructura del Sistema de Identificación. 31
3.2. Red perceptrónica multicapa. 33

5.1. Histogramas de la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V. 43
5.2. Histograma combinado para la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V. 43
5.3. Histogramas del retraso (τ) en la reconstrucción de las señales en C_E^V. 45
5.4. Histograma combinado del retraso (τ) en la reconstrucción de las señales en C_E^V. 45

6.1. Definición de bloques en el Espacio de Fase Reconstruido. 47
6.2. Segunda instancia de vocal *a*. 49
6.3. Tercera instancia de vocal *a*. 49
6.4. Primera instancia de vocal *u*. 50
6.5. Segunda instancia de vocal *u*. 50

6.6. Tercera instancia de vocal *u*. 51
6.7. Parametrización de un dígito. 54

7.1. Tasas de reconocimiento para vocales. 60

ÍNDICE DE CUADROS

2.1. Cualidades de la señal verbal. 16
2.2. Taxonomía de los sistemas de identificación. 19
2.3. Taxonomía de los sistemas. 20
2.4. Ejemplo de reconstrucción del Espacio de Fase. 25

5.1. Valores obtenidos por el Método de Mínima Entropía Diferencial para la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V. 42
5.2. Valores obtenidos por el Método de Mínima Entropía Diferencial para el retraso (τ) en la reconstrucción de las señales en C_E^V. 44

7.1. Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 0). 56
7.2. Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 0). 56
7.3. Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 1). 56
7.4. Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 1). 57
7.5. Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 2). 57
7.6. Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 2). 57
7.7. Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 0). 58
7.8. Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 0). 58
7.9. Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 1). 58
7.10. Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 1). 59

7.11. Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 2). 59
7.12. Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 2). 59
7.13. Tasas de reconocimiento para vocales de *hblA* con aproximación híbrida (nivel 1). 60
7.14. Tasas de reconocimiento para vocales de *hblB* con aproximación híbrida (nivel 1). 61
7.15. Tasas de reconocimiento para vocales independientes del hablante en C_E^V. 61
7.16. Tasas de reconocimiento para vocales independientes del hablante en C_P^V. 62
7.17. Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^V. 62
7.18. Tasas de reconocimiento para vocales independientes del hablante en C_E^V, con nueva función de activación. 63
7.19. Tasas de reconocimiento para vocales independientes del hablante en C_P^V, con nueva función de activación. 63
7.20. Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^V, con nueva función de activación. 63
7.21. Tasas de reconocimiento dígitos de *hblA*. 65
7.22. Tasas de reconocimiento dígitos de *hblB*. 65
7.23. Tasas de reconocimiento para dígitos independientes del hablante con C_P^D. 66
7.24. Tasas de reconocimiento para dígitos independientes del hablante con C_{PP}^D. 66
7.25. Tasas de reconocimiento para vocales independientes del hablante en C_P^V tratadas con wavelets. 67
7.26. Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^V tratadas con wavelets. 68
7.27. Tasas de reconocimiento para dígitos independientes del hablante con C_P^D tratados con wavelets. 68
7.28. Tasas de reconocimiento para dígitos independientes del hablante con C_{PP}^D tratados con wavelets. 69

1. PROBLEMA DE INVESTIGACIÓN

1.1. Motivación

La investigación, el continuo desarrollo de productos y las nuevas aplicaciones de las tecnologías del habla se han incrementado dramáticamente en los años recientes, en búsqueda de una interfaz hombre-máquina natural, sucinta y versátil. Más interesante resulta la atestiguada masificación de productos, particularmente en el ámbito de la electrónica de consumo, que incluyen la facultad de reconocimiento de voz. Sin embargo, las actuales tecnologías de reconocimiento aún pueden mejorarse, considerando que, en general, persisten con las técnicas lineales. Además, las tecnologías del habla se encuentran relativamente rezagadas en el país, comparando con el nivel de las mismas en las naciones desarrolladas.

Por tal razón, los estudios alternativos, como el presente, constituyen una buena oportunidad para el fortalecimiento de las nuevas tendencias en el reconocimiento, o como complemento a las ya establecidas.

En este sentido, destacan los siguientes aportes del estudio:

- Se utiliza un enfoque relativamente nuevo, y teóricamente plausible, para el análisis no lineal de la señal. En general, se trata de aplicar técnicas de caracterización no lineal de dicha señal, considerándola como salida de una *caja negra*: el sistema fonatorio. En efecto, la única información disponible sobre este sistema será la propia señal de voz.

- La pronunciación venezolana mantiene diferencias con las de otros países, abarcando también los hispanohablantes. Incluso existen variaciones entre las regiones geográficas de Venezuela. Así, ésta es la primera investigación que aplica técnicas no lineales al análisis de señales verbales de voces venezolanas, en específico aquellas en la base de datos SpeechDat venezolana.

- Hasta ahora, las investigaciones realizadas con el método del Espacio de Fase Reconstruido han recurrido a señales cortas y sencillas. No hay antecedentes sobre el uso de señales más complejas, correspondientes a dígitos. Por ende, la investigación permitirá comprobar si este método puede abordar señales de magnitud superior.

- Los resultados del estudio fungirán de precedentes para futuros trabajos en cuanto al cálculo de parámetros del Espacio de Fase Reconstruido, en el contexto del SpeechDat venezolano.
- La implementación del sistema de identificación permitirá a posteriores investigaciones disponer de una plataforma de partida para los experimentos.

Finalmente, en el apartado académico, esta investigación pretende cumplir con un requisito parcial de la Maestría en Computación. Además, trabajos de este tipo, dadas las variadas corrientes cognoscitivas en las cuales pueden aplicarse las técnicas de reconocimiento, resultan muy convenientes para consolidar la formación profesional.

1.2. Planteamiento del Problema

La identificación de señales verbales constituye un problema de reconocimiento de patrones, en el cual se analizan señales acústicas, de origen vocal y digitalmente codificadas, con la finalidad de clasificarlas según categorías linguísticas predefinidas y dependientes de la aplicación, por ejemplo, fonemas, sílabas o palabras. La decisión sobre la pertenencia de una señal a una categoría particular se basa en *características* o *parámetros* extraídos de la señal en cuestión. De esta forma, cada categoría se encuentra definida por un conjunto de características, y la identificación se reduce a la aplicación de alguna métrica para relacionar las características de señales con las de categorías.

Usualmente, la caracterización de estas señales se efectúa mediante técnicas de sistemas lineales [21, 37, 39], entre otras razones, por las abundantes y bien establecidas herramientas para el análisis de datos, y por las facultades de superposición y transformación entre los dominios del tiempo y la frecuencia, que facilitan, en cierta medida, las operaciones con señales. No obstante, tal proceder recurre a modelos lineales del sistema fonatorio, con lo cual, en última instancia, sólo podrá arribarse a aproximaciones de la señal verbal. En este sentido, el enfoque más popular es el *fuente-filtro* [9], presentado en la Figura 1.1. En este modelo la idea es que el sistema fonatorio puede separarse, a grandes rasgos, en dos componentes, a saber, la fuente de excitación y el tracto vocal, típicamente bajo la forma de un filtro. Así, la voz se produce por el filtrado o cambios espectrales que el tracto vocal induce en la señal de excitación.

A pesar del éxito logrado, los sistemas de identificación de este tipo deben apoyarse fuertemente en modelos linguísticos y ambientales que respalden su clasificación de una señal. Igualmente, se desdeña la capacidad discriminante que pudiera aportar la información sobre la dinámica interna del sistema fonatorio.

Como alternativa a las técnicas convencionales, en esta investigación se recurre al análisis no lineal de características del Espacio de Fase Reconstruido. Con tal fin,

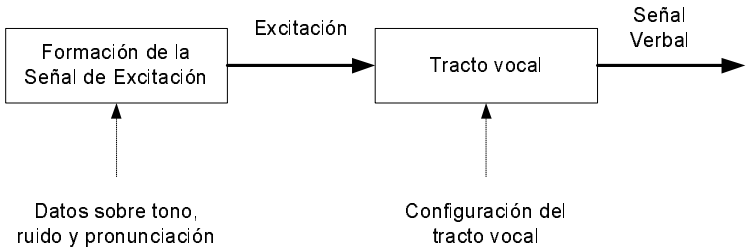

Fig. 1.1: Modelo Fuente-Filtro de la producción de voz.

se presupone el carácter no lineal del sistema fonatorio. Esta primera hipótesis se encuentra respaldada por algunos estudios que proporcionan evidencia matemática sobre el carácter no lineal de la producción de la voz [49]. Biológicamente, se le objeta al modelo *fuente-filtro* que en el caso de los fonemas sordos, la fuente de excitación es la turbulencia que se origina en el tracto vocal mismo, por lo que la fuente natural, en este caso, no corresponde con la del modelo [40]. Aún más, para apoyar la hipótesis de no linealidad, también se puede argumentar informalmente que:

- Un hablante difícilmente emite pronunciaciones idénticas consecutivamente, ni siquiera bajo voluntad. Incluso entre distintos segmentos o tramas de la señal correspondiente a un fonema suele apreciarse variabilidad.

- Hasta el momento ha resultado imposible la obtención de un modelo matemático lineal que se ajuste plenamente a la alta variabilidad de la señal verbal, en cuanto a sus cualidades fisiológicas.

- La vasta mayoría de los sistemas naturales, y aún los construidos por el hombre, no son lineales.

En consecuencia, el modelo lineal resulta aceptable sólo como aproximación del proceso no lineal que se desarrolla durante las emisiones de voz, suponiendo además que la señal verbal se mantiene casi invariable si las tramas seleccionadas para el análisis son suficientemente pequeñas[1].

Recientemente, ha emergido una nueva técnica para la identificación de señales verbales mediante el análisis de un espacio matemático derivado de la salida, o fluctuaciones en la salida, del sistema fonatorio [21, 53, 54]. Recuérdese que, para un reconocedor típico, este sistema no resulta plenamente observable. De hecho, la

[1] Esto establece un compromiso: no basta con que las tramas sean de tamaño reducido, sino también significativas, para permitir que el análisis extraiga información útil.

propia señal verbal es la única información disponible sobre el mismo. Convenientemente, bajo ciertas condiciones teóricas, la reconstrucción del espacio de fase, a partir de las observaciones de un único sensor, equivale cualitativamente a la dinámica interna del sistema [1, 44]. La hipótesis fundamental es que la representación de la señal en el Espacio de Fase Reconstruido posee suficiente capacidad discriminante para capturar la dinámica del sistema fonatorio. Los antecedentes han alcanzado tasas de reconocimiento regulares, aunque los casos de mayor alcance sólo han trabajado con fonemas aislados, sin superar los registros de efectividad de las técnicas lineales. Por otra parte, se carece de estudios que comprueben la aplicabilidad del método con voces venezolanas.

1.3. Trabajo realizado

El objetivo general de la investigación se circunscribe a la construcción, en el ambiente MATLAB, de un sistema de identificación basado exclusivamente en el análisis de la señal verbal en el Espacio de Fase Reconstruido. El sistema procesa señales correspondientes a vocales y dígitos de hablantes venezolanos, en dos vertientes, a saber, una versión dependiente del hablante, y otra independiente. En el caso dependiente del hablante, el sistema se entrena y verifica con voces grabadas por el autor. Con la otra opción, se recurre a vocales y dígitos en la base de datos SpeechDat de voces venezolanas, construida en la Universidad de Los Andes [30]. Una vez definidos los corpus de entrenamiento y prueba, la solución involucra la representación de las señales en el Espacio de Fase Reconstruido, y la sucesiva extracción de vectores de características. Luego, arreglos de redes neuronales de retropropagación [13] conforman un clasificador de señales a partir de los vectores obtenidos. Esto exige la resolución de problemas subalternos inherentes a este espacio matemático, como el cálculo de parámetros de inmersión y la susceptibilidad al ruido. Para lo primero, se utiliza el método de entropía diferencial, por su completitud [8], y para el ruido, algunos estudios sugieren el uso de wavelets [2, 3, 7]. Finalmente, hay que acotar que el sistema de identificación corresponde a un reconocedor *fuera de línea* para confirmar la viabilidad del método en la discriminación de señales vocálicas, y no como prototipo o implantación de un reconocedor en tiempo real.

2. MARCO TEÓRICO

2.1. La Señal Verbal

Sonido equivale a vibración, o de forma concisa, constituye una onda de presión longitudinal formada por compresiones y rarefacciones de las moléculas de aire [12]. Intuitivamente, las compresiones son concentraciones de las moléculas del aire, en cierto espacio, como resultado de la aplicación de energía. Por el contrario, las rarefacciones constituyen zonas donde las moléculas se encuentran menos estrechamente aglomeradas. De este modo, las ondas de presión que emanan de la boca y nariz del hablante producen la voz. Este cambio de la presión de aire se propaga hasta el tímpano, membrana elástica que obtura el oído, permitiendo la percepción del sonido.

Los elementos anatómicos que intervienen en la producción de la voz conforman el *sistema fonatorio* (ver Figura 2.1). En dicho sistema, los pulmones representan la fuente de aire. Durante la articulación de un fonema[1], las cuerdas vocales pueden vibrar (fonemas sonoros), o encontrarse en un estado demasiado laxo o tenso que impide su vibración periódica (fonemas sordos). Otros componentes del sistema, a saber, paladar, velo del paladar, mandíbula, lengua, dientes y labios, pueden modificar la emisión sonora en su trayecto hacia el exterior, a través de los tractos nasal y vocal [12].

Para nuestros propósitos, la emisión verbal debe definirse como una señal, es decir, como la salida de un sistema, en este caso, el sistema fonatorio. Una señal es una función que contiene información, normalmente, bajo la forma de patrones espaciales y temporales [20]. Así, la señal verbal S_v no es más que una función

$$S_v : \text{Tiempo} \rightarrow \text{Presión de aire} \ .$$

La cantidad de componentes en el sistema fonatorio, los cambios en los mismos a lo largo del tiempo, y las complejas interacciones entre ellos, ocasionan variabilidad en S_v, de tal manera que un hablante no pronuncia dos veces seguidas una voz vocal o una consonante exactamente de la misma manera [31].

[1] Un fonema es la imagen mental de un sonido, o también, denota cualquiera de las unidades mínimas del sonido verbal en un lenguaje, que pueden servir para distinguir una palabra de otra.

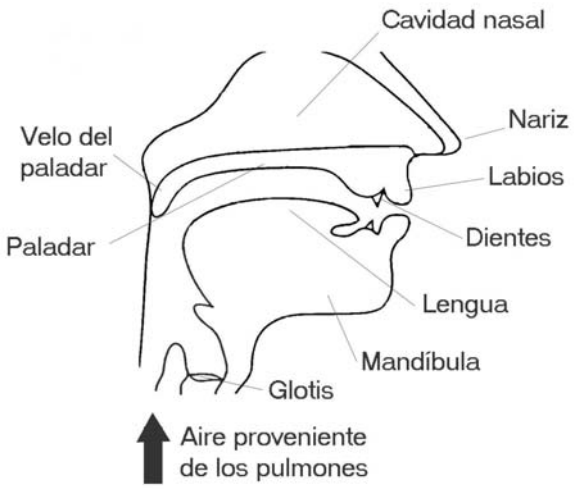

Fig. 2.1: Sistema fonatorio.

Como ejemplo, la Figura 2.2 exhibe la señal verbal correspondiente a una pronunciación de la vocal *a*. Se ha utilizado una velocidad de muestreo de 16 kHz, y un factor de cuantización de 16 bits; estos conceptos se presentarán en breve. Una simple inspección visual de la figura revela la periodicidad de la señal (repitiéndose aproximadamente cada 120 muestras), aunque se aprecian mínimas perturbaciones. Debe aclararse que sólo los fonemas sonoros evidencian este comportamiento periódico. Nótese también que la gráfica muestra valores tanto positivos como negativos, promediando el cero, aunque la presión de aire sólo admite valores positivos: por claridad, la señal suele normalizarse, sustrayendo la presión ambiental[2], dado que el oído humano es insensible a esta presión.

Ahora bien, mediante el proceso de *codificación*, la señal verbal, acústica e inherentemente análoga, se convierte en una representación digital que permite analizarla computacionalmente [9]. Esta nueva representación constituye una aproximación de la señal original. La Figura 2.3 ilustra el proceso.

El primer paso consiste en capturar la señal mediante un transductor acústico (micrófono) y convertirla en una corriente eléctrica ①, sobre la cual, posteriormente, un Convertidor Analógico Digital (CAD) aplica algún enfoque de codificación digital ②. Existen diversas técnicas de codificación, pero aquí se considerará PCM[3] lineal,

[2] Aproximadamente $10^5 newtons/m^2$
[3] Pulse Code Modulation

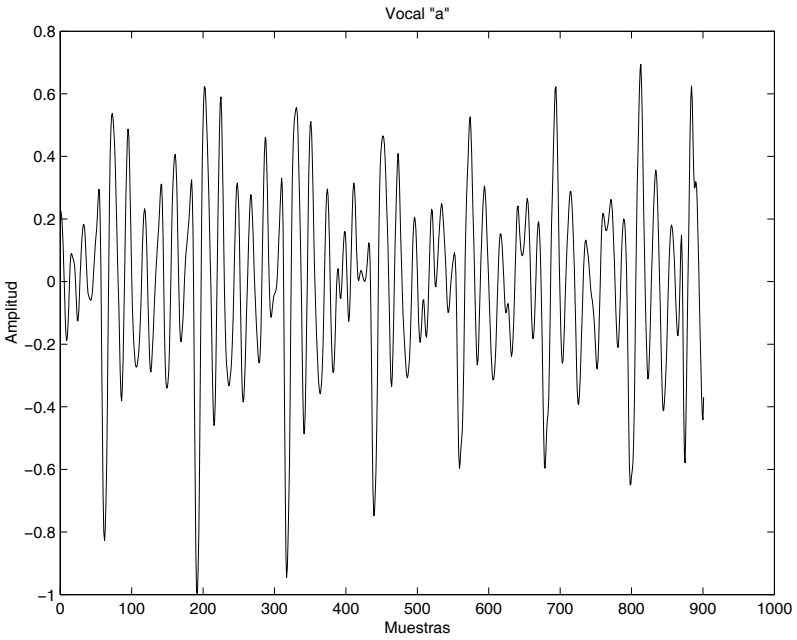

Fig. 2.2: Gráfica de una señal verbal correspondiente a la vocal "a", con velocidad de muestreo igual a 16KHz, y factor de cuantización de 16 bits.

por su sencillez teórica y popularidad. Con PCM, la señal se muestrea según una tasa periódica y constante. Para cada muestra, se cuantifica la amplitud de la señal. Como se observa, existen dos factores determinantes:

- **Velocidad de Muestreo**: Es la frecuencia de observación de la señal. Por ejemplo, en la Figura 2.2, el muestreo de 16 kHz implica una muestra cada 1/16000 segundos.
- **Factor de cuantización**: En bits, describe la precisión con la que se graba la energía en cada punto de muestreo.

En los sistemas de identificación, las velocidades de muestreo por lo general se ubican entre los 8 kHz y los 16 kHz [12]. Por su parte, la cuantificación suele emplear de 8 a 12 bits [43]. El proceso de identificación opera sobre esta señal discretizada. En lo sucesivo, S_v se referirá a la señal verbal discretizada.

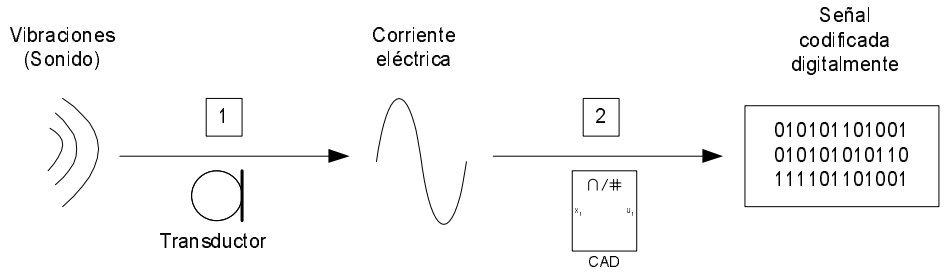

Fig. 2.3: Proceso de codificación de una señal acústica.

2.1.1. Cualidades físicas de la señal verbal

La intensidad, la altura, el timbre y la duración constituyen las cualidades físicas básicas de la señal verbal, evidentemente, de naturaleza acústica [42]. La Tabla 2.1 describe cada una.

Intensidad	Se refiere a la magnitud de la sensación experimentada al percibir la voz, y mediante la cual puede discernirse entre sonidos *fuertes* o *débiles*. Esta cualidad depende de la amplitud de la señal, y resulta afectada, principalmente, por la distancia y el medio de transmisión.
Altura	Depende de la frecuencia de la señal, y permite, por ejemplo, catalogar unos sonidos como *graves* (baja frecuencia) y otros como *agudos* (alta frecuencia).
Timbre	El timbre es la cualidad mediante la cual pueden distinguirse dos sonidos con igual intensidad y altura, emitidos por fuentes sonoras distintas. Se debe a que la onda sonora no es *pura*, y presenta una o varias periodicidades superpuestas, por lo que el oído percibe un sonido más complejo.
Duración	Es el intervalo temporal en el cual el sonido persiste sin discontinuidad [10].

Tab. 2.1: Cualidades de la señal verbal.

Cuando se hace referencia a la variabilidad de la señal verbal, implícitamente se abordan estos factores. En efecto, entre diversos hablantes, y aún en un mismo hablante, las pronunciaciones revelan diferencias significativas en intensidad, altura, timbre y duración. De esta forma, la efectividad de un sistema de identificación se encuentra supeditada a su capacidad para confrontar esta variabilidad en la señal,

dependiente de las cualidades físicas.

2.1.2. La señal verbal como una serie de tiempo

La señal verbal, claramente, también puede concebirse como una *serie de tiempo*, por cuanto consta de observaciones de una variable (en este caso, presión de aire) a lo largo del tiempo. Las observaciones o muestras que conforman a S_v se denotan

$$S_v[1], S_v[2], ..., S_v[L(S_v)]$$

donde $L(\cdot)$ representa la cantidad de muestras u observaciones.

Esta perspectiva resulta de sumo interés, porque existen técnicas de reconstrucción de la dinámica de los sistemas que proceden a partir de las series temporales. En concreto, S_v refleja los cambios del sistema fonatorio durante un período de tiempo. Luego, si puede caracterizarse la dinámica del sistema fonatorio, debería ser factible la identificación de sus salidas [4, 32].

2.2. La Identificación de la Señal Verbal

Hasta este punto se ha discutido la *forma* de S_v. Sin embargo, la señal representa alguna unidad de reconocimiento predefinida, por ejemplo, fonemas, palabras u oraciones. Estas unidades constituyen las categorías $\aleph$ en las cuales el sistema de identificación clasifica sus entradas. En otras palabras, un sistema de identificación implica un mapa de S_v en $\aleph$, y su estructura general es [12]:

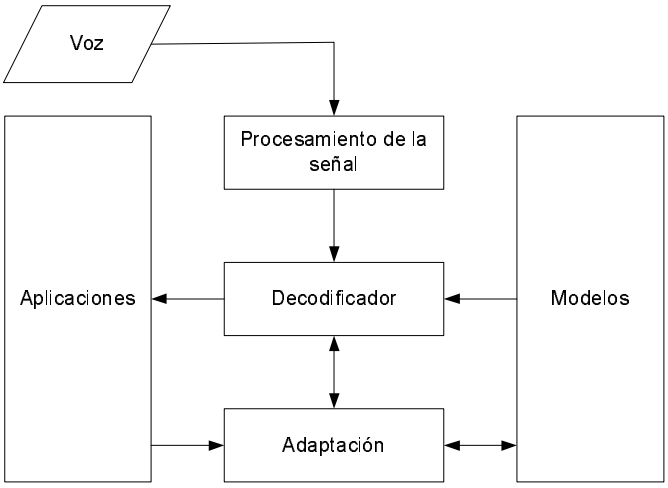

Fig. 2.4: Estructura general de un sistema de identificación.

El módulo de procesamiento de la señal extrae los vectores de características V_C para el decodificador. Éste recurre a los modelos, acústico y linguístico, para generar la secuencia de palabras más probable. También puede proporcionar información al componente de adaptación para modificar los modelos, de tal forma que se mejore el desempeño de las aplicaciones. Naturalmente, el decodificador debe conocer con antelación, o aprender por su cuenta, cuáles son los rasgos que distinguen una señal de otra, para poder decantarse por alguna categoría en ℵ. Típicamente, esto exige sesiones de entrenamiento a partir de corpus de señales. Para una revisión de las variedades de sistemas de identificación, consultar la Tabla 2.2.

El proceso inicia con la extracción de las *características* C_i que permitirán ubicar la señal en alguna de las categorías ℵ del sistema. Las técnicas lineales en el dominio de la frecuencia suelen usar como características la distribución de la energía en los diversos rangos de frecuencia, y a lo largo de tramas superpuestas. El modelo general de identificación establece que, dada una observación acústica $X = C_1C_2...C_n$, el objetivo de la identificación es encontrar la secuencia correspondiente de palabras $\hat{W} = w_1w_2...w_m$ con la mayor probabilidad $p(W \mid X)$:

$$\hat{W} = \underset{w}{argmax}\ p(W \mid X) = \underset{w}{argmax}\ \frac{p(W)p(X \mid W)}{p(X)} \tag{2.1}$$

Por cuanto la maximización transcurre con una X fija, el denominador p(X) puede removerse de la expresión.

A $p(W)$ se le denomina *modelo del lenguaje*, y contiene la información sobre la manera en que se combinan las palabras para formar frases. Por ejemplo, el modelo del lenguaje permite saber, cuando no hay seguridad, que en un determinado contexto, *ensalada* es una alternativa más probable que *ensenada*. Por otro lado, $p(X \mid W)$ representa el *modelo acústico*, el cual proporciona información sobre las propiedades y características de los sonidos asociados a cierta cadena de palabras. Este modelo consta de dos niveles. En el primero se establece la descripción de cada palabra como una secuencia de fonos[4]. El siguiente nivel indica la forma en que se relacionan estos fonos con los diversos C_i de la observación acústica. Por sí solos, los modelos linguísticos y acústicos constituyen problemas complejos, profusamente investigados, con métodos establecidos, como los n-gramas [12] y los Modelos Ocultos de Markov [35, 36], respectivamente.

[4] Un fono es la realización acústica de un fonema.

[Propiedad]	[Descripción]
Dependencia del hablante	Los sistemas dependientes del hablante se entrenan para reconocer el habla de un conjunto invariable de individuos [26]. Obviamente, resultan más fáciles de construir. Por su parte, los sistemas independientes del hablante, en teoría, se proponen el reconocimiento de la voz de cualquier persona, lo cual resulta difícil por el *sesgo* de estos sistemas hacia las voces con las que fueron entrenados. Sin tener que decirlo, el error de reconocimiento en estos sistemas suele ser más elevado que en los dependientes del hablante.
Continuidad del habla	Los reconocedores de palabras aisladas exigen pausas entre las palabras, lo que facilita la construcción, y promueve el rendimiento del reconocedor. Por su parte, los reconocedores de habla continua, aunque más amigables para el usuario, deben afrontar los problemas de segmentación y sensibilidad al contexto. En consecuencia, el entrenamiento y operación del sistema se torna complicado, por la dificultada para detectar el inicio y final de cada palabra, y por las diferencias en la pronunciación de las palabras dependiendo del contexto de las mismas.
Unidad de reconocimiento	Esta categoría se basa en la selección de la mínima unidad comunicacional reconocida por el sistema. Por ejemplo, ciertos sistemas de reconocimiento de órdenes pueden apoyarse en un conjunto relativamente pequeño de palabras predefinidas. Sin embargo, este enfoque no es escalable, pues resulta inútil cuando se abordan dominios con ingentes cantidades de palabras. En tales casos, favorece utilizar una unidad con menor granularidad, como fonemas, sílabas, trifonos, entre otras. Posteriormente, la secuencia de unidades identificadas permitirá la reconstrucción de la palabra original.
Robustez ambiental	Algunos sistemas son entrenados especialmente para reconocer señales alteradas por el ruido. Por ejemplo, el reconocimiento incorporado a un dispositivo móvil como un teléfono celular, debería funcionar efectivamente aún en presencia de sonidos de fondo como tráfico y murmullos. Una misma voz podría capturarse por diversas entradas, para una mayor fidelidad.

Tab. 2.2: Taxonomía de los sistemas de identificación.

2.3. El Espacio de Fase Reconstruido

2.3.1. Sistemas Dinámicos

Un sistema es una combinación de elementos que interactúan para lograr un fin determinado [24]. Básicamente, mediante dicha interacción se pretende la transformación de las señales de entrada en otras señales de salida, correspondientes al objetivo específico del sistema [20]. En la Tabla 2.3 se aprecia una taxonomía de los sistemas [20, 24].

[Tipo]	[Descripción]
Dinámico vs Estático	Un sistema es dinámico si su salida, en un instante dado, depende de entradas previas; si la salida sólo depende de la entrada del momento, es un sistema estático.
Lineal vs No Lineal	En los sistemas lineales, aplica el *principio de superposición*. Es decir, la respuesta producida por aplicaciones simultáneas de dos entradas diferentes es la suma de dos respuestas individuales. En los sistemas no lineales, no aplica este principio, por lo que resultan muy difíciles de abordar matemáticamente; suelen aproximarse mediante modelos lineales.
Continuos vs Discretos	En los sistemas continuos, las señales involucradas son continuas en el tiempo. En los discretos, una o más variables cambian en instantes discretos de tiempo.
Deterministas vs Probabilistas	En los sistemas deterministas, la obtención de las señales de salida sigue un proceso perfectamente delimitado. Por el contrario, en los probabilistas interviene la aleatoriedad.

Tab. 2.3: Taxonomía de los sistemas.

De acuerdo con la anterior clasificación, el sistema fonatorio es continuo, aunque la señal sobre la que procede la identificación se discretiza. También es dinámico porque, por ejemplo, la articulación de un sonido puede verse afectada por las circundantes, predecesora y sucesora, efecto conocido como *coarticulación*. Por otra parte, decidir en cuanto a Lineal - No Lineal y Determinista - Probabilista resulta más difícil, pero algunas investigaciones evidencian no linealidad del sistema fonatorio [49]

y determinismo, al menos en las vocales inglesas [23].

En particular, un sistema dinámico exhibe un comportamiento que evoluciona con el tiempo, determinado plenamente por las k variables físicas bajo estudio, conocidas como *variables de estado* [28]. Así, un sistema de k variables se denomina k-dimensional. El espacio $\Re^k$, o un subconjunto apropiado de éste, se denomina *espacio de fase* y contiene todos los posibles estados de un sistema dinámico. También se conoce como *espacio de estados*, *espacio de configuraciones*, o *espacio de fase abstracto*. En las representaciones gráficas, cada coordenada se asocia a una variable de estado. Implícitamente, debe existir alguna regla que especifique cómo proceden las transiciones entre estados. En el análisis de señales provenientes de sistemas físicos suele suponerse que dicha regla adquiere la forma de ecuaciones diferenciales o de diferencias. Así, típicamente suele emplearse la forma canónica

$$\frac{dx}{dt} = \dot{x} = f(x) \tag{2.2}$$

donde f es una función $f : U \rightarrow \Re^k$, y U es un subconjunto abierto de $\Re^k$. Las variables de estado se agrupan en el vector $x = (x_1, x_2, ..., x_k)^T$. Dichas variables dependen del tiempo $t \in \Re$. Luego, t es la variable independiente. Cuando la función f no depende de t directamente, sino a través de las variables de estado, como en este caso, el sistema se denomina *autónomo*.

Luego, el *camino* delineado en el espacio de fase por una sucesión de estados, a partir de algún estado inicial, representa una *trayectoria*. El análisis del sistema mediante estas estructuras geométricas o *atractores* en el espacio de fase constituye un enfoque alternativo para abordar sistemas complejos, no lineales [55, 56]. A grandes rasgos, un atractor es un conjunto hacia donde convergen todas las trayectorias cercanas. En términos formales, es un conjunto A que satisface estos postulados [27]:

- A es un conjunto invariante: cualquier trayectoria que inicia en A permanece en él.
- A atrae a todas las trayectorias que comienzan *suficientemente cerca* de él. Formalmente, existe un conjunto abierto U que contiene a A, tal que si $x \in U$, $distancia(x, A) \rightarrow 0$, a medida que $t \rightarrow \infty$.
- A es minimal: no existe ningún subconjunto de A que satisfaga los postulados precedentes.

Para ejemplificar, ahora se revisarán las ecuaciones de Lorenz [25], un sistema autónomo clásico en el ámbito del análisis caótico. El objetivo de estas ecuaciones es el modelado de la convección atmosférica. Dicho sistema viene dado por:

$$\begin{aligned} \frac{dx(t)}{dt} &= \sigma(y(t) - x(t)) \\ \frac{dy(t)}{dt} &= -x(t)z(t) + rx(t) - y(t) \\ \frac{dz(t)}{dt} &= x(t)y(t) - bz(t) \\ \sigma, r, b &> 0 \end{aligned} \tag{2.3}$$

Este sistema ciertamente no es lineal, por la presencia de los términos $x(t)z(t)$ y $x(t)y(t)$. Por otro lado, la aproximación lineal resulta inviable [1]. En relación con los parámetros, σ es el número de Prandtl o proporción entre la disipación térmica y la viscosa, r es la proporción entre el número de Rayleigh y el número de Rayleigh crítico, y b es la escala de una celda convectiva. La Figura 2.5 ilustra una solución del sistema[5], con los valores usados originalmente por Lorenz, a saber, parámetros $\sigma = 10$, $b = 8/3$, y $r = 28$, y condiciones iniciales $(0, 1, 0)^T$.

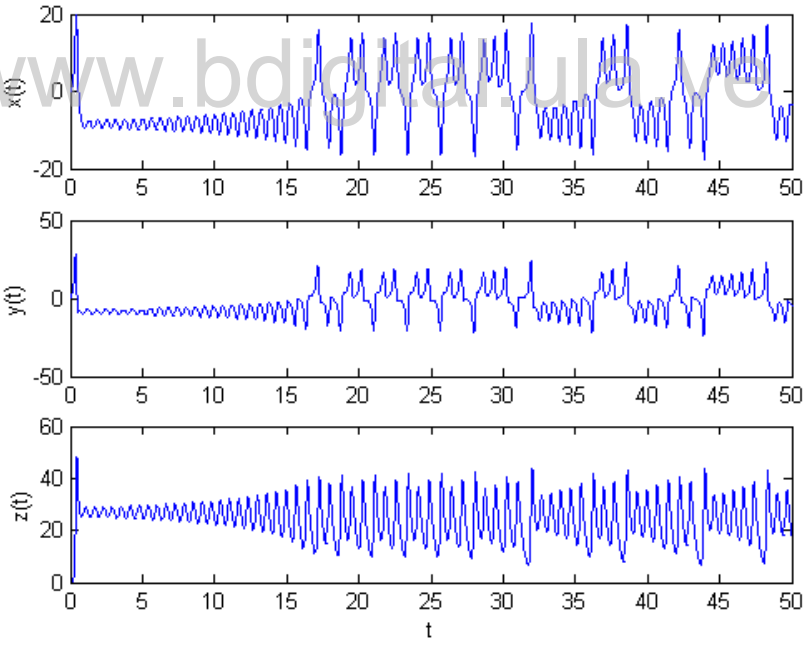

Fig. 2.5: Series de tiempo caóticas generadas por las ecuaciones de Lorenz.

[5] Computada con el ODE Solver de MATLAB.

Cuando estas trayectorias se trasladan al espacio tridimensional se obtiene un atractor extraño, con la forma de alas de mariposa (Figura 2.6). Más adelante se contrastará este atractor con el recuperado en el Espacio de Fase Reconstruido.

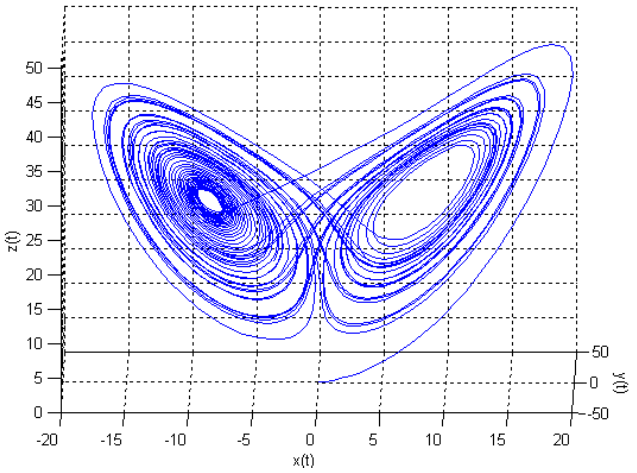

Fig. 2.6: Espacio de Fase tridimensional con el atractor de Lorenz.

2.3.2. *El Espacio de Fase*

En el caso de los sistemas no lineales, con datos incompletos, la extracción de información nueva a partir de los datos resulta más difícil que en la contraparte lineal [4]. Si el sistema en cuestión es altamente complejo (eg., el sistema fonatorio), pero sólo una de sus propiedades (eg., la señal verbal) está al alcance de algún sensor, los procedimientos de análisis tradicionales resultarán muy limitados. Como alternativa, la reconstrucción del espacio de fase permite recuperar la dinámica de un sistema no lineal a partir de una única serie de tiempo [1]. Naturalmente, el espacio reconstruido no equivale completamente a la dinámica interna del sistema, pero bajo ciertas restricciones teóricas, preserva la topología de la misma. Esto permite que las conclusiones obtenidas en la dinámica reconstruida resulten válidas también en la verdadera e inaccesible dinámica interna (caja negra) [1, 44, 47]. Además, el Espacio de Fase Reconstruido facilita la detección de estructuras que en la serie de tiempo podrían pasar desapercibidas [52].

A continuación se describe la obtención del Espacio de Fase Reconstruido. Considérese un conjunto de muestras uniformemente espaciadas de una única variable, como S_v. El Espacio de Fase Reconstruido es una representación multidimensional

de la señal contra versiones demoradas de sí misma (subseries). En términos más formales, el Espacio de Fase Reconstruido se forma mediante la definición de vectores $\vec{X}_n$ en $\Re^k$, donde

$$\vec{X}_n = \{S_v[n], S_v[n+\tau], ..., S_v[n+(m-1)\tau]\}$$

o

$$\vec{X}_n = \{S_v[n], S_v[n-\tau], ..., S_v[n-(m-1)\tau]\}$$

$S_v[i]$ es el valor de la señal en el tiempo (muestra) i, mientras que m es una constante fundamental para la reconstrucción, denominada **dimensión de inmersión**. La dimensión de inmersión puede apreciarse de las siguientes maneras:

1. Es la cantidad total de series de tiempo (S_v y sus subseries) involucradas en el análisis.

2. Gráficamente, es la cantidad de ejes, y analíticamente, la cantidad de variables.

Por su parte, el desplazamiento τ en las subseries se conoce como **retraso**, y en conjunción con la dimensión de inmersión, impone severas precondiciones a la reconstrucción. El teorema de Takens [47], que relaciona el Espacio de Fase Reconstruido con la verdadera dinámica interna del sistema, expresa que dadas una dimensión de inmersión suficiente, y el retraso apropiado, la dinámica real y el Espacio de Fase Reconstruido resultan topológicamente idénticos. Esta equivalencia permite extraer conclusiones sobre la dinámica de un sistema k-dimensional usando la salida de un único sensor. Sin embargo, para obtener el Espacio de Fase Reconstruido, se requieren m y τ, y no existen, a la fecha, métodos para derivar los valores correctos a partir de las muestras, ni directa ni indirectamente, de modo que los investigadores del área suelen recurrir a heurísticas y aproximaciones empíricas [4]. Sin embargo, recientemente se presentó una nueva técnica basada en entropía diferencial [8], la cual permite obtener a la vez m y τ a partir de las muestras, con buenos resultados prácticos reportados, si bien no puede garantizar que los parámetros calculados sean correctos. Vista la forma de reconstruir el espacio, en la Figura 2.7 se exhibe el Espacio de Fase Reconstruido correspondiente a una señal verbal arbitraria.

Anteriormente se refirió que, considerando las densidades de puntos en diversas zonas del espacio, las trayectorias en el Espacio de Fase Reconstruido representan patrones gráficos o atractores, sobre los cuales puede operar un clasificador. En concreto, estos patrones corresponden a la distribución de los puntos en el espacio, a medida que $i \to \infty$. Sin embargo, con una dimensión de inmersión muy alta[6] habrá

[6] En concreto, el problema se presenta cuando $m > 3$, porque no puede graficarse el espacio reconstruido directamente.

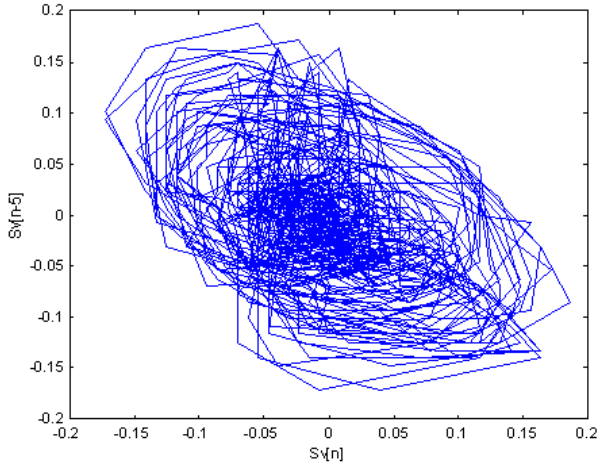

Fig. 2.7: Espacio de Fase Reconstruido a partir de una señal verbal arbitraria, con $m = 2$ y $\tau = 5$.

que aplicar alguna técnica para reducir la dimensionalidad de los datos, como por ejemplo PCA[7] [54] y/o ISOMAP [50], pero en general, la técnica dependerá de las m y τ utilizadas.

Con fines ilustrativos, supóngase que la serie dada para reconstruir el espacio consta de los primeros 10 valores calculados para $x(t)$ en las ecuaciones 2.3: 0, 0.0001, 0.0001, 0.0002, 0.0002, 0.0005, 0.0007, 0.0010, 0.0012, 0.0025. Luego, con $m = 4$ y $\tau = 1$, se mapea la serie contra algunas de sus subseries retrasadas (Tabla 2.4).

x(t)	**x(t-1)**	**x(t-2)**	**x(t-3)**
0.0000	0.0001	0.0001	0.0002
0.0001	0.0001	0.0002	0.0002
0.0001	0.0002	0.0002	0.0005
0.0002	0.0002	0.0005	0.0007
0.0002	0.0005	0.0007	0.0010
0.0005	0.0007	0.0010	0.0012
0.0007	0.0010	0.0012	0.0025

Tab. 2.4: Ejemplo de reconstrucción del Espacio de Fase.

Retornemos ahora a las ecuaciones 2.3, con el objetivo de presentar un ejemplo

[7] Análisis del Componente Principal.

gráfico. Supóngase que sólo se tiene acceso a la serie $x(t)$, lo cual resulta consistente con nuestra reiterada premisa: S_v es la única información disponible sobre el sistema fonatorio. Para recuperar información sobre la dinámica del sistema, la Figura 2.8 exhibe la reconstrucción del espacio de fase con $m = 3$, y arbitrariamente, $\tau = 5$.

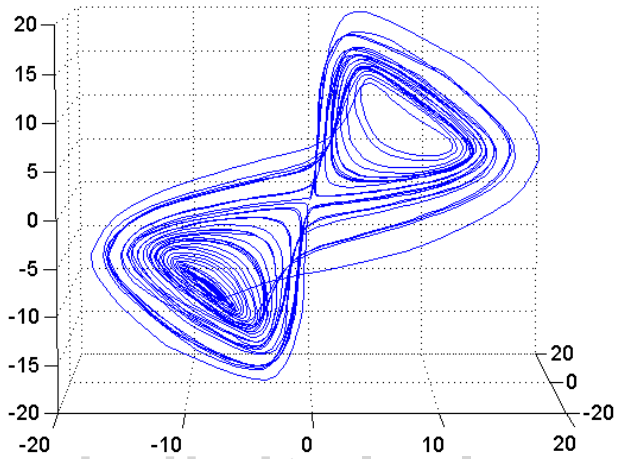

Fig. 2.8: Espacio de Fase Reconstruido a partir de $x(t)$ en las ecuaciones de Lorenz, con $m = 3$ y $\tau = 6$.

Nótese que, a pesar de no disponer de $y(t)$ y $z(t)$, el atractor obtenido resulta similar al de la Figura 2.6: las trayectorias preservan su comportamiento. De este modo, puede recuperarse alguna información sobre la dinámica.

2.4. La Identificación en el Espacio de Fase Reconstruido

Existen antecedentes importantes y recientes en cuanto al análisis de señales verbales en el Espacio de Fase Reconstruido, en su mayoría, tratando pronunciaciones inglesas. Sin embargo, no se han encontrado estudios que aborden el análisis de dígitos en dicho espacio. Iniciemos la reseña con [54], investigación en la cual se identifican fonemas del corpus TIMIT mediante un clasificador bayesiano ingenuo. El Espacio de Fase Reconstruido se caracteriza estadísticamente, mediante un histograma bidimensional, para estimar las masas de probabilidad. Además, se establece $m = 2$, y aunque se obtiene una tasa de reconocimiento aceptable con las consonantes fricativas (58.94 %), los resultados de las vocales y consonantes nasales son pobres (33.00 % y 16.67 %, respectivamente). No se especifica el tamaño de los corpus, aunque sí que

constan exclusivamente de pronunciaciones de locutores masculinos, seis en el corpus de entrenamiento, y tres en el de prueba.

Un estudio relacionado es [21], que caracteriza la distribución natural de los puntos en el espacio. Allí se usa $m = 5$, y el vector de características incluye tanto los datos de la señal (original y subseries con retraso) como la trayectoria del atractor a lo largo del tiempo. La estructura de reconocimiento es un Modelo Oculto de Markov. El entrenamiento y las pruebas también emplean TIMIT, y de nuevo, aunque se alcanzan resultados prometedores (un máximo de 38.06 %), no logra superarse el rendimiento con las técnicas lineales (un máximo de 54.86 % sobre el mismo corpus). No se proporciona información sobre las dimensiones de los corpus.

En las dos investigaciones anteriores se efectúa una normalización de los vectores de características, para minimizar el efecto de la inconsistencia en las amplitudes de las señales verbales. Ambas comprobaron que el reconocimiento mejora un poco con la normalización.

En [22] se utiliza un enfoque de reconstrucción del flujo global para generar una descripción cualitativa de la estructura y trayectoria de los atractores de vocales en el Espacio de Fase Reconstruido. Luego, define una métrica para cuantificar la similitud entre los atractores. Se logra un satisfactorio 58.1 % contra un 60.1 % de las técnicas lineales en el mismo corpus. En forma similar a las anteriores referencias, tan sólo se afirma que los corpus de entrenamiento y prueba incluyen pronunciaciones de 24 locutores masculinos, sin mayor detalle.

En [53], se apela al análisis del componente principal para afrontar el problema de la elevada dimensionalidad de los datos, sin que se perciban mejoras notables en el reconocimiento. Se efectúa un experimento dependiente del hablante, con 417 ejemplares de fonemas, distribuidos en 48 categorías. En el caso independientes del hablante, las categorías se reducen a 3, con pronunciaciones de 9 locutores masculinos, y no se proporciona más información sobre la dimensión de los corpus.

En [38] se combina el Espacio de Fase Reconstruido con técnicas de aprendizaje artificial para identificar arritmias y ritmos cardíacos normales, señales distintas a la verbal. Se usa el enfoque del histograma bidimensional, para trabajar con una red neuronal de 100 entradas, y una salida, por lo que naturalmente, se disponía de redes neuronales individuales para cada ritmo. La exactitud promedio de reconocimiento fue de un 83 %, lo cual respalda al método de análisis en el Espacio de Fase Reconstruido, aunque las investigaciones descritas anteriormente no hayan alcanzado un éxito rotundo con las señales verbales.

En [40, 41] se recurre a una representación de las trayectorias de salida usando conjuntos temporales difusos (conjuntos difusos construidos a partir de un universo cuyos elementos están ordenados en el tiempo [18]). Luego, la similitud entre segmentos de señales verbales se determina a partir de métricas de similitud entre las representaciones de conjuntos temporales. El estudio, entre otras cosas, concluye que

la escogencia del algoritmo de clustering es más significativo que la de métricas de similitud. El reconocedor construido logra detectar correctamente la similitud entre las señales verbales, aunque el corpus sólo incluía 3 vocales pronunciadas por 2 hablantes rumanos.

No puede dejar de mencionarse el trabajo pionero en el reconocimiento con el SpeechDat venezolano [26], aunque pertenezca al espectro de las técnicas lineales. Como se utiliza esta base de datos, el rendimiento del sistema debe contrastarse, en la medida de lo posible, con los resultados de la referida investigación. En ésta, el reconocedor se basa en los Modelos Ocultos de Markov, con el uso de coeficientes cepstrales para la extracción de características, y se procesan secuencias de dígitos y oraciones. En todo caso, el reconocimiento en secuencias de dígitos, afín a un grupo de las señales usadas en el presente estudio, nunca es inferior al 95 %.

2.5. Tratamiento de la señal

Se denomina *ruido* a un conjunto de perturbaciones aleatorias en una señal, de diverso origen. Por ejemplo, la actividad eléctrica externa a una línea de transmisión puede incidir sobre la señal que viaja por ésta. Existe otro tipo de ruido, independiente de la actividad externa, y siempre presente en las líneas de transmisión que operan en temperaturas superiores al cero absoluto: el *ruido térmico*. Su origen en la agitación térmica de los electrones en cada átomo de la línea de transmisión [11]. Consiste de componentes aleatorios de frecuencia, y también es conocido como *ruido blanco*.

La dificultad con el ruido es que puede entorpecer la identificación, por lo que se requiere preprocesar la señal con el fin de restaurarla. La Figura 2.9 ilustra el tratamiento de la señal.

El modelo supuesto para una señal con ruido tiene la forma

$$S_v^R(n) = S_v(n) + e(n)$$

$S_v^R(n)$ es la señal verbal con ruido, y $e(n)$ es un ruido blanco Gaussiano N(0,1). Existen diversas técnicas para intentar suprimir el ruido de la señal. Sin embargo, aquí se apelará a los wavelets, por cuanto S_v exhibe pocos cambios abruptos, recomendable para esta técnica [29], y porque investigaciones recientes demuestran que el uso de wavelets mejora notablemente la calidad de la señal verbal [2, 3, 7]. Otros estudios recomiendan que siempre debería procurarse la restauración de la señal verbal, por cuanto los efectos del ruido pueden resultar muy drásticos en los reconocedores [6].

Un wavelet es una forma ondular cuya duración se encuentra perfectamente delimitada en el tiempo, y que presenta un valor promedio de cero [29]. En el análisis de Fourier se descompone una señal en sinusoidales de diversas frecuencias. Mientras

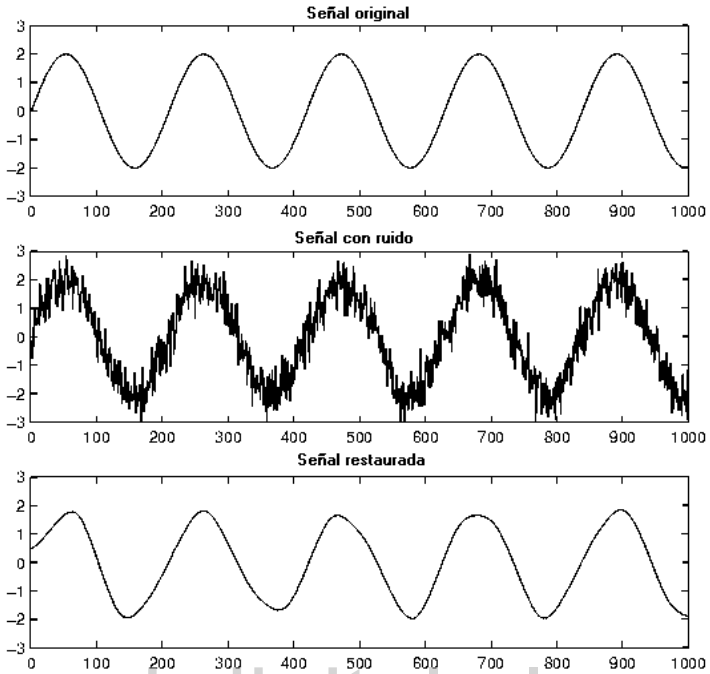

Fig. 2.9: Tratamiento del ruido.

las sinusoides del análisis de Fourier son predecibles y *suaves*, los wavelets tienden a ser irregulares y asimétricos. El análisis por wavelets consiste en descomponer una señal en versiones escaladas y desplazadas del wavelet original. Intuitivamente, las señales con cambios agudos deberían ser mejor analizadas con wavelets irregulares que con las sinusoides suaves del análisis de Fourier.

2.6. Métricas de reconocimiento

Un aspecto crítico de toda investigación de este tipo es la evaluación de la efectividad del sistema de identificación. En este caso, la efectividad depende de los errores en el reconocimiento, en cada uno de los dos niveles de identificación (letras y dígitos). El proceso es el siguiente: en primer lugar se crea un corpus de entrenamiento C_E, con señales, y en segundo lugar, un corpus de prueba, C_P, que contiene señales completamente distintas a las de C_E [12]. Luego, uno de los aspectos preliminares en la construcción de un sistema de identificación es la constitución de los corpus.

Un primer paso, como medida de seguridad, consiste en tomar una muestra de

C_E, y probar el sistema con eso. Es de esperar que el rendimiento con estas muestras sea superior al obtenido con C_P. Este paso se hace para verificar posibles errores en la implementación. El paso siguiente consiste en procesar C_P. Se utiliza una métrica simple y directa, en la cual, para cada unidad de reconocimiento, se contabilizan las clasificaciones correctas e incorrectas. Esta información suele presentarse bajo la forma de una matriz de confusión, la cual contiene información sobre las clasificaciones reales (correctas) y las efectuadas por el clasificador [17]. Ejemplificando con las vocales, una matriz de confusión típica[8] tiene la forma:

Matriz de Confusión Bidimensional		Categorías Predichas				
		a	e	i	o	u
Categorías Correctas	a	aa	ae	...		
	e	ea	ee	...		
	i	...	...	...		
	o					
	u					

Fig. 2.10: Matriz de confusión bidimensional.

Así, *aa* indica la cantidad de señales *a* que el clasificador reconoció. *ae* y *ea* indican las *a* que fueron clasificadas como *e*, y las *e* que fueron clasificadas como a, respectivamente. La matriz, además de indicar cuántas clasificaciones son válidas, permite observar entre cuáles categorías hay mayores problemas en la clasificación. Finalmente, las tasas de reconocimiento para cada categoría simplemente corresponden al porcentaje de señales correctamente clasificado.

En los experimentos a realizar posteriormente, se considerará que la técnica de análisis aplicada evidencia *capacidad discriminante* si se alcanzan las tasas de los antecedentes, o si se obtiene al menos un 50 % de efectividad en el reconocimiento. En la mejor de las circunstancias, los valores en la diagonal principal de la matriz de confusión asociada a un experimento deben ser los más altos en su respectiva fila, o cuando menos, debe manifestarse esa tendencia.

[8] La nomenclatura original se refiere a las categorías como negativos y positivos, restringida a matrices bidimensionales.

3. ESTRUCTURA DEL SISTEMA DE IDENTIFICACIÓN

3.1. Generalidades

En este contexto, *estructura* comprende el conjunto de etapas o actividades requeridas para la formación y operación de los clasificadores de señales verbales. Como ilustra la Figura 3.1, el sistema de identificación se ha organizado secuencialmente.

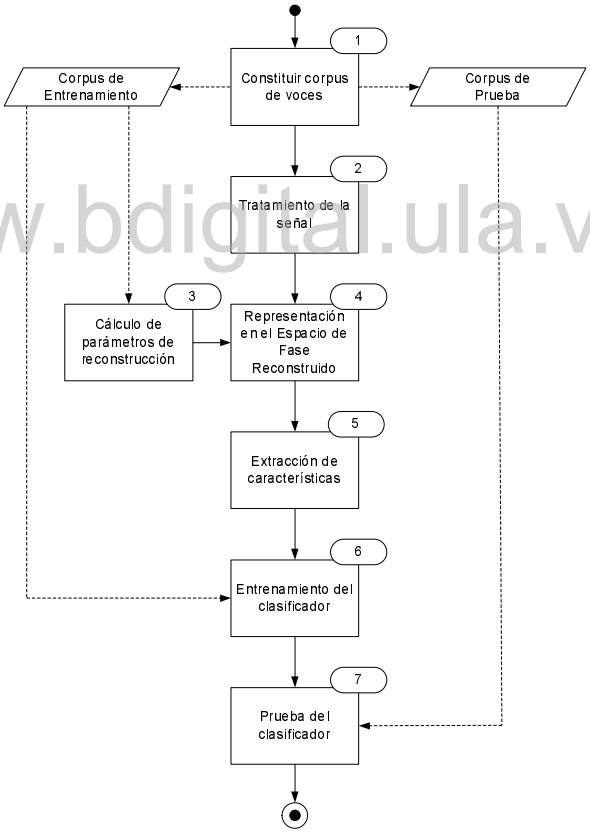

Fig. 3.1: Estructura del Sistema de Identificación.

La primera etapa consiste en definir los corpus de voces a utilizar para el entrenamiento y prueba del clasificador. Una segunda etapa, opcional, corresponde al preprocesamiento con wavelets de las señales en los corpus, con el propósito de reducir el ruido presente en las mismas. Por su parte, la tercera etapa concierne en específico al cálculo de los parámetros a emplear en la etapa siguiente: la reconstrucción del espacio de fase. La antepenúltima etapa es la obtención de vectores de características, donde se procede a cuantificar los *rasgos distintivos* del espacio de fase. Las dos últimas etapas abordan el entrenamiento y prueba del clasificador.

En esta investigación se desarrollan experimentos de reconocimiento dependientes e independientes del hablante, con vocales y dígitos, ensayando con diversos algoritmos de caracterización del Espacio de Fase Reconstruido. Luego, a cada experimento corresponde una instancia de la estructura sistémica descrita en el párrafo precedente. En otras palabras, cada experimento amerita el seguimiento de las etapas en la Figura 3.1. Los Capítulos 4 y 6 describen en detalle los experimentos y los algoritmos de caracterización, respectivamente.

Ahora resta discurrir sobre la naturaleza del clasificador del sistema, el cual está basado en redes neuronales. Tal es el propósito de la siguiente sección, que además incluye una somera revisión teórica de este modelo matemático. En alguna medida, la elección de las redes neuronales resulta circunstancial, no mandatoria. Pero se han preferido por su relativa sencillez, y su efectividad comprobada en el reconocimiento de patrones.

3.2. Revisión de Redes Neuronales

Las redes neuronales constituyen un modelo matemático, de inspiración biológica, compuesto por unidades de cálculo y conexiones entre ellas, alegóricas de las neuronas y del mecanismo sináptico, respectivamente [13]. Cada conexión recibe un peso numérico, el cual constituye el principal recurso de memoria a largo plazo en las redes neuronales, y el aprendizaje usualmente se realiza mediante la actualización de dichos pesos. Ciertas unidades reciben los estímulos del ambiente, y en consecuencia, se designan como **unidades de entrada**. Por su parte, las **unidades de salida** comunican el resultado de la operación de la red al mundo externo. La Figura 3.2 muestra la estructura de una sencilla red perceptrónica multicapa, con 4 unidades de entrada, 2 ocultas, y una de salida.

Formalmente, la salida de cada neurona a_i se computa mediante la expresión

$$a_i = f\left(\sum_j W_{j,i} a_j\right) \tag{3.1}$$

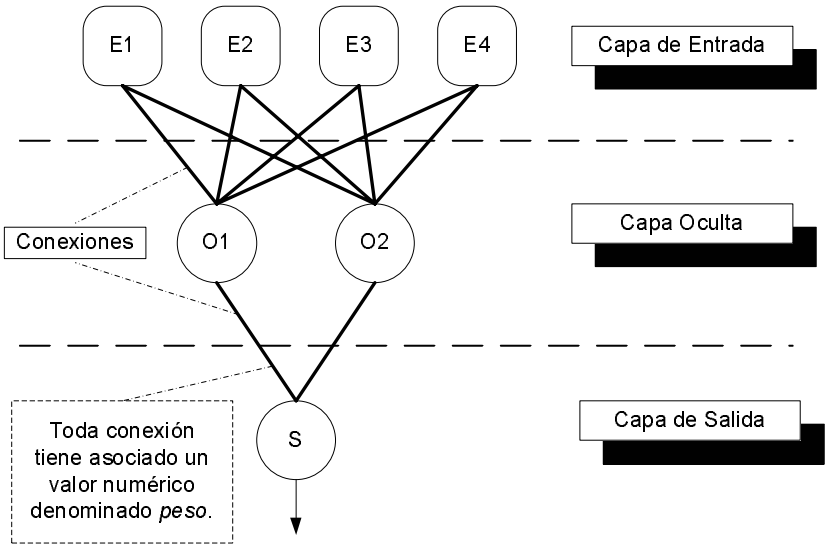

Fig. 3.2: Red perceptrónica multicapa.

donde f es la función de activación[1], $W_{j,i}$ denota el peso de la conexión entre la neurona j y la i, y a_j es la salida o vector de activación de la neurona j. Otra de las características de las redes neuronales es que el aprendizaje suele llevarse a cabo a través de sesiones de entrenamiento o épocas, donde los pesos se alteran para intentar alcanzar la relación entrada-salida exigida por el ambiente. En general, dicho entrenamiento consiste en un ajuste de pesos que persigue minimizar el error entre la salida producida por la red, y la salida deseada. De esta forma, la red generaliza sobre pares $(Entrada, SalidaDeseada)$, o en otras palabras, *aprende* a mapear *Entrada* en *SalidaDeseada*.

Las redes neuronales usadas en el clasificador son perceptrónicas multicapa, y constan de 3 capas. La cantidad de unidades en la entrada dependerá del tamaño del vector de características. Luego, si dicho vector incorpora n componentes, entonces se dispondrá de n unidades de entrada. Por ejemplo, si se particiona el espacio de fase en 100 bloques (consultar Capítulo 6), y se calcula la densidad de puntos en cada uno, el vector de características, y por ende la capa de entrada de la red, constará de 100 elementos. Por su parte, en la capa oculta se ubican 5 unidades[2], y en la capa

[1] Las más populares son *escalón*, *signo* y *sigmoide*.

[2] Aunque tales pruebas no se incluyen en este documento, empíricamente se verificó que aumentar la cantidad de neuronas en la capa oculta sólo contribuye a incrementar el tiempo de entrenamiento, mas no la efectividad de la identificación. Además, como se verá en el Capítulo 7, 5 neuronas resultan suficientes para alcanzar el error mínimo deseado.

de salida se coloca una sola unidad.

Como la capa de salida consta sólo de una unidad, resulta factible emplear el algoritmo de Levenberg-Marquardt [45] para el entrenamiento de la red, ya implementado en MATLAB. Éste es un algoritmo avanzado para la optimización no lineal, y suele converger al mínimo error más rápidamente que la retropropagación clásica, aunque su consumo de memoria resulta notoriamente elevado. El algoritmo parte del supuesto de linealidad de la función modelada, y en consecuencia, puede determinar su mínimo en un único paso. Esta suposición, cerca de un mínimo, resulta ventajosa, aunque no en puntos más lejanos. Si el nuevo punto disminuye el error, continúa a partir de él; en caso contrario apela al descenso por gradiente. Las sucesivas iteraciones proceden ajustadas a este compromiso entre la hipótesis de linealidad y el descenso por gradiente. En términos prácticos, relativos a la implementación en MATLAB, inicialmente el error objetivo se ha establecido en 10^{-3}, con sigmoides logarítmicas como funciones de activación. Esta es la configuración original, pero por motivos que se verán posteriormente, el curso de los experimentos obligó a alterarla. Con el resto de parámetros concernientes al algoritmo de Levenberg-Marquardt se han utilizado los valores por omisión que establece MATLAB R12.1.

3.3. Funcionamiento del clasificador

Sea $n = cardinalidad(\aleph)$ la cantidad de categorías a reconocer[3]. Luego, para cada experimento, el clasificador consiste en un arreglo $[R_1 R_2 ... R_n]$ de n redes neuronales R_i. Una vez definido el corpus C_E, puede procederse con la sesión de entrenamiento del clasificador con el algoritmo 1. Básicamente, R_i $(1 \leq i \leq n)$ se entrena con todas las señales j $(1 \leq j \leq cardinalidad(C_E))$ pertenecientes a C_E. A todas las entradas de entrenamiento en las que se verifique categoría(j) = categoría(i), se les asocia una salida igual a 1; en el otro caso, la salida es 0.

La ecuación 3.1 presentaba la forma de computar la salida de una neurona cualquiera. Antes de proseguir, conviene definir la función *salida* de una red neuronal R_i como la salida de la única neurona en la capa de salida de R_i. Es decir,

$$salida\,(R_i, Entrada) = a_{salida}$$

una vez que *Entrada* se propaga desde las neuronas de la capa de entrada hasta la de salida.

Posteriormente, al momento de clasificar una señal S_v^E, ésta se caracteriza y el vector resultante se administra a cada una de las n redes. La red con la salida más alta determina la categoría en la que se clasifica la señal. El algoritmo 2 formaliza

[3] En el caso de las vocales, $n = 5$. Con los dígitos, $n = 10$.

Algoritmo 1 Entrenamiento de un clasificador

`funcion` **entrena**($[R_1 R_2 ... R_n]$, C_E)

$\forall R_i (1 \leq i \leq n)$
  $\forall C_E[j]$ $(1 \leq j \leq cardinalidad(C_E))$
    si categoría($C_E[j]$) = categoría(R_i)
      SalidaDeseada := 1
    sino
      SalidaDeseada := 0
    fin-si
    Aplicar Levenberg-Marquardt a R_i con el siguiente
    par entrada-salida: (características($C_E[j]$), SalidaDeseada)

estos pasos. Allí, el valor de la variable max_i al término de la corrida determina la categoría en la que se clasifica S_v^E.

Algoritmo 2 Operación de un clasificador

`funcion` **clasifica**(S_v^E) retorna max_i

max := salida(R_1, características(S_v^E))
max_i := 1
$\forall R_i (2 \leq i \leq n)$
  temp := salida(R_i, características(S_v^E))
  si temp >max
    max := temp
    max_i := i
  fin-si

4. CONFORMACIÓN DE LOS CORPUS DE VOCES

Una de las primeras etapas en la construcción de un sistema de identificación consiste en definir los conjuntos o corpus de voces que se emplearán para el entrenamiento y prueba del clasificador. En la presente investigación, el corpus de entrenamiento, C_E, comprende aquellas señales de audio de las cuales se extraerán las características para el entrenamiento de la red neuronal. Por su parte, en el corpus de prueba, C_P, se encuentran las señales que permitirán verificar el comportamiento del sistema ante entradas no vistas. En líneas generales, la idea es que, a partir de las observaciones en C_E, la red pueda abstraer las características que definen a cada patrón, de tal forma que clasifique adecuadamente las señales conocidas y también las desconocidas. Dicho proceder corresponde a un esquema *orientado a datos*[1], en el cual se recurre al análisis de diversos ejemplares con el fin de generalizar las regularidades que relacionan a unos con otros y los ubican en determinadas categorías.

Esta actividad de conformación de corpus debe abordarse desde dos perspectivas: experimentos dependientes e independientes del hablante.

4.1. Conformación de corpus para experimentos dependientes del hablante

Recordemos que las vocales y dígitos constituyen las dos clases de señales abordadas en el presente trabajo. Por ende, se definen los corpus para los experimentos dependientes de dos hablantes: *hblA* (locutor masculino) y *hblB* (locutor femenino), por separado.

- C^V_{E-hblA}: Corpus de entrenamiento para vocales de *hblA* $(15 \times 5 = 75)$.
- C^V_{P-hblA}: Corpus de prueba para vocales de *hblA* $(10 \times 5 = 50)$.
- C^D_{E-hblA}: Corpus de entrenamiento para dígitos de *hblA* $(15 \times 10 = 150)$.
- C^D_{P-hblA}: Corpus de prueba para dígitos de *hblA* $(10 \times 10 = 100)$.

[1] La otra alternativa, *orientado a reglas*, suele resultar menos propensa a errores, siempre y cuando las reglas reflejen fielmente la realidad del dominio. El problema en el caso de los sistemas de identificación verbal es que no ha sido posible determinar dichas reglas, por la complejidad de los sistemas biológicos subyacentes.

- C^V_{E-hblB}: Corpus de entrenamiento para vocales de *hblB* ($15 \times 5 = 75$).
- C^V_{P-hblB}: Corpus de prueba para vocales de *hblB* ($10 \times 5 = 50$).
- C^D_{E-hblB}: Corpus de entrenamiento para dígitos de *hblB* ($15 \times 10 = 150$).
- C^D_{P-hblB}: Corpus de prueba para dígitos de *hblB* ($10 \times 10 = 100$).

Entre paréntesis aparece la cardinalidad de cada corpus. Por ejemplo, C^V_{E-hblA} incluye 75 señales de entrenamiento, 15 por cada una de las 5 categorías. Todas las señales fueron grabadas bajo PCM lineal, a 16 kHz, con un factor de cuantización de 16 bits.

4.2. Conformación de corpus para experimentos independientes del hablante

4.2.1. Revisión del SpeechDat Venezolano

En este caso, C_E y C_P se extraen de un conglomerado de voces de mayor extensión y riqueza fonética: el SpeechDat Venezolano, el cual es una base de datos exclusivamente constituida por pronunciaciones de hablantes venezolanos. Los hablantes registrados en el SpeechDat se encuentran geográficamente dispersos, por lo que la base de datos se enriquece con los diversos estilos de pronunciación en el país. Para este estudio, tal propiedad significa que la generalización de características resulta más difícil, pero al mismo tiempo, el logro de dicha generalización debería posibilitar un mejor rendimiento ante entradas no vistas, pues se estarían procesando las *mínimas* informaciones dinámicas que permiten distinguir la pronunciación de un "3" como tal, por ejemplo.

Las actividades de diseño del corpus y de recolección de las voces en el SpeechDat Venezolano fueron ejecutadas en la Universidad de los Andes[2], Mérida, Venezuela [30]. La base de datos comprende 44000 registros de voz, de 1000 hablantes. Por cada hablante se grabaron, a través de la línea telefónica, 44 tipos de pronunciaciones, con una velocidad de muestreo de 8kHz, y factor de cuantización de 8 bits. Información más abundante puede encontrarse en [30]. El esquema de codificación empleado en el SpeechDat es el μ-law [51], común en las aplicaciones telefónicas. Dicho esquema es del tipo no lineal, y aplica compresión logarítmica ajustada a la sensibilidad del oído humano.

4.2.2. Definición de C_E y C_P

De los 44 tipos en el SpeechDat Venezolano, para esta investigación resultan de interés sólo los concernientes a vocales y dígitos. Afortunadamente, se dispone de

[2] Con el patrocinio y soporte técnico de la Universidad Politécnica de Cataluña, España.

un tipo de pronunciación correspondiente a los dígitos[3]. Sin embargo, no hay un tipo asociado a vocales aisladas, por lo que ha resultado necesaria la edición manual de grabaciones donde los hablantes deletrean algunas palabras[4], para extraer, uno a uno, los fragmentos de señal correspondientes a vocales. Luego, estas vocales son del tipo *libres de contexto*, por cuanto la señal no se encuentra afectada por las pronunciaciones de fonemas circundantes. De esta forma, el análisis del Espacio de Fase Reconstruido se referiría en concreto a la dinámica de la producción de vocales, sin el efecto de otras articulaciones. Por el contrario, los dígitos representan señales de superior complejidad, que involucran diversos fonemas, cuya duración y tono no es consistente entre los diversos hablantes. Así, es de esperar que el análisis de los dígitos confronte mayores dificultades que el de las vocales.

Resta por definir cómo se eligen las señales que integrarán definitivamente los corpus, y más importante aún, el tamaño de éstos. La primera decisión se resuelve trivialmente, seleccionando las señales al azar. La segunda requiere un poco más de consideración, pues el tamaño de los corpus resulta importante para el clasificador: con mayor cantidad de muestras la red neuronal generalizaría mejor. Empero, tamaños muy altos incrementarían el tiempo de entrenamiento de la red, y sobre todo, los requisitos de memoria, exigiendo así demasiados recursos de cómputo. Por otra parte, los antecedentes no son generosos con el tamaño de los corpus, ni con la cantidad de hablantes en los mismos. Aquí resulta importante señalar una diferencia con los trabajos predecesores: las señales en los corpus de esta investigación son de hablantes distintos, es decir, en un corpus dado no habrá dos o más pronunciaciones de un mismo hablante. Los corpus también incluyen pronunciaciones tanto de hablantes masculinos como femeninos. Esto deviene en más variedad en las señales de los corpus, y por consiguiente, se dispone de más información sobre la dinámica del sistema fonatorio.

Concretando, con las vocales el corpus de entrenamiento consiste de 20 señales, y el de prueba, de 10, por cada vocal. Se utiliza también un corpus de prueba de distinta naturaleza, C_{PP} conformado por 10 pronunciaciones no pertenecientes al SpeechDat Venezolano, recolectadas por el autor, a fin de comprobar el rendimiento del sistema de identificación con señales de otra variedad. Con los dígitos, por la complejidad de éstos, C_E y C_P se aumentan en 10 señales, en relación con las vocales. Luego, según como está orientado el trabajo, en total se opera con 6 corpus:

- C_E^V: Corpus de entrenamiento para vocales ($20 \times 5 = 100$).
- C_P^V: Corpus de prueba para vocales ($10 \times 5 = 50$).
- C_{PP}^V: Segundo corpus de prueba (no SpeechDat) para vocales ($10 \times 5 = 50$).

[3] En la estructura de SpeechDat, se trata del corpus con identificación *I1*.
[4] Con identificadores *L1*, *L2* y *L3*.

- C_E^D: Corpus de entrenamiento para dígitos ($30 \times 10 = 300$).
- C_P^D: Corpus de prueba para dígitos ($20 \times 10 = 200$).
- C_{PP}^D: Segundo corpus de prueba (no SpeechDat) para dígitos ($10 \times 10 = 100$).

4.2.3. Codificación de los Corpus de Voces

Finalmente, las señales en los corpus se han transformado de μ-law (8 kHz, 8 bits) a PCM lineal (16 kHz, 16 bits). De esta manera, cada señal se constituye en una secuencia de muestras u observaciones con espaciado constante, necesario para la reconstrucción del espacio de fase. Por otro lado, el incremento de la velocidad de muestreo y del factor de cuantización permite disponer de más observaciones para un mejor análisis. Por último, en el caso del corpus C_{PP}, las señales se han grabado directamente con los parámetros referidos, a través de un micrófono.

En la práctica, todos los archivos de audio en los corpus se editaron, incluyendo los correspondientes a los dígitos, con la finalidad de suprimir los silencios en los extremos, y para retirar otro tipo de señales sin importancia para el estudio[5].

[5] Por ejemplo, algunos hablantes, en las grabaciones de dígitos, han interpuesto pronunciaciones descartables como en "Número nueve" y "Este...seis", donde no interesan los fragmentos de señal correspondientes a "Número" y a "Este...".

5. RECONSTRUCCIÓN DEL ESPACIO DE FASE

Este capítulo concierne al cómputo de los parámetros para la reconstrucción del espacio de fase, sobre la base de las variaciones de la entropía diferencial en la representación de la señal verbal en dicho espacio. El cálculo se aplica sólo sobre las vocales porque los dígitos son más complejos en términos de la dinámica sistémica subyacente. Recuérdese que en particular la dimensión de inmersión se refiere a la cantidad de variables de estado requeridas para generar la señal. Evidentemente, en los dígitos dicho valor debe ser alto por la cantidad de componentes articulatorios que participan en la pronunciación. Así, dimensiones muy elevadas complican en demasía el análisis, por la enorme cantidad de datos, y en consecuencia, la representación de la señal completa de un dígito en el Espacio de Fase Reconstruido resulta prohibitiva en términos de tiempo y potencia de cómputo. Además, resultados preliminares arrojan un tiempo de aproximadamente 2 horas en el cálculo de los parámetros para una sola señal de dígito, lo cual constituye una evidencia empírica contra el uso del método con estas señales. Como se verá en el cápítulo siguiente, la caracterización de los dígitos apela a una métrica que evita este problema. Por el contrario, con las vocales los costos no resultan tan elevados, al ser éstas señales más cortas y simples. Además, el hecho de que las vocales sean fonemas sonoros implica que no resulta necesario analizar toda la señal. Así, en unas 1000 muestras (62.5 ms) de cada señal ya deben estar presentes las características dinámicas que distinguen una vocal de otra. De esta forma, al trabajar con menos datos, se reduce el tiempo requerido para calcular los parámetros.

5.1. Método basado en Entropía Diferencial para el cálculo de parámetros del Espacio de Fase Reconstruido

Se trata de un método novel para determinar m y τ en una representación del espacio de fase a partir de una serie de tiempo, propuesto por Gautama, Mandic y Van Hulle [8]. Con este propósito, se recurre a un único criterio: *la razón de entropía* entre la representación de una señal en el espacio de fase, y un grupo de datos

sustitutos[1], derivados de la señal.

Los métodos convencionales computan m y τ por separado. Así, primero se determina el retraso τ como el mínimo de la información mutual entre muestras distanciadas por τ; la justificación del cálculo reside en la búsqueda de ejes independientes en $\Re^2$. Una vez obtenido el τ óptimo, se determina la dimensión de inmersión como aquella en la cual la cantidad de falsos vecinos más cercanos es pequeña. Este último cómputo verifica que la estructura topológica de la señal en $\Re^m$ se preserve, lo cual implica una fuerte dependencia entre las dimensiones en el espacio de fase, contradiciendo de esta manera el supuesto (de independencia) en el cálculo de τ.

Por el contrario, el método basado en entropía diferencial unifica el cálculo de ambos parámetros, suprimiendo también las inconsistencias. En primer lugar, se cuantifica la *cantidad de desorden*, con base en la función de densidad de probabilidad de los datos $p(x)$, según la entropía diferencial:

$$H(x) = -\int_{-\infty}^{+\infty} p(x) \ln\left(p(x)\right) dx \tag{5.1}$$

En [8], no se trabaja directamente con la ecuación 5.1, sino con la estimación de Kozachenko-Leonenko:

$$H(x) = \sum_{j=1}^{N} \ln\left(N p_j\right) + \ln 2 + C_E \tag{5.2}$$

donde N es el número de muestras en la serie, p_j es la distancia euclidiana del j-ésimo vector con retraso a su vecino más cercano, y C_E es la constante de Euler ($\approx$ 0,5572). Luego, $H(x, m, \tau)$ denota la entropía diferencial computada para una reconstrucción del espacio de fase correspondiente a la señal x, con parámetros m y τ. De esta forma, los valores óptimos para los parámetros de reconstrucción son aquellos que minimizan $H(x)$.

Para dotar de robustez al cálculo, $H(x, m, \tau)$ se estandariza con respecto a un grupo de datos sustitutos obtenidos a partir de la permutación aleatoria de las observaciones en x. Se obtienen así las nuevas series $x_{s,i}$, $i = 1, 2, ..., N_s$. Finalmente, la minimización, en vez de proceder sobre $H(x)$, aplica sobre $I(m, \tau)$, donde:

$$I(m, \tau) = \frac{H(x, m, \tau)}{\left\langle H(x_{s,i}, m, \tau)\right\rangle_i} \tag{5.3}$$

y $\langle\cdot\rangle_i$ representa el promedio con todos los i ($1 \leq i \leq N_s$).

[1] En inglés, *surrogate data*. Son datos generados artificialmente, que preservan las propiedades estadísticas de los datos reales.

5.2. Algoritmos para el cálculo de parámetros

Los autores del método proporcionan una librería en MATLAB que simplifica notoriamente el cálculo de los parámetros [8]. Las funciones `calcula_mtau` y `mtau`, en el Capítulo 9 (páginas 77 y 78 respectivamente), se apoyan en rutinas incluidas en dicha librería, y permiten computar m y τ para la señal proporcionada.

5.3. Parámetros para las vocales

Los algoritmos se aplican sobre el corpus independiente del hablante, de mayor variedad[2], y por ende, más proclive a capturar la dinámica del sistema. Por otra parte, debe ejecutarse sólo sobre el corpus de entrenamiento C_E^V, porque en un ambiente real no pueden conocerse, a priori, los parámetros de las señales de entrada al sistema de identificación (eg., aquellas en C_P^V).

La ejecución del algoritmo sobre 90 señales de C_E^V (18 por cada vocal) arroja los siguientes resultados para la dimensión de inmersión, presentados en forma tabular (Tabla 5.1) y gráfica (Figuras 5.1 y 5.2).

Vocal	m=2	m=3	m=4	m=5	m=6	m=7	m=8	m=9
a	4	3	6	4	1	0	0	0
e	6	6	3	3	0	0	0	0
i	9	1	7	1	0	0	0	0
o	3	7	6	2	0	0	0	0
u	7	7	2	2	0	0	0	0
	29	24	24	12	1	0	0	0

Tab. 5.1: Valores obtenidos por el Método de Mínima Entropía Diferencial para la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V.

Considerando los valores totales para cada $m\,(2 \leq m \leq 9)$, se tiene:

- **Media** = 3.2444
- **Moda** = 2
- **Desviación Estándar** = 1.0842

La mejor medida para decidirse por un valor de m es la moda, por cuanto sugiere que la mayoría de las señales pueden caracterizarse naturalmente en el plano. En [21]

[2] Esta *variedad* se justifica porque cada señal en dichos corpus fue generada por una instancia de sistema fonatorio (el aparato fonador del hablante).

también se prefiere la moda por un razonamiento similar. Así, se fija $m = 2$ para los experimentos subsiguientes.

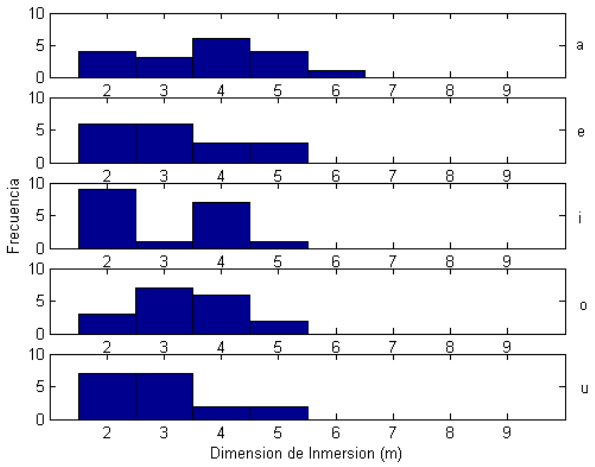

Fig. 5.1: Histogramas de la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V.

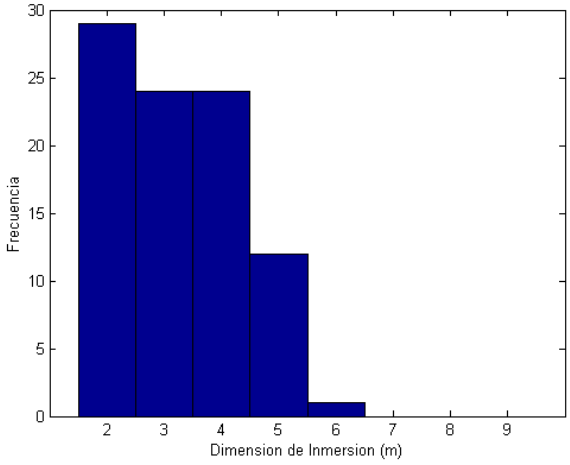

Fig. 5.2: Histograma combinado para la dimensión de inmersión (m) en la reconstrucción de las señales en C_E^V.

En cuanto al retraso, se obtuvo:

Vocal	τ=1	τ=2	τ=3	τ=4	τ=5	τ=6	τ=7	τ=8	τ=9	τ=10	τ=11	τ=12
a	3	1	3	2	0	1	0	2	3	1	2	0
e	3	0	3	0	3	1	0	1	1	0	2	4
i	6	1	1	0	2	1	1	3	1	0	2	0
o	6	1	0	1	0	0	1	2	2	0	1	4
u	4	1	0	0	0	0	0	1	3	3	2	4
	22	4	7	3	5	3	2	9	10	4	9	12

Tab. 5.2: Valores obtenidos por el Método de Mínima Entropía Diferencial para el retraso (τ) en la reconstrucción de las señales en C_E^V.

Nótese que se ha probado con un rango más amplio para $\tau\,(1 \leq \tau \leq 12)$. Las medidas de tendencia central son:

- **Media** = 7.5000
- **Moda** = 1
- **Desviación Estándar** = 5.5841

De nuevo, utilizando la moda, se establece $\tau = 1$. Coincidencialmente, es el mismo valor usado por Takens [47] en su estudio. Luego, definitivamente, los valores a usar son $m = 2$ y $\tau = 1$. Es posible que con más muestras los valores tiendan a aumentar y reflejar el carácter multidimensional del aparato fonador[3]. Sin embargo, para las tareas de clasificación de vocales (es decir, con sólo 5 categorías), estos valores en los parámetros deberían resultar suficientes.

[3] Por ejemplo, obsérvese en la Figura 5.4 como $\tau = 12$ secunda en frecuencia a $\tau = 1$.

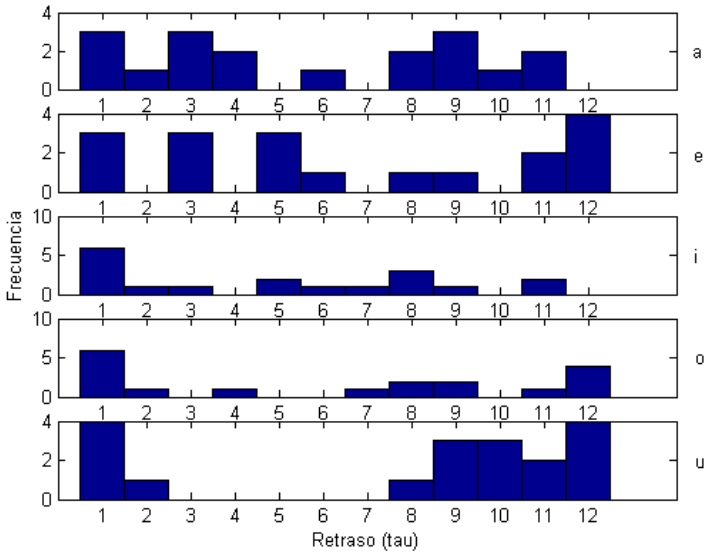

Fig. 5.3: Histogramas del retraso (τ) en la reconstrucción de las señales en C_E^V.

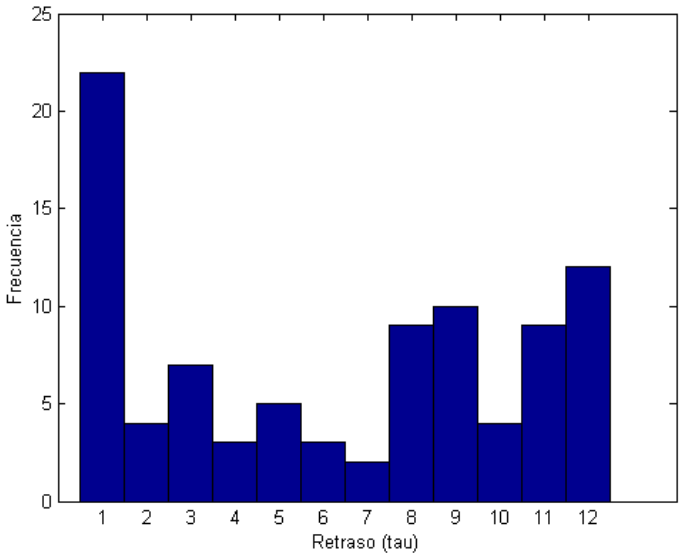

Fig. 5.4: Histograma combinado del retraso (τ) en la reconstrucción de las señales en C_E^V.

6. ANÁLISIS DE SEÑALES EN EL ESPACIO DE FASE RECONSTRUIDO

Una vez definidos los corpus de voces, se procede a especificar las métricas o algoritmos de caracterización de las señales en el Espacio de Fase Reconstruido. En otras palabras, se trata de extraer un vector $V_C = [C_1 C_2 ... C_n]$ de características para asociar S_v con alguna categoría en $\aleph$. Debido a las cualidades particulares de cada categoría, los algoritmos presentan diferencias importantes según se trabaje con vocales o dígitos. No obstante, independientemente del tipo, a toda señal se le aplica previamente una función de normalización, con el fin de minimizar el efecto perturbador de las inconsistencias en la intensidad entre señales de los corpus. Las secciones siguientes profundizan en estos detalles.

6.1. Primera Normalización

Considerando que la dimensión de inmersión para las vocales se ha establecido en 2 ($m = 2$), la señal puede analizarse en el plano. Luego, la normalización constituye un proceso bastante sencillo, con dos propósitos:

1. **Centrar la señal**: Se deriva una secuencia con media 0 y desviación estándar 1. Para ello, se reemplazan los valores originales (muestras) de S_v mediante la asignación
$$S_v = \frac{S_v - \langle S_v \rangle}{desviacionEstandar(S_v)}$$
donde $\langle S_v \rangle$ denota el promedio de las muestras.

2. **Restringir los valores de amplitud al intervalo** $[-1, +1]$: Para ello, simplemente se dividen todas las muestras que conforman a S_v entre la muestra con el mayor valor absoluto.

6.2. Análisis de Vocales

En este caso, $\aleph = \{a, e, i, o, u\}$. Por cuanto la normalización restringe la amplitud de la señal al intervalo $[-1, +1]$, los ejes de abcisas y ordenadas pueden dividirse en r intervalos, definidos por los $r + 1$ puntos $-1 + \frac{2}{r} * i$, $0 \leq i \leq r$. La intersección de

estos intervalos sobre el plano define r^2 *bloques*, como ilustra la Figura 6.1, donde $r = 10$. En lo sucesivo, V_C^V denota el vector de características para vocales.

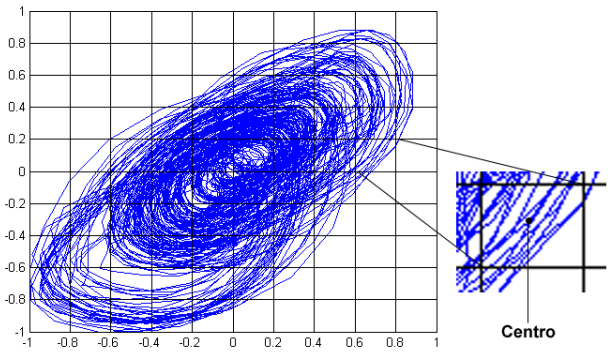

Fig. 6.1: Definición de bloques en el Espacio de Fase Reconstruido.

Los antecedentes demuestran la factibilidad de numerosas técnicas para identificar las regiones del plano hacia donde tienden las trayectorias o donde los puntos del atractor se agrupan. En este estudio se prefiere reducir el problema a la caracterización de los bloques, mediante las alternativas de *densidad espacial* y/o *compactación*. Para tales efectos, sea B_i un bloque cualquiera sobre el plano, y sea $P_k(x_k, y_k)$ un punto del atractor. Evidentemente, un bloque conglomera cero o más de estos puntos P_k. Luego, las métricas a estudiar se definen:

1. **Densidad Espacial de** B_i. Básicamente, consiste en determinar la proporción de puntos contenidos en B_i:

$$densidadEspacial(B_i) = \frac{cardinalidad\left(\{P_k / P_k \in B_i\}\right)}{L(S_v)}$$

2. **Compactación promedio de** B_i. Es una medida de lo dispersos que se encuentran los puntos del atractor en B_i. Sea P_C^i el punto centro de B_i. Luego, la compactación viene dada por

$$compactacion(B_i) = \sum_k distancia(P_k, P_C^i) \quad \forall P_k \in B_i$$

Una vez completados los cálculos, conviene normalizar dividiendo todas las compactaciones entre la mayor de ellas, con la finalidad de trabajar con menores magnitudes.

3. **Aproximación híbrida**. En búsqueda de una mayor robustez, se combinan los dos enfoques anteriores.

Cualquiera de las métricas anteriores se denotará mediante la función $C(S_v)$. Resulta interesante apreciar que con las dos primeras métricas, el vector de características generado por C posee r^2 elementos. Por su parte, con el enfoque híbrido se tendrían $2r^2$ componentes.

Adicionalmente, dependiendo del caso, pudiera incorporarse al vector de características la *rapidez* de variación en la señal, mediante una aproximación con las primeras diferencias. También puede hacerse lo propio con la *aceleración*, a través de segundas diferencias. En ambas aproximaciones, se calculan las diferencias, y se reconstruye y caracteriza el espacio de fase de cada una de estas nuevas secuencias de datos. Con más precisión, tanto la métrica de densidad como la de compactación se abordan desde tres niveles. En cada nivel, V_C^V posee una configuración distinta:

- **Nivel 0**: V_C^V sólo consta de los r^2 elementos derivados de la métrica de análisis aplicada. Formalmente,
$$V_C^V = [C(S_v)] \ .$$
- **Nivel 1**: Además de los elementos del nivel anterior, V_C^V se agranda con otros r^2 elementos más, provenientes de aplicar la métrica de análisis sobre $y_1(n) = S_v(n) - S_v(n-1)$. Así, el total de componentes es $2r^2$:
$$V_C^V = [C(S_v) \ C(y_1)] \ .$$
- **Nivel 2**: Además de los elementos del nivel anterior, V_C^V se agranda con otros r^2 elementos más, provenientes de aplicar la métrica de análisis sobre $y_2(n) = y_1(n) - y_1(n-1)$. De esta forma, V_C consta de $3r^2$ elementos:
$$V_C^V = [C(S_v) \ C(y_1) \ C(y_2)] \ .$$

La Figura 6.1, presentada anteriormente, es la reconstrucción de una vocal *a*, con los bloques. Resulta útil apreciar dos instancias más de señales de esta vocal (Figuras 6.2 y 6.3), a fin de adquirir una noción sobre las posibilidades de éxito de las métricas descritas. Todas las reconstrucciones de esta sección se obtuvieron fijando $\tau = 3$, para *abrir* el atractor, y así resultasen más evidentes las propiedades gráficas del espacio.

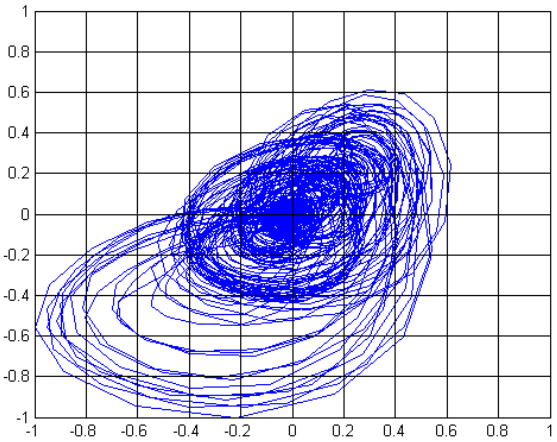

Fig. 6.2: Segunda instancia de vocal *a*.

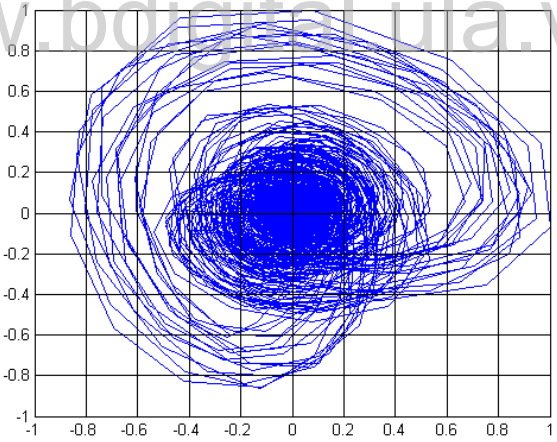

Fig. 6.3: Tercera instancia de vocal *a*.

La variabilidad entre los bloques de las Figuras 6.1, 6.2, y 6.3 es alta, aunque ciertamente los patrones exhiben alguna regularidad difícil de precisar en palabras. Como constraste, se muestran a continuación tres instancias de señales *u*.

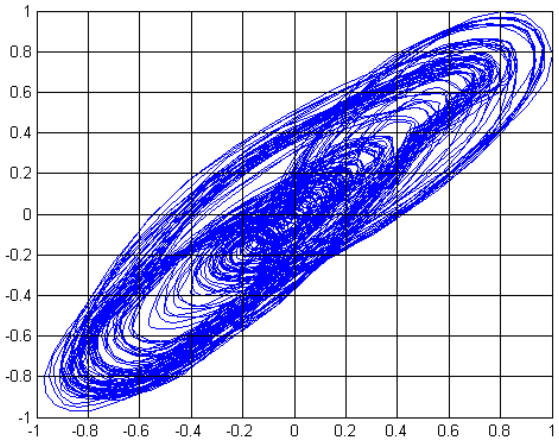

Fig. 6.4: Primera instancia de vocal *u*.

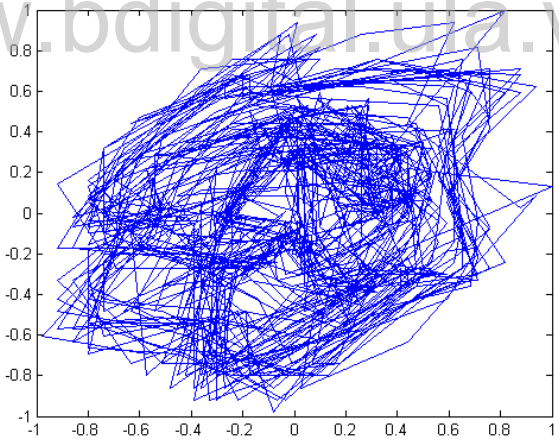

Fig. 6.5: Segunda instancia de vocal *u*.

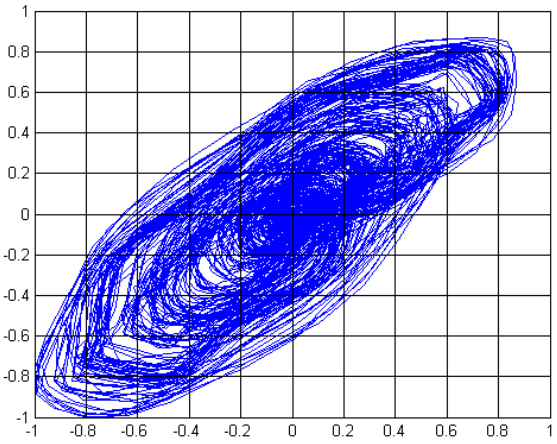

Fig. 6.6: Tercera instancia de vocal *u*.

Nuevamente, hay alguna similitud entre señales de la misma clase, y más importante aún, se aprecian diferencias con las señales de la clase *a*. Empero, la Figura 6.5 es una excepción porque el atractor se encuentra muy deformado, debido a la elección $\tau = 3$. Aunque en los experimentos se utilizará el valor computado por entropía diferencial, $\tau = 1$, la Figura 6.5 es una muestra del compromiso que significa la elección del parámetro de retraso[1].

6.3. Análisis de Dígitos

Aquí $\aleph = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Los dígitos destacan por su longitud y complejidad, en relación con las vocales. En consecuencia, vista la variabilidad que experimentan señales tan simples como las vocales, resulta ingenuo suponer que la representación de una señal de dígito completa en el espacio de fase bidimensional bastará para tareas de clasificación. Una reducción de dimensionalidad tampoco es adecuada porque, en definitiva, estas señales no exhiben tendencias claras en el espacio de fase que permitan, sobre el plano, distinguir entre diez categorías. Además, se presenta el problema de la inevitable pérdida de datos acarreada por la reducción; ya el antecedente [53] confirma que la reducción no aporta mucho. Como opción, podría abordarse la caracterización directamente en el espacio multidimensional, pero tal

[1] Compromiso, porque si τ es muy pequeño el atractor no se *abrirá* y tenderá a parecerse a una recta afín sobre el plano. Por el contrario, valores apenas altos *deforman* el atractor, como es el caso ya visto en la figura.

labor consumiría mucho tiempo y cómputo, y ciertamente requiere técnicas analíticas especializadas.

En el resto de la sección se asume que S_v corresponde a la pronunciación de algún dígito. La alternativa seguida aquí consiste en aplicar un análisis similar al de las vocales[2] sobre ns segmentos de la señal S_v. La longitud de cada segmento es $le = \lfloor L(S_v)/ns \rfloor$. Todo segmento $SS_v[s]$ $(1 \leq s \leq ns)$ inicia en la muestra $1 + le \times (s-1)$ de S_v. Luego, para la caracterización se computa la desviación estándar de la función de análisis sobre cada bloque: $desviacionEstandar(C(SS_v[s]))$. Es decir, se obtiene una medida de la dispersión de los puntos sobre el Espacio de Fase Reconstruido, en cada segmento.

También para mayor robustez, el vector de características de dígitos, V_C^D, se aumenta con la distribución energética en la señal, resultante de aplicar a ésta el Operador Nolineal Discreto de Energía de Teager [15, 16]:

$$\Psi_D[S_v(t)] = S_v(t)^2 - S_v(t+1)S_v(t-1) \ . \tag{6.1}$$

Este operador cuantifica la energía en el sistema que generó la señal, en vez de la energía de la señal en sí misma. Es de esperar que la distribución de la energía presente similitudes entre pronunciaciones de un mismo dígito, y diferencias con las de otros dígitos. Sin embargo, un problema corriente con este operador es que Ψ_D puede ser negativo si $S_v(t)^2 < S_v(t+1)S_v(t-1)$, lo cual resulta intolerable para una señal de energía [19]. Para enfrentar esta dificultad, $\Psi_D[S_v]$ se pasa por un filtro de mediana[3] de orden 80. La salida de dicho filtrado puede contener algunas irregularidades en la forma de la señal, por lo que se aplica un segundo filtrado, de alisamiento, con un filtro de Savitzky-Golay[4], de orden 3, y longitud de trama igual a 199.

La secuencia resultante del segundo filtrado se referirá como SG. Los datos resultantes del filtrado también deben normalizarse, pero en relación a la longitud. Esto es así porque la cantidad de neuronas de entrada en el clasificador será fija para todas las señales, y sin embargo, muy probablemente $L(S_v)$ será distinto para cada señal a considerar, tanto de entrenamiento, como de prueba. La idea es normalizar el SG de cada señal de dígito a la misma longitud L_f, para que el vector de características tenga siempre un tamaño fijo, independiente de la longitud de la señal. Por ende, el problema consiste en, dada una secuencia de datos X de longitud $L(X)$,

[2] La función de análisis C puede tomar muchas formas, todas presentadas en la sección precedente. La decisión de cuál métrica usar se posterga hasta la obtención de los resultados con vocales, en el capítulo siguiente. Teóricamente, cualquier elección es plausible.

[3] Un filtro de mediana es un filtro no lineal que para cada muestra de señal genera como salida la mediana de las muestras en un entorno predefinido y centrado en la muestra bajo consideración. Se emplea para eliminar muestras ruidosas de tipo impulsivo [46].

[4] Se trata de un filtro pasabajos para el alisamiento de datos, en el dominio del tiempo [33].

se desea obtener una nueva secuencia Y de L_f muestras que conserve la distribución de los datos. Con este fin, se introduce la función $transforma(X, L_f)$, la cual reemplaza $X(1), X(2), ..., X(L(X))$ por la serie $Y(1), Y(2), ..., Y(L_f)$, mediante las Ecuaciones 6.2 y 6.3.

$$Y(j) = X(i) + [X(i+1) - X(i)]\,(r_j - i) \tag{6.2}$$

$$r_j = \left(\frac{(j-1)(L(X)-1)}{L_f - 1}\right) + 1 \tag{6.3}$$

con i igual a la parte entera de r_j. La anterior es una normalización que simplemente selecciona valores regularmente espaciados de la señal original. Finalmente:

$$\begin{aligned} V_C^{Seg} &= [desviacionEstandar(C(SS_v[s]))], \forall s\,(1 \leq s \leq ns) \\ V_C^D &= [transforma(V_C^{Seg}, L_{f1})\ \ transforma(SG, L_{f2})] \end{aligned} \tag{6.4}$$

La Figura 6.7 ilustra parcialmente el proceso de parametrización de un dígito, con la función C definida como la densidad espacial de nivel 0. Allí, **A** es la señal original. **B** y **C** son $transforma(V_C^{Seg}, 100)$ y $transforma(SG, 100)$, respectivamente. Por último, V_C^D corresponde a **D**.

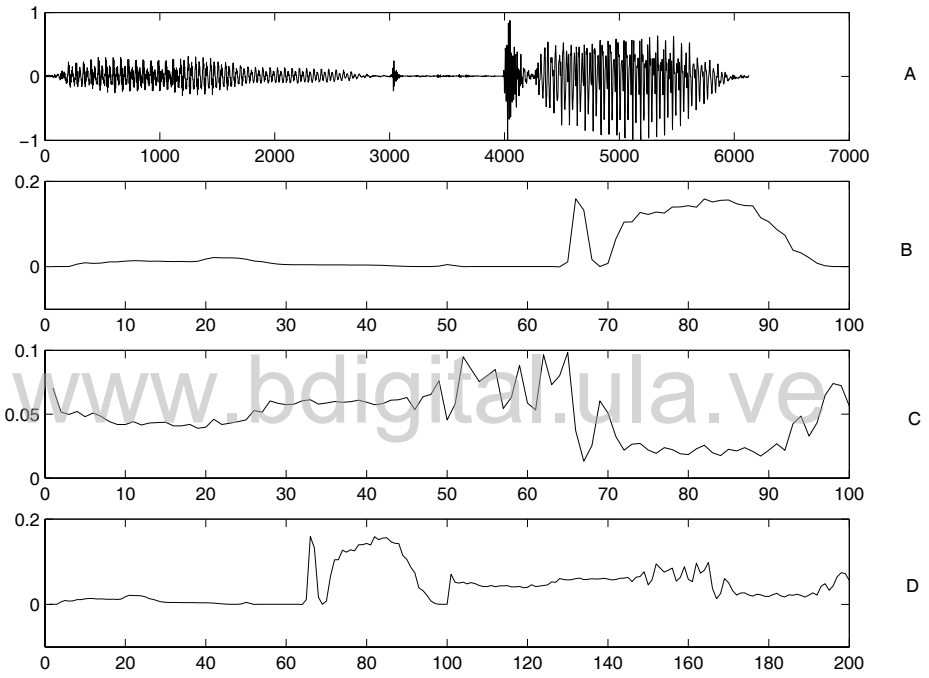

Fig. 6.7: Parametrización de un dígito.

7. EVALUACIÓN DE RESULTADOS

Este capítulo refleja el comportamiento de los clasificadores al presentarles las señales de los corpus de entrenamiento y prueba. En primer lugar se tabulan los resultados con vocales, y luego con los dígitos, ambos obtenidos a partir de ensayos dependientes e independientes del hablante. A medida que se recopilaron resultados, las conclusiones derivadas de los mismos influyeron en los experimentos subsiguientes.

Todos los algoritmos y métricas discutidos con anterioridad se han codificado en MATLAB. El Capítulo 9 contiene las rutinas desarrolladas. Allí, resulta de interés el script `dinam.m` (página 73), el cual ofrece una interfaz para invocar a las otras funciones, según el experimento que se desee realizar.

7.1. Vocales

Con anterioridad, la sección 2.6 sugería la revisión del funcionamiento de los clasificadores proporcionando a éstos, como entrada, aquellas señales con las que fueron entrenados. En el caso particular de las vocales dependientes del hablante, se reconoció correctamente el 100 % de las señales de entrenamiento, y por tal razón, se omiten las tablas con las tasas de C_E. Ahora resta presentar la secuencia de matrices de confusión para cada prueba realizada. Con estas matrices se ha tomado la licencia de incluir, en cada fila, datos del entrenamiento (épocas y error alcanzado) de la red neuronal responsable de dicha categoría.

7.1.1. Reconocimiento dependiente del hablante

A continuación, los resultados para *hblA* y *hblB*. Primeramente se muestran las tasas con la métrica de densidad espacial, y luego con la compactación. En ambas ocasiones, con los tres niveles discutidos en la sección 6.2. Para capturar con mayor detalle el comportamiento de los datos en el centro del atractor, se ha establecido el parámetro r en 10 (ie., $r = 10$) en todos los experimentos, y así se obtienen 100 bloques sobre el plano.

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	9	1	0	0	0	90.00	7	0.000436809
e	0	7	0	3	0	70.00	26	0.000201969
i	0	4	6	0	0	60.00	9	0.000322007
o	1	0	0	8	1	80.00	11	0.000199754
u	0	0	0	1	9	90.00	13	0.000176901
						78.00		

Tab. 7.1: Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 0).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	8	0	0	2	0	80.00	7	0.000535334
e	0	10	0	0	0	100.00	23	0.000931215
i	0	0	10	0	0	100.00	9	0.000140074
o	1	0	0	9	0	90.00	8	0.000805474
u	0	0	1	0	9	90.00	5	0.000658593
						92.00		

Tab. 7.2: Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 0).

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	6	0	0	4	0	60.00	9	0.000402987
e	0	9	0	0	1	90.00	14	0.000376099
i	0	0	10	0	0	100.00	6	0.000417917
o	1	0	0	9	0	90.00	10	0.000186717
u	0	0	0	0	10	100.00	7	0.000158674
						88.00		

Tab. 7.3: Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 1).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	10	0	0	0	0	100.00	7	0.000242645
e	1	9	0	0	0	90.00	20	0.000173305
i	0	1	9	0	0	90.00	6	0.000423575
o	0	0	0	10	0	100.00	8	0.000551849
u	0	0	0	0	10	100.00	8	0.000775742
						96.00		

Tab. 7.4: Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 1).

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	6	0	0	4	0	60.00	8	0.000143502
e	0	7	0	3	0	70.00	17	0.000287976
i	0	1	9	0	0	90.00	6	0.000911167
o	1	0	0	9	0	90.00	9	0.000687481
u	0	0	0	0	10	100.00	7	0.000269995
						82.00		

Tab. 7.5: Tasas de reconocimiento para vocales de *hblA* con densidades espaciales (nivel 2).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	10	0	0	0	0	100.00	22	0.000863145
e	0	10	0	0	0	100.00	12	0.000265555
i	0	0	10	0	0	100.00	7	0.000819627
o	2	0	0	8	0	80.00	9	0.000316724
u	0	0	0	0	10	100.00	8	0.000254685
						96.00		

Tab. 7.6: Tasas de reconocimiento para vocales de *hblB* con densidades espaciales (nivel 2).

Ahora se presentan los resultados para la compactación, de nuevo, con los tres niveles. Se mantiene $r = 10$.

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	7	2	0	1	0	70.00	9	0.00064613
e	0	6	3	1	0	60.00	8	0.000771797
i	1	4	3	2	0	30.00	9	0.000264206
o	0	0	0	6	4	60.00	8	0.000254413
u	0	0	0	0	10	100.00	6	0.000433465
						64.00		

Tab. 7.7: Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 0).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	5	4	1	0	0	50.00	8	0.000147638
e	2	4	3	0	1	40.00	6	0.000530496
i	0	7	1	0	2	10.00	15	0.000156377
o	0	0	0	10	0	100.00	7	0.000162701
u	0	0	0	0	10	100.00	6	0.000284934
						60.00		

Tab. 7.8: Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 0).

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	7	0	0	3	0	70.00	7	0.000153674
e	0	6	2	2	0	60.00	6	0.000640895
i	2	1	5	2	0	50.00	7	0.000152382
o	0	0	0	10	0	100.00	8	0.000223563
u	0	0	0	1	9	90.00	6	0.000185586
						74.00		

Tab. 7.9: Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 1).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	6	1	2	0	1	60.00	7	0.000244675
e	4	3	2	0	1	30.00	6	0.000272305
i	1	7	2	0	0	20.00	9	0.000334984
o	0	0	0	10	0	100.00	5	0.000726076
u	0	0	0	0	10	100.00	7	0.000797912
						62.00		

Tab. 7.10: Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 1).

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	8	0	0	2	0	80.00	6	0.000552657
e	1	5	3	0	1	50.00	18	0.000285992
i	1	3	6	0	0	60.00	6	0.000751317
o	1	0	0	9	0	90.00	6	0.000398114
u	0	0	0	0	10	100.00	7	0.000339169
						76.00		

Tab. 7.11: Tasas de reconocimiento para vocales de *hblA* con compactación (nivel 2).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	9	1	0	0	0	90.00	9	0.000034909
e	5	3	2	0	0	30.00	6	0.000385412
i	2	1	7	0	0	70.00	8	0.000263323
o	0	0	0	10	0	100.00	7	0.000155770
u	0	0	0	0	10	100.00	7	0.000333417
						78.00		

Tab. 7.12: Tasas de reconocimiento para vocales de *hblB* con compactación (nivel 2).

Las tasas promedio evidencian capacidad discriminante con vocales dependientes del hablante. La Figura 7.1 presenta los resultados en forma gráfica.

Los mejores resultados se logran con el nivel 1, y en general, con las señales de *hblB*. El nivel 2 no aporta suficiente información, y de hecho, en el experimento de las densidades espaciales con *hblA*, los resultados desmejoran en relación con el nivel 1. Además, recuérdese que en el nivel 2 se trabaja con un vector de características de $3 \times r^2 = 3 \times 100 = 300$ elementos, cantidad muy elevada para la poca efectividad

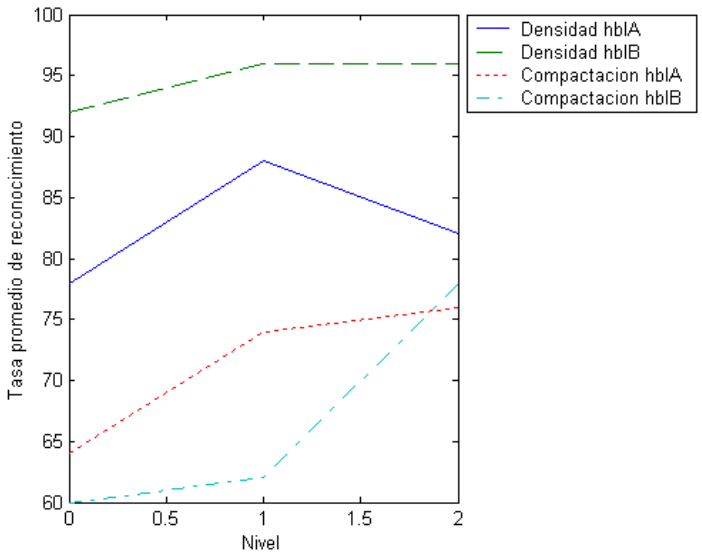

Fig. 7.1: Tasas de reconocimiento para vocales.

obtenida. Por su parte, el nivel 0 es bueno, considerando el pequeño tamaño del vector (100 elementos), pero decididamente el nivel 1 ofrece mejores tasas. En vista, pues, de estos resultados, se ha decidido emplear el nivel 1 en los restantes experimentos con vocales.

Ahora se procede a determinar la efectividad de la aproximación híbrida. En esta oportunidad, el vector de características alcanza los 400 elementos, disgregados así: 200 pertenecen al nivel 1 del análisis por densidades espaciales, y los restantes al nivel 1 de la compactación. Luego, los resultados de tal combinación son:

hblA	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	7	2	0	1	0	70.00	7	0.000342154
e	1	7	1	1	0	70.00	14	0.000254758
i	0	1	7	0	2	70.00	8	0.000582853
o	2	2	0	6	0	60.00	7	0.000611060
u	0	0	1	3	6	60.00	12	0.000222670
						66.00		

Tab. 7.13: Tasas de reconocimiento para vocales de *hblA* con aproximación híbrida (nivel 1).

hblB	**a**	**e**	**i**	**o**	**u**	%	Épocas	Error
a	7	2	0	1	0	70.00	6	0.000134627
e	0	8	1	1	0	80.00	8	0.000150303
i	1	3	4	2	0	40.00	7	0.000572765
o	0	0	0	8	2	80.00	12	0.000235261
u	0	0	0	1	9	90.00	9	0.000411279
						70.00		

Tab. 7.14: Tasas de reconocimiento para vocales de *hblB* con aproximación híbrida (nivel 1).

En verdad, las tasas promedio de 66,00 % y 70,00 % son muy pobres, en comparación con los resultados precedentes, incluso a pesar del superior tamaño de V_C^V. Así, este enfoque resulta plenamente descartable, por su costo computacional, y por la disposición de un enfoque más efectivo, como lo es el análisis por densidades espaciales de nivel 1.

7.1.2. Reconocimiento independiente del hablante

Preservando la consistencia con las conclusiones extraídas hasta este punto, en los ensayos independientes del hablante se utiliza el nivel 1 en las densidades espaciales. De este modo, las Tablas 7.15, 7.16 y 7.17 exhiben el rendimiento del clasificador entrenado con C_E^V, y probado con C_P^V y C_{PP}^V. Primero se verifica el clasificador, procesando las señales de entrenamiento:

	a	**e**	**i**	**o**	**u**	%	Épocas	Error
a	20	0	0	0	0	100.00	12	0.000266666
e	0	20	0	0	0	100.00	17	0.000187407
i	0	0	20	0	0	100.00	15	0.000432659
o	0	0	0	20	0	100.00	15	0.000286546
u	0	0	0	1	19	90.00	29	1.000000000
						98.00		

Tab. 7.15: Tasas de reconocimiento para vocales independientes del hablante en C_E^V.

Aunque no se ha satisfecho el error en el caso de la vocal **u**, la tasa del 98,00 % es aceptable. Con los corpus de prueba, se obtiene:

	a	**e**	**i**	**o**	**u**	%
a	8	1	1	0	0	80.00
e	0	6	3	1	0	60.00
i	0	2	7	0	1	70.00
o	0	0	0	9	1	90.00
u	0	1	1	6	2	20.00
						64.00

Tab. 7.16: Tasas de reconocimiento para vocales independientes del hablante en C_P^V.

hblB	**a**	**e**	**i**	**o**	**u**	%
a	4	2	3	1	0	40.00
e	0	4	5	1	0	40.00
i	0	0	10	0	0	100.00
o	0	3	0	7	0	70.00
u	0	1	1	8	0	0.00
						50.00

Tab. 7.17: Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^V.

Como era de esperarse, las tasas promedio resultan inferiores a las de los ensayos dependientes del hablante. Esto es así porque los corpus presentan mayor variedad entre señales, pues cada pronunciación pertenece a un hablante distinto. Empero, satisfactoriamente se ha logrado superar al antecedente [54], que obtiene apenas un 33,00 % con vocales[1]. Y los resultados son similares a los de [22], a pesar de que allí se emplea una técnica mucho más complicada que el análisis por densidad.

Un detalle preocupante en la Tabla 7.17 es el fracaso de la clasificación de la vocal **u**, reconociendo las señales, en su mayoría, como instancias de **o**. Presumiblemente, tal desempeño se debe a la imposibilidad de la red neuronal de la categoría **u** para lograr la cota de error durante el entrenamiento. En búsqueda de alternativas, se cambió la función de transferencia de la neurona en la capa de salida a una función de transferencia lineal[2]. Afortunadamente, sesiones de entrenamiento preliminares confirmaron la efectividad del cambio, por lo que se aprovechó para exigir

[1] En un corpus de tamaño desconocido.

[2] En términos de MATLAB, se reemplazó la función `logsig` por `purelin`. Uno de los problemas con las redes neuronales es la dificultad para definir con anticipación la mejor topología para abordar un problema, lo cual con frecuencia amerita estos *ensayos* que conduzcan a una configuración adecuada.

un error mucho menor de 10^{-5}. En consecuencia, todos los experimentos en el resto del capítulo emplean esta nueva configuración.

Los resultados obtenidos ahora para C_E^V, C_P^V y C_{PP}^V son:

	a	**e**	**i**	**o**	**u**	%	Épocas	Error
a	20	0	0	0	0	100.00	32	7.16434e-006
e	0	20	0	0	0	100.00	310	9.97692e-006
i	0	0	20	0	0	100.00	77	9.14957e-006
o	0	0	0	20	0	100.00	38	9.5894e-006
u	0	0	0	0	20	100.00	119	9.91752e-006
						100.00		

Tab. 7.18: Tasas de reconocimiento para vocales independientes del hablante en C_E^V, con nueva función de activación.

	a	**e**	**i**	**o**	**u**	%
a	6	0	0	4	0	60.00
e	0	4	3	3	0	40.00
i	0	3	4	1	2	40.00
o	1	0	0	7	2	70.00
u	0	1	0	2	7	70.00
						56.00

Tab. 7.19: Tasas de reconocimiento para vocales independientes del hablante en C_P^V, con nueva función de activación.

	a	**e**	**i**	**o**	**u**	%
a	2	2	2	4	0	20.00
e	0	4	6	0	0	40.00
i	0	1	8	1	0	80.00
o	1	2	0	7	0	70.00
u	0	0	3	1	6	60.00
						54.00

Tab. 7.20: Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^V, con nueva función de activación.

Aunque se han logrado los errores objetivo, y los promedios de C_P^V y C_{PP}^V son más

parejos entre sí, no se observan mejoras significativas en la clasificación. De hecho, las tasas disminuyen un poco. No obstante, estos resultados bastan para confirmar las observaciones anteriores sobre la capacidad discriminante del método.

Ya para cerrar la sección, se compara con la identificación en el espacio de información difuso [40, 41]. Allí, los reconocimientos son del 100 %, pero la diversidad de hablantes es muy baja (sólo dos locutores). Por ende, el estudio es más análogo a los ensayos anteriores, dependientes del hablante. En tal caso, las mejores tasas promedio de 88,00 % (*hblA*) y 96,00 % (*hblB*) se encuentran cercanas al 100,00 %.

7.2. Dígitos

La atención se centra ahora sobre estas señales, notoriamente más complejas que las vocales, por cuanto constan de diversos fonemas, y sus realizaciones acústicas exhiben duraciones irregulares incluso en las pronunciaciones de un mismo hablante. Recuérdese, de la sección 6.3, que V_C^D requiere definir los parámetros ns, L_{f1} y L_{f2}. Luego, arbitrariamente se fijaron $ns = 100$ y $L_{f1} = L_{f2} = 100$. Y la función de análisis C es por densidades espaciales, de nivel 0. Como se analizan 100 segmentos, los datos obtenidos con el nivel 0 en C deberían resultar suficientes. De esta forma, $L(V_C^D) = 200$. El tamaño del vector de características es similar al de las vocales con densidad espacial y nivel 1, aunque los dígitos son señales que constan de muchas más muestras.

Las dos secciones siguientes contienen los resultados para los experimentos dependientes e independientes del hablante.

7.2.1. Reconocimiento dependiente del hablante

Las pruebas de verificación con los corpus de entrenamiento de *hblA* y *hblB* arrojaron una efectividad del 100 %. En cuanto a los corpus de prueba se han obtenido los resultados recolectados en las Tablas 7.21 y 7.22, presentadas en la página siguiente por razones de espacio.

Ciertamente, los resultados son inferiores a los de las vocales, pero resultan justificables dada la complejidad de la señal tratada. De cualquier manera, tasas de 68,00 % y 84,00 % permiten afirmar que existe capacidad discriminante.

hblA	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%	Épocas	Error
0	5	0	0	0	0	0	3	1	0	1	60.00	32	8.22635e-006
1	0	9	0	1	0	0	0	0	0	0	90.00	29	9.74978e-006
2	0	0	2	5	1	0	2	0	0	0	20.00	22	2.12476e-006
3	0	0	0	8	0	1	0	1	0	0	80.00	13	6.13903e-006
4	0	1	0	0	8	0	0	0	1	0	80.00	219	8.54561e-006
5	1	0	0	0	0	7	0	0	2	0	70.00	13	8.34605e-006
6	1	0	0	2	0	0	7	0	0	0	70.00	59	8.83234e-006
7	1	0	1	1	1	1	1	4	0	0	40.00	233	9.94429e-006
8	0	1	0	0	0	0	0	0	9	0	90.00	10	4.81487e-006
9	0	0	0	1	0	0	0	0	0	9	90.00	111	9.88215e-006
											68.00		

Tab. 7.21: Tasas de reconocimiento dígitos de *hblA*.

hblB	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%	Épocas	Error
0	10	0	0	0	0	0	0	0	0	0	100.00	41	8.16896e-006
1	0	6	0	0	0	1	0	0	2	1	60.00	10	3.08538e-006
2	0	0	7	0	1	0	0	0	0	2	70.00	145	9.76913e-006
3	0	0	0	8	0	0	0	0	0	2	80.00	132	9.4419e-006
4	0	0	0	0	9	0	0	0	1	0	90.00	15	4.10166e-006
5	0	0	0	0	0	10	0	0	0	0	100.00	12	7.35731e-006
6	0	0	0	2	0	0	8	0	0	0	80.00	12	4.60862e-006
7	0	0	0	0	0	0	0	10	0	0	100.00	21	9.99481e-006
8	0	1	0	0	3	0	0	0	6	0	60.00	8	5.65039e-006
9	0	0	0	0	0	0	0	0	0	10	100.00	14	5.07524e-006
											84.00		

Tab. 7.22: Tasas de reconocimiento dígitos de *hblB*.

7.2.2. *Reconocimiento independiente del hablante*

Nuevamente, se obtiene un 100 % en el reconocimiento de las señales de entrenamiento en C_E^D. Las Tablas 7.23 y 7.24 exhiben los resultados con experimentos independientes del hablante, en los corpus C_P^D y C_{PP}^D.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%	Épocas	Error
0	12	2	0	0	0	2	2	0	0	2	60.00	37	9.66003e-006
1	2	6	1	0	0	2	1	2	5	1	30.00	190	9.99604e-006
2	1	3	10	1	0	1	2	0	0	2	50.00	23	1.60161e-006
3	0	2	2	11	0	3	1	0	0	1	55.00	21	9.36203e-006
4	0	0	0	0	11	2	1	5	1	0	55.00	21	4.84047e-006
5	0	0	0	1	1	17	1	0	0	0	85.00	50	9.84132e-006
6	5	1	0	1	1	0	11	0	1	0	55.00	24	5.89059e-006
7	1	0	0	1	1	2	0	13	2	1	65.00	21	8.12499e-006
8	1	0	0	0	1	4	0	0	14	0	70.00	306	9.54569e-006
9	2	1	0	1	0	1	1	1	0	13	65.00	77	9.76257e-006
											59.00		

Tab. 7.23: Tasas de reconocimiento para dígitos independientes del hablante con C_P^D.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%
0	7	0	0	0	0	0	0	0	0	3	70.00
1	0	10	0	0	0	0	0	0	0	0	100.00
2	0	1	4	1	0	0	2	0	0	2	40.00
3	0	0	1	8	0	0	1	0	0	0	80.00
4	0	1	0	0	9	0	0	0	0	0	90.00
5	0	0	0	0	1	9	0	0	0	0	90.00
6	1	0	0	1	0	0	6	1	0	1	60.00
7	0	1	0	0	0	0	1	7	0	1	70.00
8	1	0	0	0	1	0	0	0	8	0	80.00
9	2	0	0	0	0	1	1	0	0	6	60.00
											74.00

Tab. 7.24: Tasas de reconocimiento para dígitos independientes del hablante con C_{PP}^D.

De forma interesante, las tasas se encuentran muy cercanas a las de la contraparte dependiente del hablante. Tampoco existen diferencias significativas entre los experimentos con C_P y C_{PP} (apenas un 15 % en promedio), a pesar del distinto origen de ambos corpus. Por último, resulta importante apreciar que en casi todas las matrices de confusión de los dígitos, los valores en la diagonal principal son los más altos de sus respectivas filas. La única excepción ocurre en la tercera fila de la Tabla 7.21. Esto es un reflejo de la tendencia que existe en el algoritmo de análisis hacia las clasificaciones correctas.

7.3. Efecto del ruido

Esta última sección pretende determinar si el ruido afecta el desempeño de los algoritmos utilizados en la clasificación. Para resolver esta cuestión, se repiten los experimentos independientes del hablante[3], con una mínima variante. Ahora el tamaño de los corpus de entrenamiento se ha duplicado para abarcar también las señales resultantes de preprocesar con wavelets[4] sus constituyentes o señales originales. Por ejemplo, C_E^V consta de 100 señales, por lo que al procesar cada una con wavelets obtendremos 100 nuevas señales, que aunadas a las originales, duplican el tamaño del corpus.

Naturalmente, al momento de las pruebas, cada señal de los corpus de prueba también es tratada con wavelets, aunque en este caso, no se dobla el tamaño de los corpus: simplemente se prueba con las señales originales preprocesadas. Las pruebas con los corpus de entrenamiento arrojaron un 100 %. Las tablas siguientes exhiben las otras tasas pertinentes.

	a	**e**	**i**	**o**	**u**	%	Épocas	Error
a	8	1	1	0	0	80.00	23	1.41314e-006
e	0	2	3	4	1	20.00	99	3.37866e-006
i	0	6	2	0	2	20.00	69	5.03754e-006
o	3	0	0	6	1	60.00	76	8.43010e-006
u	0	1	1	5	3	30.00	78	3.16749e-006
						42.00		

Tab. 7.25: Tasas de reconocimiento para vocales independientes del hablante en C_P^V tratadas con wavelets.

[3] Es decir, experimentos con vocales (C_P^V y C_{PP}^V) y dígitos (C_P^D y C_{PP}^V).

[4] Este preprocesamiento se reduce a la aplicación, sobre cada señal, de la función `wden` del toolbox de Wavelets de MATLAB. Esta función retorna una versión *sin ruido* de la señal provista, a partir de la descomposición de la misma en estructuras wavelets.

	a	**e**	**i**	**o**	**u**	%
a	3	2	4	1	0	30.00
e	4	0	4	0	2	0.00
i	0	0	2	0	8	20.00
o	1	2	4	0	3	0.00
u	0	0	1	0	9	90.00
						28.00

Tab. 7.26: Tasas de reconocimiento para vocales independientes del hablante en C_{PP}^{V} tratadas con wavelets.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%	Épocas	Error
0	13	0	1	3	1	0	0	0	0	2	65.00	21	7.09344e-006
1	0	3	2	2	3	3	1	0	5	1	15.00	82	7.54441e-006
2	1	0	12	1	0	0	3	0	1	2	60.00	131	9.99476e-006
3	1	0	0	14	0	1	0	0	0	4	70.00	32	9.93259e-006
4	0	1	1	2	10	4	0	0	2	0	50.00	12	8.21805e-006
5	0	0	0	1	1	12	1	1	1	3	60.00	25	1.78474e-006
6	1	1	1	2	1	0	11	0	2	1	55.00	28	4.02596e-006
7	0	0	0	0	1	3	1	10	2	3	50.00	21	9.63778e-006
8	0	2	0	0	7	3	0	0	8	0	40.00	14	8.09495e-006
9	1	1	0	1	0	2	0	0	0	15	75.00	19	2.42513e-006
											54.00		

Tab. 7.27: Tasas de reconocimiento para dígitos independientes del hablante con C_{P}^{D} tratados con wavelets.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	%
0	8	0	1	0	0	1	0	0	0	0	80.00
1	0	10	0	0	0	0	0	0	0	0	100.00
2	0	0	4	1	0	0	2	0	0	3	40.00
3	0	0	1	5	0	0	2	0	0	2	50.00
4	0	0	0	0	8	1	0	0	1	0	80.00
5	0	0	0	1	2	7	0	0	0	0	70.00
6	1	0	0	0	0	0	6	1	0	2	60.00
7	1	0	0	0	0	0	0	8	0	1	80.00
8	1	0	0	2	0	0	0	0	8	1	80.00
9	3	0	0	0	0	0	1	0	0	6	60.00
											70.00

Tab. 7.28: Tasas de reconocimiento para dígitos independientes del hablante con C_{PP}^{D} tratados con wavelets.

Sin tener que decirlo, la efectividad en la clasificación desmejora. Por ende, el uso de wavelets resulta improcedente en aras de incrementar las tasas de reconocimiento, específicamente con los algoritmos desarrollados en esta investigación.

8. CONCLUSIONES Y RECOMENDACIONES

La principal conclusión, justificable por las tasas de reconocimiento logradas, es que el análisis de la señal verbal en el Espacio de Fase Reconstruido proporciona información para distinguir entre algunas categorías de señales, en particular, vocales y dígitos de voces venezolanas. En el caso dependiente del hablante los resultados son aceptables, con tasas de hasta 88 % y 96 % en las vocales, y de 68 % y 84 % en los dígitos. Por el contrario, el objetivo de independencia del hablante requiere más información, presumiblemente bajo la forma de características extraídas del dominio de la frecuencia. No obstante, las tasas obtenidas en el caso independiente, siempre superiores al 50,00 %, para ambos tipos de señales, verifican positivamente la presencia de *información discriminante* en la reconstrucción de la dinámica de las pronunciaciones. Si no existiera tal capacidad discriminante, las matrices de confusión no mostraran la tendencia a colocar los valores más altos de cada fila en las celdas pertenecientes a la diagonal principal. Por otro lado, obsérvese que en general no se presentan diferencias significativas en el reconocimiento con C_P y C_{PP}, por lo que puede afirmarse que los algoritmos de análisis son robustos, independientes de la naturaleza de los corpus.

A continuación se discuten las métricas empleadas. Nótese que todas proceden en el dominio del tiempo, sin apelar en ningún momento a datos extraídos del dominio de la frecuencia. Concretando, la superioridad del análisis por densidades espaciales, en relación con la compactación, indica que para las tareas de clasificación resulta más importante la forma general del atractor, entendida como la distribución de sus puntos en el espacio, que las variaciones a nivel local representadas por las distancias entre puntos en trayectorias cercanas, específicamente, residentes en los subespacios denominados bloques. La escasa efectividad de la compactación se ha comprobado al integrarla con la densidad en el vector de características, circunstancia en la cual no han experimentado ninguna mejoría las tasas de reconocimiento. Ahora bien, pudiera resultar que la compactación se torne más efectiva en dimensiones superiores ($m > 2$), pero en todo caso, al involucrar cálculos de distancias, resulta una métrica computacionalmente costosa. Independientemente, en general los experimentos con vocales han superado las tasas exhibidas por los antecedentes.

Ahora se discuten los dígitos. La reconstrucción directa de estas señales en el plano se ha descartado anticipadamente porque dos dimensiones no bastan para

capturar toda la dinámica involucrada. Luego, la solución elegida, la más natural, consistió en un análisis por segmentos. Una complicación evidente, es que el análisis en el Espacio de Fase Reconstruido arroja al menos r^2 datos, y en consecuencia, si la cantidad de segmentos es muy alta, el vector de características resultaría impráctico para el entrenamiento del clasificador por la gran cantidad de datos. Esta es la razón de apelar a la desviación estándar de la secuencia de datos obtenida en cada segmento. Así, se reduce el tamaño del vector a justamente la cantidad de segmentos definidos. Y al mismo tiempo, se está computando la dispersión en las densidades de los bloques a lo largo de la señal. Empero, como se trata de una reducción de datos, se optó por complementar el vector con la distribución de energía de Teager, para disponer de mayor información sobre la *estructura* de la señal. Gracias a este procedimiento, se han obtenido tasas que sólo son superadas por las vocales en el caso dependiente del hablante, a pesar de la mayor complejidad de los dígitos. Y por otro lado, distinguir entre 10 categorías es más difícil que entre 5. Considérese también que quizás $m = 2$ no basta para capturar *completamente* la dinámica de segmentos de señales verbales sensibles al contexto. Con los resultados obtenidos, en presencia de estas limitantes, podemos concluir que el reconocimiento de dígitos resulta satisfactorio.

Por otra parte, no puede menospreciarse en este análisis la importancia sutil de la normalización. Todas las señales presentan variaciones de amplitud, que trasladadas al espacio de fase, introducen una variabilidad innecesaria y confusa, en cuanto a la distribución de los puntos, incluso entre señales de la misma categoría. Por tal razón, al normalizar, todas las señales se sitúan en el rango $[+1, -1]$, lo cual ha permitido la descomposición regular del plano en bloques. A su vez, la segunda normalización constituye una alternativa válida para reducir el tamaño del vector de características.

Un punto que intencionalmente se ha soslayado en el presente trabajo es la importancia en la naturaleza del clasificador. En ambientes más exigentes, como en el reconocimiento continuo, el simple arreglo de redes neuronales empleado aquí no bastará, porque hay que considerar muchos fonemas, el silencio, y los tiempos de entrenamiento y reconocimiento. Tampoco podrá aplicarse el enfoque adoptado con los dígitos, a menos que el reconocimiento deba proceder sobre un conjunto restringido de palabras aisladas, en cuyo caso, igualmente, habría que tomar consideraciones para el silencio, y para la detección de extremos, problema que acá hemos evitado mediante una ardua edición manual de todos los archivos de audio. Adicionalmente, se requiere la construcción de modelos linguísticos porque la audición humana se apoya en el contexto y el conocimiento sobre la formación de palabras. Es probable que la incorporación de tal información, y el uso de Modelos Ocultos de Markov, permitan obtener mejores resultados con los dígitos.

Antes de cerrar, se discuten otras tres posibilidades de investigación abiertas por este estudio:

1. La ventaja principal de las técnicas de análisis investigadas es que la extracción de características resulta muy sencilla, comparando con las técnicas del dominio de la frecuencia. El lado negativo es que el vector de características incluye una cantidad muy grande de elementos. Por consiguiente, debe indagarse sobre la manera de reducir la cantidad de datos. A modo de sugerencia, podría probarse con cuantización vectorial[1].

2. En principio, puede concluirse que el uso de wavelets no contribuye con los algoritmos de identificación. Sin embargo, recuérdese que en el fondo, se trata de identificar un proceso físico, como lo es la generación de voz. Al procesar una señal de los corpus con wavelets, sus propiedades acústicas se distorsionan, y el oído ya no percibe un sonido similar al original, aunque ciertamente, la *forma* de la señal, en cuanto a picos y suavidad en las transiciones, mejora. Esto sugiere cierta conexión entre las propiedades del dominio de la frecuencia y el análisis en el Espacio de Fase Reconstruido, que también pudiera constituir un tema posterior de estudio.

3. Por último, los anteriores razonamientos podrían emplearse como punto de partida para un problema inverso: la síntesis de voz. En este sentido, la idea sería construir un sintetizador comparando la salida de varios prototipos con una salida deseada, real, aplicando las métricas de análisis en el Espacio de Fase Reconstruido. Sucesivamente, se refinaría aquél prototipo más cercano al objetivo. Esta constituye una línea de investigación a seguir en el futuro.

Finalmente, la mayor dificultad reside en que, con el tipo de análisis desarrollado en esta investigación, lo que se ha hecho es transformar el problema de clasificación de señales de voz en uno de identificación de un proceso físico, a partir de diversos perfiles del mismo. Evidentemente, el sistema en cuestión, el aparato fonador humano, resulta excesivamente complejo para ser abordado por un pequeño grupo de algoritmos. Y por otra parte, el sistema auditivo, en el otro extremo, es más sensible a las fluctuaciones de frecuencia que a las de fase. En consecuencia, estas técnicas no pueden competir con las del dominio de la frecuencia en aplicaciones *reales*, al menos por el momento. Sin embargo, la dirección seguida resulta prometedora.

[1] La cuantización vectorial es una forma de codificación que persigue reducir la cantidad de bits necesaria para transmitir un mensaje [12]. En la práctica, se aplica una *medida de distorsión* a los vectores de entrada para compararlos con las entradas en un libro o arreglo de códigos. Posteriormente, el vector puede reemplazarse por el índice de aquella entrada del libro con la menor distorsión.

9. RUTINAS EN MATLAB DEL SISTEMA DE IDENTIFICACIÓN

```
%//////////////////////////////////////
%SCRIPT dinam
%\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
%clc;
fprintf('-- dinam 0.01\n');
fprintf('-- [Jose A. Brito B.] 15/03/2004\n\n');
fprintf('DESCRIPCION GENERAL:\n');
fprintf('Sistema para identificacion de señales verbales en el EFR.\n');
fprintf('Permite el calculo de parametros de reconstruccion (m y tau)\n');
fprintf('junto con la aplicacion de varios algoritmos de caracterizacion\n');
fprintf('de la señal. Adicionalmente, existe la posibilidad de cuantificar\n');
fprintf('la incidencia del ruido a traves del tratamiento de la señal\n');
fprintf('con wavelets y la sucesiva comparacion con los valores corrientes.\n\n');
fprintf('TIPO DE SEÑALES ACEPTADAS:\n');
fprintf('Por el momento, solo se aceptan archivos WAV (16bits, 16kHz)\n');
fprintf('correspodientes a vocales y digitos, en dos vertientes:\n');
fprintf('dependientes(hblA y hblB) e independientes del hablante.\n\n');
fprintf('CLASIFICADOR:\n');
fprintf('El clasificador a utilizar es un arreglo de redes neuronales. Consultar\n');
fprintf('el Capitulo III de la tesis para detalles sobre su funcionamiento.\n\n');
fprintf('NOTAS:\n');
fprintf('Los directorios y nombres de archivos deben delimitarse con comillas\n');
fprintf('simples (''). Los directorios deben terminar con backslash (\\).\n\n');

global DIRECTORIO_TRABAJO;  %Los scripts necesitan saber donde residen las señales
global ALGORITMO_DENSIDAD;  %Densidades[1] o Compactacion[i<>1]
global DIFERENCIAS;         %Ninguna[0], Primeras[1], Primeras y Segundas[2]
opcion = -1;                %Controlador de menu

while (opcion ~= 9)

while ~ismember(opcion,[1:9])
    fprintf('\n--------------------------------\n');
    fprintf('OPCIONES:\n');
    fprintf('--------------------------------\n');
```

```
    fprintf('1.- Calcular m y tau para vocales\n');
    fprintf('2.- Calcular m y tau para digitos\n');
    fprintf('3.- Construir identificador para vocales\n');
    fprintf('4.- Construir identificador para digitos\n');
    fprintf('5.- Guardar espacio de trabajo\n');
    fprintf('6.- Cargar espacio de trabajo\n');
    fprintf('7.- Probar clasificador para vocales\n');
    fprintf('8.- Probar clasificador para digitos\n');
    fprintf('9.- Salir (Forzar con CTRL+C)\n');

    opcion = input('Seleccionar opcion [1-9]: ');
end

switch opcion
case 1
    %Calcular m y tau para vocales
    fprintf('ATENCION: El metodo de entropia diferencial puede demorar mucho,\n');
    fprintf('incluso dias, dependiendo del tamaño y cantidad de las señales. Ademas, \n');
    fprintf('requiere bastante memoria, y no puede cancelarse el proceso mientras\n');
    fprintf('se analiza una señal (archivo de audio).\n\n');

    DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales\n'...
        '(Ej: ''c:\\tmp\\vocales\\ce\\''): ']);

    if (exist(DIRECTORIO_TRABAJO,'dir') == 7)
        script_mtau_voc;

        figure;
        hist([am,em,im,om,um],2:9);
        ylabel('Frecuencia');
        xlabel('Dimension de Inmersion');

        figure;
        hist([atau,etau,itau,otau,utau],1:12);
        ylabel('Frecuencia');
        xlabel('Retraso');
    else
        fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
    end
case 2
    %Calcular m y tau para digitos
    fprintf('ATENCION: El metodo de entropia diferencial puede demorar mucho,\n');
    fprintf('incluso dias, dependiendo del tamaño y cantidad de las señales. Ademas, \n');
    fprintf('requiere bastante memoria, y no puede cancelarse el proceso mientras\n');
    fprintf('se analiza una señal (archivo de audio).\n\n');
```

```
    DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales\n' ...
        ' (Ej: ''c:\\tmp\\digitos\\ce\\''): ']);

    if (exist(DIRECTORIO_TRABAJO,'dir') == 7)
        script_mtau_dig;

        figure;
        hist([cerom unom dosm tresm cuatrom cincom seism sietem ochom nuevem],2:14);
        ylabel('Frecuencia');
        xlabel('Dimension de Inmersion');

        figure;
        hist([cerotau unotau dostau trestau cuatrotau cincotau seistau...
                sietetau ochotau nuevetau],1:12);
        ylabel('Frecuencia');
        xlabel('Retraso');
    else
        fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
    end
case 3
    %Entrenamiento vocales
    DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales de '...
        'entrenamiento\n (Ej: ''c:\\tmp\\vocales\\ce\\''): ']);
    ALGORITMO_DENSIDAD = input(['¿Caracterizar con Densidades Espaciales [1] o '...
        'Compactacion de bloques [Cualquier otro]?: ']);
    DIFERENCIAS = input(['¿Ampliar vector de caracteristicas con primeras y '...
        'segundas diferencias?\n (Ninguna [0], Primeras [1], Primeras y Segundas'...
        '[Cualquier otro]): ']);

    if (exist(DIRECTORIO_TRABAJO,'dir') == 7)
        script_ent_voc;
    else
        fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
    end
case 4
    %Entrenamiento digitos
    DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales de'...
        ' entrenamiento\n (Ej: ''c:\\tmp\\digitos\\ce\\''): ']);

    if (exist(DIRECTORIO_TRABAJO,'dir') == 7)
        script_ent_dig;
    else
        fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
    end
```

```
case 5
    %Guardar espacio de trabajo
    workspace_save = input(['Nombre del archivo donde guardar el espacio de '...
        'trabajo (Ej: ''esp_tr'')): ']);
    save(workspace_save);
    fprintf('Espacio guardado en %s\n', workspace_save);
case 6
    %Cargar espacio de trabajo
    workspace_load = input(['Nombre del archivo desde donde cargar el espacio de'...
        ' trabajo (Ej: ''esp_tr'')): ']);
    load(workspace_load);
    fprintf('Espacio cargado de %s\n', workspace_load);
case 7
    %Probar vocales
    if isempty(who('na'))
        fprintf(['Actualmente, el espacio de trabajo no contiene un clasificador'  ...
            ' de vocales.\n']);
        fprintf('Entrene uno con la opcion 3, o cargue un espacio de trabajo que\n');
        fprintf('incluya un clasificador de este tipo\n');
    else

        DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales de'...
            'prueba\n (Ej: ''c:\\tmp\\vocales\\cp\\''): ']);

        if (exist(DIRECTORIO_TRABAJO,'dir') == 7)
            fprintf(['ATENCION: Los dos parametros que siguen deben coincidir con '...
                'los que se eligieron cuando se entreno\n']);
            fprintf('el clasificador de vocales actualmente cargado en memoria.\n');
            ALGORITMO_DENSIDAD = input(['¿Caracterizar con Densidades Espaciales [1] o'...
                ' Compactacion de bloques [Cualquier otro]?: ']);
            DIFERENCIAS = input(['¿Ampliar vector de caracteristicas con primeras y '...
                'segundas diferencias?\n (Ninguna [0], Primeras [1], Primeras y '...
                'Segundas [Cualquier otro]): ']);

            [m_voc,tasas_voc] = confusion([na ne ni no nu]', DIRECTORIO_TRABAJO, ...
                [struct('nombre','a') struct('nombre','e') struct('nombre','i') ...
                    struct('nombre','o') struct('nombre','u')]');

            fprintf('Matriz de Confusion:\n');
            display(m_voc);
            display(tasas_voc);
        else
            fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
        end
    end
```

```
case 8
    %Probar digitos
    if isempty(who('ncero'))
        fprintf(['Actualmente, el espacio de trabajo no contiene un clasificador' ...
            ' de digitos.\n']);
        fprintf(['Entrene uno con la opcion 4, o cargue un espacio de trabajo que' ...
            ' incluya\n']);
        fprintf('un clasificador de este tipo\n');
    else

        DIRECTORIO_TRABAJO = input(['Introduzca el directorio con las señales de'...
            ' prueba\n (Ej: ''c:\\tmp\\digitos\\cp\\''): ']);
        if (exist(DIRECTORIO_TRABAJO,'dir') == 7)

            [m_dig,tasas_dig] = confusion([ncero nuno ndos ntres ncuatro ...
                ncinco nseis nsiete nocho nnueve]', DIRECTORIO_TRABAJO, ...
                [struct('nombre','cero') struct('nombre','uno') ...
                struct('nombre','dos') struct('nombre','tres') ...
                struct('nombre','cuatro') struct('nombre','cinco')...
                struct('nombre','seis') struct('nombre','siete')...
                struct('nombre','ocho') struct('nombre','nueve')]',1);

            fprintf('Matriz de Confusion:\n');
            display(m_dig);
            display(tasas_dig);
        else
            fprintf('\nERROR: El directorio %s no existe.\n', DIRECTORIO_TRABAJO);
        end
    end
end

if (opcion ~= 9)
    opcion = -1;
    fprintf('\nPresione ENTER para volver al menu...');
    input('');
end

end %While mas externo
```

```
function [mat_m, mat_tau] = calcula_mtau(directorio, patron, intervalo_m, intervalo_tau)
% Obtiene la lista de valores m y tau para todos los archivos
% (señales) en el directorio que se ajustan al patron
%
% IN:
```

```
%       directorio      Alojamiento de los archivos (señales)
%       patron          Patron para filtrar archivos
%       intervalo_m     Rango de valores de prueba para m
%       intervalo_tau   Rango de valores de prueba para tau
%
% OUT:
%       mat_m           Valores m para cada señal en directorio
%       mat_tau         Valores tau
%
% NOTAS: directorio y patron son cadenas.

if (nargin == 2)
    intervalo_m   = 2:9;
    intervalo_tau = 1:12;
end
%Recuperar solo las señales indicadas
prueba = filtro_archivos(directorio,patron);

mat_m   = [];
mat_tau = [];

%Extraccion de parametros
for i=1:length(prueba)
    fprintf('[Inicio: %s] ', DATESTR(NOW));
    fprintf('Procesando: %s', prueba(i).nombre);
    %Lee señal
    w = normaliza(wavread(prueba(i).nombre));
    %Computa valores
    [m,tau] = mtau(w(200:900), intervalo_m, intervalo_tau);
    fprintf('[m=%d, t=%d] ', m, tau);
    %Almacena m y tau
    mat_m   = [mat_m m];
    mat_tau = [mat_tau tau];
    fprintf('[Fin: %s]\n', DATESTR(NOW));
end
```

```
function [l_m, l_tau] = mtau(s, intervalo_m, intervalo_tau)
% Calcular dimension de inmersion (m) y retraso (tau) para
% la señal provista s
%
% IN:
%       s               Señal
%       intervalo_m     Rango de valores de prueba para m
```

```
%       intervalo_tau   Rango de valores de prueba para tau
%
% OUT:
%       l_m             Dimension de inmersion sugerida
%       l_tau           Retraso sugerido
%
% NOTAS: La señal (s) debe estar normalizada.
%

%Para que los resultados sean consistentes entre
%diversas invocaciones
rand ('seed', 1);
randn('seed', 1);

mmin = min(intervalo_m);%Min y max dimension de inmersion
mmax = max(intervalo_m);
T = intervalo_tau;      %Min y max retraso
Ns = 3;                 %Cantidad de secuencias sustitutas

%Invocar DLL para calcular entropia diferencial
%con la aproximacion de Kozachenko-Leonenko
[H,M,T] = sweep_kl (s, mmin, mmax, T);

%Sustitutos
for i=1:Ns
  Xs = generate_surrogate(s,0,0);           %Secuencias sustitutas
  Hs(:,:,i) = sweep_kl (Xs, mmin, mmax, T); %Entropia diferencial
end
%Estandarizar respecto a los sustitutos
D = (H./mean(Hs,3))';
p1 = length(s) - max(T)*mmax;
D =  D + log(p1) * repmat(intervalo_m,length(T),1) ./ p1;

%Determinar los minimos m y tau
[a b c] = minmin(D);

%Resultados
l_m   = mmin+c-1;
l_tau = T(b);
```

```
function datos=efr(s,m,tau)
% Reconstruye el espacio de fase a partir de la señal s,
% usando los parámetros m y tau
%
% IN:
```

```
% s Señal (vector columna)
% m Dimensión de inmersión
% tau Retraso (lag)
% OUT:
% datos Matriz con las componentes del espacio de
% fase reconstruido. Cada fila corresponde
% a una dimensión del espacio de fase.

largo_s = size(s,1);
datos = [];
for i=1:m
datos = [datos; s(1+(i-1)*tau:largo_s - tau*(m-i))'];
end
```

```
function lista = filtro_archivos(directorio, patron)
% Recupera todos los archivos en directorio cuyo nombre
% inicia con la secuencia de caracteres patron
%
% IN:
%       directorio      Directorio a leer
%       patron          Cadena para filtrar archivos
%
% OUT:
%       lista           Lista de archivos filtrados
%
% NOTAS: lista contiene estructuras (struct), cuyo unico
% campo es 'nombre'. Este nombre incluye tambien la ruta.
%

lista        = [];
direct       = dir(directorio);
largo_patron = length(patron);
for i=1:length(direct)
    if (length(direct(i).name) >= largo_patron) ...
            & (all(patron == direct(i).name(1:largo_patron)))
        lista = [lista; struct('nombre',strcat(directorio,direct(i).name))];
    end
end
```

```
function net=enredar(alphabet, targets)
% Entrenamiento de una red BKP, con tres
% capas (5 neuronas en la capa intermedia).
% Algoritmo de Levenberg-Marquardt.
```

```
%clf;
%figure(gcf)
%echo on

[R,Q] = size(alphabet);
[S2,Q] = size(targets);

%Definir la red

S1 = 5;
net = newff(minmax(alphabet),[S1 S2],{'logsig' 'purelin'},'trainlm');
net.LW{2,1} = net.LW{2,1}*0.01;
net.b{2} = net.b{2}*0.01;

%Entrenar la red
net.performFcn = 'sse';        % Funcion de rendimiento: Suma de Errores cuadraticos
net.trainParam.goal = 0.00001; % Nivel de error buscado: 1x10e-3 (1x10e-5)
net.trainParam.show = 20;      % Mostrar progreso cada 20 epocas
net.trainParam.epochs = 1000;  % Numero maximo de epocas
net.trainParam.mc = 0.95;      % Momentum

%...
P = alphabet;
T = targets;

[net,tr] = train(net,P,T);
```

```
function sn = normaliza(s, tratar_ruido)
% Normaliza la amplitud de la señal s
%
% IN:
%       s               La señal de voz
%       tratar_ruido    ¿Pasar la señal por wavelets?
%                       (Si = 1, No = Otro) [OPCIONAL <- 0]
%
% OUT:
%       sn      Señal normalizada
%

if (nargin == 1)
    tratar_ruido = 0;
end

%Emplear solo un canal (por si la señal es estereo)
```

```
s = s(:,1);

%Centrar señal
ts = s;
s = s - mean(s);
s = s ./ std(ts);
s = s(:);

% Normaliza la amplitud de la señal s entre +1 y -1
maxi = max(abs(s));
% Precaucion con señales cero (maxi == 0). Simplemente
% se divide cada valor de la señal entre el valor mas
% alto.
sn = s / (maxi + (maxi==0));

% Si se efectuan evaluaciones con ruido
%tratar_ruido=1;
%if (tratar_ruido == 1)
%    sn = denoise(sn);
%end
```



```
%Reconocimiento de vocales

a=[];
e=[];
i=[];
o=[];
u=[];

if isempty(DIFERENCIAS) | ~ismember(DIFERENCIAS, [0,1])
   nivel = 2;
else
   nivel = DIFERENCIAS;
end

display(nivel);
ar_a = filtro_archivos(DIRECTORIO_TRABAJO, 'a');
ar_e = filtro_archivos(DIRECTORIO_TRABAJO, 'e');
ar_i = filtro_archivos(DIRECTORIO_TRABAJO, 'i');
ar_o = filtro_archivos(DIRECTORIO_TRABAJO, 'o');
ar_u = filtro_archivos(DIRECTORIO_TRABAJO, 'u');
```

```
largo = size(ar_a,1);   % El tamaño de todos los ar* debe ser
                        % el mismo

for idx=1:largo
    display(idx);
    a = [a; calcular_densidad(normaliza(wavread(ar_a(idx).nombre)), nivel)'];
    e = [e; calcular_densidad(normaliza(wavread(ar_e(idx).nombre)), nivel)'];
    i = [i; calcular_densidad(normaliza(wavread(ar_i(idx).nombre)), nivel)'];
    o = [o; calcular_densidad(normaliza(wavread(ar_o(idx).nombre)), nivel)'];
    u = [u; calcular_densidad(normaliza(wavread(ar_u(idx).nombre)), nivel)'];
end

%a = [a; gdenoise(ar_a)];
%e = [e; gdenoise(ar_e)];
%i = [i; gdenoise(ar_i)];
%o = [o; gdenoise(ar_o)];
%u = [u; gdenoise(ar_u)];

mat_ent = [a; e; i; o; u];
%largo = size(a,1);
```



```
na = enredar(mat_ent', [ones(1,largo) zeros(1,largo*4)]);
ne = enredar(mat_ent', [zeros(1,largo) ones(1,largo) zeros(1,largo*3)]);
ni = enredar(mat_ent', [zeros(1,largo*2) ones(1,largo) zeros(1,largo*2)]);
no = enredar(mat_ent', [zeros(1,largo*3) ones(1,largo) zeros(1,largo)]);
nu = enredar(mat_ent', [zeros(1,largo*4) ones(1,largo)]);
```

```
% Reconocimiento de digitos

cero  =[];
uno   =[];
dos   =[];
tres  =[];
cuatro=[];
cinco =[];
seis  =[];
siete =[];
ocho  =[];
nueve =[];

nivel = 0;
ALGORITMO_DENSIDAD = 1;
```

```
ar_cero   = filtro_archivos(DIRECTORIO_TRABAJO, 'cero');
ar_uno    = filtro_archivos(DIRECTORIO_TRABAJO, 'uno');
ar_dos    = filtro_archivos(DIRECTORIO_TRABAJO, 'dos');
ar_tres   = filtro_archivos(DIRECTORIO_TRABAJO, 'tres');
ar_cuatro = filtro_archivos(DIRECTORIO_TRABAJO, 'cuatro');
ar_cinco  = filtro_archivos(DIRECTORIO_TRABAJO, 'cinco');
ar_seis   = filtro_archivos(DIRECTORIO_TRABAJO, 'seis');
ar_siete  = filtro_archivos(DIRECTORIO_TRABAJO, 'siete');
ar_ocho   = filtro_archivos(DIRECTORIO_TRABAJO, 'ocho');
ar_nueve  = filtro_archivos(DIRECTORIO_TRABAJO, 'nueve');

largo = size(ar_cero,1);   % El tamaño de todos los ar* debe ser
                           % el mismo

for idx=1:largo
    display(idx);
    cero   = [cero;  parametrizar_dig(normaliza(wavread(ar_cero(idx).nombre)))'];
    uno    = [uno;   parametrizar_dig(normaliza(wavread(ar_uno(idx).nombre)))'];
    dos    = [dos;   parametrizar_dig(normaliza(wavread(ar_dos(idx).nombre)))'];
    tres   = [tres;  parametrizar_dig(normaliza(wavread(ar_tres(idx).nombre)))'];
    cuatro = [cuatro;parametrizar_dig(normaliza(wavread(ar_cuatro(idx).nombre)))'];
    cinco  = [cinco; parametrizar_dig(normaliza(wavread(ar_cinco(idx).nombre)))'];
    seis   = [seis;  parametrizar_dig(normaliza(wavread(ar_seis(idx).nombre)))'];
    siete  = [siete; parametrizar_dig(normaliza(wavread(ar_siete(idx).nombre)))'];
    ocho   = [ocho;  parametrizar_dig(normaliza(wavread(ar_ocho(idx).nombre)))'];
    nueve  = [nueve; parametrizar_dig(normaliza(wavread(ar_nueve(idx).nombre)))'];
end

mat_ent = [cero; uno; dos; tres; cuatro; cinco; seis; siete; ocho; nueve];

total=9;
ncero   = enredar(mat_ent', [ones(1,largo)    zeros(1,largo*total)]);
nuno    = enredar(mat_ent', [zeros(1,largo)   ones(1,largo) zeros(1,largo*(total-1))]);
ndos    = enredar(mat_ent', [zeros(1,largo*2) ones(1,largo) zeros(1,largo*(total-2))]);
ntres   = enredar(mat_ent', [zeros(1,largo*3) ones(1,largo) zeros(1,largo*(total-3))]);
ncuatro = enredar(mat_ent', [zeros(1,largo*4) ones(1,largo) zeros(1,largo*(total-4))]);
ncinco  = enredar(mat_ent', [zeros(1,largo*5) ones(1,largo) zeros(1,largo*(total-5))]);
nseis   = enredar(mat_ent', [zeros(1,largo*6) ones(1,largo) zeros(1,largo*(total-6))]);
nsiete  = enredar(mat_ent', [zeros(1,largo*7) ones(1,largo) zeros(1,largo*(total-7))]);
nocho   = enredar(mat_ent', [zeros(1,largo*8) ones(1,largo) zeros(1,largo*(total-8))]);
nnueve  = enredar(mat_ent', [zeros(1,largo*9) ones(1,largo) zeros(1,largo*(total-9))]);
```

```
function vc = parametrizar_dig(s)
```

```
% Extrae caracteristicas del digito en s
%
% IN:
%           s       Señal
%
% OUT:
%           vc      Vector de caracteristicas
%

% Filtrados
vc = transforma(sgolayfilt(medfilt1(teager(s),80),3,199),100);
% Analisis por segmentos o tramas
vp = transforma(analisis_por_tramas(s),100);
% Formacion del vector de caracteristicas
vc = [vc(:); vp(:)];
return;
```

```
function d=obtener_den(s, retraso)
% Reconstruye el espacio de fase de la señal
% s y retorna su densidad.
%
% IN:
%       s       Señal.
%
% OUT:
%       d       Vector de densidades.
%
% NOTAS: Solo reconstruccion bidimensional.
% s debe estar normalizada de tal manera
% que su amplitud se encuentre entre -1
% y +1.

if (nargin < 2)
    retraso = 3;        %Computado experimentalmente
end

part    = (-1:0.2:1)';  %Particiones del EFR
part    = part';

%Obtener EFR
v = efr(s, 2, retraso);
%Calculo de Densidades
d = densidad([v(1,:)' v(2,:)'], part);
return;
```

```
function d=calcular_densidad(s, nivel)
% Calcula las densidades segun el algoritmo implementado
% en obtener_den.
%
% IN:
%       s           Señal
%       nivel       Diferencias a procesar. Opcional.
%
% OUT:
%       d           Vector de densidades
%
% NOTAS: Si nivel == 0, el vector de densidades
% incluye solo el resultado de obtener_den sobre
% s. Si nivel == 1, se anexan las densidades
% de diff(s). Si nivel == 2, se agregan las de
% diff(diff(s)).

if (nargin==1)
    nivel = 0;
end

switch nivel
case 0
    d  = obtener_den(s);
    d  = d(:);
case 1 %Primeras diferencias
    d1 = obtener_den(s);
    d2 = obtener_den(diff(s));
    d  = [d1 d2];
    d  = d(:);
case 2 %Segundas diferencias
    d1 = obtener_den(s);
    d2 = obtener_den(diff(s));
    d3 = obtener_den(diff(s,2));
    d  = [d1 d2 d3];
    d  = d(:);
end
```

```
function sp=teager(s)
% Aplica operador de energia de Teager sobre s
%
% IN:
%           s       Señal
%
```

```
% OUT:
%            sp      Distribucion de la energia
%

for i=2:length(s)-1
    sp(i)=(s(i)*s(i)) - (s(i+1)*s(i-1));
end
return;
```

```
function xd = denoise(x)

deb = x(1);
xd = wden(x-deb,'sqtwolog','s','mln',3,'db3')+deb;
```

```
function gd = gdenoise(gn)
% Tratamiento del ruido a todas las señales cuyas
% rutas y nombres conforman gn

gd = [];
for i=1:length(gn)
    gd = [gd; parametrizar_dig(denoise(normaliza(wavread(gn(i).nombre))))'];
    %gd = [gd; calcular_densidad(denoise(normaliza(wavread(gn(i).nombre))),1)'];
end
```

```
function nserie = transforma(serie, R)
% Transforma la serie en otra secuencia de
% tamaño R

p1 = size(serie,1);
nserie = zeros(R, 1);
for j=1:R
    r_j = (((j-1) * (p1 - 1)) / (R-1)) + 1;
    i = fix(r_j);
    if (i >= R) i=i-1; end
    nserie(j) = serie(i) + (serie(i+1)-serie(i))*(r_j-i);
end
```

```
function densidades = densidad(mat, part)
% Recibe el espacio de fase bidimensional y
% calcula las densidades en los intervalos
% definidos por part.
%
% IN:
```

```
%       mat     EFR bidimensional.
%       part    Particion del rango de valores
%               en mat.
%
% OUT:
%       densidades  Vector de densidades
%
% NOTAS: Si part incluye k elementos, entonces
% el EFR se particiona en (k-1) x (k-1) bloques.
% Por ejemplo, si mat toma valores entre
% -1 y +1, part pudiera ser [-1 -0.5 0 0.5 1],
% caso en el cual el EFR se particiona en 16
% bloques.
% Hay dos esquemas para el calculo de densidades:
%   1.- Hallar la cantidad de puntos en cada
%       bloque y dividirla entre el total de puntos
%       en el EFR (densidad de bloque).
%   2.- Hallar la distancia euclidiana de los puntos
%       en cada bloque con el centro del bloque
%       (compactacion de bloque).

%if (nargin == 2)
%    esquema = 0;
%end

global ALGORITMO_DENSIDAD;

if isempty(ALGORITMO_DENSIDAD) | (ALGORITMO_DENSIDAD ~= 1)
    esquema = 1;
else
    esquema = 0;
end

%Resultados en cero
densidades1 = zeros(length(part)-1);

%Separar los ejes del EFR
mat1=mat(:,1);
mat2=mat(:,2);

%esquema = 0;

for i=1:length(part)-1
    q1=find(mat1 >= part(i) & mat1 <= part(i+1));
    for j=1:length(part)-1
```

```
        q2=find(mat2 >= part(j) & mat2 <= part(j+1));
        %Hallar los puntos en la interseccion de
        %los intervalos i y j
        q = intersect(q1,q2);

        switch esquema
        case 0
            %Densidad
            densidades(i,j) = length(q) / length(mat1);
        case 1
            %Compactacion
            if (q==[])
                densidades(i,j)=0;
            else
                densidades(i,j)= sum(disteusq([mat1(q) mat2(q)], ...
                    mean([mat1(q) mat2(q)]))) / size(q,1);
            end
        end
    end
end

if (esquema == 1)
    densidades = densidades ./ max(densidades(:));
end
```

```
function [tabla, tasas]=confusion(redes, directorio, patrones, vocdig)
% Matriz de confusion para los archivos en directorio
% segun los patrones.
%
% IN:
%       redes       Vector columna con los clasificadores para
%                   cada categoria (n filas = n categorias)
%       directorio  Alojamiento de las señales con las que se
%                   probaran los clasificadores. Es una cadena.
%       patrones    Vector columna con los patrones de filtrado
%                   de archivos (señales) en directorio.
%       nivel       Parametro de calcular_densidad.
%       vocdig      Vocales==0, Digitos==1
%
% OUT:
%       tabla       Matriz de confusion. Cada fila corresponde
%                   a un patron, y cada columna, a una categoria
%                   o clasificador.
%
```

```
% NOTAS: En lineas generales, se filtra el directorio con cada
% patron, y los archivos (señales) resultantes de un filtrado
% pasan a cada red para determinar cual lo clasifica mejor.
% Anticipadamente el patron nos señala a cual categoria pertenece
% una señal, por lo que se puede saber si la clasificacion
% hecha por las redes es correcta.

if (nargin == 3)
    vocdig = 0;
end

global DIFERENCIAS;
if (vocdig == 0)
    if isempty(DIFERENCIAS) | ~ismember(DIFERENCIAS, [0,1])
        nivel = 2;
    else
        nivel = DIFERENCIAS;
    end
end

tabla = [];
%nivel = 1;
%Para cada patron, se comprueba la efectividad
%en la clasificacion de las redes
tasas = zeros(size(redes,1),1);
for ind_patron = 1:size(patrones,1)

    %Solo los archivos en directorio asociados al patron
    %actual
    prueba = filtro_archivos(directorio,patrones(ind_patron).nombre);
    display(ind_patron);
    %Resultados de clasificacion en cero para este patron
    acum  = zeros(size(redes,1),1);

    for i=1:size(prueba,1)
        max = 0;
        if (vocdig == 0)
            dd = calcular_densidad(normaliza(wavread(prueba(i).nombre)),nivel);
        else
            dd = parametrizar_dig(normaliza(wavread(prueba(i).nombre)));
        end
        %La categoria de la red que arroje la mayor
        %salida es la que se toma como la asignada al
        %patron. Las clasificaciones correctas cumplen
        %idx==ind_patron.
```

```
        for j=1:size(redes,1)
            sim_r = sim(redes(j), dd);
            if (sim_r > max)
                max = sim_r;
                idx = j;
            end
        end
        acum(idx) = acum(idx) + 1;
    end
    %acum = [acum; acum(ind_patron) / sum(acum)];
    tasas(ind_patron) = acum(ind_patron) / sum(acum);
    tabla = [tabla; acum'];
end
```

BIBLIOGRAFíA

[1] H. Abarbanel, R. Brown, J. Sidorowich, y L. Tsimring, "The analysis of observed chaotic data in physical systems", *Reviews of Modern Physics*, vol. 65, No. 4, 1993.

[2] M. Bahoura, y J. Rouat, "Wavelet speech enhancement based on the Teager energy operator", *IEEE Signal Processing letter*, Vol. 8, No. 1, pp. 10-12, 2001.

[3] M. Bahoura, y J. Rouat, "Wavelet noise reduction: Application to speech enhancement", Canadian Acoustics, Vol. 28, pp. 158-159, 2000.

[4] E. Bradley, "Time series analysis", en *Intelligent Data Analysis: An Introduction*, Springer, 1999.

[5] D. Cairns, J. Hansen, y J. Riski, "Detection of hypernasal speech using a nonlinear operator", *Proceedings of the 16th Annual International Conference of the IEEE*, Vol. 1, pp. 253-254, 1994.

[6] A. de la Torre, A. M. Peinado, J. C. Segura, J. L. Pérez, C. Benítez, y A. Rubio, "Histogram equalization of the speech representation for robust speech recognition", URL: citeseer.nj.nec.com/544712.html.

[7] Q. Fu, y E. A. Wan, "A novel speech enhancement system based on wavelet denoising", Center of Spoken Language Understanding, OGI School of Science and Engineering at OHSU, 2003.

[8] T. Gautama, D. Mandic, y M. Van Hulle, "A differential entropy based method for determining the optimal embedding parameters of a signal", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2003.

[9] R. Goldberg, *A Practical Handbook of Speech Coders*, CRC Press, 2000.

[10] Grupo de Acústica de la Universidad del País Vasco, "Características del Sonido: Intensidad, Timbre, Tono y Duración", URL: http://www.ehu.es/acustica/bachillerato/casoes/casoes.html.

[11] F. Halsall, *Data communications, computer networks and Open Systems*, Fourth Edition, Addison-Wesley, 1998.

[12] X. Huang, A. Acero, y H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, New Jersey: Prentice Hall PTR, 2001.

[13] A. K. Jain, J. Mao, y K. M. Mohiuddin, "Artificial Neural Networks: A Tutorial", *IEEE Computer*, vol. 29, No. 3, pp. 31-44, 1996.

[14] F. Jabloun, A. Cetin, y E. Erzin, "Teager energy based feature parameters for speech recognition in car noise", *IEEE Signal Processing Letters*, Vol. 6, No. 10, 1999.

[15] J. Kaiser, "On a simple algorithm to calculate the energy of a signal", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 381-384, 1990.

[16] J. Kaiser, "On Teager's energy algorithm and its generalization to continuous signals", *Proceedings of the 4th IEEE Digital signal processing workshop*, 1990.

[17] Kohavi, R., y Provost, F. *Glossary of Terms*, Machine Learning, Nº30, Vol. 2/3, pp. 271-274, 1998.

[18] B. R. Kosanović, *Signal and system analysis in fuzzy information space*, Ph.D. Dissertation, University of Pittsburgh, 1995.

[19] E. Kvedalen, *Signal processing using the Teager Energy Operator and other nonlinear operators*, Cand. Science Thesis, University of Oslo, 2003.

[20] E. A. Lee, y P. Varaiya, *Structure and interpretation of Signals and Systems*, California: Addison-Wesley, 2000.

[21] A. C. Lindgren, M.T. Johnson, y R. J. Povinelli, "Speech recognition using reconstructed phase space features", *International Conference on Acoustics, Speech and Signal Processing*, 2003.

[22] X. Liu, M. T. Johnson, y R. J. Povinelli, "Vowel classification by global dynamic modeling", *Proceedings of ISCA Tutorial and Research Workshop on Non-Linear Speech Processing (NOLISP 2003)*, en revisión.

[23] X. Liu, R. J. Povinelli, y M. T. Johnson, "Detecting determinism in english phonemes".

[24] L. Ljung, y T. Glad, *Modeling of dynamical systems*, Prentice Hall, New Jersey: 1994.

[25] Lorenz, E., *Deterministic nonperiodic flow*, J. Atmos. Sci., 20, 130-141, 1963.

[26] J. L. Maldonado, *Tratamiento y reconocimiento automático de señales de la voz venezolana*, Disertación doctoral, Universidad de Los Andes, 2003.

[27] Mazo Torres, J., "Apuntes de Introducción a la teoría del caos: Ecuaciones de Lorenz", URL: http://wzar.unizar.es/acad/fac/cie/condmat/T/juanjo/caos/caos.html.

[28] A. Medio, y M. Lines, *Nonlinear Dynamics: A Primer*, Cambridge University Press, 2001.

[29] M. Misiti, G. Oppenheim, J.-M. Poggi, y Y. Misiti, *MATLAB Wavelet Toolbox User Guide*, The MathWorks, 2003.

[30] A. Moreno, y E. Mora, *Speechdat Spanish Venezuelan database for the fixed Telephone network*, Universidad politécnica de Cataluña, España y Universidad de Los Andes, Venezuela, 1999.

[31] G. Mounin, *Claves para la Linguística*, Editorial Anagrama, 1970.

[32] R. J. Povinelli, *Time Series Data Mining: Identifying Temporal Patterns for Characterization and Prediction of Time Series Events*, Ph.D. Dissertation, Marquette University, 1999.

[33] W. Press et al. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, 1992.

[34] T. Quatieri, C. Jankowski, y D. Reynolds, "Energy onset times for speaker identification", *IEEE Signal Processing Letters*, Vol. 1, pp. 160-162, 1994.

[35] L. Rabiner, y B. Juang, "An introduction to hidden Markov models". *IEEE Acoustics, Speech and Signal Processing Magazine*, vol. 3, pp. 4-16, 1986.

[36] L. Rabiner, "A tutorial on hidden Markov models", *Proceedings of the IEEE*, vol. 77, pp. 257-286, 1989.

[37] L. Rabiner, y R. Schafer, *Digital processing of speech signals*, Prentice-Hall, 1978.

[38] F. M. Roberts, R. J. Povinelli, y K. M. Ropella, "Identification of ECG Arrhythias using Phase Space Reconstruction", *5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01)*, pp. 411-423, 2001.

[39] M. A. Rodríguez, I. Cortázar, D. Tapias, y J. Relaño, "Estado del arte en tecnologías de voz", *Comunicaciones de Telefónica I+D*, No. 20, pp. 117-136, 2001.

[40] W. Rodríguez, H.-N. Teodorescu, F. Grigoras, A. Kandel y H. Bunke, "A fuzzy information space approach to speech signal non-linear analysis", *International Journal of Intelligent Systems*, vol. 15, No. 4, pp. 343-363, 2000.

[41] W. Rodriguez, "Similarity of Dynamical Systems", Ph.D. Thesis, University of South Florida, 1998.

[42] S. Rosen y P. Howell, *Signals and Systems for Speech and Hearing*, Academic Press, 1991.

[43] S. Russell, y P. Norvig, *Inteligencia Artificial: Un enfoque moderno*, México: Prentice-Hall, 1996.

[44] T. Sauer, J. A. Yorke, y M. Casdagli, "Embedology", *Journal of Statistical Physics*, vol. 65, pp. 579-616, 1991.

[45] Shepherd, A. J. *Second-Order Methods for Neural Networks*, Springer-Verlag, 1997.

[46] N. Sundaram, B. Smolenski, y R. Yantorno, "Instantaneous nonlinear teager energy operator for robust voiced - unvoiced speech classification", [URL: http://www.temple.edu/speech_lab/sundaram.PDF].

[47] F. Takens, "Detecting strange attractors in turbulence", *Dynamical Systems and Turbulence*, Warwick, 1980.

[48] C. Tan Keng Yan, *Speaker adaptive phoneme recognition using Time Delay Neural Networks*, M. Sc. Thesis, National University of Singapore, 2000.

[49] H. Teager, y S. Teager, "Evidence for nonlinear sound production mechanisms in the vocal tract", *Proceedings NATO ASI on Speech Production and Speech Modeling*, pp. 241-261, 1990.

[50] J. B. Tenenbaum, V. de Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction", *Science*, vol. 20, 22 de Diciembre de 2000.

[51] M. Villeret et. al, "A New Digital Technique for Implementation of Any Continuous PCM Companding Law", IEEE Int. Conf. on Communications, 1973, vol. 1, pp. 11.12-11.17.

[52] G. Williams, *Chaos Theory Tamed*, Joseph Henry Press, 1997.

[53] J. Ye, M. T. Johnson, y R. J. Povinelli, "Phoneme Classification using Naive Bayes Classifier in Reconstructed Phase Space", *10th IEEE Digital Signal Processing Workshop*, 2002.

[54] J. Ye, M. T. Johnson, y R. J. Povinelli, "Phoneme classification over the reconstructed phase space using principal component analysis", *ISCA Tutorial and Research Workshop on Non-linear Speech Processing (NOLISP)*, 2003.

[55] F. Zhao, "Extracting and Representing Qualitative Behaviors of Complex Systems in Phase Space", *Artificial Intelligence*, vol. 69, pp. 51-92, 1994.

[56] F. Zhao, "Automatic Analysis and Synthesis of Controllers for Dynamical Systems Based on Phase Space Knowledge", Ph.D Thesis, MIT, MIT AI Lab Technical Report AI-TR-1385, 1992.