

AI CHATBOT FROM SCRATCH – COMPLETE GUIDE (Beginner → Technical → Full Nerd Mode)

SECTION 1 — BEGINNER FRIENDLY

What is an AI chatbot?

An AI chatbot is a program that reads text, processes it, and generates a response. Some chatbots use rules, some use machine learning, and some use neural networks like GPT.

How chatbots think (simple):

Input text → break into words → decide what the user means → generate reply.

SECTION 2 — RULE-BASED CHATBOT

Rule-based chatbots use fixed patterns to respond. Example logic:

if "hello" in message:

 respond("Hi there!")

This is very simple and cannot create new phrases.

SECTION 3 — ML CHATBOT (MARKOV CHAINS) - DETAILED EXPLANATION WITH DIAGRAMS

Machine learning chatbots go beyond rule-based responses by learning patterns from text data. They can generate responses they haven't seen before.

How Markov Chains Work

A Markov Chain predicts the next item in a sequence based on the current state. For chatbots, each 'state' is a word.

Step-by-Step Example:

1. Collect training text: "hello world hello ai world ai makes phrases hello ai"
2. Tokenize into words: ['hello', 'world', 'hello', 'ai', 'world', 'ai', 'makes', 'phrases', 'hello', 'ai']
3. Build Markov chain dictionary:
{

```
'hello': ['world', 'ai'],  
'world': ['hello', 'ai'],  
'ai': ['world', 'makes', 'hello'],  
'makes': ['phrases'],  
'phrases': ['hello']  
}
```

Diagram:

hello → world → hello → ai → world → ai → makes → phrases → hello → ai

4. Generate text: start with a word, pick the next word randomly from the dictionary.

Example output: "hello ai makes phrases hello ai world"

Why Important:

- Introduces probabilistic text generation
- Responses are dynamic
- Teaches contextual word relationships

Limitations:

- Cannot understand long-term context
- May produce nonsensical sentences
- Limited vocabulary

Enhancements:

- Higher-order chains (consider last 2-3 words)
- Text preprocessing
- Combine with neural networks for smarter outputs

SECTION 4 — NEURAL NETWORK CHATBOT (WITH DIAGRAMS)

Neural networks can learn complex patterns and generate smarter responses.

Step 1 — Tokenization (Words to Numbers)

`tokens = [vocab[word] for word in text.split()]`

Diagram: [hello, world, ai] → [0, 1, 2]

Step 2 — Training Pairs

Input: current word → Output: next word

(0 → 1), (1 → 2), (2 → 0)

Diagram: Node 0 --predict--> Node 1 --predict--> Node 2

Step 3 — Neural Network Forward Pass

- Input layer: one-hot vector of current word
- Hidden layer: small size (e.g., 16 neurons)
- Output layer: probability of next word (softmax)

Diagram: Input vector → Hidden layer → Output probabilities → Pick next word

Step 4 — Training (Gradient Descent)

- Predict next word
- Calculate error (predicted vs actual)
- Update weights to minimize error

Step 5 — Generate Sentences

- Start with a word
- Predict next word using the trained network
- Append and repeat for desired length

Why Important:

- Learns patterns, not just probabilities
- Can generalize to new phrases

- Forms the basis for modern AI chatbots

SECTION 5 — TRANSFORMERS (GPT-LIKE)

Transformers understand long text and generate coherent sentences.

- Self-Attention: Lets the model focus on important words
- Multi-Head Attention: Multiple perspectives on context
- Positional Encoding: Knows word order
- Feed-Forward Layers: Combine features to predict next word

Diagram:

Input text → Embedding → Attention layers → Feed-forward → Output probabilities → Generated text

SECTION 6 — MINI IMPLEMENTATION FROM SCRATCH

1. Create vocabulary
2. Convert text to tokens
3. Build small neural network weights
4. Forward pass to get predictions
5. Train with gradient descent
6. Generate new sentences

This guide explains all essential concepts, from beginner-friendly rule-based bots to mini GPT-like neural networks, with diagrams and examples for easier understanding.