Conciencia Tecnológica No. 32, Julio-Diciembre 2006

Reconocimiento de Voz con Redes Neuronales, DTW y Modelos Ocultos de Markov

Investigación

De Luna Ortega Carlos Alejandro, Martínez Romo Julio César, Mora González Miguel
Universidad Politécnica de Aguascalientes, Instituto Tecnológico de Aguascalientes, Universidad de Guadalajara (Centro Universitario de los Lagos)
alejandro.deluna@upa.edu.mx, jucemaro25@hotmail.com, mmora@culagos.udg.mx

Resumen

En este artículo se aborda el diseño de un reconocedor de voz, con el idioma español mexicano, del estado de Aguascalientes, de palabras aisladas, con dependencia del hablante y vocabulario pequeño, empleando Redes Neuronales Artificiales (ANN por sus siglas en inglés), Alineamiento Dinámico del Tiempo (DTW por sus siglas en inglés) y Modelos Ocultos de Markov (HMM por sus siglas en inglés) para la realización del algoritmo de reconocimiento.

Introducción

Los sistemas de reconocimiento de voz con el tiempo han tenido un gran auge en la sociedad moderna, ante la creciente necesidad de tener sistemas que se puedan controlar de manera no física, para un porcentaje creciente de personas discapacitadas en el mundo.
Un reconocimiento de voz tiene la dificultad principal en la extracción de características de la voz, debido a que cuenta con diversos problemas a solucionar, como lo es que una persona no puede pronunciar dos veces igual la misma palabra, debido entre otras cosas: *al estado de ánimo, la salud, y la fuerza de pronunciación, el tiempo, la entonación, etc.*

Aún y con estas dificultades, se han desarrollado algoritmos para poder determinar un nivel de coincidencia entre las pronunciaciones para poder realizar un reconocimiento eficaz, logrando encontrar algunos algoritmos que lo realizan con mayor desempeño, debido a su construcción, y sus características internas de proceso para el reconocimiento de patrones, entre ellos destacan las ANN, el DTW y Los HMM.

En nuestro diseño se ha decidido utilizar los tres algoritmos, debido a que las ANN debido a que muestran un grado notable de aprendizaje sobre las señales de entrada, dando adaptabilidad en las variantes que presenta la voz, lo que nos dará un gran apoyo a nuestro algoritmo, el DTW porque ha sido una técnica comprobada como eficiente en diversos trabajos y que nos permita ser de referencia en nuestro trabajo, y los HMM para tener de manera completa el algoritmo y poder usar un modelo de mayoría.

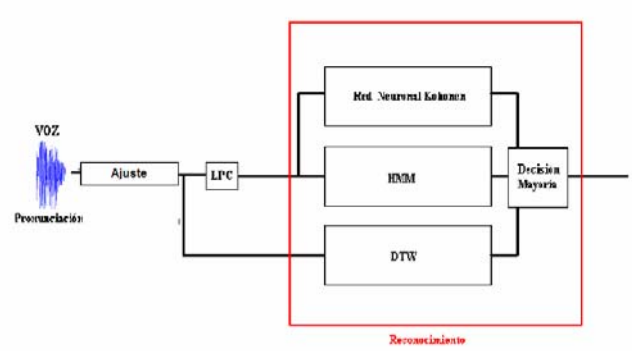

Fig. 1. Etapas del sistema de reconocimiento de voz a emplear para el algoritmo del reconocedor.

Reconocimiento de voz

El algoritmo de reconocimiento de voz en el que se trabajó se buscó que reconociera a un hablante de una diversidad de ellos mediante un modelo de mayoría, donde se identifica la pronunciación por cada uno de los algoritmos de ANN, DTW y HMM, y tener mediante una mayoría la palabra pronunciada.

El sistema del reconocimiento de voz se muestra en la figura 1, consiste en capturar la voz para pasarla a través de una normalización, la cual al terminar tendrá una transformación del tiempo a frecuencia, la cual segmentaremos en fonemas para tener así ya lista la palabra a reconocer y determinar si es la palabra esperada.

A. Captura de la Voz

La captura de voz se realiza en la computadora, aplicando el software Simulink de Matlab®, utilizando entre 3 o 4 segundos para la grabación que se va a procesar, tomando una resolución de 8 bits, que hacen que nuestra muestra sea pequeña en espacio pero con una información suficiente para analizar.

La figura 2 muestra la gráfica de un archivo capturado durante las pruebas de la captura de la voz.

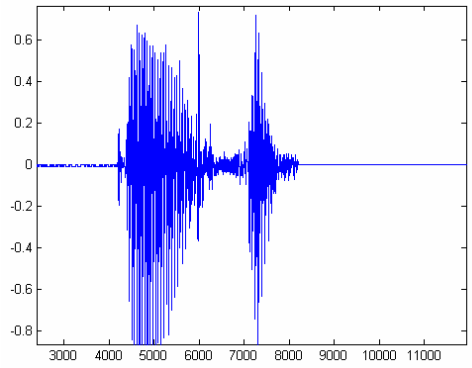

Fig. 2. Archivo de audio de la palabra "casa" capturado en simulink.

Una vez capturada la voz dentro de la computadora se da paso a seguir con el procesamiento en base a algoritmos realizados en Matlab®.

B. Etapa de Ajuste

Una vez obtenida la señal del micrófono y transportada a nuestro ambiente, es decir, tener un vector de sonido, se realiza el siguiente paso que es la normalización, ya que es necesario ajustar los tamaños temporales de los grupos de estudio [1].

El ajuste consiste en quitar la parte de silencio de la entrada a todos los archivos y dejarlos en un comienzo semejante para poder aplicar el algoritmo a señales con un comienzo en una palabra, sin que el silencio del inicio intervenga.

El algoritmo que se empleó para la normalización fue simplemente una comparación entre los puntos del vector de sonido y verificar dónde se encontraría el inicio de la pronunciación mediante un cambio dramático de valor.

Utilizando las instrucciones de Matlab® se pudo llegar a obtener el ajuste como el que se presenta en la figura 3.

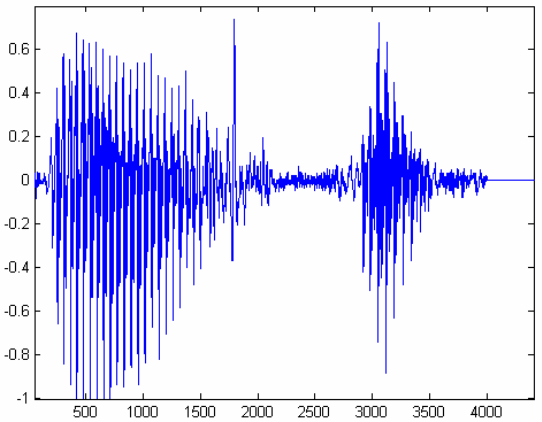

Fig. 3. Archivo de audio normalizado de la palabra "casa" con el algoritmo en Matlab®.

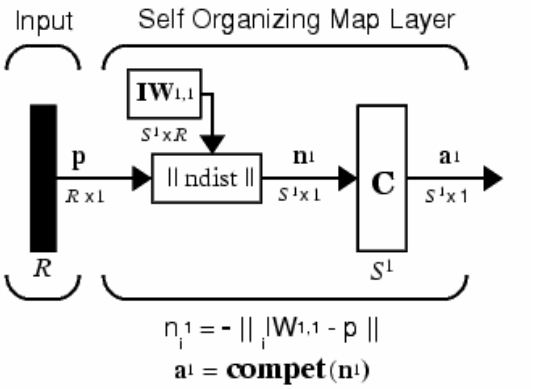

Fig. 4. Estructura de una SOFM.

Teniendo ya las señales alineadas en un valor inicial igual, el siguiente proceso necesario para nuestro sistema es el dar un cambio de manejo de variables, en lugar de tener la señal en el tiempo, la tendremos que manejar en términos de la frecuencia, lo que implica realizar una transformación, usando una Codificación de Predicción Lineal (LPC), con la cual obtendremos unos coeficientes que nos servirán para el entrenamiento de la red, los valores de transición del HMM.

C. Obtención de los coeficientes por LPC

La recursión de Levinson-Durbin es una importante solución para la obtención de coeficientes de filtros FIR[2] es por lo que S}se toma la señal después del ajuste para realizar el cambio a la frecuencia, mediante la obtención de los coeficientes por LPC, mediante dicho algoritmo.

Se ajustó sobre el número de coeficientes para obtener con dicho algoritmo, para dejarlo establecido con 30 coeficientes de cada palabra, de esta manera, si la palabra era muy larga en pronunciación o corta, se aseguraba que los coeficientes de todas fueran iguales y con ello poder realizar una comparación y determinar la concordancia o diferencia entre las pronunciaciones.

D. Algoritmo de ANN

El modelo de red neuronal que se ocupó en el desarrollo del reconocedor fue el tipo de red autoorganizada, o comúnmente llamada mapas de Kohonen.

Los mapas autoorganizados (SOFM por sus siglas en inglés) o mapas de Kohonen son redes neuronales que tienen la peculiaridad de tener un mapa organizado de sus pesos, mediante el algoritmo de entrenamiento.

Los SOFM poseen un gran potencial de aplicabilidad práctica, clasificando patrones,

cuantificando vectorialmente, reduciendo dimensiones, extrayendo rasgos, etc.

Este modelo tiene dos variantes denominadas LVQ (Learning Vector Quantization) y TPM (Topology Preserving Map) o SOM (Self Organizing Map), ambas se basan en el principio de formación de mapas topológicos para establecer características comunes entre las informaciones (vectores) de entrada a la red, aunque difieren en las dimensiones de éstos, siendo de una sola dimensión en el caso de LVQ y bidimensional o tridimensional en la red SOM. Estas redes se tratarán con mayor profundidad en secciones posteriores.

El aprendizaje en el modelo de Kohonen es de tipo Off-line, por lo que se distingue una etapa de aprendizaje y otra de funcionamiento. En la etapa de aprendizaje se fijan los valores de las conexiones (feedforward) entre la capa de entrada y la salida. Esta red utiliza un aprendizaje no supervisado de tipo competitivo, las neuronas de la capa de salida compiten por activarse y sólo una de ellas permanece activa ante una determinada información de entrada a la red, los pesos de las conexiones se ajustan en función de la neurona que haya resultado vencedora [3] (ver figura 4).

En fase de aprendizaje cada neurona del mapa sintoniza con diferentes rasgos del espacio de entrada. El proceso es el siguiente: tras la presentación y procesamiento de un vector de entrada x(t), la neurona vencedora modifica sus pesos de manera que se parezca un poco más a x(t). De este modo, ante el mismo patrón de entrada, dicha neurona responderá en el futuro todavía con más intensidad.

Sin embargo el modelo de mapa de Kohonen aporta una importante novedad, pues incorpora a este esquema relaciones entre las neuronas próximas en el mapa [4]. Esto quiere decir que el mapa puede clasificar en la fase de aprendizaje algunas variantes del patrón, como por ejemplo la pronunciación de la palabra "A" más de una vez por la misma persona, que dicha palabra no es igual en ninguna de las veces. Con ello la ANN puede clasificar de manera correcta las variaciones que se pueden presentar cuando un hablante pronuncie la misma palabra.

El algoritmo de aprendizaje autoorganizado basa su funcionamiento en las distancias euclídeas para determinar la neurona ganadora y la cual se asemejará a la entrada, actualizando sus pesos con la regla de la siguiente ecuación [5]:

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t).h(|i-g|,t).(x_k(t) - w_{ijk}(t)) \quad (1)$$

Obteniendo con este aprendizaje un mapa autoorganizado que nos dejará en cada neurona una entrada, así si lo aplicamos en el reconocimiento de voz tendríamos un peso y una neurona exclusiva para cada fonema y sus variantes del patrón, teniendo con ello una gran posibilidad de segmentación.

Con esto tomamos 30 muestras de cada palabra para pasar los coeficientes de todas las neuronas al algoritmo de entrenamiento de la red neuronal, y con esto obtener índices y pesos de cada palabra, que serán la base para el reconocimiento de las palabras, dichos índices son autoorganizativos.

El algoritmo de reconocimiento de la red llamada mapa autorganizativo (SOM por sus siglas en Inglés), se realiza cuando entra una palabra con los coeficientes del LPC y comienza a compararla con las ya entrenadas para ver en que parte del mapa cae y así poder determinar la pronunciación de la palabra.

E. Algoritmo de DTW

Para utilizar el algoritmo DTW, se tomaron las voces capturadas sin pasarlas por la conversión del LPC, ya que como el algoritmo está basado en el tiempo, no serviría tener los coeficientes del LPC que representan la frecuencia.

Se tomó un patrón como base para poder comparar con las voces que se van a reconocer, por lo cual se hizo una selección meticulosa sobre cada una de las palabras, por lo que se decidió tomar un promedio de cada una de las palabras a reconocer para tenerlas como patrón de referencia.

F. Algoritmo de HMM

A primera instancia se realizó una conversión de coeficientes de LPC por letras para un fácil manejo en la evaluación del algoritmo de Viterbi con las probabilidades de transición. Dicha conversión que toma los promedios obtenidos de cada una de las palabras de la base de datos, comparando cada coeficiente de los promedios con cada coeficiente recibido de la pronunciación efectuada, para obtener una distancia euclídea, y así la menor distancia entre el coeficiente de la palabra recibida y los promedios es la letra a la que se convertirá dicha secuencia.

Una vez tenida la secuencia de letras de la palabra pronunciada, se introduce al algoritmo de Viterbi para evaluar el comportamiento de la palabra y su probabilidad de que corresponda a una de las palabras guardadas en la base de datos. El HMM que se utilizó para la evaluación del algoritmo de Viterbi se presenta en la figura 5.

El algoritmo de Viterbi se inicializa con una de las letras establecidas, dependiendo de cual de ellas tuvo el mayor número de repeticiones en la secuencia de letras, por ejemplo si se pronunció la palabra casa y al realizar la conversión a letras se obtuvo que en la secuencia se tienen más letras A, se inicializa con las probabilidades de A, si hubiera sido B, se inicializaría con las probabilidades de B, y así sucesivamente.

Finalizando la evaluación se obtienen once probabilidades, donde la mayor probabilidad será la palabra pronunciada.

G. Modelo de Mayoría

Una vez que se tienen los reconocimientos por parte de cada uno de los algoritmos establecidos, se procede a realizar una decisión de mayoría que nos dará el resultado final del sistema y determinará que palabra se pronunció.

En la figura 6 se presenta el diagrama de flujo del reconocimiento de voz empleado en este documento, tomando en cuenta que XI es un vector ya ajustado y Xi es el vector de los coeficientes de LPC

Resultados

Se realizaron varias pruebas al sistema de manera individual para cada algoritmo, y de forma conjunta, tomando como base las pronunciaciones de los números del 0 al 9 y la palabra fin, para poder tener a futuro una aplicación en concreto, por ejemplo poder digitar un teléfono con la voz, o mover una silla de ruedas, etc.

Los resultados encontrados se muestran en la tabla 1, donde se puede observar que de manera individual que se está estableciendo cada reconocedor alrededor del 80%, pero al juntarlos en el modelo de mayoría se obtuvo un promedio del 97%, que era el modelo que nos interesaba. Cabe hacer mención que las pruebas aquí presentadas se realizaron a un solo hablante y se utilizaron os algoritmos para reconocer la palabra en tiempo discontinuo.

Conclusiones

Algunas de las conclusiones que se encontraron en el desarrollo del trabajo son las que se nombran en los siguientes párrafos.

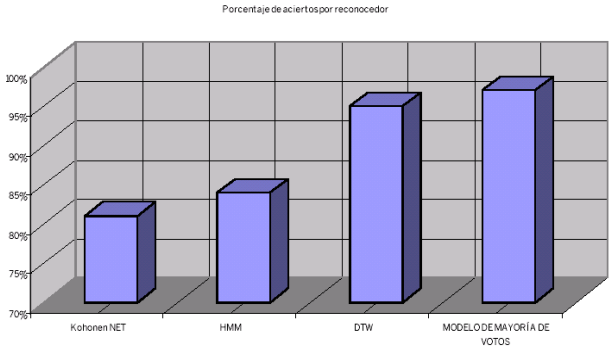

Fig. 7. Porcentaje de aciertos por reconocedor.

Evidentemente un algoritmo de red neuronal aplicado a hardware presenta el problema del entrenamiento por el tiempo y cálculos que conlleva, pero después de ello el algoritmo es rápido, y aunque el sistema presentado en este trabajo al llevarlo a hardware necesitará más equipo y más proceso de datos, también nos dará mejor resultado que usar un algoritmo individual para el reconocimiento.

La ventaja de usar un sistema híbrido como el que se presenta, es que se encuentra con mayor eficiencia en el acierto de las pronunciaciones, porque dos algoritmos pueden cubrir a uno del error y seguir dando una eficiencia en el reconocimiento, como se puede observar en la Figura 7.

La Codificación por Predicción Lineal (LPC) es una de las técnicas más eficaces de parametrización de la voz, cabe señalar que el análisis por LPC proporciona un conjunto de parámetros que representan la señal de voz y que permiten su adecuada reconstrucción. El modelo de producción de voz simula las cavidades del tracto vocal mediante un filtro lineal modelando los modos de excitación mediante dos tipos de señales. Estas señales son: ruido blanco que representa los segmentos no sonoros de la señal y trenes de pulso que representan los segmentos sonoros.

El algoritmo de *DTW* ofrece una gran eficiencia y nos da la oportunidad de comparar dos señales que en tiempo son distintas pero en contexto son iguales, ofreciendo una mayor precisión en el reconocimiento, a lo que se suma la conclusión de que para poder tener

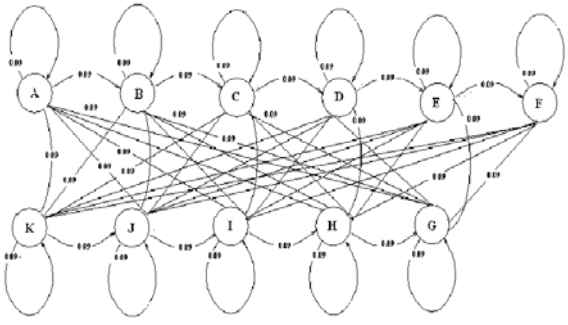

Fig. 5. Modelo Oculto de Markov utilizado para la evaluación del algoritmo de Viterbi.

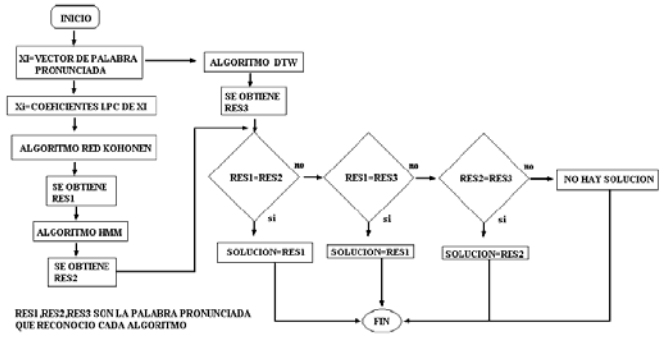

Fig. 6. Diagrama de flujo del Modelo de mayoría empleado en el reconocimiento de voz.

un mejor desempeño del algoritmo de *DTW* se debe tener en cuenta el modelo que se va a utilizar como patrón de comparación debido a que una selección errónea de éste puede ocasionar errores en el reconocimiento.

Como se puede ver en la figura 7, el *DTW* es el algoritmo que mejor porcentaje de aciertos tiene, pero aun y cuando se vea que es mayor el porcentaje se tiene que compensar la parte que el algoritmo no cubre por ejemplo en algunos casos donde es muy bajo el porcentaje, los otros dos algoritmos cubren esa parte dando una mejor eficiencia en el reconocimiento de voz del sistema.

El uso de una base de datos adecuada es importante, debido a que la pronunciación de las personas no siempre es igual y se debe tratar de dejar una base de datos uniforme y no tomar en cuenta las características más extremas, como lo puede ser la palabra más rápida ó la palabra que se grita, debido a que en el cálculo de los promedios con los que se maneja tanto la red neuronal y el HMM, puede verse afectado de una manera considerable por valores extremos que nos pueden dar un reconocimiento erróneo en la aplicación.

El entrenamiento de la red *Kohonen* debe ser el suficiente y adecuado para poder tener una clasificación adecuada de índices de cada palabra y tener un reconocimiento correcto, al igual que se debe tener cuidado con no sobre entrenar a la red porque en lugar de traer beneficios puede ofrecernos desventajas.

El algoritmo de HMM debe tener probabilidades adecuadas para la evaluación del algoritmo de Viterbi, ya que si no se tienen bien planteadas se pueden tener errores en el reconocimiento, por lo cual la rutina de conversión de coeficientes LPC a letras es de suma importancia ya que es la que determina el tipo de probabilidades a usar y con ello se debe tener bien establecidos los promedios para un desempeño eficiente.

Palabra Pronunciada	ANN	HMM	DTW	Modelo de Mayoría
1	80%	75%	95%	100%
2	90%	90%	85%	100%
3	80%	90%	100%	90%
4	85%	90%	95%	100%
5	95%	90%	100%	100%
6	65%	75%	100%	100%
7	85%	85%	95%	95%
8	70%	70%	95%	95%
9	85%	85%	100%	95%
0	65%	75%	80%	100%
Fin	95%	100%	100%	95%
Porcentaje de aciertos	81%	84%	95%	97%

TABLA I

PORCENTAJE DE ACIERTOS POR PALABRA Y POR ALGORITMO DE RECONOCIMIENTO.

El sistema de reconocimiento de voz con los tres algoritmos ofrece una decisión simple y de mucha utilidad llamada mayoría de votos, que nos da la certeza de que dos o los tres algoritmos reconocieron la misma pronunciación y de una manera alta es correcta, evitando en algunas partes el error que nos presentaría un solo algoritmo.

La taza de reconocimiento del sistema de reconocimiento de voz supera en amplia forma al uso de las redes neuronales y HMM, con lo cual se puede concluir que es más eficiente que estos dos algoritmos y al analizar el *DTW* comparado con el sistema de reconocimiento de voz podemos decir que son casi similares pero que le da una ventaja al sistema de reconocimiento y es que si el algoritmo de *DTW* falla, el sistema compensa con los otros dos algoritmos la falla, cosa que no sucedería en un algoritmo individual de *DTW*.

Referencias

[1] Bernal Bermúdez Jesús, Bobadilla Sancho Jesús, *Reconocimiento de Voz y Fonética Acústica*, Ed. Alfaomega, pp 8-9

[2] Zaknich Anthony, *Principles of Adaptive Filters And Self-Learning Systems*, Editorial Spinger, U.S.A, pp 142.

[3] Acosta B. María Isabel, Zuluaga M. Camilo A., Salazar I. Harold, "Tutorial de Redes Neuronales" Universidad Tecnológica de Pereira, http://ohm.utp.edu.co/neuronales/Capitulo2/Competitivas/Kohonen.htm

[4] Martín del Brío Bonifacio, Sanz Molina Alfredo, *Redes Neuronales y Sistemas Difusos*, Ed. Alfaomega, pp 89-96

[5] Martín del Brío Bonifacio, Sanz Molina Alfredo, *Redes Neuronales y Sistemas Difusos*, Ed. Alfaomega, pp 90