Red neuronal artificial backpropagation para el control de un Androide autónomo

Sergio MONDRAGÓN CONTRERAS
Escuela Profesional de Ingeniería de Sistemas e Informática, Universidad Tecnológica de los Andes
Abancay, Apurímac 03001/Apurímac, Perú

Marleny PERALTA ASCUE
Escuela Profesional de Ingeniería de Sistemas e Informática, Universidad Tecnológica de los Andes
Escuela Académico Profesional de Ingeniería Informática y Sistemas, Universidad Nacional Micaela Bastidas de Apurímac
Abancay, Apurímac 03001/Apurímac, Perú

José Luis MERMA ARONI
Escuela Profesional de Ingeniería Civil, Universidad Tecnológica de los Andes
Escuela Académico Profesional de Ingeniería Informática y Sistemas, Universidad Nacional Micaela Bastidas de Apurímac
Abancay, Apurímac 03001/Apurímac, Perú

RESUMEN

Esta investigación consiste en desplegar un sistema que interactúe con componentes mecánicos a través de consolas lógico-electrónicas, para emular singularidades humanas. Se construyó un robot androide, cuyas extremidades son remolcados por motores mecánicos. La unidad principal de procesos es interpretado, por redes neuronales backpropagation, delegado para recibir, procesar y enviar datos al controlador principal de motores. La unidad de salida de audio del computador y audios grabados representan las voces generadas del androide. La unidad de entrada de audio es delegado para capturar los sonidos para ser procesados por redes neuronales, representado el audio del androide.

Palabras Claves: Red neuronal, androide, red neuronal backpropagation, audio

1. INTRODUCCIÓN

La inteligencia artificial es una disciplina que permite elaborar sistemas autómatas con capacidades para emular y superar la inteligencia y talento humano, reciben percepciones del entorno, y ejecutan acciones. Se apoyan en formulaciones matemáticas en tres áreas: lógica, computacional y probabilística [1]. Las redes neuronales son sistemas dinámicos auto adaptativos, se adaptan debido a la capacidad de auto ajustarse de los elementos procesales (neuronas) que componen el sistema. Son dinámicos pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones. Cuando las redes neuronales se utilizan para reconocer ciertas clases de patrones, ellas auto organizan la información usada [2]. La red llamada backpropagation creará su propia representación característica mediante la cual puede reconocer ciertos patrones.

Las redes neuronales backpropagation son procesadores de cálculos matemáticos que pueden ser representados para emular el razonamiento humano, crean su propia representación característica mediante la cual reconocen ciertos patrones. Es un tipo de red de aprendizaje supervisado, que aplica un ciclo de propagación – adaptación de dos tareas. Una vez que se ha empleado un patrón a la entrada de la red como estímulo, este se propaga desde la primera capa a través de las capas superiores de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas. Teniendo la señal de error, se procede a actualizar los pesos de conexión de cada neurona, para hacer que la red converja hasta un estado que permita clasificar todos los patrones de entrada sin errores [3].

Alguna de las técnicas informático-electrónicas como el razonamiento de redes neuronales, aprendizaje automático-profundo, reconocimiento de patrones, algoritmos genéticos, razonamiento basado en modelos, son los que se aplican para el razonamiento de los androides [1]. Un androide es un autómata antropomorfo que procura asemejar una figura humanizada, procura emular características de hábito humano, está compuesto de articulaciones mecánicas inspiradas en las articulaciones humanas que permite emular un movimiento humano. Las articulaciones mecánicas toman la funcionalidad de articular basada en los fundamentos de la biomecánica articular humana [4].

La investigación consiste en desplegar un sistema de emulación para el control de un androide que está compuesto por: 1) un coaching (entrenador de redes neuronales), que ayuda a entrenar las redes neuronales, y ponerla a punto para su uso; 2) reconocimiento de señales de sonido (MFCC) permite capturar sonidos y obtener un resumen de patrones de un sonido capturado; 3) motor de inferencia, se encarga de captura los patrones de sonido e identificar su significado y su respuesta como movimiento o sonido; 4) mapa de redes neuronales, es la agrupación de redes que se cargan en la memorial del computador; 5) diccionario de palabras (texto) en un contenedor de archivos tipo texto con un identificador que la red neuronal obtiene como resultado de una ejecución; 6) Diccionario de palabras (sonidos) es un contenedor de archivos tipo audio con un identificador que la red neuronal obtiene como resultado de una ejecución; 7) configuración y captura de palabras, es una panel de control donde se vincula un texto con un sonido en particular, ya sea para patrón de reconocimiento o síntesis; 8) Síntesis de voz, permite unir varios sonidos de palabras y reproducirlo en una oración, 9) control hardware, permite enviar y recibir una matriz de patrones para controlar los motores del androide y recibir datos de los sensores del androide.

2. MATERIALES Y MÈTODO

El diseño de la investigación es experimental, debido a que se recolectan datos, para obtener información acerca del fenómeno estudiado (Androide autónomo) con información clara y suficiente, sin error o con un error para poder controlar. Reproduce el fenómeno en una situación controlada llamada experimento [5].

En la Tabla 1 se muestra las herramientas y técnicas aplicadas en la investigación alineados a cada resultado esperado.

Tabla 1. Herramientas utilizadas en la investigación

Resultado esperado	Herramienta utilizadas
Capturar e identificar patrones de sonidos.	Java, C++. Mel Frequency Cepstral Coefficients (MFCC).
Emular razonamiento basado en hechos. Desarrollar un motor de inferencia, basada en redes neuronales.	Java. Red neuronal Backpropagation. Pic 18f4550.

En el experimento se consideró algunas capacidades humanas para realizar las diferentes emulaciones, como es el reconocimiento de voz y el razonamiento basado en hechos para emular los movimientos (cabeza, brazos y piernas).

Sistema de biomecánica articular androide

La parte electrónica del androide tiene como plataforma principal, una placa madre, que contiene zócalos, para conectar el arreglo de drivers de los motores CC y recibir datos de los diferentes sensores que se puedan incluir. La placa madre está gobernada por un microcontrolador PIC 18f4550 que recibe órdenes del computador a través del puerto USB.

Los drivers tienen la funcionalidad de encender, apagar, cambiar de polaridad a los motores CC, esa gama de funcionalidades proporcionan al androide la animación necesaria para emular el comportamiento mecánico de un ser humano (ver figura 1).

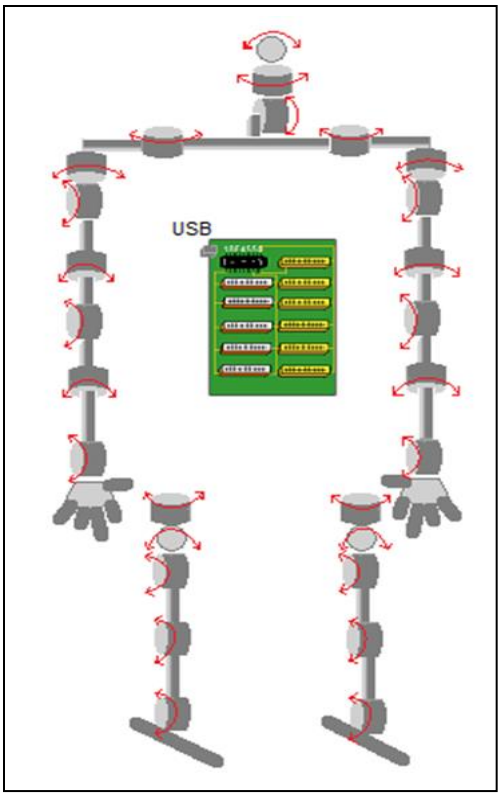

Figura 1. Esquema de la mecánica de movimientos del androide.

3. RESULTADOS

Captura he identificación de patrones de sonidos

Se utilizó el micrófono de una computadora personal, se desarrolló un algoritmo que captura 512 patrones de sonido, para luego ser procesados por el algoritmo MFCC (*Mel Frequency Cepstral Coefficients*). Se procede a analizar los patrones de sonidos con el algoritmo de MFCC el cual genera un arreglo de 12 patrones para luego ser identificado por las redes neuronales (ver figura 2).

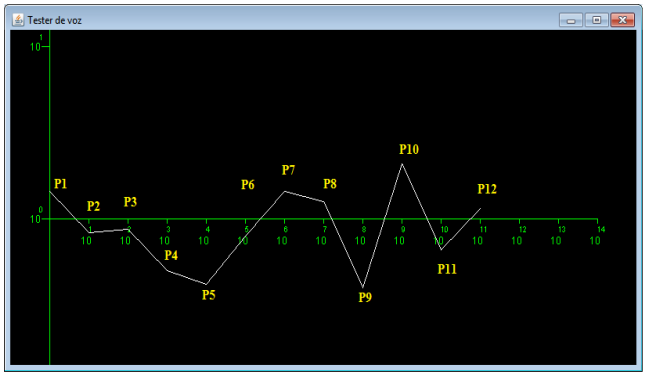

Figura 2. Patrón de sonido generado por el algoritmo MFCC.

Luego de obtener los 12 patrones, se inyectan a variables globales, las cuales son consultadas por diferentes redes neuronales.

En la figura 3, se muestra un diagrama de flujo de interacción entre el algoritmo de parametrización de habla MFCC y los algoritmos de redes neuronales recibiendo los patrones de sonido, para la identificación del mismo. Existen varias docenas de redes neuronales que tiene como objetivo identificar un determinado patrón, las salidas de las redes neuronales deben mostrar un margen de error no superior al límite (0.1), si es que se cumple el caso se abra identificado satisfactoriamente el patrón de sonido.

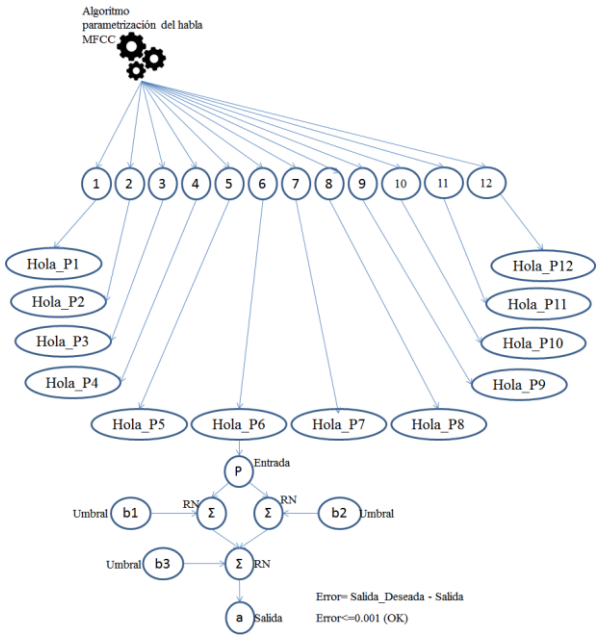

Figura 3. Diagrama de flujo del algoritmo de captura he identificación de patrones de sonidos.

Se realizaron una serie de pruebas para verificar la reacción del prototipo computacional en la captura he identificación de patrones de sonidos. Se consideró una lista de 8 palabras, de las que se realizó 10 intentos por cada palabra teniendo como resultados lo se aprecia en la Tabla 2.

Tabla 2. Resultados de la pruebas de captura he identificación de patrones.

Palabras	**Intentos satisfactorios**	**Intentos fallidos**	**Total Intentos**
Hola	7	3	10
Como	8	2	10
Estas	6	4	10
Tú	8	2	10
Nombre	6	4	10
Avanza	7	3	10
Retrocede	7	3	10
Saluda	6	4	10
TOTAL	**55**	**25**	**80**

Del 100% de pruebas realizadas se obtiene que el 31% de las pruebas son fallidas, equivalentes a 25 pruebas y un 69% de las pruebas son satisfactorias equivalentes a 55 pruebas de 8 palabras donde:
El Error promedio: **3.125**.
La desviación estándar: **0.78062475**
El error estándar de la prueba: **0.275992527**

Emulación del razonamiento basado en conocimiento de hechos

Se cuenta con algoritmos que almacenen la información en una jerarquía que se pueda asemejar al sistema de organización de tareas de un ser humano. La información se organiza en una jerarquía de directorio raíz de configuración de parámetros y de almacenamiento de redes neuronales y ficheros WAV. Se tiene tres directorios: **Config** almacena los parámetros de configuración del sistema. **Dbsound** almacena los ficheros WAV los sonidos grabados para su posterior concatenación y reproducción. Los ficheros WAV están organizados en orden alfabético. Y **Network** en este directorio raíz se almacena los eventos, sistemas, y redes neuronales y sus parámetros, dentro de este se encuentran ficheros como **event_mover_brazo, event_mover_pierna, event_oir**, y el fichero **network_humano,** donde se parametrizan los eventos que se ejecutan en el androide.
Se organizan los ficheros **system_brazo**, **system_oido**, **system_pierna,** donde se parametrizan las redes neuronales que se ejecutan en androide.

Se cuentan con directorios raíz de las entradas (input) de las redes neuronales para su entrenamiento (12 patrones procesados por el algoritmo MFCC). Directorio raíz de las salidas (output). Directorio raíz de los umbrales (threshold). Directorio raíz de los pesos (weight) que se generan después del entrenamiento de las redes neuronales.
Como ejemplo se cuenta con un fichero **Hola02016-35-04-02-35-46-463v0,** donde se parametriza la arquitectura ([1,2,1]), el puerto de entrada (1), puerto de salida (13), pesos (w_*v0), umbrales (t_*v0), entradas (i_*v0) y salida (o_*v0). La arquitectura ([1,2,1]) indica que tiene una entrada, dos neurona intermedias y una salida. Los puertos ([1:13]) indican los identificadores de la variables globales donde recolecta la información (puerto 1) y el puerto de depósito (13) donde almacena la información procesada.

Algoritmo puesta en producción de red neuronal backpropagation.

```
public class Core extends Thread {

static Logger log = Logger.getLogger(Core.class.getName());

private Net net=null;
private CommunicationObject comObjects=null;
private Integer presicione[][] = null;
private Integer arquitecture[] = null;
private double[][][] weight = null;
private double[][][] threshold = null;

public Core(Net net){
this.net=net;
comObjects=CommunicationObject.getInstance();
presicione = net.getNetoutput().getPresicione();
arquitecture = net.getNetarquitecture();
weight = net.getNetWeight();
threshold = net.getNetThreshold();
}

public void run(){
double a[][]=new double [arquitecture.length][1];

FunctionTransfer ft = new FunctionTransfer();
HashMap<Integer,Double>
port=comObjects.getMapports();

int capa=0;

NetIntput netIntput = net.getNetintput();
a[capa] = netIntput.getIntputProvabel()[0];

NetOutput outPort = net.getNetoutput();
Integer outPorts[] = outPort.getOutputport();

do{
a[capa+1] =new double [weight[capa].length];
log.debug("====CAPA=================="+(capa
+1));
for(int f=0;f<weight[capa].length;f++){
for(int c=0;c<weight[capa][f].length;c++){
a[capa+1][f] = a[capa+1][f] +
a[capa][c]*weight[capa][f][c];

log.debug("columna="+c+" fila="+f);
log.debug("entrada:"+a[capa][c]+" *
peso:"+weight[capa][f][c]+" = salida:"+a[capa+1][f]);
}

if(capa!=arquitecture.length-2){
//log.debug("Red="+net.getKey()+";
Previo="+a[capa+1][f]);
a[capa+1][f] =
ft.geRound2Decim(ft.getTanSig(a[capa+1][f]+threshold[
capa][f][0]));
log.debug("columna="+0+" fila="+f);
log.debug("entrada:"+a[capa+1][f]+" *
umbral:"+threshold[capa][f][0]+" =
salida:"+(a[capa+1][f]+threshold[capa][f][0]));
log.debug("aplica funcion de activacion TanSig:"+
a[capa+1][f]);
}else{
a[capa+1][f] =
ft.geRound3Decim(a[capa+1][f]+threshold[capa][f][0]);
//log.debug("Red="+net.getKey()+";
Resultado="+a[capa+1][f]);
log.debug("columna="+0+" fila="+f);
log.debug("entrada:"+a[capa+1][f]+" *
umbral:"+threshold[capa][f][0]+" =
salida:"+a[capa+1][f]);
net.setResultOutput(a[capa+1][f]);
}
}

capa++;
}while(capa<arquitecture.length-1);

comObjects.setNet(net.getKey(), net);

for(Integer ip:outPorts){
port.put(ip, net.getResultOutput());
}

}
}
```

Toma de decisión autónoma.
Se desarrolló un algoritmo Autómata Finito Determinista (AFD) que analizar el mejor resultado del modelo de red neuronal, obtiene el título de la etiqueta, he identifica el estado del nodo del resultado. En la figura 4, se muestra el grafo AFD y en la tabla 3 se muestra el estado de las transmisiones de los nodos.

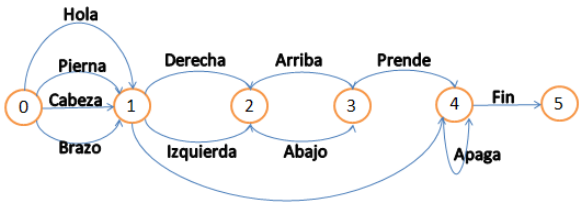

Figura 4. Grafo AFD de toma de decisiones.

Tabla 3. Tabla de transiciones del AFD de toma de decisiones

	Nodo 0	Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
hola	error	**0**	error	error	error	error
brazo	error	**0**	error	error	error	error
cabeza	error	**0**	error	error	error	error
pierna	error	**0**	error	error	error	error
izquierda	error	error	**1**	error	error	error
derecha	error	error	**1**	error	error	error
arriba	error	error	error	**2**	error	error
abajo	error	error	error	**2**	error	error
prende	error	error	error	error	**3**	error
apaga	error	error	error	error	**4**	error
fin	error	error	error	error	error	**4**

Interface de entrenador de Redes Neuronales

En la figura 5, se muestra la pantalla que permite la captura de sonidos en formato WAV, y obtener un patrón de 12 valores que luego serán la entraras de las redes neuronales. En esta pantalla se puede configurar el evento, sistema, grupo de red neuronal donde se agruparan las redes neuronales clasificadas por patrón (palabras, movimientos) de acuerdo al criterio del administrador. Esta pantalla incluye la funcionalidad de entrenar manualmente las redes neuronales.

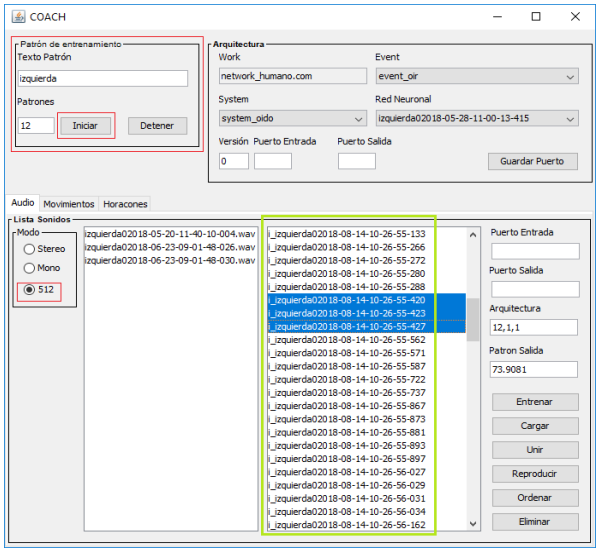

Figura 5 Interfaz de usuario para grabar archivos planos de parámetros de sonido.

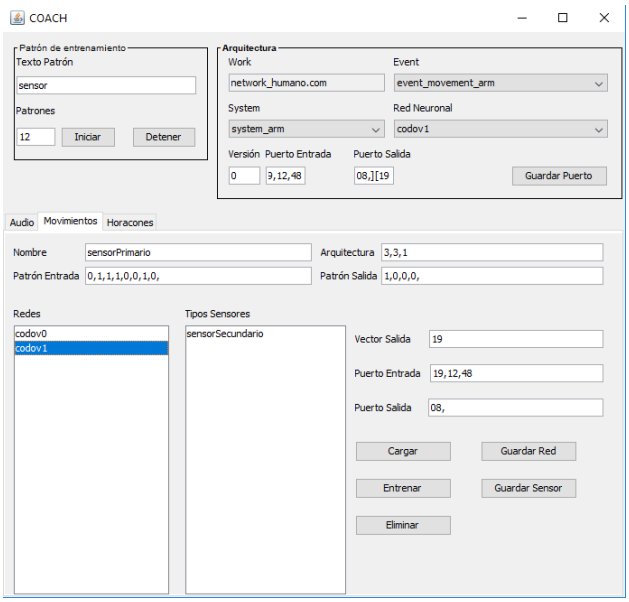

Figura 6. Interfaz entrenador de redes neuronales.

Cuando el interlocutor humano habla, da una orden o realiza una pregunta, el androide procede a revisar su base de conocimientos e identifica sus respuestas, estas respuestas pueden ser palabras (sonidos) o movimientos. Se entrenó redes neuronales para almacenar algunas respuestas (identificador de respuestas) respecto a diferentes entradas (identificador de preguntas / identificador de palabras). En la Tabla 6, se muestra las diferentes reacciones del androide respecto al interlocutor humano, teniendo en cuenta que los patrones de sonido han sido identificados correctamente, en esta prueba solo se evalúa la reacción del androide a órdenes y preguntas correctamente identificadas.

Tabla 4. Resultados de la prueba de razonamiento y autonomía.

Interlocución		**Intentos satisfacto-rios**	**Intentos fallidos**	**Total intentos**
Humano	**Androide**			
Hola	Hola cómo estás	10	0	10
Hola cómo estás	bien	10	0	10
Tu nombre	Grievous	10	0	10
Avanzar	movimiento de motores de piernas hacia adelante	10	0	10
Retroceder	Movimiento de motores de piernas hacia atrás	10	0	10

Saludar	Movimiento de brazos	10	0	10
	TOTAL	**60**	**0**	**60**

De los resultados apreciados en la tabla 4, se observa que de las pruebas realizadas no se presentan errores, la razón es que para cada pregunta u orden recibida por el interlocutor humano existe una red neuronal asignada asegurando el resultado correcto.

Figura 7. Androide construido.

4. CONCLUSIONES

El emular el comportamiento racional, mecánico, humano es un desafío que muchas empresas, centros de investigación han emprendido, en estos últimos años. Así como, la construcción de androides con capacidades de autonomía aceptables como el androide Atlas de Dynamic Boston.

Las redes neuronales son una técnica viable para poder procesar información probabilística, muy habitual en sistemas con inteligencia artificial, por la enorme variedad de información imprecisa, ambigua. Si se acompaña de algoritmos que gestionen su correcto funcionamiento existen una enorme probabilidad de generar un agente con inteligencia artificial.

La investigación ha demostrado que se puede desarrollar un sistema autónomo que permite emular algunos comportamiento básicos de un humano, como escuchar, tomar decisiones en base a experiencia de hechos, y hablar reproduciendo sonidos previamente recordados, dado que el ser humano es casi la perfección en cuanto a evolución se trata, no es fácil traducir todo el raciocinio de un ser humano en fórmulas matemáticas dada que el ser humano equilibra sus decisiones en base a recuerdos, datos obtenidos del entorno y trivialidades del momento que no necesariamente son racionales. Uno de los métodos que nos aproxima en plasmar el comportamiento de un ser humano son las redes neuronales ya que trabajan con datos atípicos y retorna decisiones esperadas o similares al de un humano.

5. REFERENCIAS

[1] L. Gomez. “Simpáticos, pero no empáticos: a la sombra de los Robots Androides”. Revista Crítica. Julio 2019. [En línea] Available: http://www.revista-critica.es/2019/07/08/ simpaticos-pero-no-empaticos-a-la-sombra-de-los-robots-androides/

[2] J. Hilera & V. Marines. Redes Neuronales Artificiales. Fundamento, modelos y aplicaciones. Madrid: Ra-Ma. 1995

[3] Universidad Tecnológica de Pereira. Facultad de Ingeniería Eléctrica. Tutorial redes neuronales. Pereira, Colombia. 2000. [En línea] Available: http://medicinaycomplejidad.org/pdf/redes/Backpropagation.pdf

[4] Instituto Medico Leloir. Fundamentos de biomecánica articular. [En línea] Available: http://imedleloir.com.ar/documentos/Biomecanica_articular.pdf

[5] S. Gómez, S. Metodología de la investigación. México: Red Tercer Milenio. 2012

.