

AI Beyond Chatbots

- By Akshat Giri

Why “AI Beyond Chatbots”

The reason I named this talk AI beyond chatbots is because the recent leap AI / machine learning tech has taken in the past 2 years is astronomical. However the it's perception has been condensed to helpful chatbot assistants as it's main usecases. It's so much more than that and we can do so many exciting things with this powerful tech. So I want to share my perspective on this tech and hopefully give you a framework to approach it with. Show some unique examples on how to integrate into your existing workflow or to spark your imagination by showing surface area of the tech that might get unnoticed usually.

I'll go over the usecase and some demo code to show how we can achieve the same things.

Sound good?

But before that

But before that I'll give a quick rundown on machine learning history.

And where we were before gpt, llama, midjourney, elevenlabs took over the tech space.

I promise it'll be quick and it'll give us a good jumping point.

Machine Learning has been around since the 50s

Machine learning has been around for a while, since 1950s. And a lot of the math and algorithms we use still were discovered in the 80s.

ChatGPT in the 80s?

No chatgpt in the 80s?

Fun Fact: There was actually a chatbot named Eliza in the 1960s that could supposedly fool humans, it was just a bunch of if statements under the hood. If you're interested in learning more about that talk to me after.

Anyone want to take a guess as to what was needed?

Hardware !



The constraint was hardware.

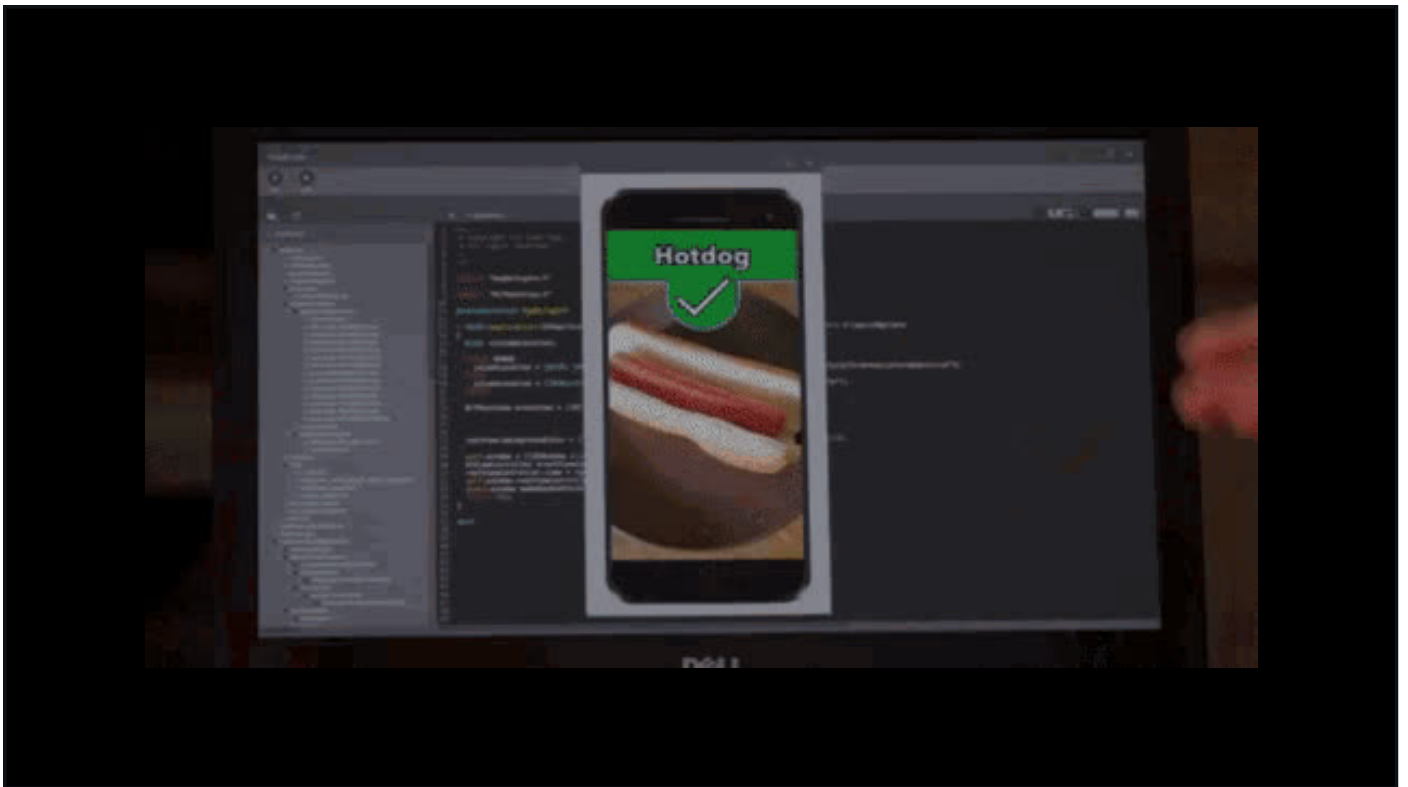
And that really changed in the 20teens with massive gpu, cpu, memory, ssd improvements.

Ofcourse there were software improvements as well, but hardware was the key ingredient. We could actually start using them in production.

Mostly Traditional AI

But even until the late 2010s we were mostly using Traditional machine learning.

What I mean by that is most of the real usecases were around narrow AIs.



The way I like to think about it is that models were mostly useful for Segmentation, classification.

The simplest example is hotdog not hotdog.

Where our input to our model is an image and the output has 2 options. Hotdog or not hotdog.

We can feed data into our model and it can learn to segment data into buckets that we define.

So we can say input is text, and we want to classify the motion. Like happy, sad, angry, surprised. In that case we are telling the model to segment data into 4 buckets.

There was some re-inforcement machine learning and, and some evolutionary algorithms but mostly the buzz was around this outside of research.

ChatGPT in 2010s?

But that still doesn't bring us Chatgpt or the boom of generative AI that we have now.

There was still 1 more key ingredient missing?

Any guesses?

Transformers!

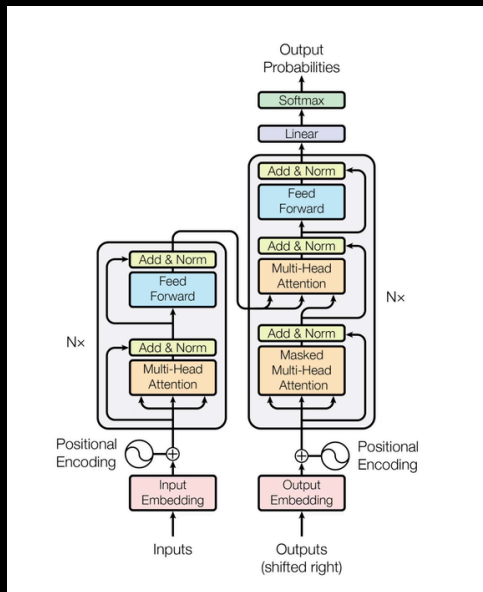


Researchers from google released a paper called "Attention is all you need" in 2017.

and in that paper they introduced the concept of A transformer model.

Attention Model

<https://deepgram.com/learn/visualizing-and-explaining-transformer-models-from-the-ground-up>



We don't need to go into detail about how the transformer model works.

But if you're curious I can share more details afterwards.

Essentially it made generative AI alot more efficient to train and run.

It just kept scaling!

Openai took this concept and just kept scaling it and it kept working.

They've done fantastic research on the subject and their technical blogposts are amazing as well.

Gpt-2, gpt3, gpt3.5 and gpt4.

A lot of other players entered the space.

Mistral AI with Mistral, Anthropic with Claude, Meta with Llama, twitter with Groq.

What's different about gen AI?

So what's different about generative AI?

Instead of defining and telling our model here segment the data into 5 buckets.

We allow it to output "tokens", you can think of them as alphabets.

So instead of segmenting emotions like happy, sad, angry, surprised which would map to 0, 1, 2, 3.

We let the model output tokens, for a simple example let's assume 26 tokens, the english alphabet a-z. We output 1 character at a time and feed it back into the model to predict the next alphabet. So it can literally generate the word "happy", "angry", "sad", "surprised".

0 - Happy
1 - Sad
2 - Angry
3 - Surprised

0 - a
1 - b
2 - c
3 - d
4 - e
5 - f
6 - g
7 - i
. . .

So instead of segmenting emotions like happy, sad, angry, surprised which would map to 0, 1, 2, 3.

We let the model output tokens, for a simple example let's assume 26 tokens, the english alphabet a-z. We output 1 character at a time and feed it back into the model to predict the next alphabet. So it can literally generate the word "happy", "angry", "sad", "surprised". t



data data data data

And if we make a model big enough and train it with enough data... let's say the entire internet corpus.

Some interesting properties start to emerge.

Fooled a rock into thinking



Yeah It can talk like a human. It can understand language, and images and videos.

But more than that, it can reason, it can do logic. It can generate images out of noise, that look completely real with understanding of people and concepts.

It's not completely at our level yet. Intelligence isn't solved, but we can start to do some very interesting things with this tech.

It's a big deal.

At a basic level the model is still just predicting the next token based on it's training data. Some might say that it's just compression. But it's clearly more than that.

LLMs = Reasoning Engine

So I think the best way to think about llms is to think of them as a reasoning engines.

They have fundamental understanding of language, logical concepts. They have a vast amount of information and references that we can use.

They can reason.

Why is that useful?

So why is that useful?

Just like previously knowing Like traditional ai models it was useful to think of them as useful for segmenting, it's useful to think of large language models as reasoning engines. They're another tool in our dev toolbelts.

When we come across a problem where we would want something / someone to reason for us before taking the next step, we can reach for an LLM or another generative ai model.

Chatbots are the most obvious use case of LLMs. The foundation models which just does text completion isn't made to be a chatbot, however we can do some training on top of the foundation

Size and Quantization

Now their capabilities and their ability to reason and have accurate data is depends on a few factors.

Their size,
architecture,
training,
data etc.

But for our purposes, when judging a model we're going to look at 2 things.

Size which is typically measured in number of parameters.

Typically billions of parameters.

Common sizes are 3billion, 7billion, 30billion, 40billion, 70billion, 120billion, 300billion etc.

Quantization refers to the precision level of the weights and activations (floating point numbers). Typically 32bit, 16bit, 8bit.

The higher the better the model is. But also more expensive to run, higher hardware requirements. So often the tradeoff is worth it.

Foundation vs Chat Models.

Foundation model is the base model that simply does text completion. You can input any text and it will try to continue that text.

These foundation models are further trained / tuned to become chat models.

Chat models are industry standard now, and foundation models are less used for some reason.

Non deterministic

One last thing before we get into some fun stuff.

The way we typically program is that we our code is deterministic. The computer is doing exactly what we tell it to. Nothing more nothing less.
Given the same input we get the same output.

LLMs and machine learning in general is non deterministic. Given the same input we can often get different outputs. In traditional ai this is less so. Our model can be wrong in it's prediction but we always get one of the defined outputs.

Here our model has free range to output pretty much anything.

**Alright, let's
have some fun!**

Let's have some fun.

We're going to try to use

Chat Moderation

We've all been in a group chat that started out as fun chat between a few people, that is slowly devolves into 2 people arguing over something off topic. And then chat slowly dies off.

This is so common in bigger community group chat with lots of people that community owners pay hundreds / thousands of dollars to chat moderators to manage community chats.

If we want to programmatically manage a chat, we need a tool that understands language. When someone is being mean, when the conversation goes off topic etc etc.

I think we have just the thing, and someone's already done this so let's look at that. My boy levels.io on twitter runs nomadlist.

Nomdalist telegram community chat

- @alevelsio



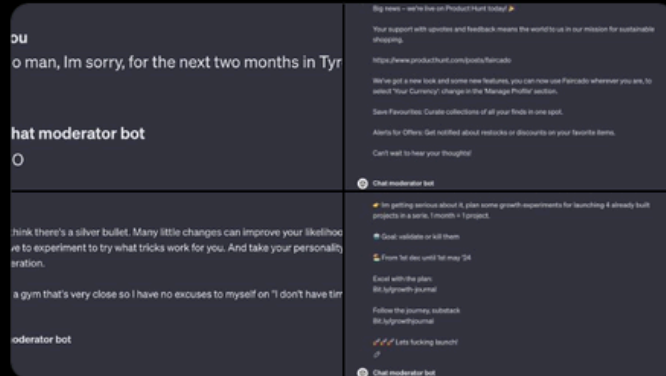


@levelsio

Subscribe

🤖 Made a ChatGPT moderator bot for Nomad List's Telegram chat, it checks every message if it's spam

"You are a chat moderator bot. You review every message to see if it's self promotion or spam. Reply to every message with YES or NO if you think it's spam and should be removed"



4:36 AM · Nov 24, 2023 · 213.3K Views



@levelsio

Subscribe

🔥 Nomad List's GPT4-based 🤖 Nomad Bot I built can now detect identity politics discussions and immediately 🚫 nuke them from both sides

Still the #1 reason for fights breaking out

This was impossible for me to detect properly with code before GPT4, and saves a lot of time modding

I think I'll open source the Nomad Bot when it works well enough

Other stuff it detects and instantly nukes (PS this is literally just what is sent into GPT4's API, it's not much more than this and GPT4 just gets it):

- links to other Whatsapp groups starting with wa.me
- links to other Telegram chat groups starting with t.me
- asking if anyone knows Whatsapp groups about cities
- affiliate links, coupon codes, vouchers
- surveys and customer research requests
- startup launches (like on Product Hunt)
- my home, room or apartment is for rent messages
- looking for home, room or apartment for rent
- identity politics
- socio-political issues
- United States politics
- crypto ICO or shitcoin launches
- job posts or recruiting messages
- looking for work messages
- asking for help with mental health
- requests for adopting pets
- asking to borrow money (even in emergencies)
- people sharing their phone number

I tried with GPT3.5 API also but it doesn't understand it well enough, GPT4 makes NO mistakes

```
"- self promotion
- spam
- surveys
- room or apartment for rent posts
- links to Whatsapp or Telegram groups
- job or work recruiting posts
- looking for work posts
- asking for help with mental health or medical conditions
- affiliate links, coupon codes, vouchers
- messages about putting pets for adoption
- messages asking for money because people lost their money
- people sharing their phone number
- non-English messages";
```

**Let's build
this ourselves**

```
1 import TelegramBot from 'node-telegram-bot-api'
2
3 const TELEGRAM_BOT_TOKEN = process.env.TELEGRAM_BOT_TOKEN
4
5 // setup telegram bot
6 const bot = new TelegramBot(TELEGRAM_BOT_TOKEN, { polling: true })
7
8 // listen for new messages
9 bot.on('message', (msg) => {
10   const chatId = msg.chat.id
11   bot.sendMessage(chatId, 'Received your message')
12 })
13
```

```
1 import Groq from 'groq-sdk'
2
3 const GROQ_API_KEY = process.env.GROQ_API_KEY
4
5 // setup groq sdk
6 const groq = new Groq({ apiKey: GROQ_API_KEY })
7
8 async function checkMessageForSpam(msg: string) {
9   // TODO:
10 }
11
```

```

1 async function checkMessageForSpam(msg: string): Promise<{
2   isSpam: boolean,
3   reason: string,
4 }> {
5   const systemPrompt = `You are a chat moderator bot. You review every message to
    see if it's self promotion or spam. Reply to every message with a YES or NO followed
    by a ":" and the reason why you think it's spam or not.
6
7   Example
8   - "YES: This message contains a link to a website."
9   - "NO: This message is a question about the product."
10
11  What we categorize as spam
12  - surveys
13  - link to whatsapp or telegram groups
14  - job or work recruiting posts
15  - looking for work posts
16  - affiliate links, coupon codes, vouchers
17  - messages asking people for money because people lost their money
18  - non-english messages
19  - etc use you understanding of spam`
20
21  .
22  .
23  .
24 }
25

```



```

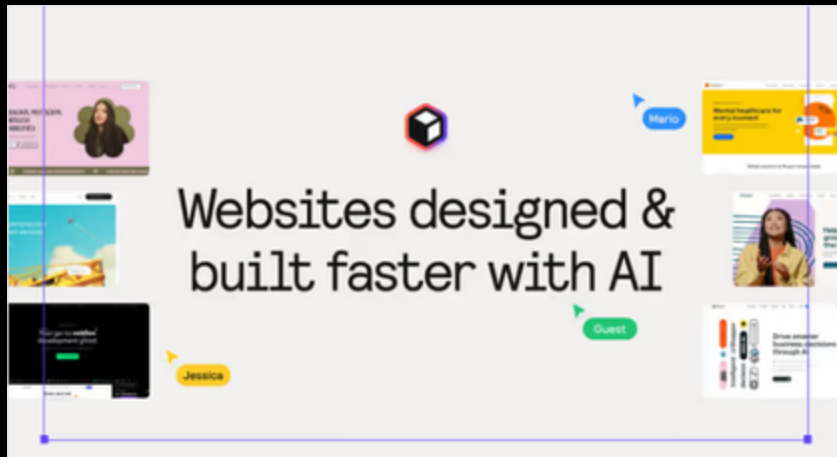
1 async function checkMessageForSpam(msg: string): Promise<{
2   isSpam: boolean,
3   reason: string,
4 }> {
5   const systemPrompt = 'You are a chat moderator bot. You ...'
6
7   const response = await groq.chat.completions.create({
8     messages: [
9       {
10        role: 'system',
11        content: systemPrompt,
12      },
13      {
14        role: 'user',
15        content: 'Is the message spam? - \n${msg}',
16      },
17    ],
18    model: 'llama-3.1-70b-versatile',
19    // randomness of the generation
20    temperature: 0.2,
21    // maximum number of tokens model is allowed to generate
22    max_tokens: 1024,
23    // likelihood-weighted options are considered. 0.5 is 50%
24    top_p: 1,
25    // string to stop the generation on
26    stop: null,
27    // whether to stream the response
28    stream: false,
29  })
30
31  const { choices } = response
32
33  const [firstChoice] = choices
34
35  const [isSpamTxt, reasonTxt] = firstChoice.message.content.split(':')
36  const isSpam = isSpamTxt.trim().toLowerCase() === 'yes'
37  const reason = reasonTxt.trim()
38
39  return {
40    isSpam,
41    reason,
42  }
43 }
44

```

```
1 // setup telegram bot
2 const bot = new TelegramBot(TELEGRAM_BOT_TOKEN, { polling: true })
3
4 // listen for new messages
5 bot.on('message', async (msg) => {
6   const chatId = msg.chat.id
7
8   const { isSpam, reason } = await checkMessageForSpam(msg.text)
9
10  if (isSpam) {
11    bot.deleteMessage(chatId, msg.message_id)
12    // store message and reason in db
13    // optionally send a message to the user with the reason
14  }
15 })
16
```

DEMO

AI Website Generator




The image displays the Relume AI website builder interface. At the top, the title "Websites designed & built faster with AI" is centered. Below the title, several website templates are shown, each with a unique color scheme and layout. User avatars are visible next to some templates, including "Mario", "Guest", and "Jessica". The Relume logo, a stylized cube, is positioned at the top center. The interface is framed by a purple border.

Websites designed & built faster with AI

Relume — Websites designed & built faster with AI | AI website builder

Use AI as your design ally, not as a replacement. Effortlessly generate sitemaps and wireframes for marketing websites in minutes with Relume's AI website builder.

 relume.io

DEMO

Function Calling

DEMO

Fake data generation

AI Agents

Data segmentation

We can use LLMs to do data segmentation.

And if you want to extract features from images or videos even that is possible.

LLAVA is great.

LLAVA For Image Classification

Questions?