Universidad Nacional Autónoma de México

Facultad de Ingeniería

Inteligencia Artificial

Semestre 2022-2

Grupo

Proyecto Final

Chatbot sobre conceptos de la materia

Integrantes:

Callejas Sanchez Juan Antonio

Coria Huerta Isaac Uriel

Romero Torres Brenda Mónica

Abstract

In recent years there has been a fast growing use of chatbots in various fields, such as education. In this work, the realization of a chatbot prototype is presented, whose purpose is to provide users with concepts on the subject of Artificial Intelligence. For this purpose, a system developed in Python has been made, which can detect key words and, using natural language processing techniques, provides the concepts to the student. This work can be implemented in the future for many more subjects and other topics, like an interactive encyclopedia.

Resumen

En los últimos años ha habido un rápido crecimiento del uso de chatbots en varios campos, entre ellos, la educación. En este trabajo se presenta la realización de un prototipo de chatbot, cuyo propósito es proporcionar a los usuarios conceptos sobre la asignatura de Inteligencia Artificial. Para este fin, se ha implementado un sistema desarrollado en Python, que puede detectar palabras clave habladas y, gracias al uso de técnicas de procesamiento del lenguaje natural, provee al estudiante los conceptos buscados. Este trabajo puede implementarse a futuro para muchas más asignaturas y otros temas, como una enciclopedia interactiva.

Introducción

Un chatbot se define como un programa de computadora diseñado para simular una conversación (ya sea escrita o hablada) con usuarios humanos, especialmente mediante internet [1]. Estos pueden responder a consultas sencillas o ser tan sofisticados como los asistentes digitales, con la capacidad de aprender para ofrecer experiencias personalizadas a sus usuarios [2]. Un chatbot proporciona respuestas basadas en una combinación de guiones predefinidos y aplicaciones de machine learning, de forma que cuando se le hace una pregunta, el chatbot responde de acuerdo con la base de datos disponible en ese momento [3].

El reconocimiento de voz es la capacidad de una máquina para escuchar palabras habladas e identificarlas. Los chatbots pueden ser desarrollados con diversas herramientas para poder tener reconocimiento de voz, una de ellas es SpeechRecognition, que es una librería para construir programas en Python que toman la entrada del micrófono, la procesa y puede convertir las palabras habladas en texto, hacer una consulta o dar una respuesta. [4].

Otra librería utilizada para el procesamiento de lenguaje es pyttsx3, que se encarga de la conversión de texto a voz en Python. A diferencia de las bibliotecas alternativas, funciona sin conexión y es compatible con Python 2 y 3 [5]. Otra herramienta utilizada para el desarrollo de este chatbot fue PyAutoGUI, que permite que sus secuencias de comandos de Python controlen el mouse y el teclado para automatizar las interacciones con otras aplicaciones. La API está diseñada para ser simple, funciona en Windows, macOS y Linux, y se ejecuta en Python 2 y 3. [6].

En los últimos años ha habido un rápido crecimiento del uso de chatbots en varios campos, como el cuidado de la salud, el marketing, los sistemas de apoyo, el patrimonio cultural, el entretenimiento y la educación, entre otros. Se han desarrollado chatbots para su uso en la educación, por ejemplo, Tutorbot, que se utiliza para entornos de educación en línea y contiene algunas características como la gestión del lenguaje natural, presentación de contenidos e interacción con un motor de búsqueda [7]. Otro ejemplo es el chatbot presentado por Clarizia et al., que funcionaba en la plataforma de educación en línea de la Universidad de Salermo, y estuvo dirigida para los alumnos de dos cursos: Fundamentos de Ciencias de la Computación y Redes de Computadores [8]. En ambos casos, se presentaban contenidos educativos a los alumnos, relacionados con palabras clave que se introducían en el chat.

El objetivo de este trabajo es desarrollar un chatbot que sirva como herramienta para aprender y repasar conceptos de la materia de Inteligencia Artificial, de acuerdo al plan de estudios actual de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México (UNAM).

Desarrollo

Para el análisis de los conceptos, se revisó el plan de estudios de la asignatura de Inteligencia Artificial de la Facultad de Ingeniería de la UNAM [9]. Para conocer los conceptos seleccionados véase el anexo de este documento.

Primero se instalaron las librerías pyttsx3 y pyautogui desde nuestra terminal,

```
C:\Users\asscs>pip install pyttsx3
Collecting pyttsx3
  Downloading pyttsx3-2.90-py3-none-any.whl (39 kB)
Collecting comtypes
  Downloading comtypes-1.1.11-py2.py3-none-any.whl (167 kB)
     ---------------------------------------- 167.4/167.4 KB 1.4 MB/s eta 0:00:00
Collecting pypiwin32
  Downloading pypiwin32-223-py3-none-any.whl (1.7 kB)
Collecting pywin32
  Downloading pywin32-304-cp39-cp39-win_amd64.whl (12.2 MB)
     ---------------------------------------- 12.2/12.2 MB 7.4 MB/s eta 0:00:00
Installing collected packages: pywin32, comtypes, pypiwin32, pyttsx3
Successfully installed comtypes-1.1.11 pypiwin32-223 pyttsx3-2.90 pywin32-304
```

Figura 1. Instalación de librería pyttsx3.

```
C:\Users\asscs>pip install pyautogui
Collecting pyautogui
  Downloading PyAutoGUI-0.9.53.tar.gz (59 kB)
     ---------------------------------------- 59.0/59.0 KB 524.6 kB/s eta 0:00:00
```

Figura 2. Instalación de librería pyautogui.

Para poder descargar la librería SpeechRecognition, uno de los requerimientos es tener instalada la librería PyAudio. Para lograr esto en Windows, se instaló la librería pipwin, que es es una herramienta complementaria para pip en Windows que permite instalar binarios de paquetes de python no oficiales para Windows [10]. Posteriormente, la librería PyAudio se instaló con el comando pipwin install pyaudio. PyAudio proporciona enlaces de Python para PortAudio, la biblioteca de E/S de audio multiplataforma. Con ella, se puede usar Python fácilmente para reproducir y grabar audio en una variedad de plataformas [11].

Figura 3. Instalación de librería pyaudio.

Ahora podremos hacer uso de la librería SpeechRecognition.

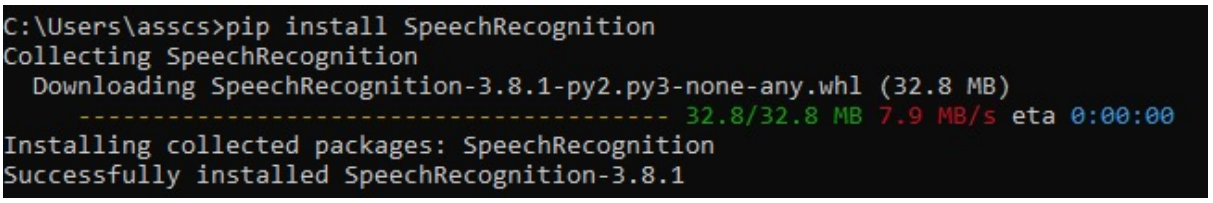

Figura 4. Instalación de librería SpeechRecognition

Una vez instaladas nuestras librerías, abrimos nuestra consola de Python. En este caso, se utilizó la versión 3.8.5.

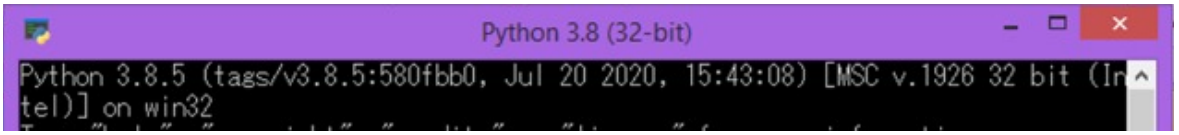

Figura 5. Terminal de Python.

Luego de haber instalado nuestras librerías necesarias comenzamos con las pruebas de estas para ver si se habían instalado correctamente.

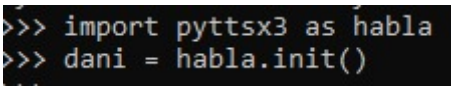

Figura 6. Creación del objeto de voz

A partir de este objeto, con las propiedades de la biblioteca pyttsx3, empezamos a hacer las pruebas necesarias con las características de la biblioteca en este caso la función que se va a utilizar para empezar a modificar es “getProperty”.

```
getProperty(self, name)
    Gets the current value of a property. Valid names and values include:

    voices: List of L{voice.Voice} objects supported by the driver
    voice: String ID of the current voice
    rate: Integer speech rate in words per minute
    volume: Floating point volume of speech in the range [0.0, 1.0]

    Numeric values outside the valid range supported by the driver are
    clipped.

    @param name: Name of the property to fetch
    @type name: str
    @return: Value associated with the property
    @rtype: object
    @raise KeyError: When the property name is unknown
```

Figura 7. Menú help del objeto propiedad getProperty

Utilizando la función getProperty podemos cambiar o modificar las características de nuestro objeto, como se puede ver en la imagen tenemos diferentes características pero primero en los que nos enfocaremos es en la voz ya que será un chatbot que podrás utilizar mediante voz lo ideal es que pueda contestarte entonces nos enfocaremos en voices para detectar cual es la voz que se usará y el acento que queremos:

```
>>> voz = dani.getProperty('voices')
>>> voz
[<pyttsx3.voice.Voice object at 0x000002102963EB50>, <pyttsx3.voice.Voice object at 0x000002102963EFD0>, <pyttsx3.voice
Voice object at 0x0000021023AEB340>]
>>> for acento in voz:
...     print(acento.id)
...
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_ES-ES_HELENA_11.0
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-US_ZIRA_11.0
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_ES-MX_SABINA_11.0
```

Figura 8. Voces disponibles del objeto “Dani”

Al entrar en las propiedades de “voices” tenemos 3 tipos, la diferencia entre éstas es el acento, como se aprecia en la imagen, una tiene acento castellano, otra es acento inglés y la última es acento español-México. Para nuestro caso utilizaremos el acento español-MX y se

la adjuntamos (utilizando toda la dirección) al objeto “dani” para que sea la voz predeterminada.

```
>>> dani.setProperty('voice', 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_ES-MX_SABINA_11.0')
```

Figura 9. Voz añadida a “dani”

Además, antes de probar como se escuchaba, lo que se hizo fue modificar la propiedad de “rate”, la cual se encarga de las palabras por minuto de nuestro objeto. Una vez actualizada esta característica, está lista para la prueba. Por último se utilizó “say” and “runAndWait” para darle la frase que la interfaz diría y luego escuchar el acento junto con la frase dada.

```
>>> dani.getProperty('rate')
200
>>> dani.setProperty('rate', 140)
>>> dani.say('Esta es la prueba 1 del Proyecto de IA')
>>> dani.runAndWait()
```

Figura 10. Actualización y prueba de la voz

Luego de probar el funcionamiento de nuestra interfaz de voz, pasaremos a la parte del código, donde se añadirán las diferentes funciones para utilizar el manejo de las características para el uso de la interfaz de voz.

Nos dirigimos principalmente a la parte medular del código, que es nuestro ciclo de reconocimiento de comando, esto genera que una vez corriendo el código, imprime en pantalla un mensaje para que el usuario sepa que ya está funcionando, y al usar los diferentes comando de voz, se obtendrá una respuesta, ya sea del concepto que se quiere, o en dado caso de que no esté el comando, se mostrará un mensaje de “no entiendo” o “no está disponible”, dependiendo del error, y continuará con el ciclo hasta que la variable cambie.

```
##escuchar el audio con el microfono
c=0
while c!=1:
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("¿Qué concepto deseas aprender?")
        audio = r.listen(source)

    try:
        #En caso de entender el audio
        comando=r.recognize_google(audio,language='es-MX')
        print ("Creo que dijiste: " + comando)
        interpretar(comando)
        #En caso de no entender
    except sr.UnknownValueError:
        print("No te pude entender")
    except sr.RequestError as e:
        #En caso de no tener conexión a internet
        print("No pude obtener respuesta del servicio de Google Speech Recognition; {0}".format(e))
```

Figura 11. Análisis de respuesta de la interfaz con respecto al comando (código)

En el caso de que el comando que se haya dicho sea aceptado, el código continuará con su ejecución, mandando a llamar a la respectiva función donde nuestra interfaz “Dani” dará la definición de lo que se pregunta, y dependiendo de cada función, proporcionará dará la definición dentro de un bloc-notas o sólo la interfaz dirá la definición solicitada con base en sus respuestas y comandos.

```
a_res1=('resolvente' or 'problema'or 'problemas') in comando_de_audio
objetivo=('objetivo') in comando_de_audio
estado=('estado') in comando_de_audio
solucion=('solución') in comando_de_audio
abstraccion=('abstracción') in comando_de_audio
bus_h=('busqueda' and 'heurística') in comando_de_audio
bus_n_i=('búsqueda' and 'no' and 'informada') in comando_de_audio
busqueda=('búsqueda') in comando_de_audio
```

Figura 12. Algunos comandos aceptados del código

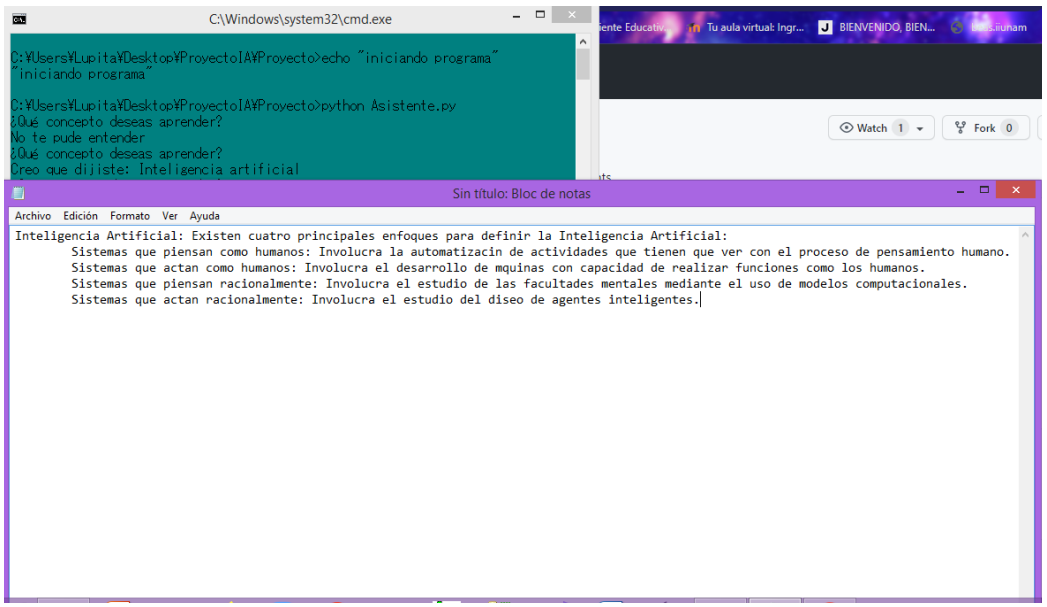

Figura 13. Ejecución del programa al decir “inteligencia artificial”.

Las dos funciones que ejecutan acciones diferentes a dar definiciones, son “abrir_codigo” y “cerrar”. En la primera, se nos abrirá una pestaña en nuestro navegador de google al detectar la palabra “código” o “Github”, y se nos mostrará el repositorio de nuestro proyecto en Github. El enlace es https://github.com/asscs15/Chatbot_IA. Aquí se encuentra el archivo Asistente.py, el archivo navegar.bat con la dirección del repositorio y el archivo asistenteVirtual.bat con el ejecutable de nuestro chatbot.

La segunda función es cerrar, que al detectar la palabra “salir” o “finalizar”, el chatbot se despedirá de nosotros de forma verbal y nos mostrará un mensaje de despedida en la consola.

```
def abrir_codigo():
    sub.call([r'C:/Users/Lupita/Desktop/ProyectoIA/Proyecto/navegar.bat'])
    return None
def cerrar():
    sleep(1.5)
    dani = voz.init()
    velocidad=dani.getProperty('rate')
    dani.setProperty('rate',velocidad-25)
    dani.setProperty('voice','TTS_MS_ES-MX_SABINA_11.0')
    dani.say('Gracias por participar')
    dani.runAndWait()
    print('Que te vaya bonito')
    exit()
```

Figura 14. Algunas funciones para los comandos aceptados.

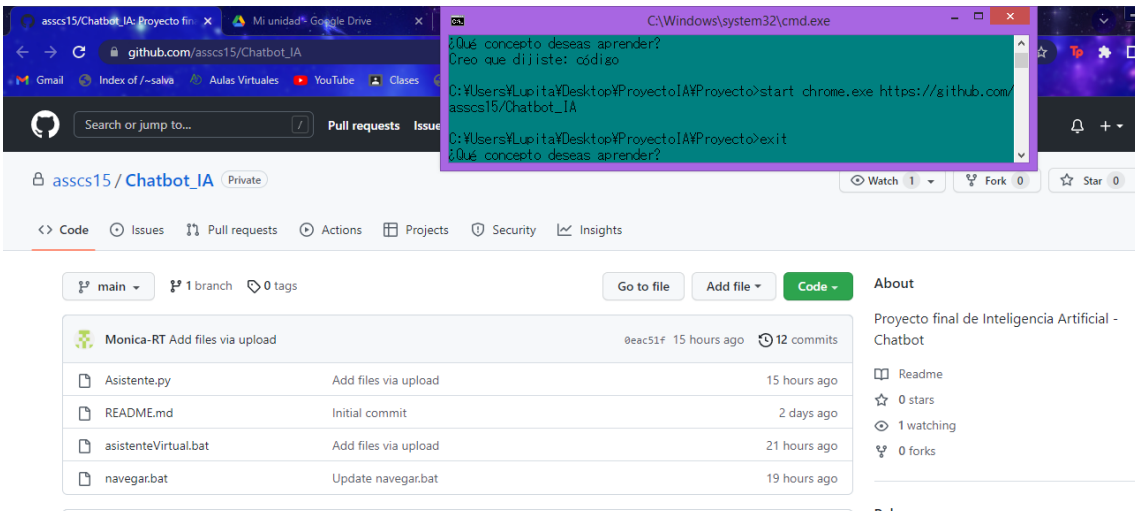

Figura 15. Ejecución del programa al decir “código”.

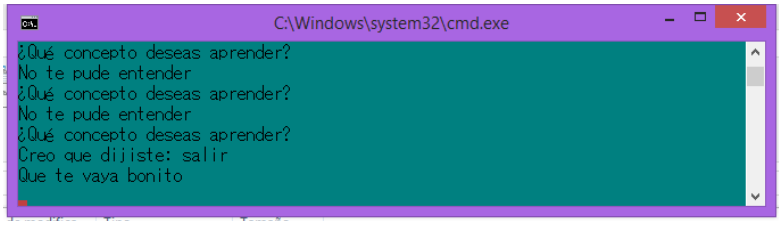

Figura 16. Ejecución del programa al decir “salir”.

Conclusión

Gracias a este trabajo, pudimos implementar un chatbot sencillo que utiliza el servicio de reconocimiento de voz de Google para detectar palabras clave dichas en el micrófono de nuestra computadora y nos devuelva definiciones en formato escrito y hablado. Este proyecto nos ayudó a su vez a repasar algunos conceptos revisados a lo largo del curso. Finalmente, a pesar de los problemas de incompatibilidad con las bibliotecas de Python, se pudo crear un código que funciona de acuerdo a lo esperado.

Trabajo a futuro

Con este primer vistazo de nuestro proyecto, podemos contemplar múltiples opciones de mejora. La más destacada es que el chatbot sea autosuficiente y tenga la capacidad de aprender, ya que para esta entrega, el diseño actual es completamente dependiente del programador y las líneas de código proporcionadas, es decir, los conceptos provienen del conocimiento del programador y no son propios del programa. Además, como ya se mencionó, no tiene la capacidad de aprender respuestas o palabras por cuenta propia, no se implementó esta capacidad, ya que es un chatbot simple. Otra característica a mejorar es el formato en el que presenta la información, podría añadirse contenido multimedia y otras fuentes de consulta en línea, para complementar el aprendizaje. Finalmente, este chatbot puede implementarse para su uso en otras asignaturas y temas de interés.

Bibliografía

[1] R. Nuseibeh. (2018, mayo 11). What is a Chatbot? [Online]. Disponible: https://chatbotsmagazine.com/what-is-a-chatbot-6dfff005bb34

[2] Oracle. (2022). ¿Qué es un chatbot? [Online]. Disponible: https://www.oracle.com/mx/chatbots/what-is-a-chatbot/

[3] V. Elupula. (2019, mayo 15). How do chatbots work? An overview of the architecture of chatbots. [Online]. Disponible: https://bigdata-madesimple.com/how-do-chatbots-work-an-overview-of-the-architecture-of-a-chatbot/

[4] SimpliLearn (2021, septiembre 17). A Guide to Speech Recognition in Python: Everything You Should Know. [Online]. Disponible: https://www.simplilearn.com/tutorials/python-tutorial/speech-recognition-in-python

[5] N. Bhat. (2020, julio 10). pyttsx3 2.90. [Online]. Disponible: https://pypi.org/project/pyttsx3/

[6] PyAutoGUI. (2019). Welcome to PyAutoGUI's documentation! [Online]. Disponible: https://pyautogui.readthedocs.io/en/latest/

[7] Satu, S., Parvez, H., AI-Mamun, S. "Review of integrated applications with AIML based chatbot" in First International Conference on Computer and Information Engineering (ICCIE) (2015). doi:10.1109/ccie.2015.7399324

[8] Clarizia, F., Colace, F., Lombardi, M., Pascale, F., & Santaniello, D. "Chatbot: An Education Support System for Student" in Lecture Notes in Computer Science, 2018, 291–302. doi:10.1007/978-3-030-01689-0_23

[9] UNAM. (2015, abril 28). Programa de estudio. [Online]. Disponible: https://classroom.google.com/c/NDYzNzgxMjQ0NzIx/m/NDY0MTIwNDExMjY5/details

[10] Lepisma. (2022, marzo 17). pipwin 0.5.2. [Online]. Disponible: https://pypi.org/project/pipwin/

[11] LearnTutorials (s.f.). Python Language pyaudio [Online]. Disponible: https://learntutorials.net/es/python/topic/10627/pyaudio

Anexo

Lista de términos

Tema 1

Inteligencia Artificial: Existen cuatro principales enfoques para definir la Inteligencia Artificial

- Sistemas que piensan como humanos: Involucra la automatización de actividades que tienen que ver con el proceso de pensamiento humano.
- Sistemas que actúan como humanos: Involucra el desarrollo de máquinas con capacidad de realizar funciones como los humanos.
- Sistemas que piensan racionalmente: Involucra el estudio de las facultades mentales mediante el uso de modelos computacionales.
- Sistemas que actúan racionalmente: Involucra el estudio del diseño de agentes inteligentes.

Prueba de Turing: Prueba diseñada para proporcionar una definición operacional y satisfactoria de inteligencia. Sirve para diferenciar entre entidades inteligentes y seres humanos. No existe interacción física entre el evaluador y el computador. Para que la entidad inteligente pase la prueba, ésta debe estar dotada de Visión computacional y Robótica.

IA débil: Hipótesis que afirma que es posible que las máquinas actúen con inteligencia. Es la idea más extendida entre los desarrolladores.

IA fuerte: Hipótesis que afirma que las máquinas son capaces de pensar.

Tema 2

Agente: Es algo que razona y está dotado de controles autónomos, son capaces de percibir su entorno, pueden persistir durante un período de tiempo prolongado, se adaptan a los cambios y son capaces de alcanzar objetivos diferentes

Percepción: Se refiere a la habilidad del agente para recibir entradas en cualquier instante.

Agente reactivo simple: Es el agente más sencillo y seleccionan las acciones a partir de las percepciones actuales, ignorando todas las percepciones pasadas.

Agente reactivo basado en modelos: Maneja un estado interno que depende de las percepciones pasadas y éstas deben irse actualizando.

Agente basado en objetivos: Además de la descripción del estado actual, el agente necesita información sobre su meta, es decir, las situaciones que son deseables y posibles.

Agente basado en utilidad: Además de la descripción del estado actual y una meta, es necesario distinguir los estados de mayor utilidad.

Tema 3

Agentes resolventes-problemas: Agentes inteligentes cuyo propósito es encontrar una secuencia de acciones para cumplir un estado objetivo.

Objetivo: Conjunto de estados del mundo que satisfacen un problema.

Estado: Descripción actual del mundo o problema.

Búsqueda: Proceso para hallar una secuencia que cumple un objetivo.

Solución: Camino de un estado inicial a un estado objetivo. La resolución de problemas se lleva a cabo por medio de búsquedas en el espacio otorgado.

Abstracción: Proceso de eliminar detalles de una representación sin perder su significado.

Búsqueda no informada: Solo se proporciona la información del estado que define el problema, genera los estados conforme se avanza al objetivo. Algunos ejemplos son: búsqueda por amplitud, búsqueda primero en profundidad y búsqueda de costo uniforme.

Búsqueda heurística: También conocida como búsqueda informada, utiliza información adicional específica para cada problema más allá de la definición. Las funciones heurísticas son fundamentales (cálculo del camino más corto del inicio al fin) para estas búsquedas. Algunos ejemplos son: búsqueda A*, búsqueda voraz y el algoritmo Dijkstra.

Problemas de satisfacción de restricciones: Las variables correspondientes poseen un dominio, impidiendo que tomen ciertos valores. Se resuelven por medio de algoritmos como la búsqueda con vuelta atrás y el ascenso de colinas.

Tema 4

SBC: Sistemas basados en conocimientos, expresan el conocimiento de forma que humanos y ordenadores puedan entender.

Conocimiento: Un concepto que se puede considerar como un conjunto de hechos o verdades sobre un tema. Desde un punto de vista computacional, es el mapeo entre objetos y relaciones del dominio de un problema.

Al representarlo, se deben cumplir cuatro condiciones: adecuación, eficiencia, extensibilidad y propiedad.

Reglas: Restricciones independientes que definen secciones del conocimiento y aseguran cumplir con un dominio para la resolución de conflictos.

Algoritmos de inferencia: Determinan cómo utilizar una base de conocimiento. Se dividen en tres grandes familias: encadenamiento hacia adelante, encadenamiento hacia atrás y sistemas de demostración de teoremas basados en la resolución.

Redes semánticas: Algoritmos eficientes para inferir propiedades de un objeto en base a su pertenencia a cierta categoría. Funcionan como apoyo gráfico para visualizar bases de conocimiento.

Razonamiento: Proceso para obtener conclusiones, juicios o inferencias a partir de hechos o premisas. Existen varios tipos como el deductivo (movimiento de lo general a lo particular) y el inductivo (basado en observaciones previas).

Razonamiento progresivo: Evoluciona a una conclusión a partir de las premisas.

Razonamiento regresivo: Empieza con la conclusión y decide si las reglas establecidas pueden obtenerla.

Lógica proposicional: Define las reglas para determinar el valor de verdad de una sentencia.

Razonamiento con incertidumbre: Problema sin modelos exactos ni garantía en su definición, esta incertidumbre se genera cuando hay un conflicto relacionado a los hechos o las reglas. Es posible representarlas con símbolos o números para obtener posibles soluciones.

Tema 5:

Modelos probabilísticos: estos son una representación de la realidad sobre la que podemos nosotros hacer inferencias. de estos podemos obtener: redes bayesianas y modelos de Markov

Las redes bayesianas: son un tipo de modelo grafo probabilístico la cual se utiliza para representar un conjunto de variables aleatorias y sus dependientes condicionales, a que nos referimos con esto las probabilidades que un evento provoque otro. Un ejemplo de la relación entre enfermedades y síntomas.

Modelos de Markov: este es un método que se utiliza básicamente para hacer estudios de coste y efectividad.

Los métodos con base en reglas: los cuales son modelos de representación del conocimiento se usan de una manera muy amplia. De aquí tenemos podemos realizar los árboles de regresión.

Los árboles de regresión: son representaciones gráficas que se realizan para resolver problemas, estos son utilizados para crear modelos predictivos de aprendizaje.

Los modelos bioinspirados: son sistemas construidos por medio de un hardware configurable y sistemas electrónicos simulan la forma de pensar, procesar y resolver problemas de sistemas biológicos. De estos surgen las redes neuronales y también la computación evolutiva

Las redes neuronales: están inspiradas en la biología ósea que con este modelo se aprende con la experiencia osea modifica su comportamiento con la respuesta del entorno, este modelo abstrae la información del conjunto de entradas que tenga.

La computación evolutiva: resuelve problemas, con el fin de encontrar el mejor resultado al problema, podemos decir o referirnos en el entorno del más fuerte o mas apto sobrevive, en este caso el mejor.

Tema 6:

Los modelos predictivos en el sector público: La gestión de infraestructuras y servicios públicos es complejo y cada decisión debe tener en cuenta muchos parámetros, los elementos como la captación, el procesamiento y el análisis de los datos permite a la (IA) realizar proyecciones más precisas, indicando los diagramas de cada decisión su aporte y coste.

También tenemos la investigación y desarrollo en el ámbito de la salud: La Inteligencia artificial es el núcleo de diferentes programas de investigación, ya que esta ofrece resultados más prometedores, ya que el análisis predictivo y el reconocimiento de imágenes puede ayudar a salvar pacientes, incluso antes de que se manifieste la enfermedad.

Seguridad informática y protección de datos: este es uno de los puntos clave de la inteligencia artificial, ya que el acceso, intercambio y uso de información son esenciales para cualquier empresa, cualquier programa basado en ciberseguridad basado en inteligencia artificial ayuda a detectar diferentes fallos en una red.

Finalmente la traducción automática: esto es igual enfocado hacia las empresas pero esto es más en cuestión de dar un buen servicio, para este caso, la inteligencia artificial puede enfocarse en las solicitudes de los clientes traduciendo así las solicitudes para dar una respuesta adaptada a lo que el cliente necesite.

Información obtenida de:

S. Russell, P. Norving. (2012). Artificial Intelligence: A Modern Approach. (3rd. edition). New Jersey. Prentice Hall.