planning

## Overview and structure

* Intro
  + I’d like to start off with a quick story
  + Empowerment params
    - What AI is and where LLMs fit in in the AI landscape
    - Basic intuition of how LLMs work
    - Common LLM usage pitfalls
      * Be able to have a sort of gut-feel for what tasks you can or cannot trust LLMs on
* Landscape slide -
  + How’re we going to do this
    - Transformer architecture
      * What is the attention mechanism
    - The common steps gone through for creating a basic LLM model
      * Training an LLM model
        + Tokenization
        + Pre-training
        + Post-training
      * Generating useful outputs
        + Autoregressive decoding
        + What is temperature?
    - Let’s build a (micro) LLM from scratch

## Start of main talk

* Question: What is AI
  + The AI we think of is called supervised learning because it learns from examples, we have inputs and outputs and the mathematical equations tune weights and biases that are little numbers that are tuned to create a mathematical mapping from those inputs to outputs in order to teach them how to predict, classify and do things that we find helpful
    - Common AI models and a landscape overview
    - Where do LLMs fit into this landscape
* Landscape slide - cycle
* Transformer architecture -
  + Explain attention - (very shallow explanation)
  + Autoregressive decoding
  + Attention graph?
  + Show attention in [Illustrated Transformer](https://jalammar.github.io/illustrated-transformer/)
* Landscape slide - cycle
* Tokenization - how does this model *know* what a word or text is?
  + Question: How would we feed in this information about the internet or whatever our sources is into something like an AI model that can only work with numbers
    - Character level vs word level quick overview
    - Char vs word level vs hybrid
  + Show [tiktoken](https://tiktokenizer.vercel.app/) example
    - My superhero name is Max Verstappen
    - Learning LLMs is as easy as 12.34567
    - Explain how typos are really bad
      * my faorite food is any food my wife makes... or chocolate
* Landscape slide - cycle
* Pretraining - how to speak, factual knowledge - glorified autocomplete
  + Question: How would we build a model that has information about everything? Where would we start? -> internet, books, private data
    - Lossy zip file of the internet
    - Compress the info of all the world’s knowledge
    - Show [fine web dataset](https://huggingface.co/spaces/HuggingFaceFW/blogpost-fineweb-v1), explain that it’s 44TB in size
    - Try to get a model that can memorise not just the whole internet but learn the concepts of what things in the real-world are
    - We want to build a model that has a lot of general world-knowledge and can answer questions for us
  + End off by showing that it’s only a text completer on [hyperbolic](https://app.hyperbolic.ai/models/llama31-405b-base-bf-16)
  + Question: How would we turn this base-model into an assistant that can answer questions
  + Hyperbolic demo
* Landscape slide - cycle
* Post training - how to answer questions, how to think, how to apply the knowledge it has as an autocomplete to autocomplete QA pairs
  + Hyperbolic demo of completing question answer
    - Question: what is the capital of japan? Answer:
    - Question: How do you get a Dr to use an e-scripting platform instead of paper scripts? Answer:
    - Question: Please repeat this number back exactly 123456 with no other text. Answer:
* Landscape slide - cycle
* Generation - How to answer MY question
  + [Ai-studio](https://aistudio.google.com/prompts/new_chat) - what’s behind this screen?
  + Temperature Allows us to get creative (random answers)
    - How how on [ai-studio](https://aistudio.google.com/prompts/new_chat) the answers change when you give different questions
      * Give me a short poem
      * Indicate how randomness increases the possibility of non-perfect next token
      * “Perfect” next token is just a statistical determination of what word occurs next to the preceding word in the dataset given the info before
* Landscape slide - cycle
* Let’s build our own model -
  + Tokenization
  + Pretrain
  + Post-train
  + Decode and show temperature’s effects
* Landscape slide - cycle
* Question: Which of these tasks do you think LLMs will be good at
  + Summarization
  + Copy-pasting
    - Show temperature issue
  + NER
    - Concept recognition
  + Generating starter code for a project
    - Many examples of that in the training data
  + Humanlike interaction
  + Mathematics
    - Show the tokenization concept breaking what numbers are
  + Factual retrieval
    - Pretrain date
    - Current date
  + Regenerating a response multiple times
    - Show own example of temperature
  + Grammar

### Overarching principles to keep sight of:

* Begin with an interesting story of AI to hook the audience
* Clearly establish the goals of the presentation and the “why you need to listen”
  + Empowerment parameters ✅
* Frequent checkpoints to the goals of the presentation
  + Cycle on the problem without coming across as condescending ✅
  + Repeat the intro and goal of the talk slide so that we can jump back on board at each new idea ✅
  + Ask a question at the end of each section so that we can get people back on track and engaged
    - Not too easy or hard ✅
* Make several onramp points for if someone loses concentration
* Take questions
* Lock in the keypoints in a short intuitive summary
* Always exhibit passion
* Express vision and passion in the first 5mins
  + This is a miracle of modern science
* Have one really cool demo that does something amazing!
  + Pretrained bible translator?
  + Joke maker?

Intro

## Bullet points

* 1989 Yann LeCun
  + Yann LeCun—now the Chief AI Scientist at Meta
  + Handwritten digits on envelopes
  + Didn’t use any hand-crafted rules to determine the characters but learnt from examples
  + This was the seed of our modern AI in its infancy
  + Fuelled the imagination of what computers could do
  + Show the video
* 1997 Deep Blue vs Gary Kaparov
  + IBM pinnacle of computing research
  + Chess programme called Deep blue to face gary
  + This program defeated Gary, the greatest player alive using simple rules and brute force computing
  + Taught people to dream of what could be possible, what other tasks could AI do
* 2012 AlexNet
  + Alex Krizhevsky, Geoffrey Hinton (godfather of AI), and Ilya Sutskever (former Chief Scientist of OpenAI, only guy to ever fire sam altman OPENAI CEO now the founder of a competing AI lab)
  + Created a model that could train recognise images better than any other
  + Breakthrough ->
    - GPU used for graphics processing to train models instead of CPU
    - 1000x faster
    - Unlocks ML that was thought infeasible or impossible
* 2016 alpha go
  + Google deepmind
    - Lead by Demis hasabis, former child chess prodigy now google deepmind CEO
  + Challenge in Lee Sedol greatest go player alive to Go - 3000-4000 year old game
  + Much more complex than chess
  + More board positions than atoms in the universe
  + Impossible to brute force
  + The model has to truly learn
  + Move during one game
    - Move 37
    - Everyone thought it was a mistake - commentators, pros alike
    - Genius move
    - Move that nobody would’ve played
  + Won the game
  + Signalled potential for AI to go beyond human ability in extremely difficult tasks
* 2017 AIAYN & GPT
  + Transformer architecture
  + Each word pays attention to each other word in the sequence
  + Originally used for translation
  + Didn’t realise the power of the architecture
  + Applied to everything and open ended questions
  + Gpt
    - Brings us to gpt launch in 30 Nov 2022
    - Everything changed
    - A general type of intelligence
    - Today we’re going to learn how LLMs based on this arch work

## 

## Full text

In the early 1990s, a young researcher named Yann LeCun—now the Chief AI Scientist at Meta—built a neural network that could do something magical: read handwritten digits on envelopes. It didn’t follow rules like ‘if it’s round, call it a zero.’ It learned directly from thousands of examples. That was the seed of modern AI at its infancy. It was the first proof that machines could learn useful things from example data.

But the world’s first public shock came in 1997. At the time, IBM was the pinnacle of computing research—the place where breakthroughs happened. And it was IBM’s Deep Blue (the name they gave to their chess programme) that sat across from Garry Kasparov, the greatest chess player alive. Over six tense games, Deep Blue defeated him. It wasn’t intelligence—it was brute force. Two hundred million positions per second, powered by hand-crafted heuristics from human grandmasters. Power, yes. But not understanding.

Then in 2012, three researchers in Toronto—Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever, the former Chief Scientist of OpenAI, now the founder of a competing AI lab —entered a visual recognition contest called ImageNet. Their system, AlexNet, trained on two gaming GPUs, demolished the competition. That moment wasn’t just an algorithmic breakthrough—it was GPUs unlocking deep learning, scaling LeCun’s original ideas into something world-changing.

But the real turning point came in 2016. Google DeepMind’s AlphaGo took on Lee Sedol, one of the greatest Go players in history. Now, Go isn’t chess... it's much much harder. Go has more possible board states than atoms in the universe. You can’t brute force it the way that Deep Blue did to Kasparov. The only way for a computer to win is to learn.

AlphaGo trained on human games, then improved through self-play on massive GPU clusters. In Game Two it played the now-legendary Move 37—so unexpected that Sedol left the room in shock. It wasn’t a blunder but brilliance, a moment that showed AI could move beyond imitation into discovery. And behind it was Demis Hassabis—once a chess prodigy, now CEO of Google DeepMind.

The very next year, 2017, a paper quietly appeared: Attention Is All You Need. The Transformer architecture (the version of the architecture behind modern LLMs) used a mechanism called attention, which lets the model focus on the most relevant parts of a sequence when making predictions. It was first applied to translating between languages, and at the time the authors didn’t fully realise the power of what they had created. Soon it became clear that Transformers were far more general than translation: a universal architecture for language, vision, biology, and beyond. That breakthrough lit the fuse for GPT and the modern era of large language models.

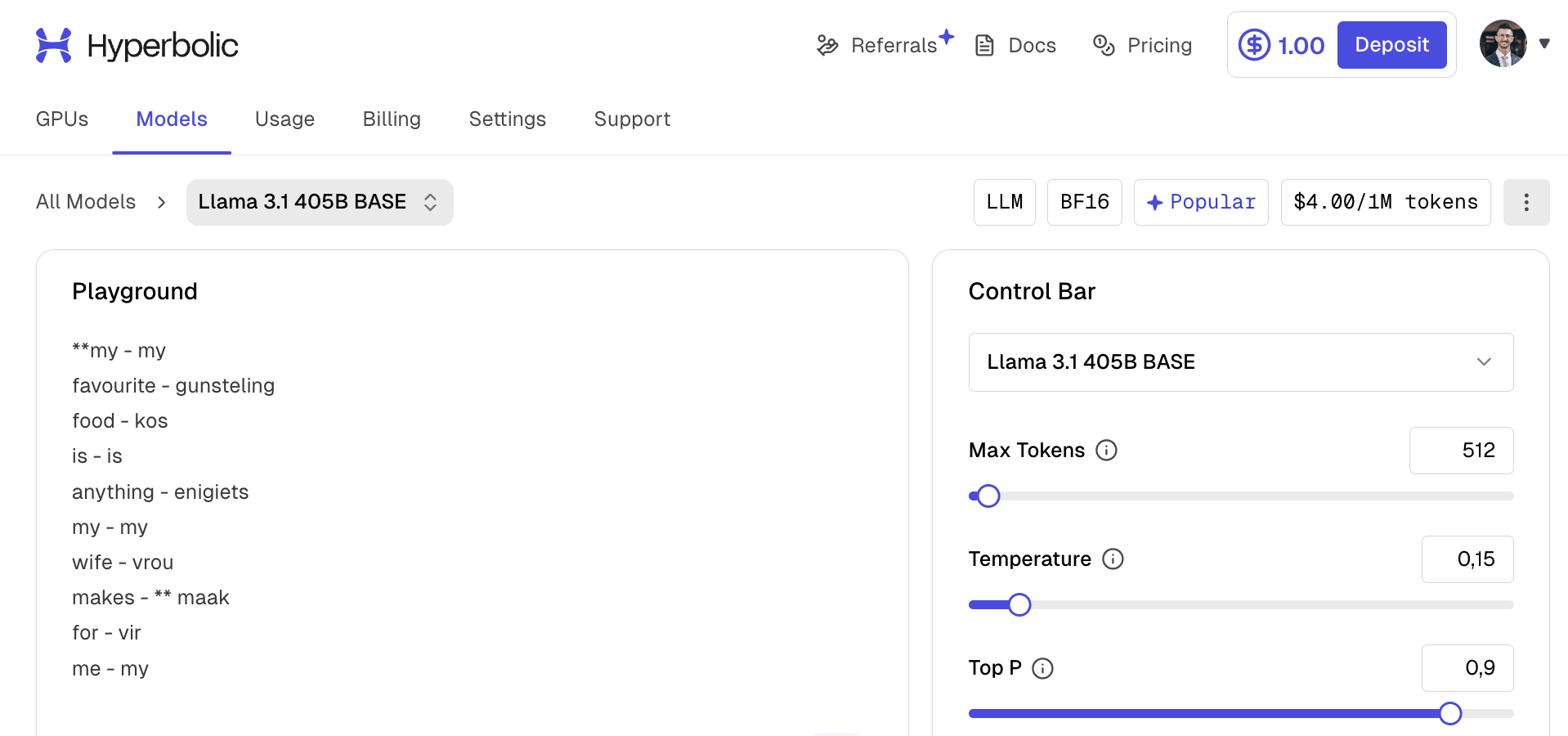
And then, in late 2022, the world met GPT. Not just researchers. Not just engineers. Everyone. Millions logged into ChatGPT and, for the first time, felt what it was like to converse with a machine that could write, reason, and explain. What began with LeCun’s digit recogniser has become the language models shaping our world today.

Today we're going to talk about how LLMs work and the intuition behind them.

not doing

## Not in this talk

* Goal of the talk
  + Teach you all the details of what an LLM is, where it fits into the broader topic of “AI” and learn how it works intuitively
  + Then we’re going to build up a tiny LLM model together in code
  + Know how to spot common pitfalls of using LLMs and the reasons why they occur to empower you to use LLMs more effectively next week when Marius and Peter Cressey give the next LLM masterclass
* How to use LLMs effectively
  + RAG/context engineering
  + How to use as a dev
    - Templates
    - AI Studio
  + Multiple attempts
  + New chats
    - Try to get it in one-shot
      * Multiple questions create confounding attention
* LLM pitfalls
  + Copy-pasting
    - Copy a very large list and show that it’s not correct
  + Mental math
    - Big numbers multiplication breaks
  + Hallucinations and factual retrieval
  + Stochasticity
    - Ask for a poem multiple times, different results
    - Temperature increases randomness
* What LLMs are very good at
  + Summarization
    - attention mechanism can capture the semantic meaning/vibe
  + Grammar
    - Pretrained on high quality data without grammar “errors”
  + Code syntax
    - Pretrained on so much code
  + Coding overall
    - Getting better every day
    - Good at common tasks or even hard tasks that are broken down easily
    - Making code work together



my - my

favourite - gunsteling

food - kos

is - is

anything - enigiets

my - my

wife - vrou

makes -