

Chatbot en MATLAB usando contains()

Sistema Experto sobre el
Descubrimiento de América
Universidad

Chatbots transaccionales

- Reservas
- Soporte técnico
- Pagos
- Ejemplo: atención bancaria

Chatbots conversacionales

- Interacción natural
- Educación
- Asistentes personales

Chatbots híbridos

- Combinan:
- Reglas + NLP
- Base de conocimiento + Generación

¿Qué es NLP?

- **NLP (Natural Language Processing) o Procesamiento de Lenguaje Natural** es la rama de la Inteligencia Artificial que permite que las máquinas **comprendan, interpreten y generen lenguaje humano** (texto o voz).

¿Qué hace exactamente NLP?

- **Preprocesamiento**
 - lower() → convertir a minúsculas
 - Eliminación de tildes y signos
 - Tokenización (tokenizedDocument en MATLAB)
 - Eliminación de stopwords
- **Representación numérica**
 - Bag of Words
 - TF-IDF
 - Word embeddings
 - Sentence embeddings
- **Modelado**
 - Clasificación de texto
 - Análisis de sentimientos
 - Reconocimiento de entidades (elementos clave)
 - Traducción automática
 - Chatbots

1. Introducción

- Un chatbot es un programa que simula una conversación humana.
 - En este proyecto usamos MATLAB.
 - Se implementa usando estructuras condicionales e identificación de palabras clave.

2. Objetivo del Proyecto

- Diseñar un chatbot educativo.
 - Aplicar programación estructurada en MATLAB.
 - Comprender el funcionamiento de sistemas basados en reglas.

3. Arquitectura del Sistema

- Entrada del usuario (input).
 - Procesamiento con `lower()`.
 - Evaluación con `contains()`.
 - Respuesta automática con `if`.

4. Función contains()

- Permite buscar palabras clave dentro de un texto.
 - Ejemplo: `contains(usuario,'colon')`
 - Devuelve verdadero o falso.

5. Estructura Principal del Código

- while true
 - usuario = lower(input('Tú: ', 's'));
 - if contains(usuario, 'colon')
 - disp('Cristóbal Colón llegó en 1492');
 - end
 - end

6. Base de Conocimiento Histórica

- Cristóbal Colón
 - Fecha: 12 de octubre de 1492
 - Carabelas: Niña, Pinta y Santa María
 - Reyes Católicos: Isabel y Fernando

7. Ventajas del Modelo

- Fácil implementación.
 - Ideal para enseñanza.
 - Permite ampliar base de conocimiento.



8. Limitaciones

- No aprende automáticamente.
 - Depende de palabras exactas.
 - No entiende contexto complejo.

9. Conclusión

- El chatbot implementa un sistema experto básico.
 - Es una base para evolucionar hacia Machine Learning.
 - Demuestra aplicación práctica de programación en MATLAB.


- como hago para que colon lo pueda el usuario escribir con tilde sin tilde con mayuscula etc
- Excelente pregunta. Esto ya es **normalización de texto**, concepto clave en NLP.
- El problema es que:
 - “Colón”
 - “colon”
 - “COLÓN”
 - “ColOn”
- son cadenas distintas para MATLAB.
- La solución profesional es **normalizar el texto antes de evaluarlo**.

-  **SOLUCIÓN 1 (Recomendada): Quitar tildes y pasar a minúscula**
- Agrega una función para eliminar acentos.
-  **Paso 1: Normalizar entrada**
- usuario = input('Tú: ', 's');

```
% Pasar a minúsculas
usuario = lower(usuario);
```

```
% Quitar tildes
usuario = replace(usuario, 'á', 'a');
usuario = replace(usuario, 'é', 'e');
usuario = replace(usuario, 'í', 'i');
usuario = replace(usuario, 'ó', 'o');
usuario = replace(usuario, 'ú', 'u');
```

- Ahora:
- “Colón”
- “COLÓN”
- “colon”
- → Todos se convierten en:
- colon
- Y tu condición funciona:
- if contains(usuario, 'colon')
 - disp('Bot: Cristobal Colon llego en 1492.');
- end

-  **SOLUCIÓN 2 (Más elegante – función auxiliar)**
- Puedes crear una función al final del script:
- `function texto = normalizar(texto)`

```
texto = lower(texto);
```

```
texto = replace(texto,'á','a');
```

```
texto = replace(texto,'é','e');
```


```
texto = replace(texto,'í','i');
```


```
texto = replace(texto,'ó','o');
```


```
texto = replace(texto,'ú','u');
```

```
end
```



- Y en tu código principal:
- `usuario = input('Tú: ','s');`
`usuario = normalizar(usuario);`
- Esto es más profesional y modular.


-  **SOLUCIÓN 3 (Nivel más avanzado – regex)**
- Si quieres algo más compacto:
- `usuario = lower(usuario);`
`usuario = regexprep(usuario, '[áàä]', 'a');`
`usuario = regexprep(usuario, '[éèë]', 'e');`
`usuario = regexprep(usuario, '[îï]', 'i');`
`usuario = regexprep(usuario, '[óòö]', 'o');`
`usuario = regexprep(usuario, '[úùü]', 'u');`
- Esto ya es nivel más técnico.

-  **En clase puedes explicar**
- Conceptos que estás aplicando:
- Normalización de texto
- Preprocesamiento NLP
- Sensibilidad a mayúsculas
- Manejo de codificación UTF-8
- Robustez del sistema

-  **Resultado final**
- Con esto el usuario podrá escribir:
- ¿Quién fue Colón?
- quien fue colon
- COLÓN
- colon
- CoLÓN
- Y el bot lo detectará correctamente.

- como hago que la pregunta del usuario tenga 2 o 3 palabras clave
- Muy buena pregunta. Ahora entramos en **lógica booleana aplicada a NLP básico** usando `contains()` con múltiples condiciones.
- Hay **tres formas profesionales** de hacerlo en MATLAB.

-   Exigir que tenga 2 palabras clave (AND lógico)
- Ejemplo: que la pregunta tenga “colon” Y “1492”
- ```
if contains(usuario,'colon') &&
contains(usuario,'1492')
 disp('Bot: Cristóbal Colón llegó a América en
1492.');
```

  
end
- Aquí:
- && = ambas condiciones deben cumplirse.
-  Ejemplo válido:
- ¿Colón llegó en 1492?

-  **2** Permitir combinaciones (OR + AND)

- Ejemplo: que tenga:

- (colon O colón)


- Y (america O 1492)

- ```
if (contains(usuario,'colon')) && ...  
    (contains(usuario,'america') ||  
contains(usuario,'1492'))
```

```
    disp('Bot: Colón llegó a América en 1492.');
```


```
end
```

- Aquí estás creando una condición compuesta más realista.

-  **3 Método profesional: contar palabras clave**
- Esta es la mejor forma académica si quieres que tenga **mínimo 2 palabras clave**.
- claves = ["colon", "america", "1492"];
contador = 0;

```
for i = 1:length(claves)
    if contains(usuario, claves(i))
        contador = contador + 1;
    end
end
```

```
if contador >= 2
    disp('Bot: Estás preguntando sobre el viaje de Colón a América en 1492.');
```


-  Ventaja:
- Escalable
- Más elegante
- Fácil de ampliar
- Se acerca a modelo de recuperación de información


- 🔥 **4 Nivel más avanzado: tokenizar primero**
- Si quieres algo más limpio:
- palabras = split(usuario);

```
claves = ["colon", "america", "1492"];  
contador = sum(ismember(palabras, claves));
```

```
if contador >= 2  
    disp('Bot: Detecté varias palabras clave históricas.');
```

- Aquí ya estás usando:
- Tokenización
- Comparación de conjuntos
- Conteo lógico
- Eso ya es nivel universidad serio

-  **Conceptos que puedes explicar en clase**
- Operadores lógicos (&&, ||)
- Matching múltiple
- Conteo de coincidencias
- Recuperación de información
- Simulación básica de clasificación de intención

-  **Recomendación docente**
- Para universidad:
- Usa el método 3 o 4.
- Para colegio:
- Usa método 1 o 2.

- como sugerir respuestas si no tiene las palabras clave completas sino algunas
- Excelente. Ahora estás entrando en **detección parcial + sugerencia inteligente**, que es un paso previo a NLP formal.
- La idea es:
- Si el usuario no escribió todas las palabras clave, pero escribió algunas, el bot debe sugerir o completar la intención.
- Te explico 3 niveles profesionales.

-  **1 Nivel básico: detectar coincidencias parciales**

- Supongamos que quieres mínimo 2 palabras clave:

- claves = ["colon", "america", "1492"];
contador = 0;

```
for i = 1:length(claves)
    if contains(usuario, claves(i))
        contador = contador + 1;
    end
end
```

```
if contador == 3
    disp('Bot: Respuesta completa sobre el viaje de Colón.');
```

```
elseif contador == 2
    disp('Bot: Tu pregunta parece relacionada con el descubrimiento de América.');
```


```
    disp('¿Quieres saber sobre la fecha o las carabelas?');
```

```
elseif contador == 1
    disp('Bot: Detecto que hablas de historia. ¿Podrías especificar más?');
```

```
else
    disp('Bot: No entendí la pregunta.');
```

```
end
```


-  Aquí estás usando lógica por niveles de coincidencia.


-  **2 Nivel intermedio: sugerir qué falta**
- Más inteligente: mostrar qué palabra clave no apareció.
- claves = ["colon", "america", "1492"];
faltantes = [];

```
for i = 1:length(claves)
    if ~contains(usuario, claves(i))
        faltantes = [faltantes claves(i)];
    end
end
```

```
if length(faltantes) == 0
    disp('Bot: Pregunta completa detectada.');
```

```
else
    disp('Bot: Tal vez quieres preguntar sobre:');
    disp(faltantes)
end
```

-  Si escribe:
- colon america
- El bot responde:
- Tal vez quieres preguntar sobre: 1492
- Eso ya es comportamiento tipo asistente.

-  **3 Nivel universitario: porcentaje de similitud**
- Más elegante: calcular porcentaje de coincidencia.
- claves = ["colon", "america", "1492"];
coincidencias = 0;

```
for i = 1:length(claves)
    if contains(usuario, claves(i))
        coincidencias = coincidencias + 1;
    end
end
```

```
porcentaje = coincidencias / length(claves);
```


```
if porcentaje >= 0.7
    disp('Bot: Entiendo casi completamente tu pregunta.');
```


```
elseif porcentaje >= 0.4
    disp('Bot: Entiendo parcialmente tu pregunta.');
```


```
else
    disp('Bot: Muy poca información detectada.');
```

```
end
```

- Esto ya introduce:
- Métrica de similitud
- Lógica probabilística simple
- Fundamentos de clasificación

-  **Nivel más técnico aún (tipo recuperación de información)**
- Tokenizas primero:
- palabras = split(usuario);
claves = ["colon", "america", "1492"];
- coincidencias = sum(ismember(palabras, claves));
- Esto evita falsos positivos como:
- colonizacion
- Porque compara palabra completa.

-  **Conceptos académicos que puedes enseñar**
- Matching parcial
- Sistema de recomendación básico
- Similitud léxica
- Conteo de intersección de conjuntos
- Umbral de decisión

-  **Si quieres subirlo más**
- Podemos hacerlo tipo:
- Ranking de posibles respuestas
- Matriz de pesos
- Mini modelo TF-IDF manual
- Sistema tipo motor de búsqueda histórico

- Pregunta estratégica:
- ¿Quieres que el bot actúe como tutor inteligente (haciendo preguntas de vuelta) o como sistema informativo directo?

- Perfecto. Entonces lo diseñamos como **tutor inteligente**, no solo informativo.
El bot ahora:
- Detecta coincidencias parciales
- Evalúa nivel de comprensión
- Hace preguntas de seguimiento
- Guía al estudiante
- Esto ya es un **sistema tutorial basado en reglas con retroalimentación adaptativa**



Diseño del Tutor Inteligente

- Tema: Descubrimiento de América
- Estrategia:
- $\geq 70\%$ coincidencia \rightarrow Respuesta completa
- 40–69% \rightarrow Respuesta parcial + pregunta guía
- $< 40\%$ \rightarrow Redirigir y sugerir tema

• Código Completo en MATLAB

```
clc;
clear;

disp('TUTOR INTELIGENTE - Descubrimiento de
América');
disp('Escribe "salir" para terminar');
disp('-----');

claves =
["colon", "america", "1492", "carabelas", "reyes", "isabel",
"fernando"];

while true

    usuario = input('Tú: ', 's');
    usuario = lower(usuario);

    % Normalizar tildes
    usuario = replace(usuario, 'á', 'a');
    usuario = replace(usuario, 'é', 'e');
    usuario = replace(usuario, 'í', 'i');
    usuario = replace(usuario, 'ó', 'o');
    usuario = replace(usuario, 'ú', 'u');

    if strcmp(usuario, 'salir')
        disp('Tutor: Excelente trabajo hoy. ');
        break;
    end

    palabras = split(usuario);
    coincidencias = sum(ismember(palabras, claves));
    porcentaje = coincidencias / length(claves);
```

```
% Nivel alto
if porcentaje >= 0.7
    disp('Tutor: Muy buena pregunta completa. ');
    disp('Colón llegó a América en 1492 con apoyo
de los Reyes Católicos. ');
end

% Nivel medio
if porcentaje >= 0.3 && porcentaje < 0.7
    disp('Tutor: Tu pregunta es parcialmente
correcta. ');


    if contains(usuario, 'colon')
        disp('¿Recuerdas en qué año ocurrió el
viaje? ');
    end

    if contains(usuario, '1492')
        disp('¿Sabes quién financió el viaje? ');
    end

    if contains(usuario, 'carabelas')
        disp('¿Puedes nombrar las tres carabelas? ');
    end
end

% Nivel bajo
if porcentaje < 0.3
    disp('Tutor: Intenta incluir palabras como:
Colón, 1492, carabelas o Reyes Católicos. ');
end

end
```

-  **Qué hace este tutor**
- ✓ Detecta coincidencia parcial
- ✓ Evalúa porcentaje
- ✓ Genera retroalimentación adaptativa
- ✓ Formula preguntas para estimular memoria
- Esto ya se acerca a un **ITS (Intelligent Tutoring System)** básico.

```
% Texto de entrada
textData = "Quiero saber el horario de tecnología";

% Tokenización (extraer palabras)
documents = tokenizedDocument(textData);

% Normalización (todo en minúscula)
documents = lower(documents);
documents = removeStopWords(documents);

% Mostrar tokens
disp(documents)

% Clasificación (ejemplo simple con palabras clave)
if contains(textData, "horario")
    respuesta = "El horario de Tecnología es martes y jueves a las 10:00 AM.";
else
    respuesta = "No entendí tu consulta, ¿puedes repetirla?";
end

disp(respuesta)
```