UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

Facultad de Ingeniería

Ingeniería en Ciencias y Sistemas

Inteligencia Artificial

Manual Técnico

ChatBot

Maria Isabel Masaya Cordova

Lesther Kevin Federico López Miculax

Kevin Uriel Ramírez

Guatemala, Guatemala, 10 de Diciembre del 2024

Introducción

Este proyecto implementa un chatbot conversacional utilizando TensorFlow.js para el aprendizaje automático y procesamiento del lenguaje natural. El chatbot está diseñado para interactuar con usuarios en lenguaje natural, comprender sus intenciones y proporcionar respuestas relevantes.

Componentes Principales

- **model/model.json:** Contiene el modelo de aprendizaje automático entrenado para clasificar intenciones.
- **words.json**: Almacena el vocabulario utilizado para vectorizar las entradas del usuario.
- **classes.json**: Contiene las posibles intenciones que el chatbot puede reconocer.
- **intents.json:** Define las intenciones, patrones de entrada y respuestas asociadas.
- **index.html:** Interfaz web para interactuar con el chatbot.
- **app.js:** Lógica principal del chatbot, incluyendo la carga de recursos, procesamiento de lenguaje natural y generación de respuestas.

Flujo de Trabajo

Carga de Recursos: Al iniciar, app.js carga el modelo, vocabulario, clases e intenciones desde los archivos JSON. Esto para llenar de información al chatbot y así poder ejecutarse de manera correcta.

```
// Cargar recursos al iniciar
async function loadResources() {
    try {
        model = await tf.loadLayersModel('model/model.json');

        const wordsResponse = await fetch('words.json');
        words = await wordsResponse.json();

        const classesResponse = await fetch('classes.json');
        classes = await classesResponse.json();

        const intentsResponse = await fetch('intents.json');
        intents = await intentsResponse.json();

        console.log("Recursos cargados");
    } catch (error) {
        console.error("Error al cargar los recursos:", error);
        alert("Hubo un problema al cargar el chatbot. Por favor, intenta recargar la página.");
    }
}
```

Procesamiento de la Entrada del Usuario

El procesamiento de la entrada del usuario es un paso crucial en el funcionamiento del chatbot. Su objetivo es convertir el texto que introduce el usuario en un formato que el modelo de aprendizaje automático pueda comprender y procesar. Esto se logra mediante dos etapas principales:

- **Tokenizar:** El texto de entrada se divide en unidades individuales llamadas "tokens". En este caso, los tokens son simplemente las palabras que componen la frase. Ejemplo: Si la entrada es "Hola, ¿cómo estás?", los tokens serían: "Hola", "¿cómo", "estás?".

```
// Tokenizar y procesar la entrada del usuario
function tokenize(sentence) {
    return sentence.toLowerCase().match(/\b(\w+)\b/g);
}
```

Bag-of-Words:

- Se crea un vector numérico (una "bolsa de palabras") que representa la entrada del usuario.
- El vector tiene la misma longitud que el vocabulario conocido por el chatbot (almacenado en words.json).
- Cada elemento del vector corresponde a una palabra del vocabulario.
- Si una palabra del vocabulario está presente en la entrada del usuario, el elemento correspondiente en el vector se establece en 1, de lo contrario, se establece en 0.
- Ejemplo: Si el vocabulario es ["hola", "cómo", "estás", "bien"] y la entrada es "Hola, ¿cómo estás?", el vector bag-of-words sería [1, 1, 1, 0].

```
function bagOfWords(sentence, words) {
    const sentenceTokens = tokenize(sentence);
    const bag = Array(words.length).fill(0);
    sentenceTokens.forEach(token => {
        const index = words.indexOf(token);
        if (index !== -1) {
            bag[index] = 1;
        }
    });
    return bag;
}
```

Predicción de Intención: El modelo de aprendizaje automático predice la intención de la entrada del usuario. ¡Claro! Detallemos la parte de la predicción de intención en el chatbot:

Una vez que la entrada del usuario ha sido procesada y convertida en un vector numérico (bag-of-words), el siguiente paso es la predicción de la intención. Aquí es donde entra el modelo de aprendizaje automático (model/model.json).

El modelo de aprendizaje automático

- Este modelo ha sido previamente entrenado con un conjunto de datos que contiene ejemplos de frases y sus correspondientes intenciones.
- El modelo ha aprendido a identificar patrones y relaciones entre las palabras y las intenciones, de manera que puede generalizar y predecir la intención de nuevas frases que no ha visto antes.

Proceso de predicción

1. **Entrada al modelo:** El vector numérico (bag-of-words) que representa la entrada del usuario se introduce como entrada al modelo de aprendizaje automático.
2. **Predicciones:** El modelo procesa la entrada y genera un conjunto de predicciones. Cada predicción corresponde a una de las posibles intenciones definidas en el chatbot (classes.json).
3. **Probabilidades:** Las predicciones son en realidad probabilidades, que indican la confianza del modelo en que la entrada del usuario pertenece a cada una de las intenciones.
4. **Selección de la intención:** Se selecciona la intención con la probabilidad más alta como la intención predicha para la entrada del usuario.

```
// Predecir la clase
async function predictClass(sentence) {
    const inputBag = bagOfWords(sentence, words);
    const inputTensor = tf.tensor([inputBag]);

    const predictions = await model.predict(inputTensor).array();
    const maxIdx = predictions[0].indexOf(Math.max(...predictions[0]));

    return predictions[0][maxIdx] > 0.25 ? classes[maxIdx] : null;
}
```

Generación de Respuesta

Una vez que el chatbot ha predicho la intención del usuario, llega el momento de generar una respuesta. Para esto, el chatbot recurre a un conjunto de respuestas predefinidas que están asociadas a cada intención.

El proceso es el siguiente:

1. **Identificar la intención:** El chatbot ya tiene la intención predicha en la etapa anterior.
2. **Acceder a las respuestas:** Busca en su base de conocimientos (en este caso, el archivo intents.json) las respuestas que están asociadas a esa intención.
3. **Seleccionar una respuesta:** De forma aleatoria, elige una de las respuestas disponibles para esa intención.
4. **Mostrar la respuesta:** Presenta la respuesta seleccionada al usuario a través de la interfaz del chatbot.

```
// Obtener una respuesta según la clase
function getResponse(intentTag) {
    const intent = intents.intents.find(i => i.tag === intentTag);
    return intent ? intent.responses[Math.floor(Math.random() * intent.responses.length)] : "No entiendo lo que dices.";
}
```

Puntos Relevantes

- **Entrenamiento del Modelo:** El modelo model.json se entrena utilizando un conjunto de datos de intenciones y patrones.
- **Vocabulario:** El vocabulario en words.json debe ser lo suficientemente amplio para cubrir las posibles entradas del usuario.
- **Intenciones:** Las intenciones en intents.json deben ser definidas con precisión para que el chatbot pueda clasificarlas correctamente.
- **Respuestas:** Las respuestas deben ser coherentes con las intenciones y proporcionar información útil al usuario.
- **Interfaz Web:** index.html proporciona una interfaz básica para la interacción. Se puede personalizar para mejorar la experiencia del usuario.

Conclusión

Este proyecto proporciona una base sólida para construir un chatbot conversacional. Con un entrenamiento adecuado y una buena definición de intenciones y respuestas, el chatbot puede ser una herramienta útil para la comunicación e información.