UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN

Manual de prácticas de Sistemas Inteligentes

Departamento: Ingeniería

Sección: Electrónica

Clave de Carrera: 130

Asignatura: Sistemas inteligentes

Clave de la asignatura: 1840

Autor: David Tinoco Varela

Fecha de elaboración: Junio de 2020

Fecha de revisión: Junio de 2024

ÍNDICE

OBJETIVOS GENERALES DE LA ASIGNATURA

- Al finalizar el curso el alumno comprenderá y aplicará las herramientas y técnicas que le permitan modelar, diseñar, integrar, programar y construir sistemas digitales con diferentes tecnologías.

OBJETIVOS DEL CURSO EXPERIMENTAL

- Aprender a generar interacciones entre hardware y software para generar sistemas que presenten inteligencia artificial.
- Analizar datos provenientes del entorno, para que un ente ingenieril tome decisiones basados en tales cúmulos de información.
- Aprender a diseñar sistemas inteligentes a partir de bases de conocimiento y PROLOG.
- Generar interacciones entre elementos sensoriales y PROLOG para adquirir entes que lleguen a conclusiones lógicas.

REGLAMENTO INTERNO DEL LABORATORIO

El presente reglamento de la sección electrónica tiene por objetivo establecer los lineamientos para el uso y seguridad de laboratorios, condiciones de operación y evaluación, que deberán de conocer y aplicar, estudiantes y profesores en sus cuatro áreas: comunicaciones, control, sistemas analógicos y sistemas digitales.

1. Queda estrictamente prohibido, al interior de los laboratorios

   a) Correr, jugar, gritar o hacer cualquier otra clase de desorden.
   b) Dejar basura en las mesas de trabajo y/o pisos.
   c) Fumar, consumir alimentos y/o bebidas.
   d) Realizar o responder llamadas telefónicas y/o el envío de cualquier tipo de mensajería.
   e) La presencia de personas ajenas en los horarios de laboratorio.
   f) Dejar los bancos en desorden y/o sobre las mesas.
   g) Mover equipos o quitar accesorios de una mesa de trabajo.
   h) Usar o manipular el equipo sin la autorización del profesor.
   i) Rayar y/o sentarse en las mesas del laboratorio.
   j) Energizar algún circuito sin antes verificar que las conexiones sean las correctas (polaridad de las fuentes de voltaje, multímetros, etc.).
   k) Hacer cambios en las conexiones o desconectar el equipo estando energizado.
   l) Hacer trabajos pesados (taladrar, martillar, etc.) en las mesas de trabajo.
   m) Instalar software y/o guardar información en los equipos de cómputo de los laboratorios.
   n) El uso de cualquier aparato o dispositivo electrónico ajeno al propósito para la realización de la práctica.
   o) Impartir clases teóricas, su uso es exclusivo para las sesiones de laboratorio.

2. Es responsabilidad del profesor y de los estudiantes revisar las condiciones del equipo e instalaciones del laboratorio al inicio de cada práctica (encendido, dañado, sin funcionar, maltratado, etc.). El profesor deberá generar el reporte de fallas de equipo o de cualquier anomalía y entregarlo al responsable de laboratorio o al jefe de sección.

3. Los profesores deberán de cumplir con las actividades y tiempos indicados en el "cronograma de actividades de laboratorio".

4. Es requisito indispensable para la realización de las prácticas que el estudiante:

a) Descargue el manual completo y actualizado al semestre en curso, el cual podrá obtener en (http://olimpia.cuautitlan2.unam.mx/pagina_ingenieria/)
b) Presente su circuito armado en la tableta de conexiones para poder realizar la práctica (cuando aplique), de no ser así, tendrá una evaluación de cero en la sesión correspondiente.
c) Realizar las actividades previas y entregarlas antes del inicio de la sesión de práctica, de no ser así, tendrá una evaluación de cero en la sesión correspondiente.

5. Estudiante que no asista a la sesión de práctica de laboratorio será evaluado con cero.

6. La evaluación de cada sesión debe realizarse con base en los criterios de evaluación incluidos en los manuales de prácticas de laboratorio y no podrán ser modificados. En caso contrario, el estudiante deberá reportarlo al jefe de sección.

7. La evaluación final del estudiante en los laboratorios será con base en lo siguiente:

a) (Aprobado) Cuando el promedio total de todas las prácticas de laboratorio sea mayor o igual a 6 siempre y cuando tengan el 90% de asistencia y el 80% de prácticas acreditadas con base en los criterios de evaluación.
b) (No Aprobado) No cumplió con los requisitos mínimos establecidos en el punto anterior.
c) (No Presentó) Cuando no asistió a ninguna sesión de laboratorio o que no haya entregado actividades previas o reporte alguno.

8. Profesores que requieran hacer uso de las instalaciones de laboratorio para realizar trabajos o proyectos, es requisito indispensable que las soliciten por escrito al jefe de sección. Siempre y cuando no interfiera con los horarios de los laboratorios.

9. Estudiantes que requieran realizar trabajos o proyectos en las instalaciones de los laboratorios, es requisito indispensable que esté presente el profesor responsable del trabajo o proyecto. En caso contrario no podrán hacer uso de las instalaciones.

10. Correo electrónico del buzón para quejas y sugerencias para cualquier asunto relacionado con los laboratorios (seccion_electronica@cuautitlan.unam.mx).

11. El incumplimiento a estas disposiciones faculta al profesor para que instruya la salida del infractor y en caso de resistencia, la suspensión de la práctica.

12. A los usuarios que, por su negligencia o descuido inexcusable, cause daños al laboratorio, materiales o equipo deberá cubrir los gastos que se generen con motivo de la reparación o reposición, indicándose en el reporte de fallas correspondiente.

13. Los usuarios de laboratorio que sean sorprendidos haciendo uso indebido de equipos, materiales, instalaciones y demás implementos, serán sancionados conforme a la legislación universitaria que le corresponda, según la gravedad de la falta cometida.

14. Los casos no previstos en el presente reglamento serán resueltos por el Jefe de Sección, de acuerdo con los lineamientos generales para el uso de los laboratorios en la Universidad Nacional Autónoma de México.

CRITERIOS DE EVALUACIÓN

C1	Actividades previas indicadas en el manual de practicas	10 %
C2	Análisis e interpretación de resultados	20%
C3	Toma de lecturas correctas	30%
C4	Reporte entregado con todos los puntos indicados	40%

INSTALACIÓN DE MATLAB

Para el desarrollo de las prácticas de este manual es necesario la descarga y uso de Matlab, el cual puede ser obtenido e instalado desde https://www.software.unam.mx/producto/matlab/, recordando que esta es una licencia libre para toda la comunidad UNAM. Para descargar el software es necesario tener una cuenta institucional de correo electrónico, lo que cubrirá la versión de trabajo *offline* y la versión *online*.

Los requisitos mínimos de instalación para la última versión (2022) son: Sistema operativo Windows 10 o mayor (Windows 7 sigue siendo soportado en versiones anteriores de Matlab); Procesador AMD o Intel de x86-64; Memoria RAM de 4 GB y recomendada 8 GB; Y una capacidad de almacenamiento de 5 a 8 GB para una instalación típica, considerando 31.5 GB para una instalación completa (Lo cual no es necesario para estas prácticas).

INTRODUCCIÓN

La inteligencia artificial (IA) ha cobrado una gran relevancia en los últimos tiempos, logrando que hoy en día casi cualquier, si no es que toda, rama del conocimiento la utilice dentro de sus campos de investigación. Hoy la podemos encontrar en sistemas de análisis biológico, financiero, mercantiles, psicológicos e incluso en el estudio de lenguajes.

Para llegar a este momento, obviamente se ha tenido que recorrer un largo camino desde 1943 cuando Warren McCulloch y Walter Pitts presentaron su modelo de neuronas artificiales, pasando por la definición del “Test de Turing” presentado en el trabajo *Computing Machinery and Intelligence* de Alan Turing en 1950, sólo por mencionar la IA moderna.

A lo largo de las décadas, se han ido generando ideas y esquemas que han ido conformando lo que hoy define el campo de la IA, sin embargo, hay que decir que todas estas ideas y representaciones abstractas son solo simulaciones “débiles” de lo que una inteligencia biológica representa, pero no por eso, estas dejan de ser útiles en diferentes aplicaciones científicas y tecnológicas.

Este campo de estudio ha tratado de emular diferentes comportamientos del cerebro humano, tales como el aprendizaje, la adaptación, la interpretación, el razonamiento lógico, el razonamiento difuso, incluso, ha tratado de imitar reacciones humanas como la intuición o las emociones, logrando con ello cantidades incontables de algoritmos y herramientas para el procesamiento inteligente de la información.

En el área de ingeniería, la IA juega un papel muy importante, ya que los ingenieros comienzan a desarrollar sistemas que recogen información del entorno por medio de sensores, procesen tal información de manera inteligente (aprendiendo, razonando, evaluando, etc) y logran que diferentes actuadores respondan de forma precisa a las situaciones particulares que el entorno les presenta. Además de la generación de esquemas que cada día requieren de más y más comunicación inteligente entre diferentes tipos de dispositivos tecnológicos, tales como computadoras, teléfonos celulares, televisiones, tarjetas bancarias, y básicamente todo dispositivo que tenga integrado una unidad de procesamiento.

Como se mencionará más adelante, una piedra angular para el correcto funcionamiento de los algoritmos existentes, es la calidad y la cantidad de información que estos reciben para “aprender” y “razonar”, por lo que los datos, así como su análisis es un área paralela muy importante de estudio.

Este manual de prácticas, está enfocado al entendimiento primario de diferentes esquemas y algoritmos relacionados a los sistemas inteligentes, tratándole de dar una perspectiva ingenieril.

Agradecimiento.

Manual apoyado con el proyecto PAPIME PE100221 y PIAPI2053.

BIBLIOGRAFÍA

1. Russell Stuart J., Norvig Peter, Artificial Intelligence: A Modern Approach, Prentice Hall, 2009.
2. F. Luger George, Artificial intelligence: structures and strategies for complex problem solving, Pearson Addison-Wesley, 2009.
3. José T. Palma Méndez y Roque Marín Morales (eds.), Inteligencia Artificial. Técnicas, métodos y aplicaciones, ed. McGraw Hill. 2008.

NOTA: La imagen que se presenta en la portada de este manual de prácticas, fue obtenida de https://pixabay.com/es/illustrations/chica-hacia-adelante-digital-2181709/ Agradecemos al autor de tal imagen.

Laboratorio de Sistemas Inteligentes

Práctica 1

Introducción: Datos

TEMAS

1. Introducción a la Inteligencia Artificial

1.1. Evolución y Campos de Estudio.

1.4. Métodos Básico

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Identificar algunas herramientas para manejo de vectores de datos.
- Identificar cuando se puede utilizar una determinada técnica, y cuando es ineficiente su utilización.
- Entender la importancia de los datos en Inteligencia Artificial.

INTRODUCCIÓN

Los datos, su obtención, procesamiento, ejecución e interpretación, presentan gran relevancia en el desarrollo de sistemas inteligentes. Básicamente, todo sistema inteligente, necesita poder tener acceso a una base de datos, o conocimiento, para poder generar aprendizaje, razonamiento, conclusiones, y decisiones. Sin el ingreso de grandes cantidades de datos, básicamente los sistemas inteligentes podrían no tener respuestas tan eficientes en su comportamiento, ya que no existirían puntos de comparación y clasificación.

A pesar de la importancia del procesamiento de datos dentro de esta área del conocimiento, estos pueden ser menospreciados por los alumnos, quienes no logran visualizar que no habría inteligencia artificial sin la adquisición y análisis de grandes volúmenes de información.

Para fines de esta asignatura, se pueden considerar dos tipos de conjuntos de datos: aquellos que forman una señal continua, y que son dependientes entre sí para encontrar un patrón, tal como los datos de una señal bio eléctrica, y aquellos datos que describen objetos, cada objeto del mismo campo será descrito por las mismas métricas, sin embargo, cada descripción es independiente de todos los demás objetos con los que se alimenta una base de conocimiento, en este caso, el patrón se encuentra analizando las similitudes de cada descripción independiente.

El procesamiento de los datos, se realiza por varios motivos, entre ellos, el identificar regiones de acción, o para suavizar los datos de entrada a un sistema inteligente para que este lo pueda "entender" más fácilmente, evitando ingresar datos crudos.

Hoy más que en cualquier otro periodo histórico, la obtención y análisis de datos juega un papel primordial para el desarrollo de sistemas inteligentes y el denominado *Deep learning*, ya que ello permite entender comportamientos científicos de diferentes áreas del conocimiento, pero también permite aprender de los comportamientos de consumo y de comportamiento de diferentes individuos y sociedades.

Un ejemplo de la importancia de los datos en nuestro mundo moderno, está fuertemente ligado a las redes sociales, ya que actualmente, son uno de los grandes acumuladores de datos de millones de usuarios, datos que van desde comportamiento externo, así como toda la identidad interna de los individuos que participan en ellas, incluyendo gustos y personalidad.

Actualmente, existen diferentes organizaciones, grupos de trabajo e individuos, que dan libre acceso a diferentes bases de datos para que puedan ser utilizadas por cualquier individuo que quiera trabajar sobre ellas, entre estos compendios de bases de datos podemos encontrar https://datos.gob.es/es/catalogo , http://deeplearning.net/datasets/ y https://archive.ics.uci.edu/ml/index.php .

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará los datos utilizado en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 , adicionalmente interpretará cada uno de los elementos y comandos que componen la práctica.
3. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una computadora con Matlab (https://www.software.unam.mx/producto/matlab/) instalado y con las paqueterías de "MATLAB Support Package for Arduino Hardware" y "Deep learning" cargadas en el software.

PROCEDIMIENTO EXPERIMENTAL

En este proceso experimental, se buscará que el alumno tenga interacción con bases de datos ya existentes y señales predefinidas, para que pueda procesarlas con procedimientos básicos. Para este fin, es necesario que el alumno descargue de https://archive.ics.uci.edu/ml/index.php el data set de "Wine quality". Cada uno de estos conjuntos de datos, presenta la descripción de la base de datos, sus clases, sus etiquetas y las características que se ingresan. Para este caso, obteniendo la descripción de dicha página web, se describe que este conjunto de datos tiene 11 variables de entrada (características cuantificables) y una variable de salida, "calidad (puntaje entre 0 y 10)".

A modo de ejemplo, en el archivo descargado “whinequality-white”, es posible observar que se han descrito 4899 muestras con las 11 características definidas. Para la primera muestra de un vino blanco se tiene que la acidez fija = 7; acidez volátil = 0.27; ácido cítrico = 0.36; azúcar residual = 20.7; cloruros = 0.045; dióxido de azufre libre = 45; dióxido de azufre total = 170; densidad = 1.001; pH = 3; sulfatos = 0.45; alcohol = 8.8. Dando un 6 como resultado de calidad general.

En la primera parte de la práctica se utilizará una base de datos con ejemplos independientes, solo con la finalidad de identificar como trabajar con tales datos.

En la segunda parte se trabajará con señales que representan un comportamiento y como se pueden ir mejorando tales señales, para que los datos procesados sean más adecuados que simples datos “crudos”. Estas señales, han sido generadas por medio de un osciloscopio/generador de funciones digital, se generaron señales con base senoidal pero con pequeños errores y estas se almacenaron como vectores para su análisis en esta práctica. El osciloscopio utilizado para la generación de señales fue el Picoscope 2203, que puede verse en la figura 1.

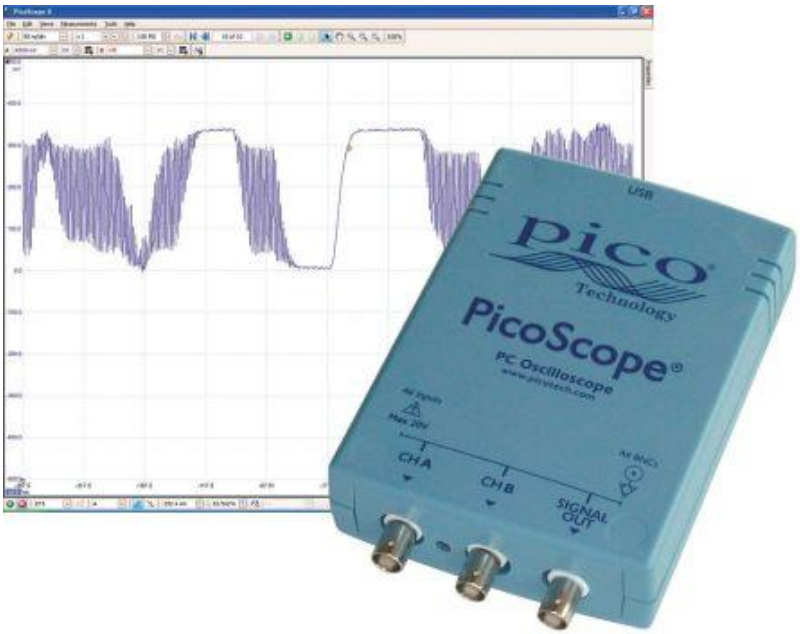

Fig. 1.- Osciloscopio digital con el que se crearon las señales para la parte 2 de esta práctica.

Una vez descrito, se procede al ejercicio práctico.

1. Ya que se ha descargado el archivo, este puede ser leído en secciones, por ejemplo, una sola columna. El alumno debe separar una columna proveniente del conjunto de datos por medio de

   columna=readvars('winequality-red.csv','Range','A1:A4899')

2. También, se puede importar todo el conjunto de datos. El alumno debe de importar este set de datos y agregarlo a la variable “datosCompletos”, por medio de

   datosCompletos= importdata('winequality-red.csv')

3. Con el paso 2, se genera un objeto completo denominado datosCompletos, que contiene los elementos siguientes, donde se muestran todos los identificadores del conjunto de datos. EL alumno debe revisar todos los indicadores e identificar su funcionalidad.

    - data
    - textdata
    - colheaders

4. Ahora el alumno puede trabajar de forma independiente con cada uno de estos elementos, por ejemplo, puede trabajar con los datos únicamente por medio de

    soloDatos=datosCompletos.data

5. El alumno debe realizar un pequeño script donde se lean los datos de una sola columna de soloDatos o de datosCompletos. Mostrar al profesor su correcto funcionamiento.

6. En la segunda parte, vamos a trabajar con datos que pertenecen a señales completas, el alumno debe de descargar el archivo senal1, y cargarlo en la variable senal1.

7. Graficar el vector correspondiente a senal1 con plot(x, senal1), donde x es un vector de 100 posiciones, al igual que señal.

8. El alumno debe abrir nuevo “live script”, identificar "task" y ejecutar “smooth data”. Selecciona como dato de entrada la variable senal1 y como método de suavizamiento "Gaussian filter". Observar lo que sucede en la pantalla derecha. Mostrar a su profesor.

9. En la parte baja del task, se despliega código generado del proceso y ahí se encuentra la variable "smoothedData", esta es la variable en la cual la nueva señal suavizada se ha almacenado, ahí podemos encontrar el vector que la representa, el alumno debe graficar ambas imágenes, la de entrada y la suavizada.

10. ¿Qué se puede observar en esta comparación?

11. Prueba diferentes métodos de suavizado. Explica que sucede.

12. Nuevamente, identificar "task" y ejecutar “Find local extrema”. Selecciona como dato de entrada la variable senal1 y como extrema type "Maxima", luego "Minima". Observar lo que sucede en la pantalla derecha. Muestra a tu profesor.

13. Identifica la variable donde se guardan cada uno de los puntos identificados y graficala

14. Descargar el archivo senal2 y cargarlo en la variable senal2.

15. Abrir un nuevo “live script”, identificar "task" y ejecutar “smooth data”. Selecciona como dato de entrada la variable senal2 y como smoothing method "Gaussian filter". Observa lo que sucede en la pantalla derecha.

16. Identifica la variable en la que se guarda la señal suavizada, imprime la señal de origen, la suavizada y compara. Enseña a tu profesor.
17. Dentro del task y ejecutar “Clean outlier data”, seleccionando como cleaning method “remove outliers” y seleccionando “detection method Generalized extreme…”. Muestra a tu profesor y describe que sucedió con la señal original.
18. Ahora selecciona en detection method, “media”.
19. Aplica el suavizado de datos nuevamente en la variable senal1, identifica la variable de salida.
20. Ingresa en el vector de la variable suavizada y aleatoriamente cambia algunas casillas de su valor numérico a NaN. Grafica el vector resultante (debe de ser una figura con espacios en blanco).
21. Selecciona nuevamente “task” y selecciona “Clean Missing Data”. En la entrada de variable, selecciona la variable de salida del paso 19, que puede tener el nombre “smoothedData”. En “cleaning method”, selecciona “fill missing”, linear interpolation.
22. En esa misma ventana, selecciona como “cleaning method”, “next value” y “provious value”.

CUESTIONARIO

1. En el punto 16, ¿Qué importancia tiene la suavización de la señal?, ¿Para qué nos es útil en un sistema inteligente?
2. En el paso 18 ¿Qué representa el comportamiento de la señal de salida?, ¿Por qué sucede tal comportamiento?
3. Estos comportamientos tan diferentes entre señales de salida, ¿Qué te indican con respecto a las herramientas??
4. ¿Todas las herramientas pueden utilizarse para todos los casos?
5. En los pasos 21 y 22, ¿Cuál de los tres casos se acerca más al caso correcto?, ¿A qué se debe esto?
6. Bajar los datos de Iris de la página https://archive.ics.uci.edu/ml/index.php, convertirlo en Excel e importarlo a Matlab, obviamente considerando matrices bien formadas y no un solo vector separado por comas.

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Inteligentes

Práctica 2

Árboles de decisión

TEMAS

4.3. Aprendizaje mediante Árboles de Decisión.

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Entender el aprendizaje artificial por medio de árboles de decisión.
- Ingresar datos desde un sistema físico para que estos sean analizados por medio de un árbol de decisión.
- Interpretar las reglas generadoras de un árbol de decisión.

INTRODUCCIÓN

Existen diferentes tipos de aprendizaje artificial, entre los que se encuentra el aprendizaje inductivo. Este aprendizaje, basa su funcionamiento en la recolección de ejemplos de un tipo particular y a partir de tales ejemplos encuentra patrones.

Es necesario seccionar cada ejemplo en sus respectivas características o atributos, ya que son estas características las que alimentaran al sistema inteligente. A partir de estos atributos, el árbol de decisión generará una clasificación de los ejemplos que alimentan el aprendizaje del sistema.

Una vez que el sistema ha aprendido y se ha generado el árbol de decisión es posible identificar patrones de nuevos ejemplos y poder predecir comportamientos futuros, es decir, una vez que el sistema ha aprendido, el usuario podrá ingresar nuevos datos para que el árbol de decisión determine la clasificación que le corresponde.

Un ejemplo sencillo de esto es que podemos definir el caso “perro”, al que le corresponden los atributos {pelo, cuatro patas, dientes, terrestre} y a este lo clasificamos dentro de mamífero; por otro lado, se define el objeto “tucán”, que cuenta con los atributos {plumas, dos patas, pico, aéreo}, el cual es clasificado con la etiqueta ave; con la información definida, un usuario podría ingresar los datos de un gato, tal que {pelo, cuatro patas, dientes, terrestre}, con lo cual el sistema fácilmente lo identificaría con la etiqueta de mamífero.

Es obvio que el ejemplo mencionado, dista mucho de ser un sistema eficiente, ya que si describimos un canguro, tal que tenga las características {pelo, dientes, dos patas, terrestre}, el sistema podría no clasificarlo adecuadamente, esto debido a que este tipo de sistemas requiere una gran cantidad de elementos de entrada para poder realizar clasificaciones de forma acertada (o al menos con un alto rango de efectividad). Esta es una de las necesidades principales de este tipo de sistemas, requieren la cantidad de datos suficientes para la clasificación, la falta de datos puede provocar un árbol ineficiente.

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará el script utilizado en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 e interpretará cada uno de los elementos y comandos que componen el script.
3. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una computadora con Matlab (https://www.software.unam.mx/producto/matlab/) instalado y con las paqueterías de “MATLAB Support Package for Arduino Hardware” y “Deep learning” cargadas en el software.
2. Una tarjeta de desarrollo Arduino (Preferentemente la versión UNO). Dependiendo de la tarjeta Arduino utilizada, original o genérica, la computadora podrá identificar diferentes tipos de puertos, siendo los puertos COM 1 y COM 3 los más comunes.
3. Dos potenciómetros de 50 ĸΩ.
4. Cables de conexión.
5. Tableta Protoboard.

PROCEDIMIENTO EXPERIMENTAL

Para este experimento, se simulará la definición de diferentes objetos, etiquetados como {1,2,..5}. Los potenciómetros 0 y 1 se encargaran de definir los atributos de cada uno de tales objetos.

1. Arme el circuito tal como se ve en la figura 1, obviamente realizando la conexión con un puerto USB de la computadora a utilizar.

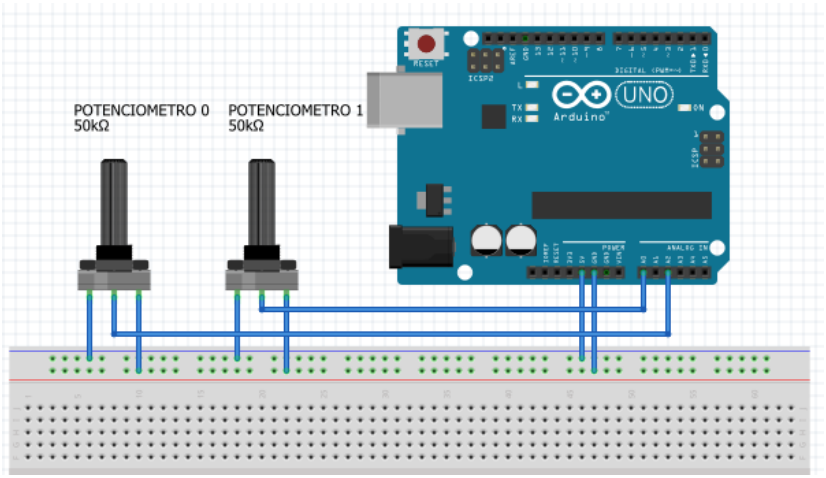

Fig. 1.- Esquema de conexiones, donde los dos potenciómetros definirán las características de las etiquetas.

2. Copiar el Script descargado de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 en un nuevo script de Matlab.

3. Se van a ingresar 30 muestras de información, para este fin, el potenciómetro 0 definirá la primera característica y el potenciómetro 1 definirá la segunda característica de las etiquetas que daremos a continuación.

4. Hay que poner los dos potenciómetros en el extremo que indique un valor cercano a 0, giro anti horario.

5. Al ejecutar el programa en Matlab, este nos pedirá el número de muestras a guardar, hay que indicar 30 muestras. Estas 30 muestras serán divididas en 5 etiquetas, enumeradas entre 1 y 5.

6. El programa nos ira pidiendo que llenemos el vector de “clase” este vector será llenado en la siguiente forma: si tenemos 30 muestras, estas serán divididas entre el número de etiquetas, es decir que corresponden a 6 posiciones del vector por etiqueta. Quedando un vector en la forma vec[1],…,vec[6]=1; vec[7],…,vec[12]=2; vec[13],…,vec[18]=3….

7. Una vez llenado el vector de las etiquetas, ahora serán llenados los vectores que representan las características de cada una de esas etiquetas (en este caso, sólo se consideran dos características, pero puede ser una cantidad *n* de ellas), por lo que el programa irá pidiendo que se ingresen los valores, cada que el programa reciba un dato proveniente de la tarjeta Arduino, es necesario indicar con un enter que puede continuar con la siguiente captura.

8. En este punto, el programa pide llenar el vector de la característica 1 del objeto a simular. Para los primeros cinco valores a ingresar relativos a la característica 1, el potenciómetro 0 debe de estar en aproximadamente 1 KΩ, para las siguientes cinco características debe de estar en aproximadamente 8 KΩ, para los siguientes 5 valores debe de estar en aproximadamente 16 KΩ, los siguientes valores debe de estar aproximadamente en 24 KΩ,

para los siguientes 5 valores debe de estar en aproximadamente 32 KΩ, para los siguientes 5 valores debe de estar en aproximadamente 40 KΩ y para los últimos 5 valores debe de estar en aproximadamente 48 KΩ. Obviamente, en la interfaz de Matlab, se verán rangos de valores entre 0 y 5, cada que modifiquemos el potenciómetro. Recordando que los valores del potenciómetro mencionados son solo supuestos y **los valores reales se van a dar paulatinamente de acuerdo al giro del potenciómetro**.

9. Una vez ingresado el vector correspondiente a la característica 1, el programa pedirá que se ingresen los datos relacionados a la característica 2 de cada uno de los elementos ingresados. De la misma forma que con la característica 1, en este caso vamos a posicionar el potenciómetro 1 cerca de 0 o 1 KΩ, durante las primeras seis solicitudes de ingreso de datos, vamos a mover muy ligeramente en sentido horario el potenciómetro 1, viendo pequeñas variaciones en los valores de entrada. Después de esto, hay que dar un movimiento un poco más grande y volver a realizar giros muy pequeños del potenciómetro, cubriendo nuevamente seis datos; nuevamente se realiza un giro más grande y cubrimos seis registros con datos pequeños, así sucesivamente hasta llenar los 30 registros del vector. Se lleva un ciclo con un giro largo (relativamente) y cinco pequeños, ciclo repetido 5 veces. Vamos a obtener una tabla semejante a la tabla 1.

Tabla 1. Tabla demostrativa de cómo quedarán aproximadamente los valores ingresados por el alumno, se puede ver que por secuencias de 6, el potenciómetro 1 permanece constante, mientras que el potenciómetro 2 da primero un salto grande y después cinco pequeños.

Etiqueta	Pot 0	Pot 1	Indicaciones
1	.01	.01	Inicio
1	.01	.09	Giro pequeño
1	.01	.1	Giro pequeño
1	.01	.13	Giro pequeño
1	.01	.15	Giro pequeño
1	.01	.21	Giro pequeño
2	1.2	1.3	Giro grande
2	1.2	1.5	Giro pequeño
2	1.2	1.7	Giro pequeño
2	1.2	1.9	Giro pequeño
2	1.2	2.01	Giro pequeño
2	1.2	2.2	Giro pequeño
3	2.3	2.5	Giro grande
3	2.3	2.6	Giro pequeño
3	2.3	2.7	Giro pequeño
3	2.3	2.8	Giro pequeño
3	2.3	2.99	Giro pequeño
3	2.3	3.01	Giro pequeño
…	…	…	…
5	4.5	4.8	Giro pequeño
5	4.5	4.9	Giro pequeño

10. Identificar las reglas obtenidas de forma escrita y ver el árbol de decisión que se forma, guardar tanto la descripción como la imagen del árbol generado.

11. Ahora bien, ingresar en la ventana de comandos la siguiente sentencia, en el vector (dato1 y dato2) ingresa dos valores aleatorios en el rango entre 0 y 5, simulando las entradas de los potenciómetros. Repite esto 4 veces.

```
respuesta=predict(ctree, [dato1, dato2])
```

12. Volver a ejecutar el script, pero esta vez indicar que solo se tomaran 10 muestras. De igual forma tendremos 5 etiquetas, por lo tanto, cada cambio corresponde a dos datos, tal como se puede ver en la tabla 2. Guardar tanto la descripción como la imagen del árbol generado.

Tabla 2. Tabla demostrativa de cómo quedarán aproximadamente los valores ingresados por el alumno, se puede ver que por secuencias de 2, el potenciómetro 1 permanece constante, mientras que el potenciómetro 2 da primero un salto grande y después uno pequeño.

Etiqueta	Pot 0	Pot 1	Indicaciones
1	.1	.12	Inicio
1	.1	.16	Giro pequeño
2	1.3	1.8	Giro grande
2	1.3	1.9	Giro pequeño
3	2.4	2.7	Giro grande
3	2.4	2.8	Giro pequeño
4	3.8	3.6	Giro grande
4	3.8	3.7	Giro pequeño
5	4.7	4.8	Giro grande
5	4.7	4.9	Giro pequeño

13. Volver a ejecutar el script, pero esta vez indicar que solo se tomaran 20 muestras. De igual forma tendremos 5 etiquetas, por lo tanto, cada cambio corresponde a cuatro datos. Guardar tanto la descripción como la imagen del árbol generado.

CUESTIONARIO

1. ¿Qué se puede deducir con la comparación de los tres árboles generados?
2. ¿Qué representa cada nodo que tiene el árbol?
3. En el paso 11, ¿Qué significan los resultados obtenidos?
4. ¿Qué clasifico el árbol?

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Inteligentes

Práctica 3

Controlador difuso

TEMAS

2. El Control en Inteligencia Artificial

2.1. Sistemas de Resolución de Problemas.

2.2. Estrategias de Control.

3.2. Sistemas de Producciones o Basados en Reglas.

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Desarrollar sistemas de control basados en reglas.
- Aplicará el conocimiento borroso en un sistema de control.
- Visualizará la forma de generar sistemas ingenieriles con un control inteligente.

INTRODUCCIÓN

Los sistemas de control son uno de los aspectos más importantes dentro de ingeniería, dentro de esta rama se pueden considerar el control de dos estados, los controladores PID y en este caso, el controlador difuso.

El controlador difuso, es un sistema de control que basa su respuesta de salida en base a un motor de inferencias lógicas, en el cual se encuentran definidas diferentes reglas que definen el comportamiento del modelo. Estas reglas se componen de estructuras IF ( ) THEN( ), se encargan de condicionar los elementos de entrada del controlador, en este caso, los valores de los sensores.

Este tipo de control se puede definir como sistemas expertos en tiempo real que implementan experiencias y conocimientos humanos, que otro tipo de controladores no pueden realizar; Sirven también como sistemas heurísticos para definir cualquier sistema de control no lineal; Con suficiente conocimiento del sistema, los controladores difusos pueden lograr un mayor grado de automatización y pueden ir mucho más allá de cualquier controlador convencional con el uso de redes neuronales y algoritmos genéticos. Y uno de los aspectos más relevantes de un controlador de este tipo, si hay mucha incertidumbre o variabilidad, un controlador difuso puede ser capaz de manejar tales circunstancias.

Sin embargo, no todo es bueno, estos sistemas también presentan desventajas en su funcionamiento, tales como que la experiencia del conocimiento tiene que estar disponible todo el tiempo, no se entiende claramente y no tiene un ajuste estándar ni criterios de estabilidad, no se puede implementar para un sistema desconocido sin información, es decir, el sistema que nunca se ha ejecutado en el pasado.

Un sistema básico de control difuso, puede verse en la figura 1.

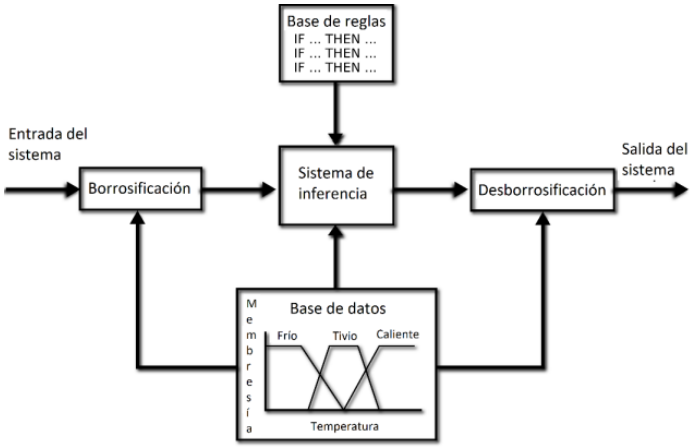

Fig. 1.- Diagrama de bloques de un controlador difuso.

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará el archivo utilizado en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 e interpretará cada uno de los elementos y comandos que componen el sistema.
3. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una computadora con Matlab (https://www.software.unam.mx/producto/matlab/) instalado y con las paqueterías de “MATLAB Support Package for Arduino Hardware” y “Deep learning” y “Fuzzy logic toolbox” cargadas en el software.
2. Una tarjeta de desarrollo Arduino (Preferentemente la versión UNO).
3. Un potenciómetro de 50 KΩ.
4. Un LDR.
5. Un LED.
6. Dos resistencias de 1 KΩ.
7. Cables de conexión.
8. Tableta Protoboard.

PROCEDIMIENTO EXPERIMENTAL

1. Armar el circuito mostrado en la figura 2.
2. Descargar el archivo .fis de la siguiente dirección web, así como copiar el script y cargarlo en Matlab: http://virtual.cuautitlan.unam.mx/intar/?page_id=786

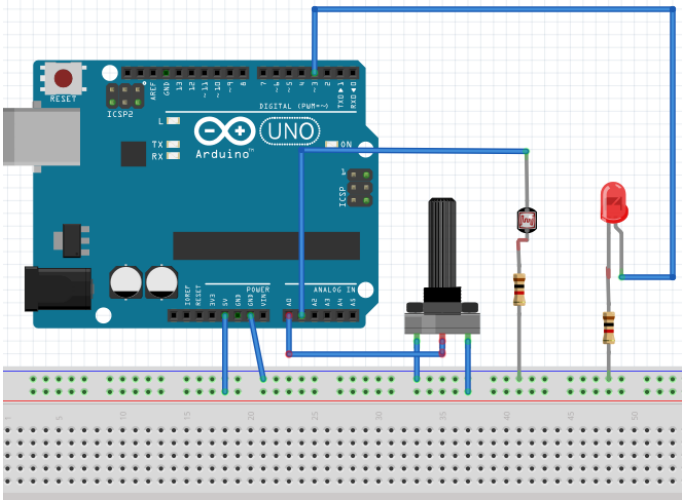

Fig. 2.- Esquema de conexión del circuito a utilizar.

3. Escribir en la ventana de comandos de Matlab, fuzzy. Lo cual desplegará una ventana en la cual se realizará el sistema difuso.
4. Por medio de la ventana desplegada, importa el archivo descargado. Lo cual mostrará la ventana de la figura 3.

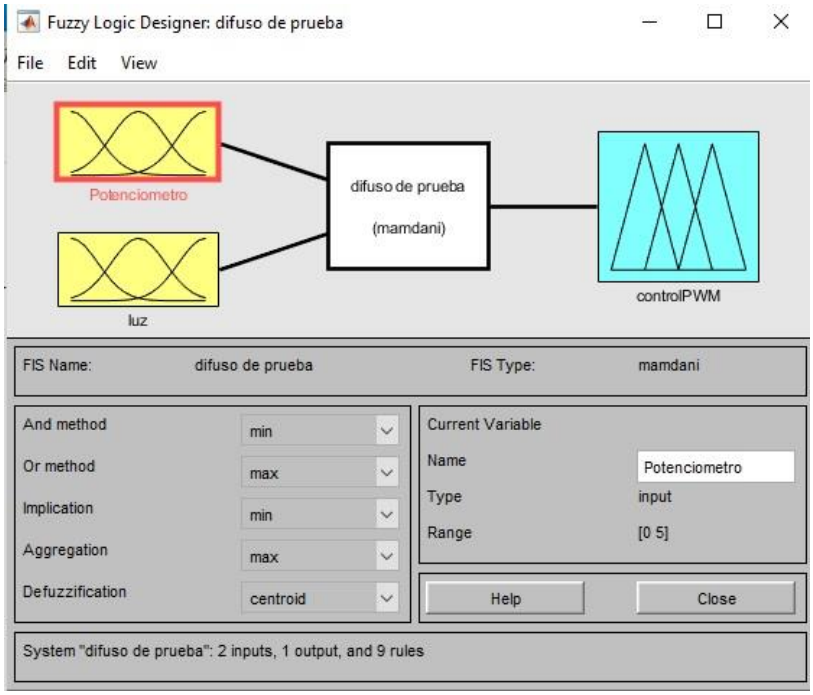

FIg.3.- Controlador pre definido

5. El alumno debe verificar los conjuntos difusos definidos, dando doble click en cada uno de los recuadros amarillos y azul.
6. Revisar las reglas que definen este controlador por medio del menú “edit -> rules”. Debe realizar el análisis de cómo se comporta el controlador y ver cuál es la relación entre los datos de entrada y de salida.
7. Modificar el script para que sólo se procese 9 veces la adquisición y respuesta de los datos. Ejecutar el script.
8. El alumno debe de modificar los valores del potenciómetro, de valores bajos a altos, realizando movimientos aleatorios, así como también debe de ingresar brillo luminoso al sensor LDR. Quedando un comportamiento aproximado al que se muestra en la tabla 1. Hay que mencionar que estos valores no son precisos, el alumno se debe de posicionar en un punto donde de una resistencia baja, una media y un alta el potenciómetro, aleatoriamente; de igual forma en el LDR, se debe buscar una incidencia de poca luz, media luz y mucha luz, para llenar la tabla con los valores obtenidos.

Tabla 1.- Comportamiento base.

Potenciómetro	**LDR**	**Salida**
Bajo	Bajo	
Bajo	Medio	
Bajo	Alto	
Medio	Bajo	
Medio	Medio	
Medio	Alto	
Alto	Bajo	
Alto	Medio	
Alto	Alto	

9. Modificar el script para que se ejecute 40 veces la adquisición y respuesta de los datos. Ejecútalo.
10. Colocar el osciloscopio, para que mida la salida del pin digital 3 de la tarjeta de desarrollo. Ir moviendo el potenciómetro lentamente, así como los valores de incidencia lumínica al LDR. Ver que es lo que sucede. Mostrar al profesor.
11. ¿Qué relación tiene este comportamiento con las reglas del controlador?
12. El alumno debe generar su propio controlador, con las mismas entradas pero definiendo 4 conjuntos difusos por entrada y definiendo sus propios conjuntos de reglas que el alumno considere lógicas y coherentes.
13. Ejecutar el controlador y observar en el osciloscopio el comportamiento del mismo. Mostrar al profesor.

CUESTIONARIO

1. Explique el controlador difuso que se realizó en la práctica, ¿Cómo actuaron las reglas en el comportamiento del mismo?
2. Suponiendo que tenemos el siguiente código, ¿Este comportamiento es el mismo que el comportamiento de las reglas difusas?

```
if(temperatura >1 && luz>1)
{
        salida=45°;
}
```

3. Genere un script que guarde todos los datos de entrada y salida en una matriz, es decir, cada valor que se ingrese con el potenciómetro y el LDR y su respectiva respuesta de salida.
4. Del rango de valores de entrada, ¿Existe algún valor para el cual el controlador no tenga respuesta de salida?

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Inteligentes

Práctica 4

Introducción a PROLOG

TEMAS

6. Lenguajes de Programación PROLOG, Common Lisp

6.1. PROLOG.

6.1.4. Objetos y relaciones.

6.1.6. Hechos.

6.1.10. Archivos.

6.1.11. Aprendizaje.

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Interpretar hechos y reglas en PROLOG.
- Implementar hechos y reglas dentro de la placa de desarrollo Raspberry Pi.
- Generar consultas dentro de Raspberry Pi.
- Generar una interacción con el entorno, a partir de las respuestas lógicas que devuelva un programa en PROLOG.
- Aprenderá la interacción entre pensamiento lógico artificial y entorno físico.

INTRODUCCIÓN

PROLOG es un lenguaje de programación nacido en la Universidad de Aix-Marseille I (Marsella, Francia), desarrollado por Alain Colmerauer y Philippe Roussel.

Aunque originalmente no estuvo pensado para ser un lenguaje de programación, ya que se buscaba que fuera una forma de procesar de forma algorítmica al lenguaje natural, este se convirtió en el primer lenguaje dirigido específicamente al desarrollo de sistemas inteligentes.

PROLOG presenta una gran potencia cuando se trata de la relación de eventos (hechos) y como estos desembocan en deducciones lógica, lo que se puede interpretar como pensamiento lógico artificial.

Este lenguaje no es secuencial, como la mayoría de los lenguajes más populares, por el contrario, este lenguaje pertenece al paradigma lógico y declarativo, es decir, importa más la descripción de lo que se debe de lograr, que la forma en la que se debe lograr.

Básicamente, un programa en PROLOG se compone de una base de conocimiento (KB), o de hechos, de un campo y la relación y reglas existentes entre tales conocimientos.

Actualmente, PROLOG es un lenguaje que puede ser ampliamente utilizado en el diseño y desarrollo de sistemas ingenieriles inteligentes.

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará los archivos utilizados en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 e interpretará cada uno de los elementos y comandos que componen el script principal.
3. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una tarjeta de desarrollo Raspberry Pi 3, cargada con el sistema operativo Raspberry Pi OS, anteriormente Raspbian (https://www.raspberrypi.org/downloads/raspberry-pi-os/).
2. Un eliminador para la tarjeta Raspberry, con un voltaje de **salida de 5 volts y 2 Amperes** para su funcionamiento adecuado (Se recomiendan 700mA, sin embargo, esta capacidad no logra cubrir los requerimientos de la tarjeta con periféricos conectados).
3. Un teclado y un mouse con entrada USB.
4. 5 LEDs de diferentes colores, preferentemente.
5. 5 resistencias de 1k.
6. Cables de conexión, macho-hembra.
7. Tableta Protoboard.

PROCEDIMIENTO EXPERIMENTAL

Este procedimiento experimental está diseñado para que el alumno interactúe con PROLOG dentro de una tarjeta de desarrollo, en este caso, Raspberry. Es importante mencionar, que se lleva a cabo de esta manera para que el alumno visualice que se puede generar una reacción en un elemento físico de salida que responda a los datos que se obtienen dentro del programa. El lenguaje busca obtener una respuesta lógica a los hechos definidos, por medio de este experimento, esta respuesta lógica puede visualizarse a través de diferentes indicadores, LEDs, y es posible verificar físicamente si existe un determinado dato verdadero dentro de las consultas realizadas a PROLOG.

1. El alumno debe de abrir una terminal, dentro de la tarjeta Raspberry, donde se ejecutaran los comandos y algunas rutinas.
2. Si no está cargado PROLOG dentro de la Raspberry, el alumno debe cargarlo por medio de:

   apt-get install swi-prolog

3. Para generar el puente entre Python y PROLOG se debe de instalar PySwip (https://pypi.org/project/pyswip/), por medio de:

sudo pip install pyswip

4. Para lograr la comunicación entre Python y PROLOG, se debe de teclear la línea:

sudo ln -s libswipl.so /usr/lib/libpl.so

5. La primera parte de este proceso, consiste en los comandos dentro de la terminal.
6. El alumno debe teclear “python”, sin las comillas, para ingresar al intérprete del lenguaje. Iniciando el proceso tecleando:

```
from pyswip import Prolog
prolog = Prolog()
prolog.assertz("tiene(tucan,plumas)")
prolog.assertz("tiene(perro,pelo)")
prolog.assertz("tiene(beta,escamas)")
```

7. Identifica el procedimiento que se está llevando a cabo en las líneas de código colocadas en el paso anterior, explica en tu reporte.
8. El alumno debe realizar la consulta “el tucán tiene…” por medio de:

```
list(prolog.query("tiene(tucan,X)"))
```

9. ¿Qué respuesta da?
10. A la muy pequeña base de conocimiento existente, hay que agregar nuevos datos, tales como

```
prolog.assertz("tiene(tucan,picoAmarillo)")
```

11. El alumno debe realizar nuevamente la consulta:

```
list(prolog.query("tiene(tucan,X)"))
```

12. ¿Se modificó el resultado? Muestra y explica a tu profesor.
13. Ingresa más características de los animales tucán, perro, beta; coloca al menos tres características que concuerden con la descripción “sujeto tiene característica”-
14. Ahora, el alumno debe teclear la siguiente línea, para ingresar una regla a la base de conocimientos:

```
prolog.assertz("tiene(X,Y,Z):-tiene(X,Y),tiene(X,Z)")
```

15. Realiza la consulta

```
list(prolog.query("tiene(tucan,A,B)"))
```

16. ¿Qué se obtiene? Muestra y explica a tu profesor.
17. El alumno debe realizar más consultas, verificando las características de los demás animales dentro de la KB, tales como:

list(prolog.query("tiene(perro,A,B)"))

list(prolog.query("tiene(beta,A,B)"))

18. El alumno debe generar dos nuevas reglas que relacionen los hechos ya existentes, o que relacione hechos nuevos que el alumno defina. Mostrar al profesor.
19. En este segundo punto, el alumno debe aprender a trabajar con un archivo o programa existente.
20. Vamos a salir de la ejecución de Python, para salir de la ejecución de algún programa en la terminal, se utiliza Ctrl + d.
21. EL alumno debe descargar el archivo "prolog.py" de la página http://virtual.cuautitlan.unam.mx/intar/?page_id=786 y colocarlo en el escritorio. Recuerden que para moverte entre directorios en la terminal, se utiliza el comando cd, en este caso "cd Desktop".
22. Armar el circuito mostrado en la figura 1, considerando las conexiones de los GPIO mostrados en la figura 2.

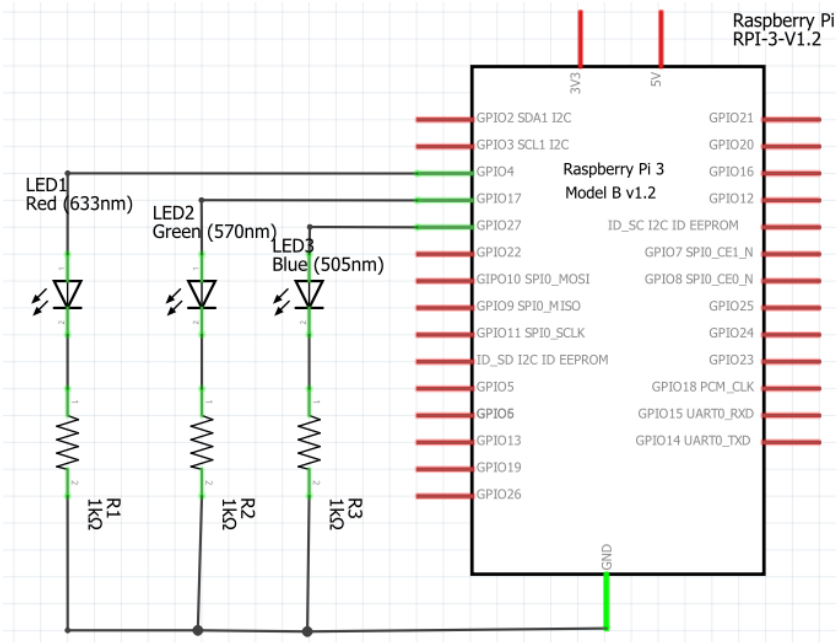

Fig. 1.- Esquema del circuito electrónico a armar.

23. Hay que posicionarse en el escritorio por medio de la terminal, recuerda que para ejecutar los programas hay que estar en el mismo directorio donde se encuentren los archivos a ejecutar o utilizar (utilizando el comando CD). Una vez posicionados en el escritorio, se ejecuta el archivo en Python, tecleando "python prolog.py"

24. Ejecuta y verifica que se llene la lista con todos los valores resultantes (valores provenientes del archivo prolog.py), verifica que se vayan definiendo cada uno de los resultados de forma individual. Verifica que los LEDs correspondan a cada uno de los resultados lógicos que da como salida la consulta realizada dentro del archivo.
25. Modifica la KB del archivo, para realizar las consultas padre(bart,Y), padre(X,bart), modificando la secuencia de los LEDs, para que cada uno de ellos indique si existe un dato determinado dentro de la KB que cumpla con la consulta requerida.
26. Realiza una modificación dentro del programa colocando la consulta que gustes, y haz que encienda un LED si la respuesta correcta existe, y que prenda un segundo LED si la respuesta correcta no existe.

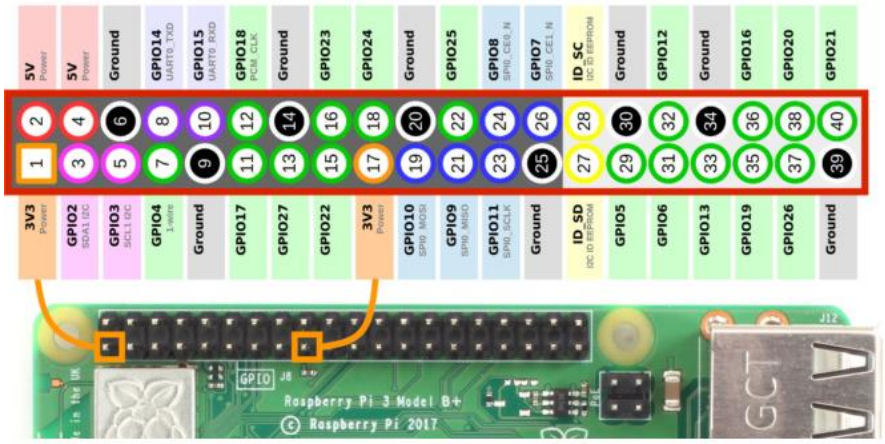

Fig. 2.- Configuración de los GPIO de la placa de desarrollo Raspberry Pi. Imagen recuperada de: https://www.raspberrypi-spy.co.uk/2012/06/simple-guide-to-the-rpi-gpio-header-and-pins/

27. El alumno debe des comentar la regla “abuelo” y ejecutar nuevamente el archivo. Mostrar al profesor los resultados de salida.
28. Agregar más personajes de la serie al archivo .py, por medio del predicado “padre(X,Y)”, para que existan más respuestas correctas en la siguiente consulta. El archivo “Simpson.pl”, contiene muchos personajes relacionados en esta forma, pueden utilizarse algunos de ellos.
29. Modifica el archivo prolog.py, para identificar cuantos resultados correctos da la consulta “abuelo” dentro del programa existen, cada respuesta correcta debe de ser indicada con un parpadeo en un LED determinado por el alumno. Es decir, si hay dos respuestas correctas, el LED debe parpadear 2 veces. Puedes utilizar banderas para identificar el número de respuestas correctas. Mostrar al profesor.
30. Genera una nueva regla que se llame “abuela” dentro del archivo .py, para esta nueva regla, será necesario agregar predicados tipo “madre(X,Y)”, utilizando los personajes de la misma serie.
31. Repita el paso 29, pero utilizando la consulta “abuela”. Mostrar al profesor.

32. Para la tercera parte, es necesario descargar el archivo Simpson.pl de la página http://virtual.cuautitlan.unam.mx/intar/?page_id=786 y colocarlo en el escritorio. Para leer este archivo, asegúrate de estar en la terminal en el fichero Desktop.
33. Teclea python.
34. Ingresa las siguientes líneas

```
from pyswip import Prolog
prolog=Prolog()
prolog.consult("simpson.pl")
```

35. Realiza las siguientes consultas desde terminal:

```
list(prolog.query("padre(X,Y)"))
list(prolog.query("padre(homero,Y)"))
```

36. Realizar consultas que muestren a los hijos de Cletus y Brandin, revisar la KB denominada simpson.pl para ver la forma en la que se deben realizar las consultas.

CUESTIONARIO

1. El alumno debe diseñar un programa .py, en donde defina una KB con al menos 50 hechos o predicados, y al menos 4 reglas dentro de él. Los hechos pueden representar, fauna y su comportamiento, comportamiento de elementos tecnológicos, identificar una falla en un vehículo, identificar alguna enfermedad, básicamente puede estar relacionado a lo que el alumno desee, cuidando que no se repitan las KB entre los alumnos que componen al grupo. En este programa, debe de existir un conjunto de identificadores (LEDs) que mostraran la cantidad de repuestas correctas que cada consulta devuelve. Así mismo, existirá un identificador aparte que se encenderá si la consulta no devuelve resultados. Este debe ser ejecutado en la placa Raspberry Pi.

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Inteligentes

Práctica 5

Decisión condicionada en PROLOG

TEMAS

6. Lenguajes de Programación PROLOG, Common Lisp

6.1. PROLOG.

6.1.4. Objetos y relaciones.

6.1.6. Hechos.

6.1.10. Archivos.

6.1.11. Aprendizaje.

6.1.12. Aplicaciones.

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Entender el uso de PROLOG en aplicaciones, tales como autómatas.
- Generar autómatas que sean capaces de mantener una conversación simple basada en condiciones.
- Definir sistemas de relación de objetos por medio de datos.

INTRODUCCIÓN

PROLOG puede ser utilizado en diferentes aplicaciones, en este caso se utilizará para la definición de autómatas, máquinas abstractas que permiten definir si una secuencia de palabras o símbolos es correcto, llegando a esta conclusión si la máquina cumple con las condiciones relacionales que existe ente cada uno de los elementos que componen el sistema.

Esta característica puede ser utilizada para la solución de problemas lógicos, es decir, un pensamiento entrelazado entre diferentes predicados que sean verdaderos para llegar a una conclusión verdadera.

En el caso de esta práctica, se buscara que la formación de ciertas palabras, construidas con un alfabeto que $\in \{0,1\}$, puedan ser identificadas como palabras verdaderas o palabras falsas dentro de un lenguaje.

Es posible observar que, con la adecuada generación de relaciones, reglas y hechos, es posible generar aplicaciones tales como transmisores/receptores en sistemas de comunicaciones, identificación de palabras de control, entre otros.

Debido a las características implícitas de PROLOG, el diseño de este tipo de sistemas se presenta de una forma simplificada.

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará el script utilizado en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 e interpretará cada uno de los elementos y comandos que componen el script.
3. El alumno descargara el programa realizado en PROLOG de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 e interpretará cada uno de los elementos y comandos que lo componen.
4. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una computadora con Matlab (https://www.software.unam.mx/producto/matlab/) instalado y con las paqueterías de “MATLAB Support Package for Arduino Hardware” y “Deep learning” cargadas en el software, adicionalmente, debe contar con el interprete SWI-PROLOG
2. Una tarjeta de desarrollo Arduino (Preferentemente la versión UNO).
3. Un switch.
4. Un pushbutton.
5. Un diodo LED.
6. Una resistencia de 1kΩ.
7. Dos resistencias de 10k Ω.
8. Cables de conexión.
9. Tableta Protoboard.

PROCEDIMIENTO EXPERIMENTAL

Para los fines de este procedimiento experimental, Matlab funcionará como el intermediario entre la placa de desarrollo y PROLOG, ingresando los datos que PROLOG posteriormente procesará. Para este fin, se generaron dos programas independientes, el script de Matlab y la base de conocimientos de PROLOG.

Se considera para la práctica un abecedario compuesto de {0,1}, que son la base para la construcción de las palabras a rechazar o aceptar.

1. Armar el circuito mostrado en la figura 1.

2. Descargar el archivo con terminación pl de la siguiente dirección web y ejecutarlo; así como también copiar el script de Matlab y ejecutarlo. http://virtual.cuautitlan.unam.mx/intar/?page_id=786

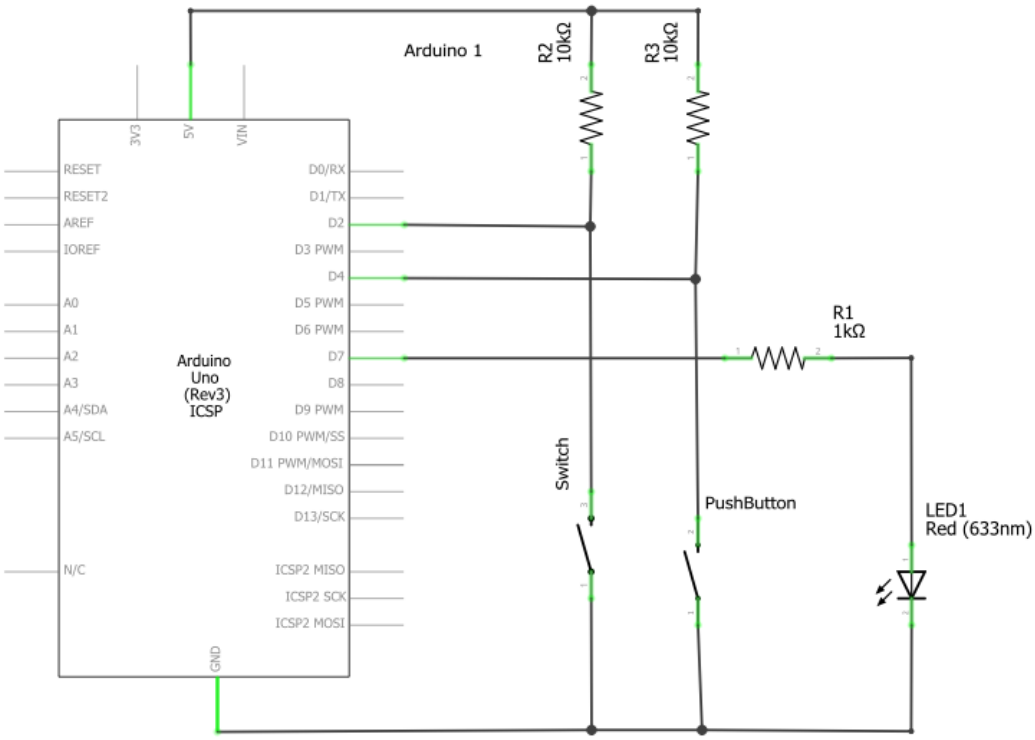

Fig. 1.- Diagrama esquemático de la práctica.

3. Como ya se mencionó, se van a ejecutar dos programas de forma independiente, por lo que se recomienda que ambas herramientas sean minimizadas para ocupar la mitad de la pantalla cada una.

   NOTA: Aquí se trabaja con archivos que utilizan rutas absolutas para su interrelación. Las rutas que están en los archivos corresponden a las rutas de la computadora en la cual se realizaron, para llevar a cabo adecuadamente esta práctica, es necesario modificar tales rutas a las rutas específicas de la computadora a utilizar. Se deben de sincronizar las rutas para buscar en los mismos archivos.

4. Primero que nada, el alumno debe ejecutar el script de Matlab, lo que definirá los datos que PROLOG utilizará.

5. El alumno debe ingresar una cadena de unos y ceros de forma aleatoria, para formar las palabras a identificar. El ingreso de estas cadenas, originalmente con una distancia de 10 bits, inicia cuando el LED se enciende, en este punto se tiene dos segundos para ingresar el dato por medio del pushbutton. Después del encendido del LED, se iniciará una secuencia de encendido y apagado, donde cada estado del LED representa el ingreso de un nuevo dato. En cuanto se cambie el estado del LED, en ese momento se ingresa el dato deseado, ya sea cero o uno.

6. El alumno debe recordar que el control de la recepción de la cadena binaria está dada por el switch, este debe de estar en 1 lógico para iniciar el ingreso de datos, y en 0 para no iniciar otro conteo. IMPORTANTE, antes de que termine la secuencia de 10 cambios de estados del LED, es importante que el alumno haya cambiado el valor del switch de 1 a 0 lógico, para evitar que se inicie un nuevo conteo de ingreso de datos y el alumno pierda la noción de la cadena ingresada.

7. Cuando se ha terminado el ingreso de datos, el alumno debe "consultar" en el intérprete de PROLOG, el archivo descargado, e iniciar la ejecución del programa con la palabra "**start.**" (**sin comillas y con el punto**). Dará la respuesta de si es una secuencia permitida en el lenguaje o no lo es.

8. PROLOG preguntará si quieres ingresar una nueva palabra, antes de indicar que sí, el alumno debe ir a Matlab y ejecutar nuevamente el script, para ingresar la nueva secuencia binaria, una vez obtenida, ya se le puede indicar que si a PROLOG. Nuevamente indicará si es una secuencia correcta o incorrecta dentro del lenguaje.

9. Repite este procedimiento hasta obtener al menos dos secuencias que si pertenecen al lenguaje y dos secuencias que no pertenezcan al lenguaje.

10. El alumno debe abrir el archivo de PROLOG, y definir el autómata que está procesando la información, y verificar que efectivamente los resultados dados por el intérprete de PROLGO son correctos.

11. El alumno debe modificar el script de PROLOG para que sea él, el que defina las longitudes de las cadenas de entrada. Este punto no afectará en nada a la base de conocimientos de PROLOG.

12. EL alumno modificará el programa en PROLOG, para que identifique si una secuencia de palabras comunes representan un saludo o no. Mostrar a su profesor.

CUESTIONARIO

1. Cómo el alumno ya habrá identificado, la secuencia binaria es pasada de Matlab a PROLOG por medio de un archivo, identificar el archivo. El alumno debe de generar un sistema de archivos, o datos, para lograr una comunicación bidireccional entre PROLOG y Matlab, es decir, que el alumno sólo deba de ejecutar una sola vez el script de Matlab, y una vez el intérprete de PROLOG. Matlab debe de identificar en que momento debe de iniciar nuevamente el ingreso de datos, y PROLOG debe identificar en que momento los datos ya están disponibles para su lectura.

2. Partiendo de lo realizado en la práctica, el alumno realizará un programa en PROLOG que intercambie al menos tres preguntas y respuestas por medio de los autómatas, es decir, el alumno saluda, PROLOG, contesta, el alumno pregunta a PROLOG, y este le contesta al alumno. Obviamente sin condicionales IF, todo por medio de la forma base dada en la práctica.

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Inteligentes

Práctica 6

Deep Learning: Detección de SPAM

TEMAS

4. Modelos y Métodos de Aprendizaje en la Inteligencia Artificial

4.1. Modelos y Paradigmas de Aprendizaje.

5.1. Conceptos y Problemas en Percepción Artificial.

5.2. Reconocimiento de Voz y Lenguaje.

6.4. Ejemplo de Aplicación.

OBJETIVOS

Al término de esta práctica el alumno podrá:

- Identificar problemas que pueden ser resueltos por medio de Deep Learning.
- Generar un identificador de SPAM en SMS que pueda despreciar los mensajes no deseados.
- Aprender el procesamiento de texto para clasificación de datos.
- Aprender a utilizar el Deep Learning para el análisis del lenguaje.

INTRODUCCIÓN

En fechas recientes, el *Deep Learning* (DL) ha ido tomando gran relevancia en el diseño de diferentes esquemas tecnológicos, esto debido a la gran capacidad de procesamiento de información que pueden ejecutar estos sistemas. El DL, es un conjunto de técnicas que permiten procesar la información como si del sistema nervioso de un mamífero se tratara, lo que las dota de una alta capacidad de interpretación y "aprendizaje".

Se define como aprendizaje profundo, debido a la cantidad que estos esquemas tienen de capas neuronales ocultas, mientras que las redes neuronales tradicionales cuentan con dos o tres capas ocultas, los esquemas de DL pueden tener hasta 150 capas ocultas. A pesar de que el concepto no

es nuevo, ha tenido un auge debido a las capacidades computacionales con las que ahora se cuenta, no sólo eso, el intercambio de información se ha vuelto masivo, lo que permite a los sistemas de DL "aprender" de forma eficiente. Esto debido a que los esquemas de DL son entrenados mediante extensos conjuntos de datos etiquetados.

Las técnicas de Deep Learning han mejorado la capacidad de clasificar, reconocer, detectar y describir. Estas son ampliamente utilizadas en aplicaciones de procesamiento de imágenes, identificación de música, lenguaje natural, procesamiento de texto, entre otros problemas de gran complejidad.

Los esquemas de DL tienen la ventaja de que estos extraen las características del problema a resolver, a diferencia de esquemas de Machine Learning (ML) "clásicos". La figura 1, muestra la diferencia de procesamiento de un sistema de ML contra uno de DL.

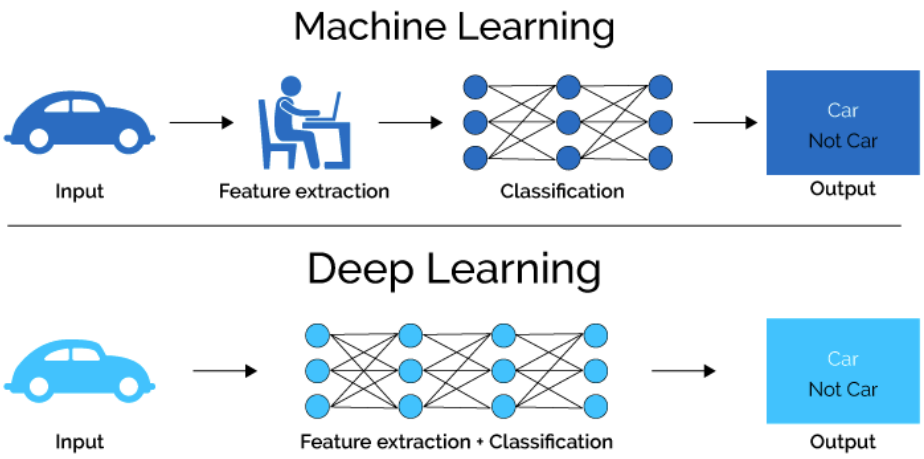

Fig. 1.- Diferencia de procesamiento de información entre machine learning y Deep learning. Imagen recuperada de https://lawtomated.com/a-i-technical-machine-vs-deep-learning/

Para esta práctica, nos vamos a centrar sobre el procesamiento de texto por medio de Deep Learning, más específicamente sobre el problema de la diferenciación de mensajes tipo SPAM y mensajes apropiados. Konstantin Tretyakov en su trabajo "Machine Learning Techniques in Spam Filtering" presenta diferentes técnicas para el procesamiento de texto y el filtrado de SPAM, lo que demuestra que el problema es un problema de interés científico.

En la figura 2, se puede apreciar una representación esquemática básica de un esquema de DL enfocado al procesamiento de texto.

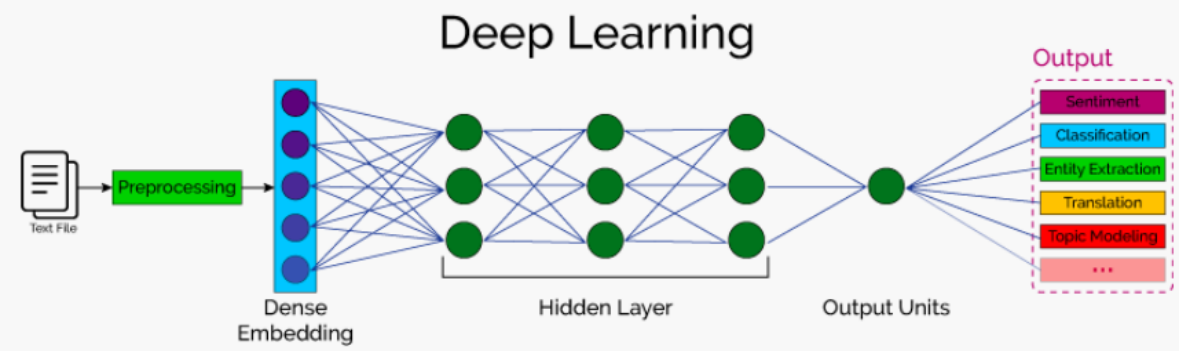

Fig. 2.- Esquema básico de una red neuronal profunda para procesar texto. Imagen recuperada de http://orionvalley.com/mldl.html

ACTIVIDADES PREVIAS A LA PRÁCTICA

1. El alumno realizará la lectura de la práctica.
2. El alumno descargará los datos utilizado en esta práctica de http://virtual.cuautitlan.unam.mx/intar/?page_id=786 , adicionalmente interpretará cada uno de los elementos y comandos que componen los scripts de la práctica.
3. El alumno debe de haber adquirido conocimiento de las herramientas a utilizar antes de la práctica.

MATERIAL Y EQUIPO

1. Una computadora con Matlab (https://www.software.unam.mx/producto/matlab/) instalado y con las paqueterías de “MATLAB Support Package for Arduino Hardware”, “Deep Learning” y “**Text Analytics Toolbox**” cargadas en el software.

PROCEDIMIENTO EXPERIMENTAL

En este procedimiento experimental se presenta el análisis de diferentes mensajes tipo SMS, entre los que se encuentran mensajes reales interpersonales y mensajes tipo SPAM. Esta es una de las muchas aplicaciones que el Deep Learning puede tener dentro de los sistemas de comunicación moderna.

EL clasificador identificara los mensajes que son reales y los mensajes tipo SPAM. Para la realización de esta práctica se utilizan los scripts y funciones provenientes directamente de la página Matworks de Matlab, solo adecuados al problema planteado. Así mismo, se utilizó una base de datos proveniente de http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/ (en inglés), a partir de esta base de datos se realizará la clasificación.

1. EL alumno debe descargar los scripts correspondientes a la práctica, de la página http://virtual.cuautitlan.unam.mx/intar/?page_id=786 y los abrirá en Matlab.

2. EL alumno descargara los archivos correspondientes al entrenamiento y verificación de información relacionada a la práctica, en este caso se tienen los archivos “spamSMS.csv”, para entrenar y “MensajesParaVerificar.csv“ para verificar el entrenamiento.

3. **Tanto archivos como scripts deben de ser colocados en la carpeta por defalut de Matlab, generalmente se encuentra dentro de “Mis documentos”.** Matlab puede trabajar directamente con los archivos .cvs, sin embargo, también se pueden copiar las celdas del archivo en una variable de Matlab, o simplemente desde la sección “fichero actual” a la sección del “espacio de trabajo” se puede realizar el llamado del archivo.

4. El alumno abrirá el archivo “spamSMS.csv” y dejara sólo 20 de los más de 5000 mensajes existentes en el archivo, estos veinte mensajes se deben de escoger de manera aleatoria, cuidando que existan al menos 10 mensajes tipo SPAM y 10 mensajes normales.

5. Ejecutar el script “ClasificadorTexto”.

6. Al realizar la ejecución, aparecerá la “Word cloud” correspondiente a los datos de entrada, guardar esta imagen. Mostrar al profesor. Continuar con le ejecución dando enter en la línea de comandos.

7. En este punto aparecerá una imagen similar a la mostrada en la figura 3.

Fig. 3.-Proceso de aprendizaje de la red neuronal.

8. Al terminar el proceso de entrenamiento, el alumno debe escoger aleatoriamente 10 mensajes del archivo “spamSMS.csv” y colocarlos en el script “probarTextoParaClasificar”, sustituyendo a los existentes. Ejecutar el script y determinar si el entrenamiento es óptimo.

9. En este nuevo punto, el alumno debe de escoger al menos 10 mensajes de forma aleatoria del archivo “MensajesParaVerificar.csv“, colocarlos en el script “probarTextoParaClasificar”, sustituyendo a los existentes. Ejecutar el script y determinar el porcentaje de éxito de la red

neuronal. Mostrar al profesor. Guardar los datos de porcentaje de éxito y error. Mostrar al profesor los resultados obtenidos.

10. Repetir los pasos 4 a 9, pero en este caso dejar dentro del archivo "spamSMS.csv" 100 mensajes aleatorios, cuidando que sean 50 spam y 50 mensajes apropiados. Verificar el entrenamiento con 10 mensajes aleatorios a la vez, 10 del archivo "spamSMS.csv" y 10 del archivo "probarTextoParaClasificar". Guardar dato de porcentajes de éxito y error. Mostrar al profesor los resultados obtenidos.

11. Repetir los pasos 4 a 9, pero en este caso dejar dentro del archivo "spamSMS.csv", 1,000 mensajes aleatorios, cuidando que sean 500 spam y 500 mensajes apropiados. Verificar el entrenamiento con 10 mensajes aleatorios a la vez, 10 del archivo "spamSMS.csv" y 10 del archivo "probarTextoParaClasificar". Guardar dato de porcentajes de éxito y error. Mostrar al profesor los resultados obtenidos.

12. Repetir los pasos 4 a 9, en este caso con la cantidad completa de mensajes dentro del archivo "spamSMS.csv". Verificar el entrenamiento con 10 mensajes aleatorios a la vez, 10 del archivo "spamSMS.csv" y 10 del archivo "probarTextoParaClasificar". Guardar dato de porcentajes de éxito y error. Mostrar al profesor los resultados obtenidos.

13. Realizar una tabla comparativa de los resultados obtenidos similar a la tabla 1.

Tabla 1.- Tabla comparativa de resultados obtenidos

Número de datos de entrenamiento	Porcentaje de Éxito	Wordcloud (imagen)	Tiempo de entrenamiento	Resultados de otros equipos del laboratorio
20				
100				
1,000				
5,000+				

14. En la tabla 1, la última columna dice "Resultados de otros equipos de laboratorio", cada equipo debe verificar los porcentajes de éxito que los otros equipos obtuvieron y realizar la comparativa.

CUESTIONARIO

1. El alumno debe escribir un script que vaya pidiendo el mensaje a clasificar y vaya decidiendo al momento si es SPAM o no. Este script debe de ser continuo, es decir, seguir evaluando mensajes hasta que el usuario le dé la orden de parar. Obviamente, este script debe de utiliza la red neuronal ya entrenada.
2. Diseñar un circuito electrónico, utilizando un GPRS para detectar mensajes tipo SPAM. Validar su funcionamiento.
3. Definir el significado de cada uno de los elementos que se encuentran en el recuadro anaranjado de la figura 3. Explicar que muestran y como se interpretan.

4. ¿Qué puedes concluir con respecto al porcentaje de éxito de los experimentos, comparado con el número de muestras de entrenamiento?
5. ¿Los porcentajes de éxito de todos los equipos fueron similares?, Explica lo observado.
6. ¿Qué representa la Word cloud de cada uno de los experimentos?

CONCLUSIONES

Escribe tus conclusiones de la práctica.

BIBLIOGRAFÍA

Laboratorio de Sistemas Basados en Redes Neuronales

Bibliografía

En este apartado se muestra bibliografía complementaria para el desarrollo de estas prácticas.

1. Santoso, B., & Azis, A. I. (2020). *Machine Learning & Reasoning Fuzzy Logic Algoritma, Manual, Matlab, & Rapid Miner*. Deepublish.
2. Ertel, W. (2018). Introduction to artificial intelligence. Springer.
3. García, A. (2012). Inteligencia Artificial. Fundamentos, práctica y aplicaciones. Rc Libros.
4. Kubat, M., & Kubat. (2017). An introduction to machine learning (Vol. 2). Cham, Switzerland: Springer International Publishing.
5. Silaparasetty, N. (2020). Machine Learning Concepts with Python and the Jupyter Notebook Environment: Using Tensorflow 2.0. Apress.

Laboratorio de Sistemas Basados en Redes Neuronales

Anexos

A continuación se presentan las páginas electrónicas en las que se encuentran las características y herramientas utilizadas en este manual de prácticas.

1. Matlab gratis para comunidad UNAM

   https://www.software.unam.mx/producto/matlab/

2. SpringerLink para poder tener acceso a algunos libros enlistados en la bibliografía, de acceso gratuito a la comunidad UNAM teniendo una cuenta activa en Bidi UNAM.

   https://link.springer.com/

3. Crear cuenta de Bidi UNAM

   https://www.bidi.unam.mx/

4. Características de la tarjeta Arduino

   https://www.arduino.cc/

5. Ayuda en los procesos de aprendizaje de Matlab

   https://www.mathworks.com/

6. Descarga de material necesario para estas prácticas

   http://virtual.cuautitlan.unam.mx/intar/?page_id=786