

# Theory, Practice and Future of LLMs

Denison University  
Computer Science and Math Departments  
2 Nov 2023

• • •

Jon Chun  
Kenyon College

[jonachun.com](http://jonachun.com)  
@jonchun2000

- Teaching
  - First human-centered AI curriculum (2016)
  - 300+ Mentored student research at [digital.kenyon.edu/dh](http://digital.kenyon.edu/dh)
- Interdisciplinary Research
  - APQ: Novel Jailbreak Ethical Reasoning Benchmarks of 12 Leading LLM
  - IJDH: Novel GreyBox Ensemble eXplainable AI
  - Spring Nature IJHAC: AI Challenges and Interdisciplinary Curriculum
  - (current): LLM Affective AI, Synthetic Data, Small 7B-13B Open LLMs
- Presentations
  - Limits of Cognition (Theoretical CS)
  - Medical Applications of AI (FDA association)
  - Manufacturing Engineering/Physics Simulation (Synthetic Time Series)
  - Higher Ed (Interdisciplinary AI)
- Consulting
  - Finance (VC, Private Equity, Industry)
  - Startup Consultant
- Industry
  - Silicon Valley Startups to Fortune 500 Director of Development (Sale Engineer to Patents)
  - Computer Security, FinTech, MedTech, InsurTech, Semiconductors: US, Asian and Latin America
- Paper/Book Reviews, Grants, Non-Profit, [jonachun.com](http://jonachun.com) and Tweets @jonchun2000

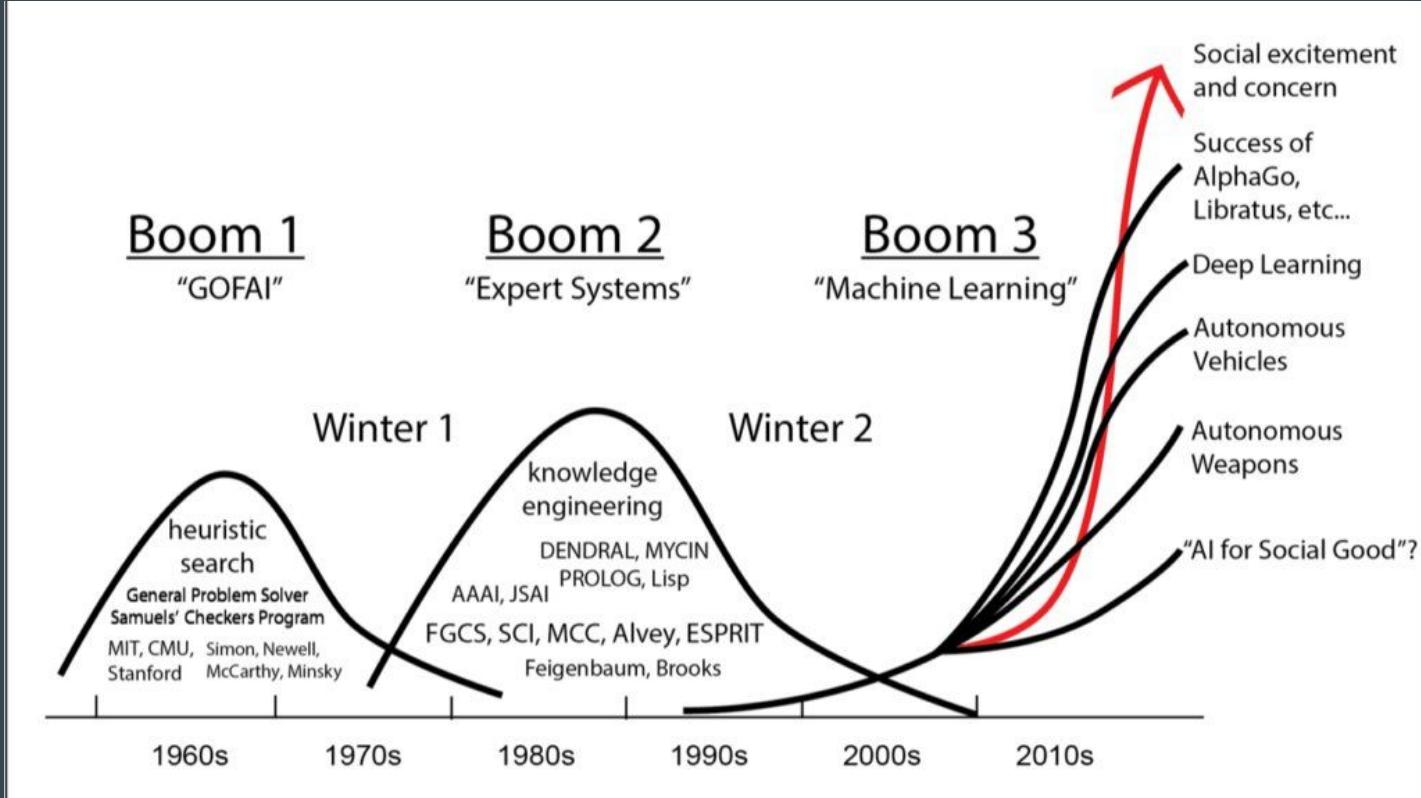
# Overview

- History
- Theory
  - Information
  - Representation
  - Learning
  - Reasoning
- Practice
  - Transformers Architecture
  - Inference vs Training
- Future

# History

# Is This Time Different?

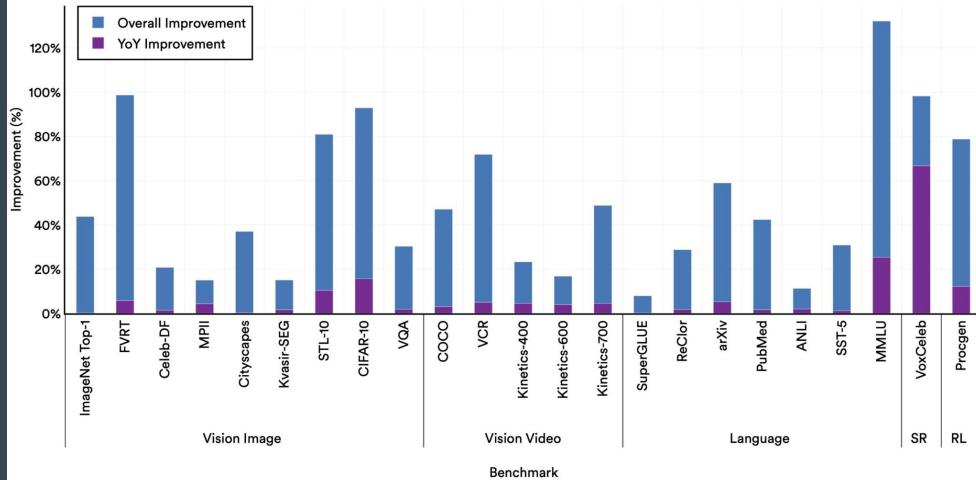
Francis Fukuyama  
**THE END OF HISTORY AND THE LAST MAN**



# Towards AGI

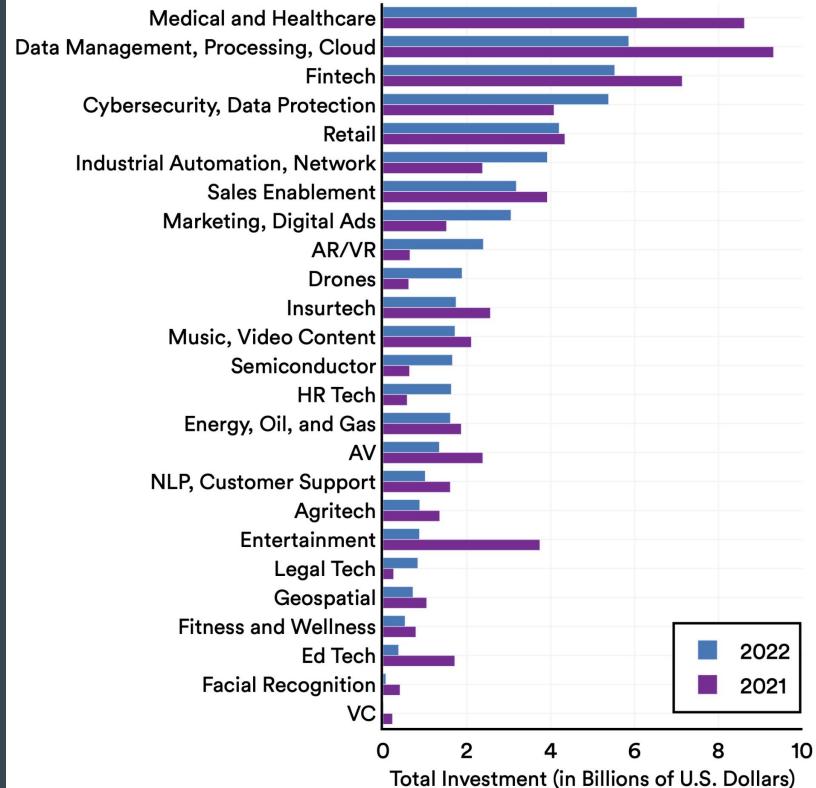
**Improvement Over Time on Select AI Index Technical Performance Benchmarks**

Source: AI Index, 2022 | Chart: 2023 AI Index Report



## Private Investment in AI by Focus Area, 2021 Vs. 2022

Source: NetBase Quid, 2022 | Chart: 2023 AI Index Report



# Playground

Your presets

Save

View code

Share

...

It was a dark and stormy night. Suddenly, a shot rang out. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it.



ir through = 98.58%

throughout = 0.86%

across = 0.24%

off = 0.19%

in = 0.05%

Total: -6.04 logprob on 1 tokens  
(99.92% probability covered in top 5 logits)

The travelers, who had been huddled around the fire in the common room, jumped and looked at each other in fear. They had been on their way to the nearby city when the storm had forced them to take shelter at the inn. They were all strangers to each other, but in that moment, they were all connected by the sense of unease that hung in the air.

One of the travelers, a young woman with long, dark hair, stepped forward. "I

## Model

gpt-3.5-turbo-ins...

Temperature 1

Maximum length 256

Stop sequences  
Enter sequence and press Tab

Top P 1

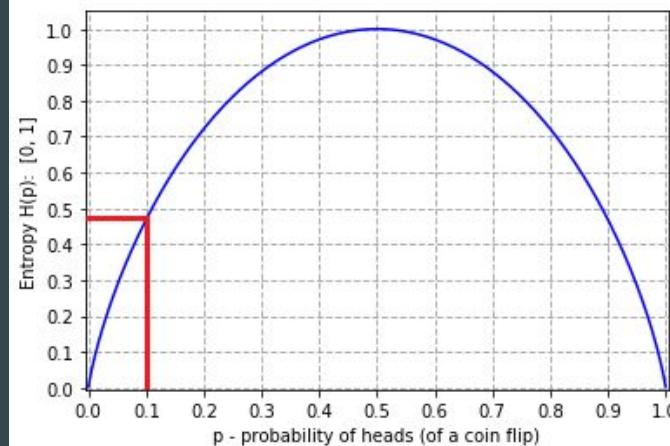
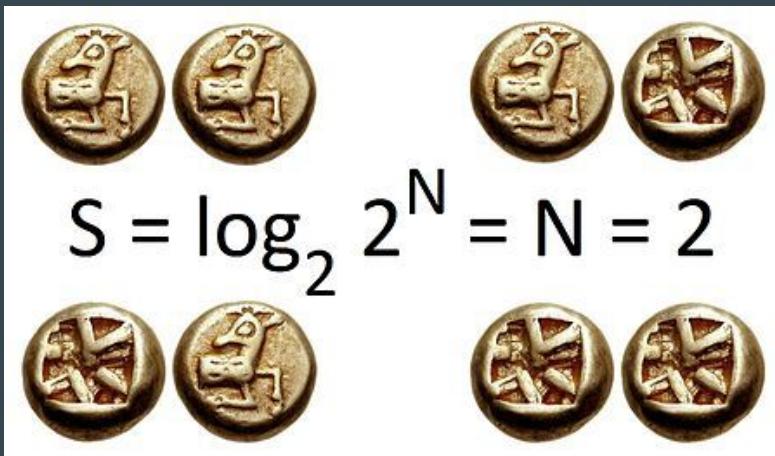
Frequency penalty 0

Presence penalty 0

>>> Information <<<  
Representation  
Learning  
Reasoning

# Entropy

- Fundamental (mass or energy)
- Metric of uncertainty/disorder (no time, isolation, events-messages)
- Compression ~ Intelligence (id patterns, efficient representations, feature extraction, prediction, generalization)

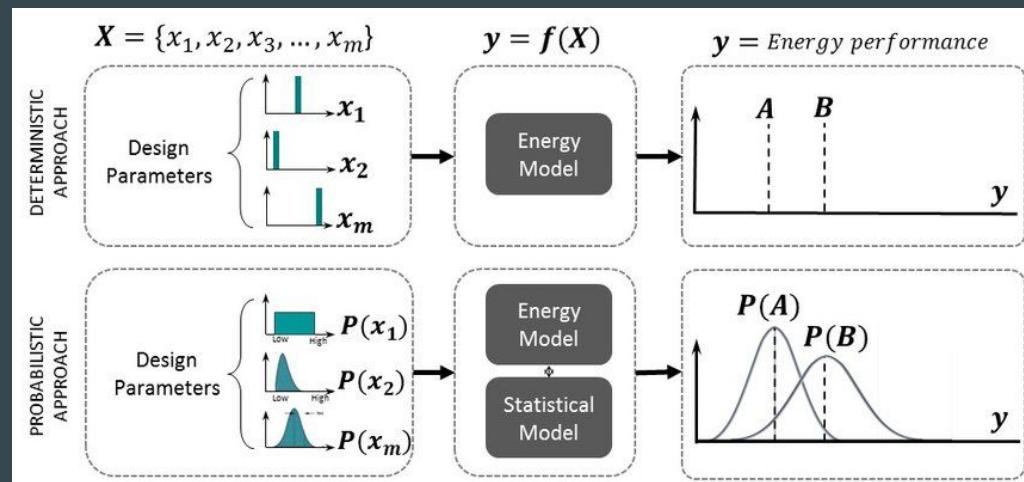
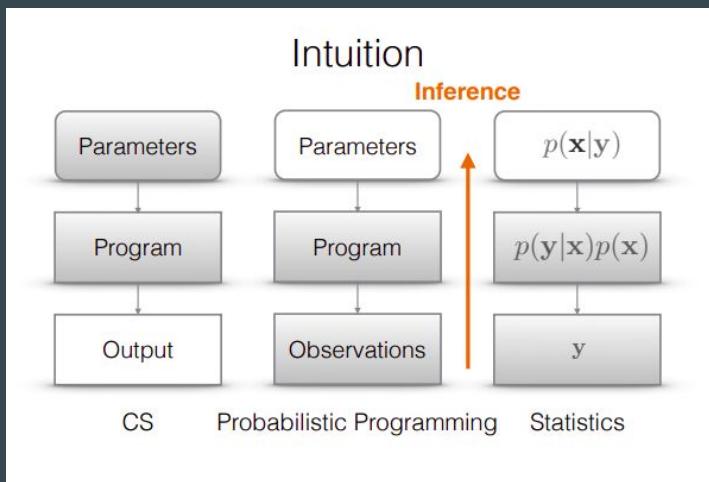


$$p = 0.5 \\ H(p) = 1$$

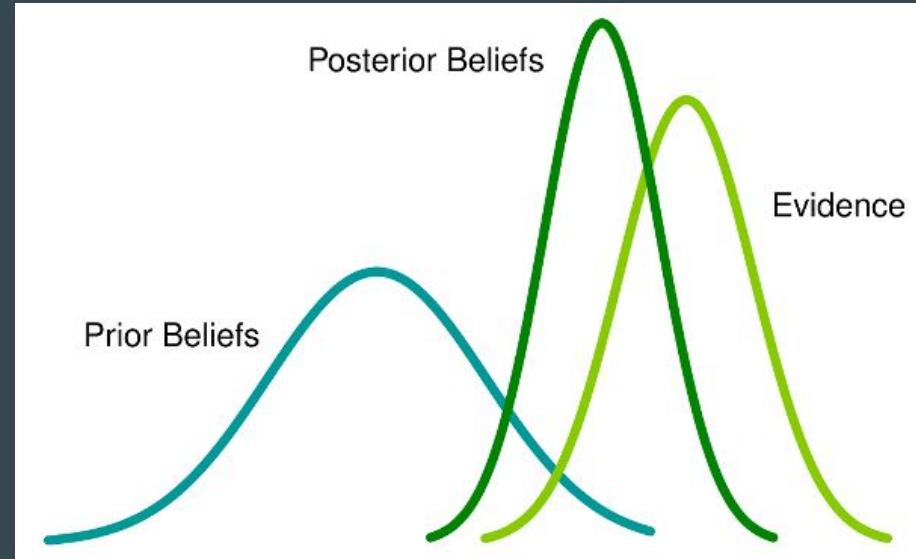
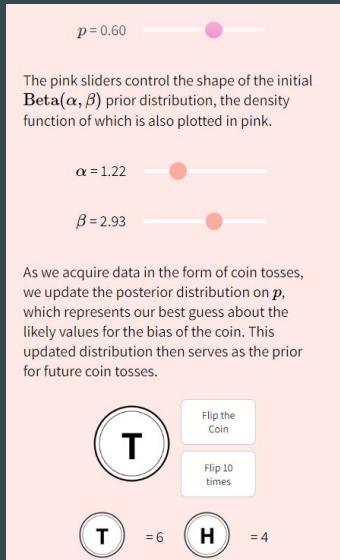
$$p = 0.1 \\ H(p) = 0.469$$

# Probabilistic Thinking

- Deterministic to Probabilistic
- Point Value to Distributions
- Implicit to Explicit Approximations



# Distribution Sampling & Types



# Information: Data is Code is Data

## Growing Neural Cellular Automata

Differentiable Model of Morphogenesis



Speed: 1x  
Step 1412 (144.0 step/s)

Model type:  
 Growing  
 Persistent  
 Regenerating

Rotation 0° [experiment 4]

Regenerating models were subject to pattern damages during training, so their regenerative capabilities are much stronger, especially in the central area. [experiment 3]

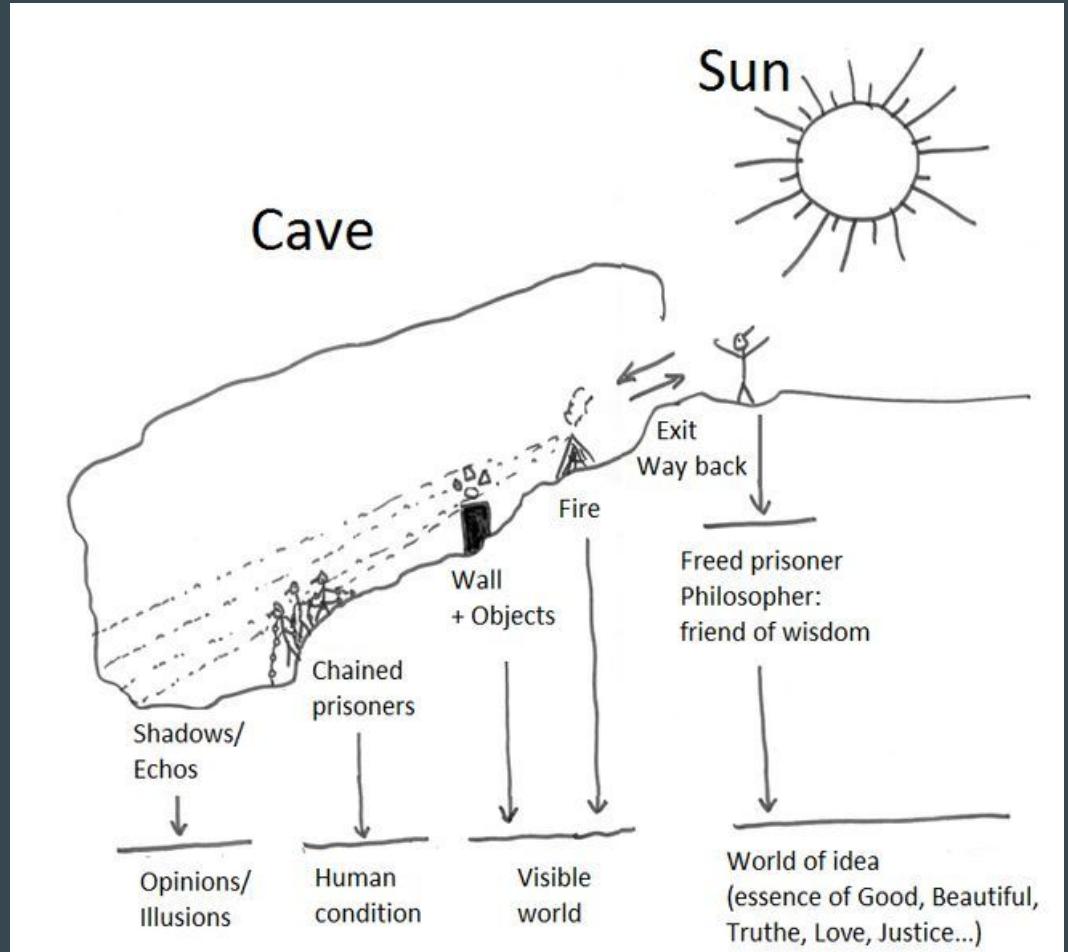
TRY IN A  NOTEBOOK

# Information > Modality: MONAIC vs TensorFlow

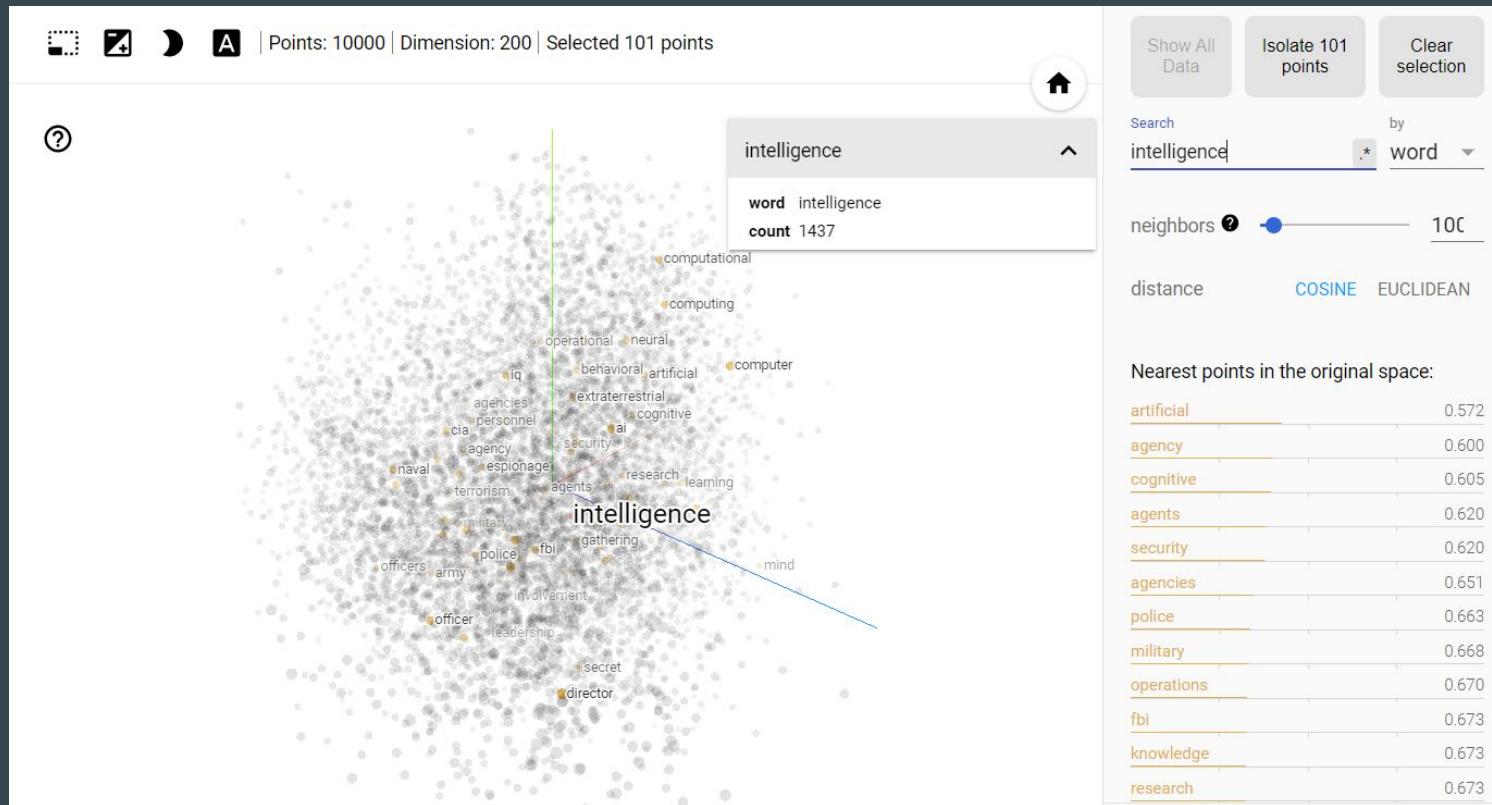


Information  
">>>> Representation <<<  
Learning  
Reasoning

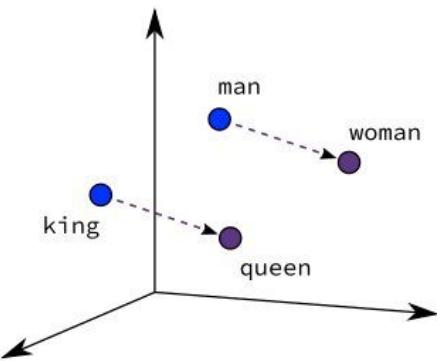
# Plato's Cave



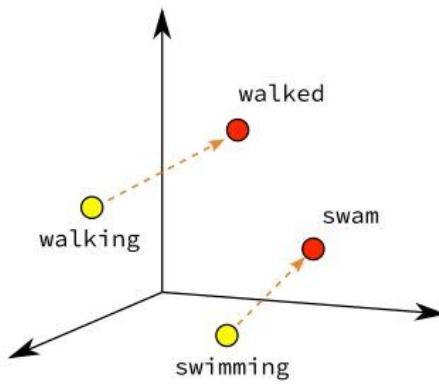
# Revealing Latent Semantic Structure in Language



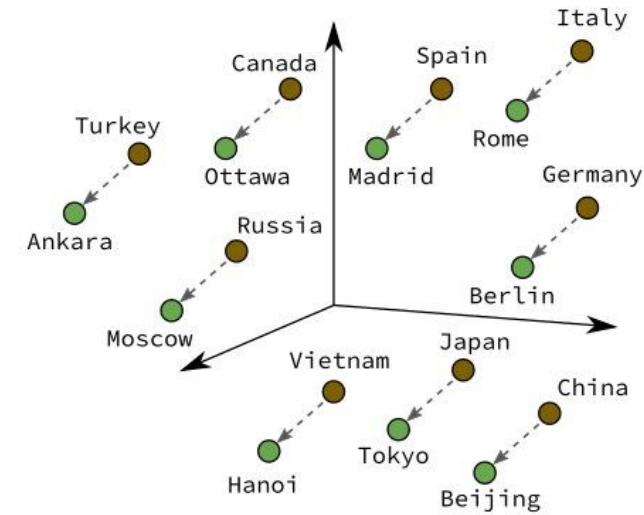
# Semantically Meaningful Dimensions



Male-Female



Verb Tense



Country-Capital

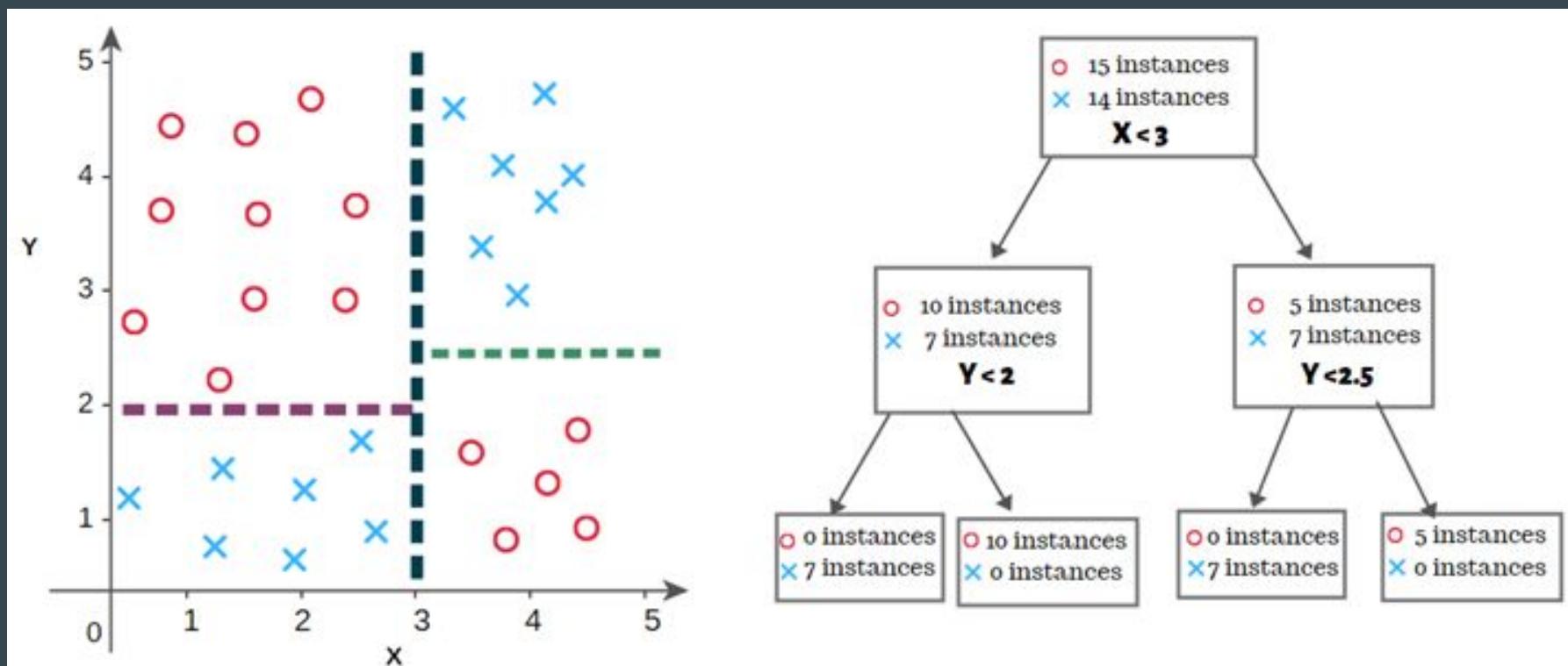
# Latent < [Semantic] < Manifolds

- OPERATION:
  - 1. Up-sampling Embeddings (Viz)
  - 2. Down-sampling Dimensionality Reduction/Projection (3D to 2D, pros/cons)
  - 3. Change of Basis
  - 4. Structure: None, Algebraic, and Geometric/Topological
- GOAL: Feature Engineering & Semantic Algebra (Viz)
- CHALLENGES:
  - Coverage
  - OOV
  - Sample Efficiency
  - High Dimensionality (sparsity/curse of dimensionality, overfitting, stability, interpretability, etc.)
- RESEARCH
  - Cognition as 1. goal-oriented transformations of in latent space and/or 2. directed movement within

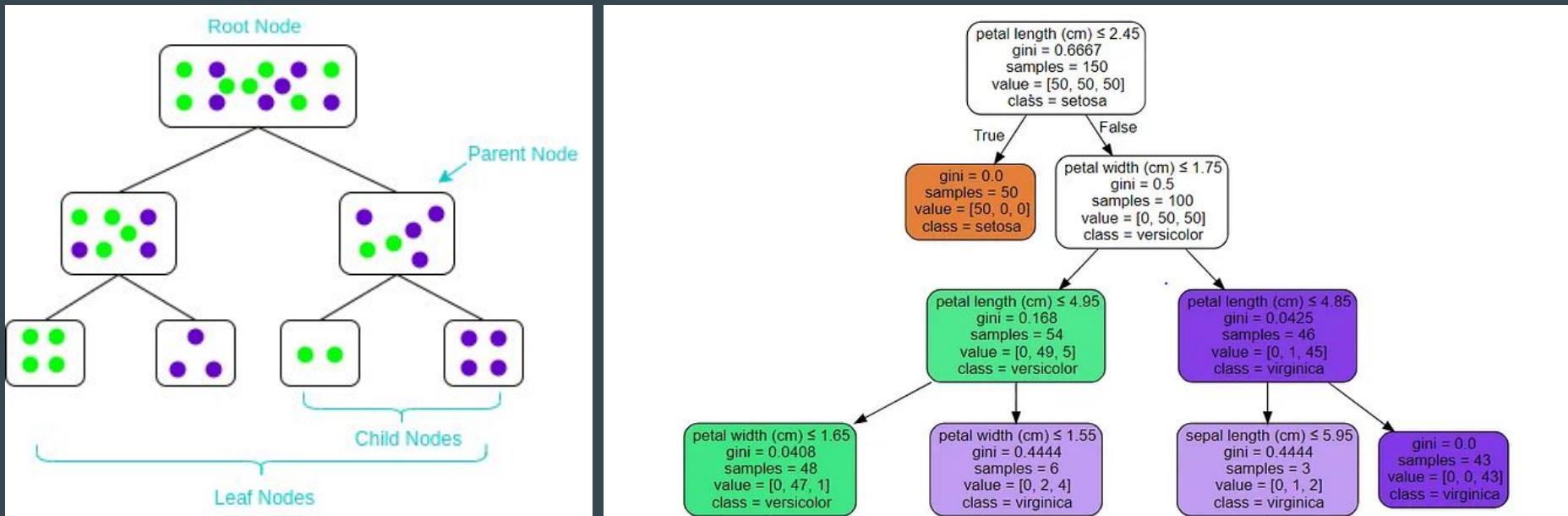
# Transformations

Information  
Representation  
>>> Transformations <<<  
Superposition  
Learning  
Reasoning

# Decision Trees (GOFAI Linear Decision Boundaries)

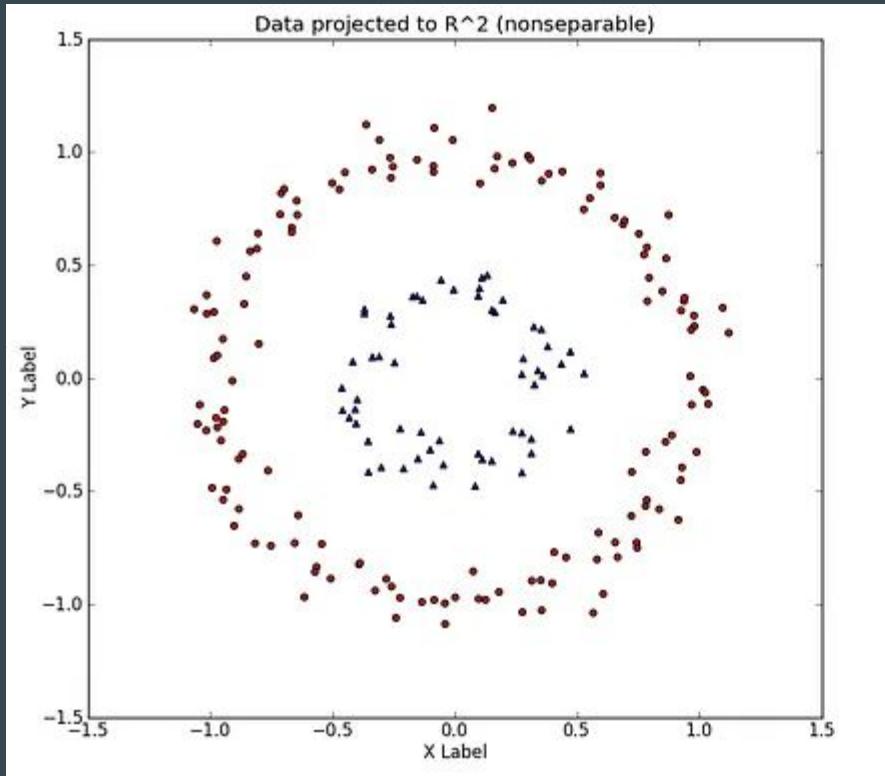


# Objective Function: Min(Entropy) in Children Nodes

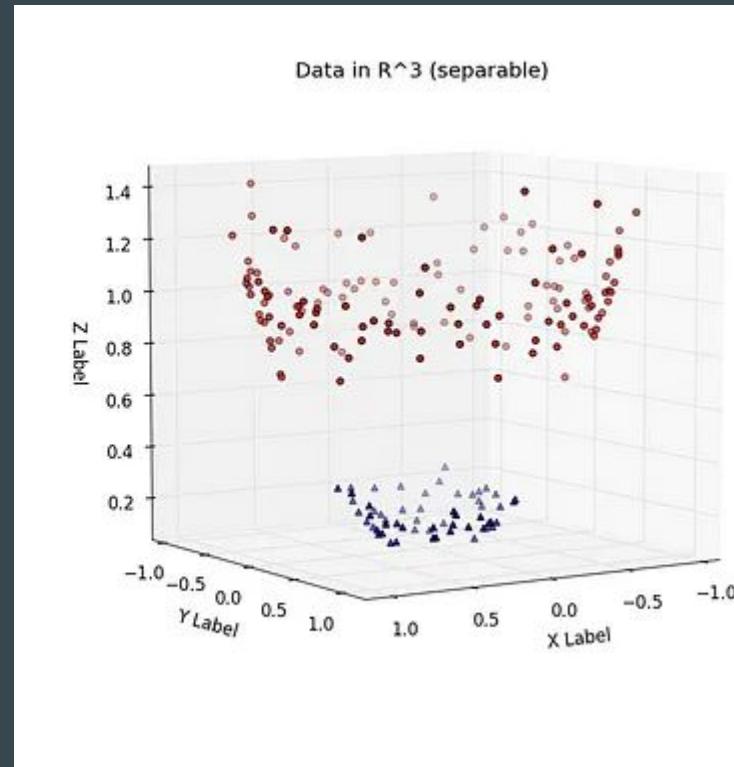


# Linear Classifier

(Transform to enable  
Linear separation)



# Transform to Higher Dimension



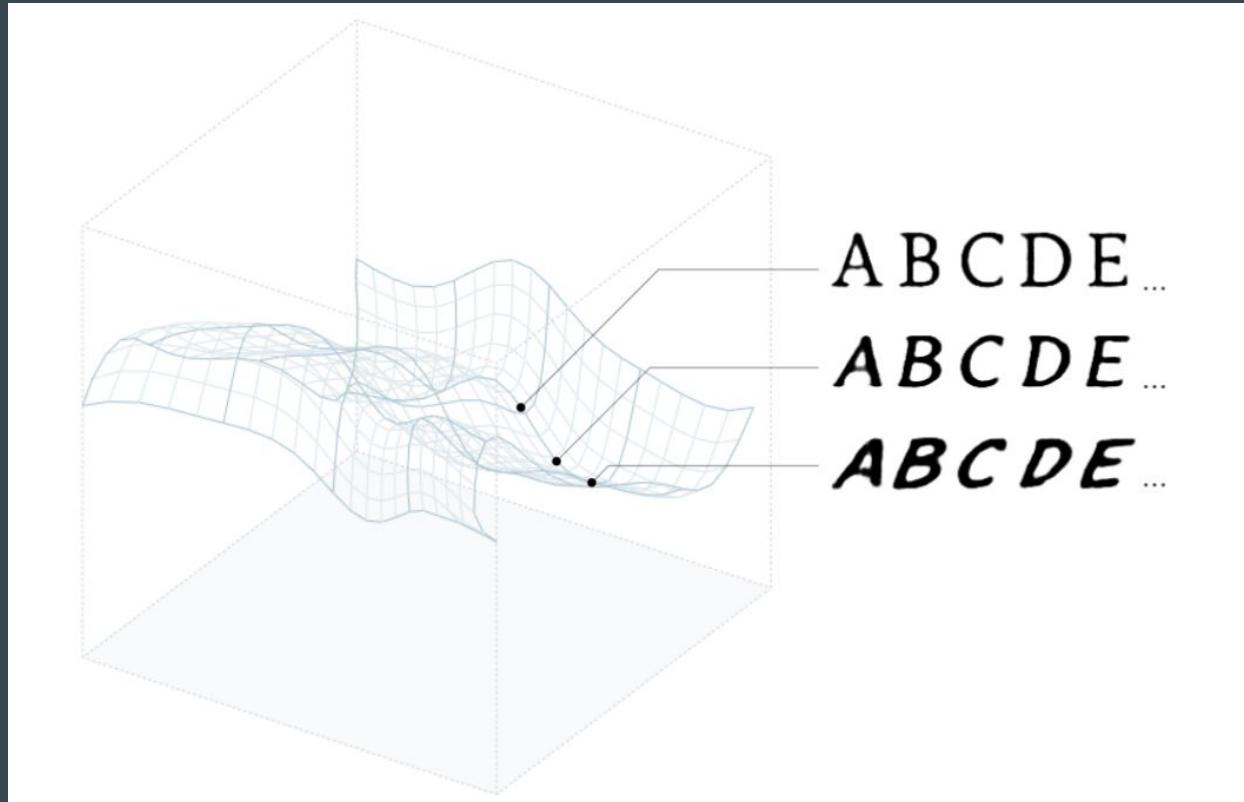
# Superposition

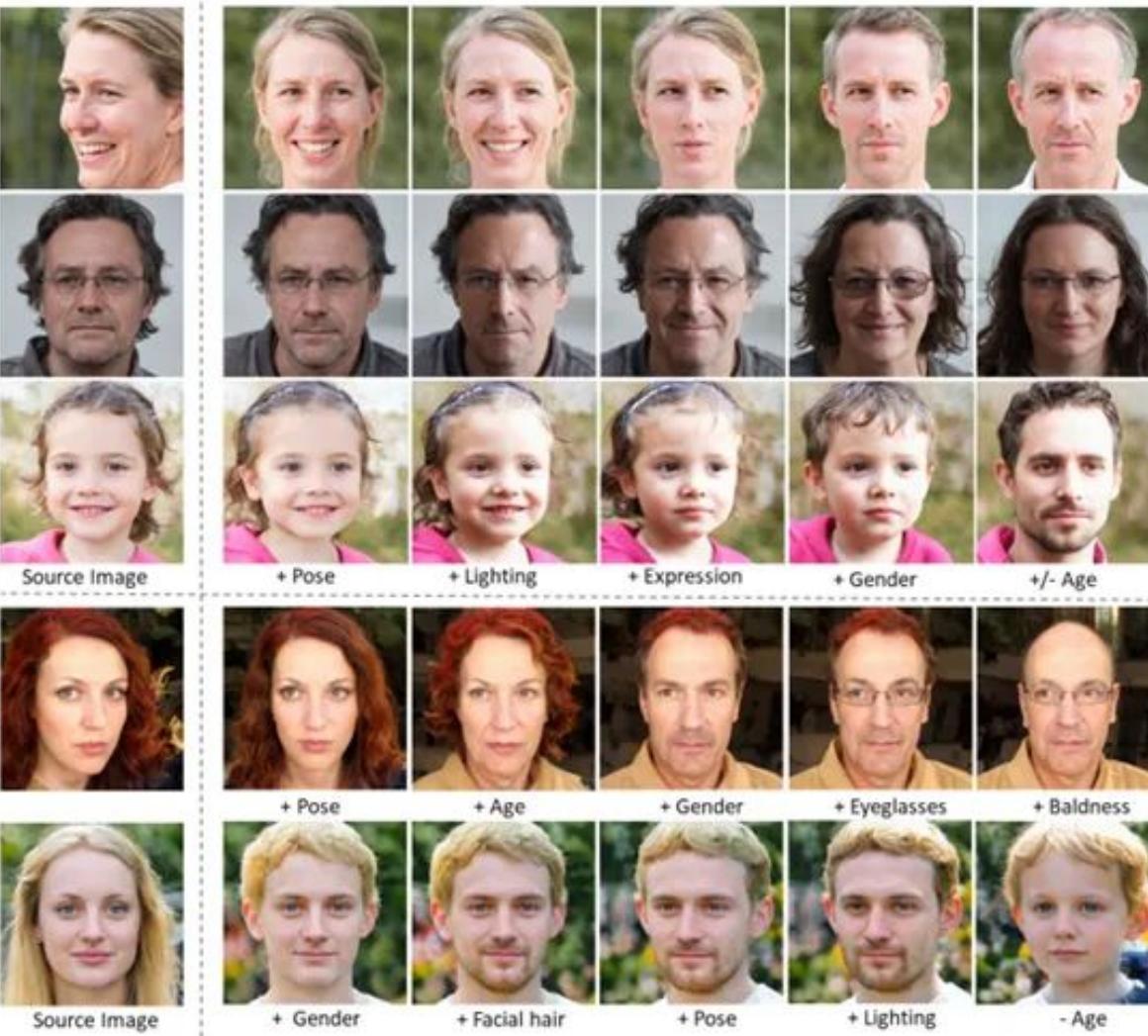


Find 3 Transformations to Meaningful Semantic Spaces

# Working in Semantic Space

# Semantic Spaces with Meaningful Dimensions





# Latent Space of Models Encoded with Language

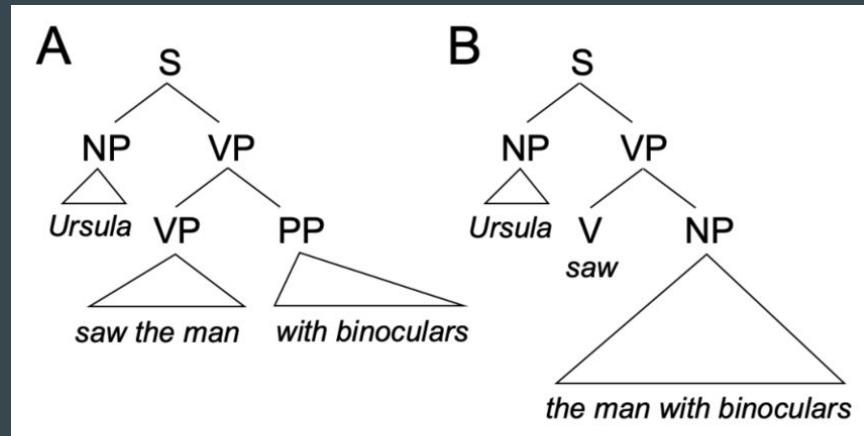
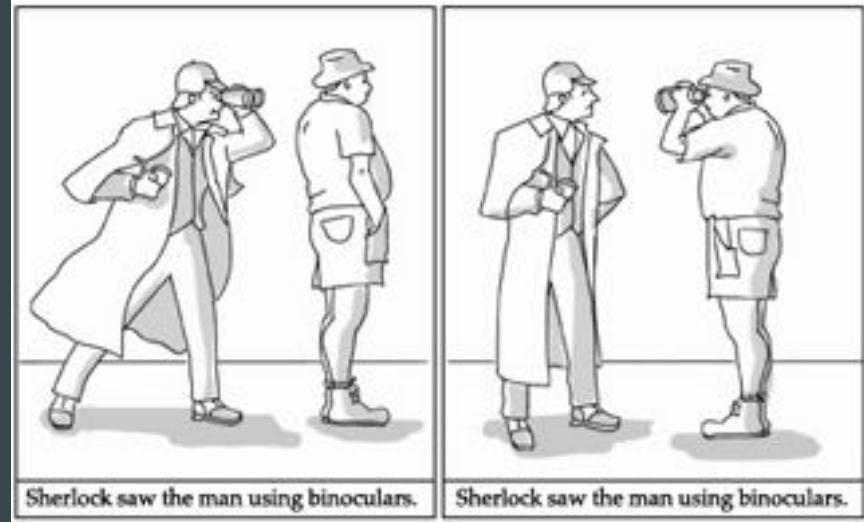
“Sherlock saw the man with binoculars.”

(1-5% Aphantasia)

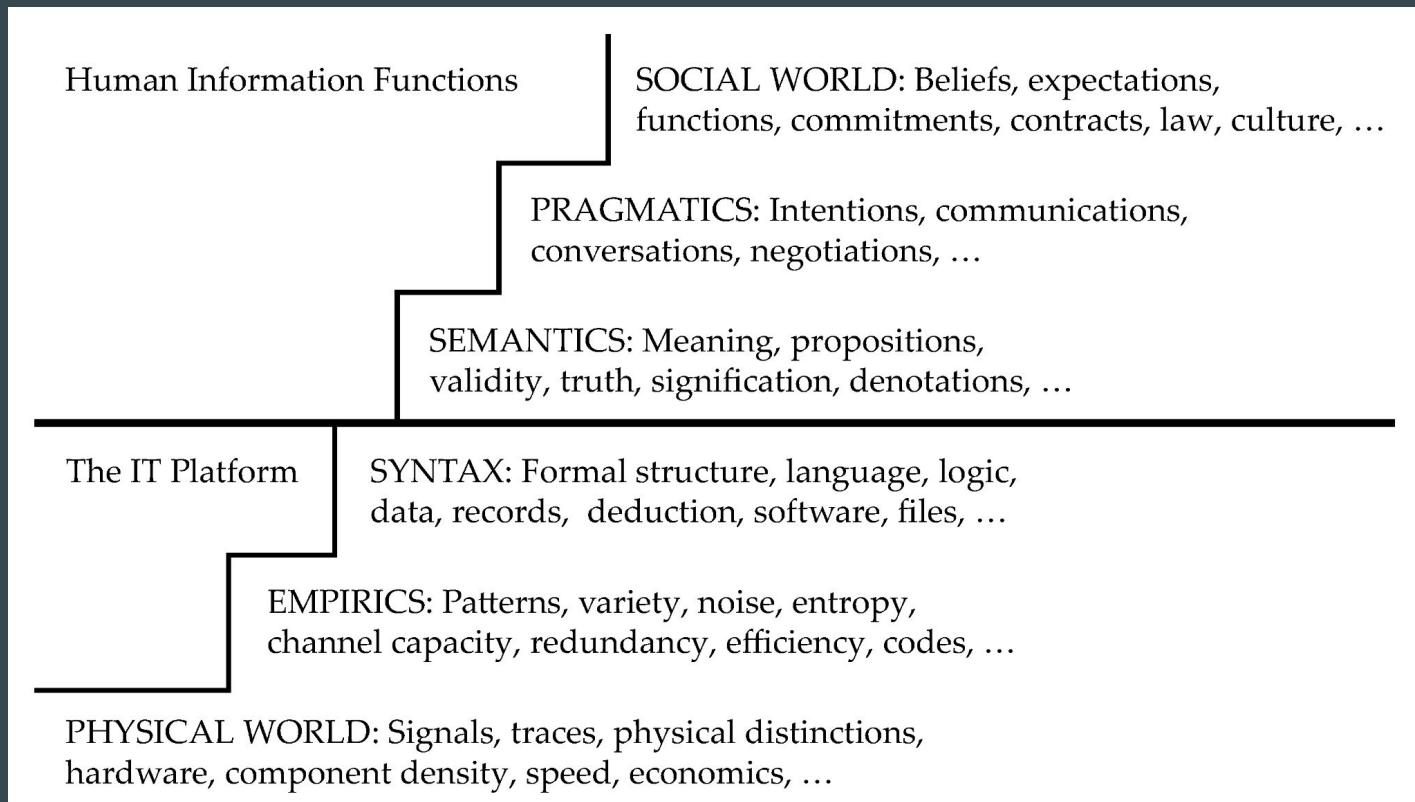
# Natural Language

Vs

## Constructed Languages (Examples?)

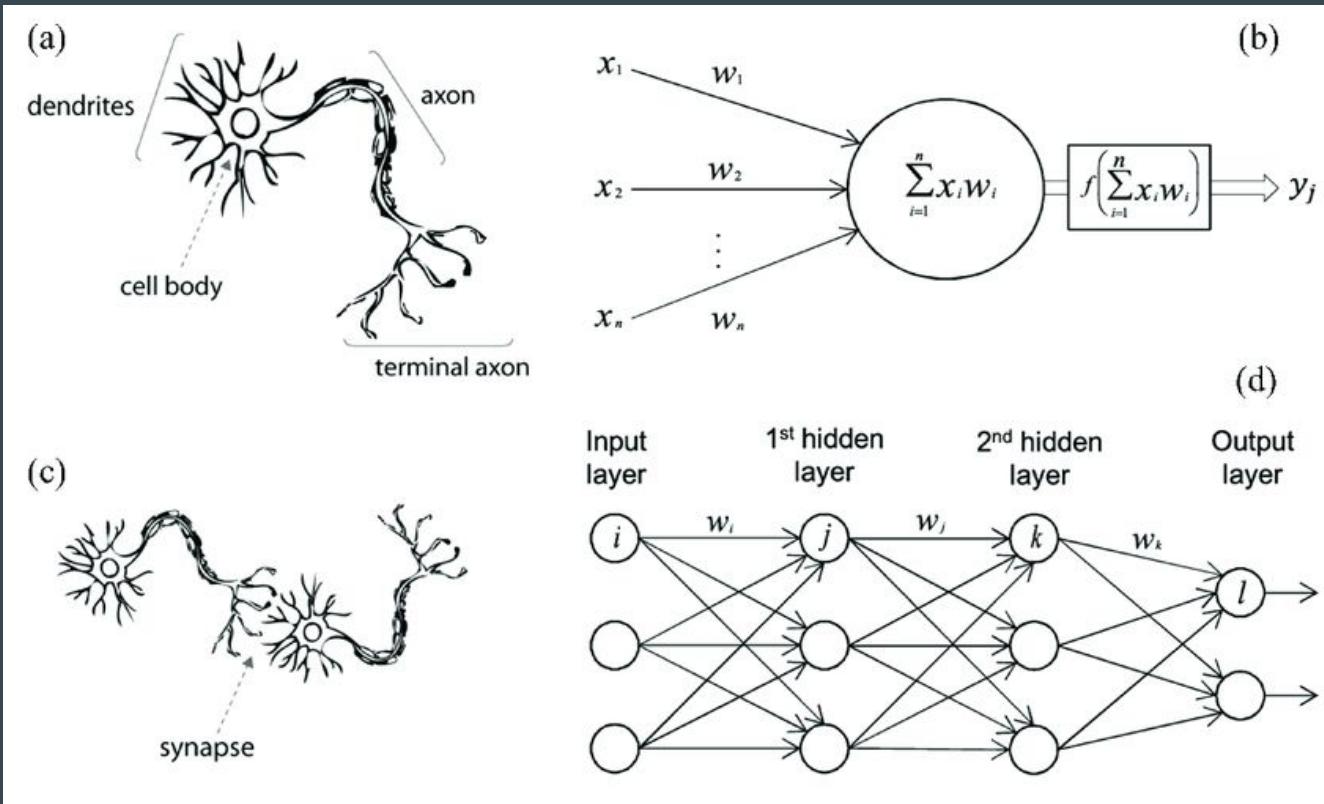


# Semiotics to Linguistics to NLP

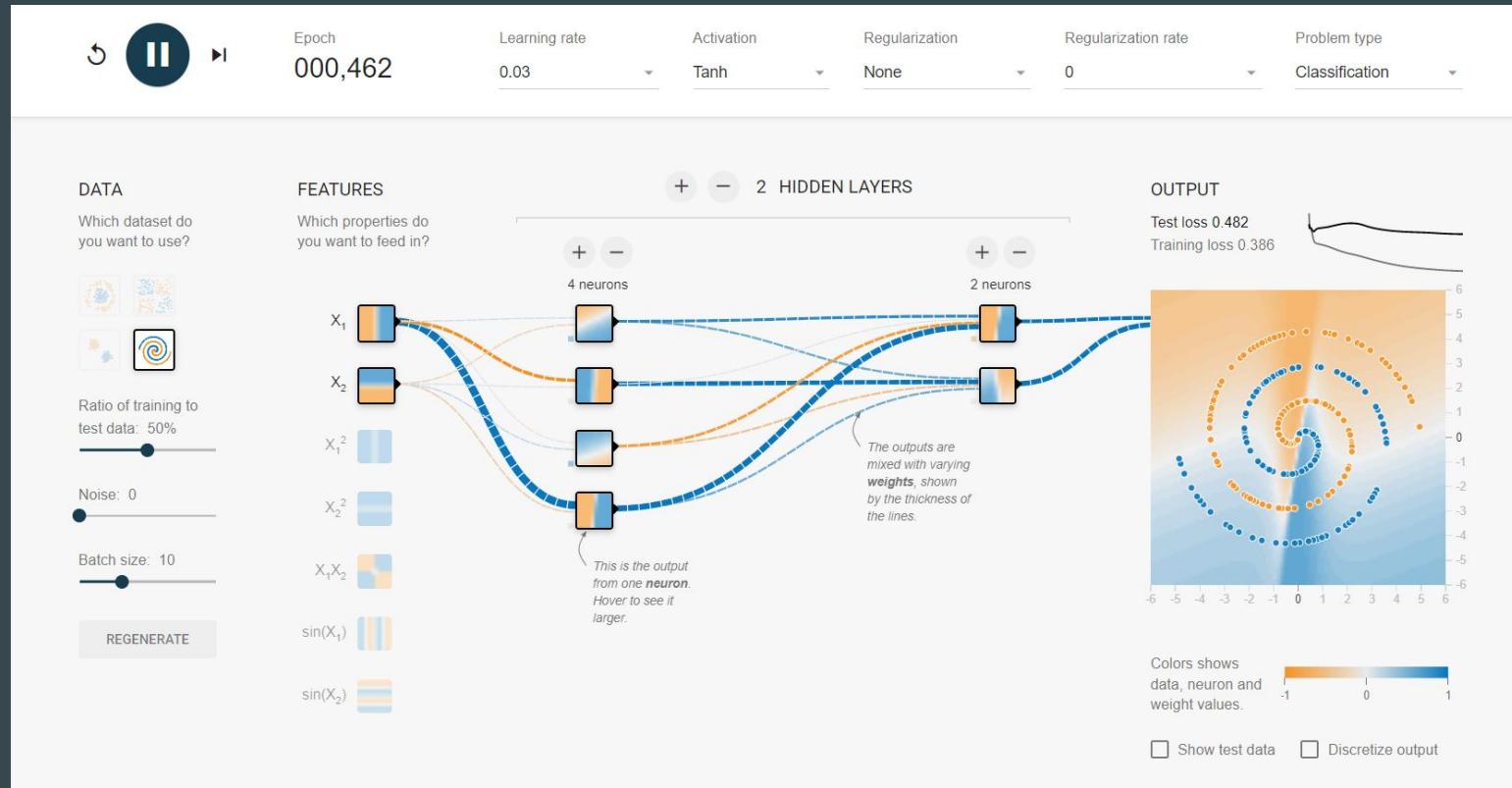


# Learning

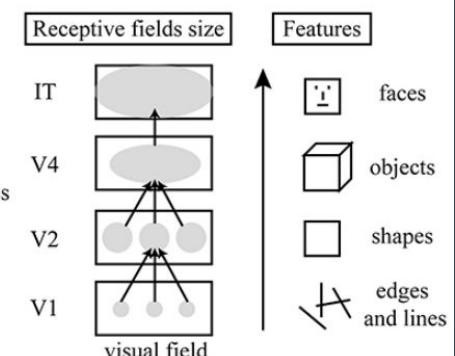
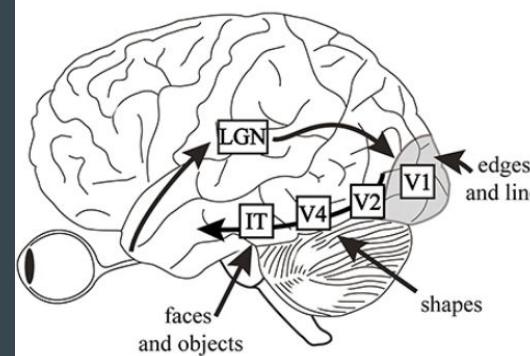
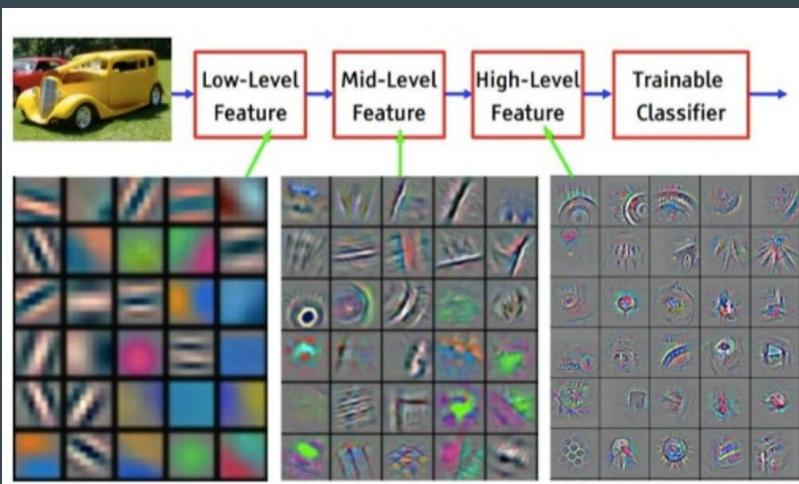
# Deep Neural Networks



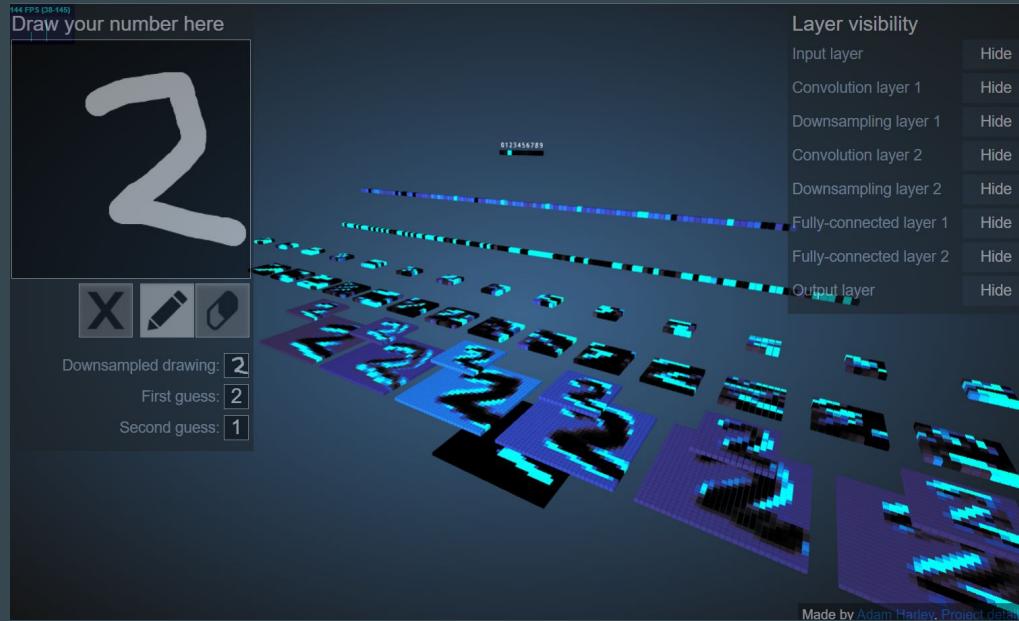
# Universal Function Approximator



# Efficient Architecture for Spatial Feature Detection (CNN)



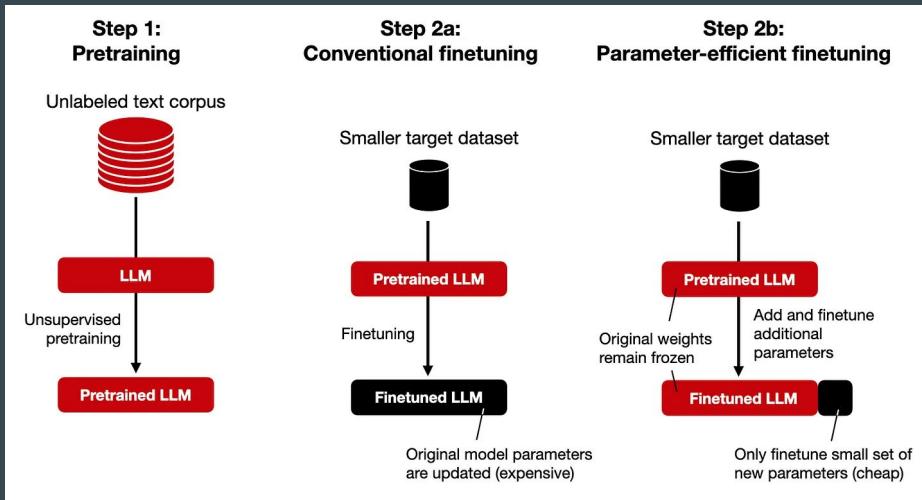
# CNN for Spatial Processing



```
model = keras.Sequential(  
[  
    keras.Input(shape=input_shape),  
    layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),  
    layers.MaxPooling2D(pool_size=(2, 2)),  
    layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),  
    layers.MaxPooling2D(pool_size=(2, 2)),  
    layers.Flatten(),  
    layers.Dropout(0.5),  
    layers.Dense(num_classes, activation="softmax"),  
]  
)  
  
model.summary()
```

# Injecting Information into LLMs

- Pre-Training: autoregressive next word prediction (Language)
- Fine-Tuning: instruct patterns (Skills/Behaviors)
- Prompting: n-shot In-Context Learning (ICL)
- Tools (RAG, DB/KB, etc)



Input: 2014-06-01  
Output: !06!01!2014!  
Input: 2007-12-13  
Output: !12!13!2007!  
Input: 2010-09-23  
Output: !09!23!2010!  
Input: 2005-07-23  
Output: !07!23!2005!

*in-context examples*

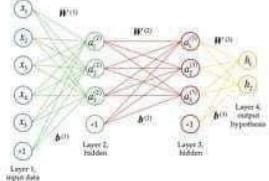
*test example*

*model completion*

# Reasoning

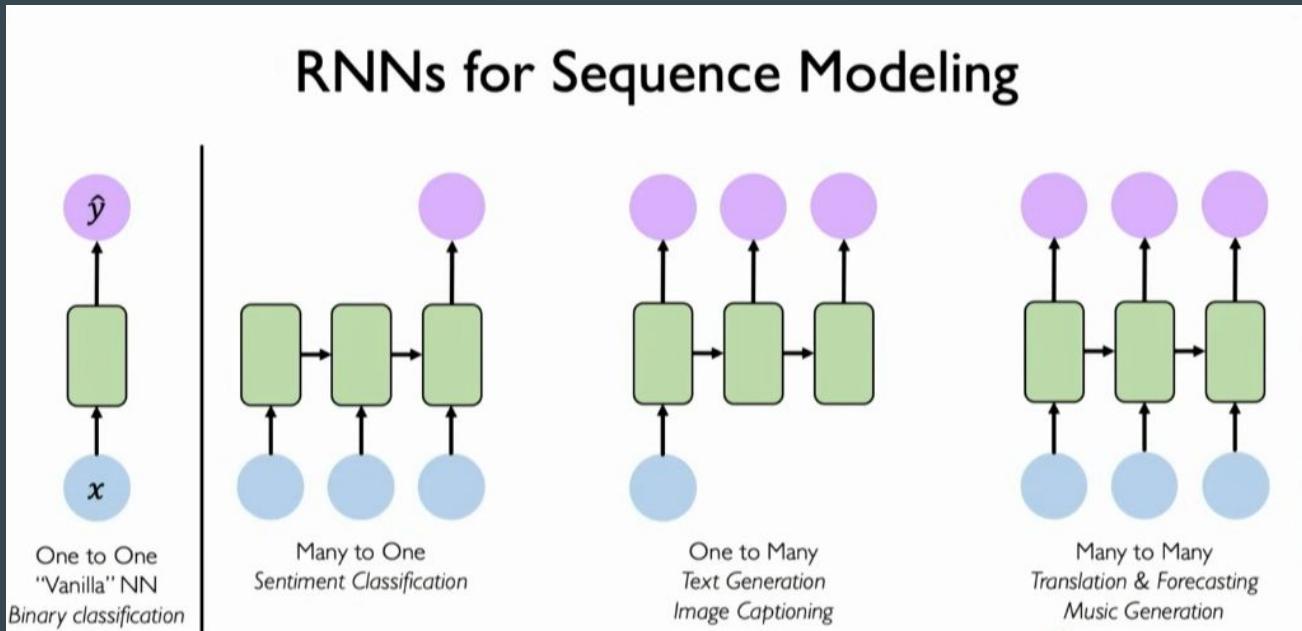
# LLM: Reasoning

- Prompt Strategies:
  - CoT
  - Reflection
  - ReAct
- Agents
  - Rational
  - Collaborative
  - Tool Use/Making
  - Embodied
  - Autonomous

	 <p>A parrot</p>	 <p>Machine learning algorithm</p>
Learns random phrases		
Doesn't understand █ about what it learns		
Occasionally speaks nonsense		
Is a cute birdie parrot		

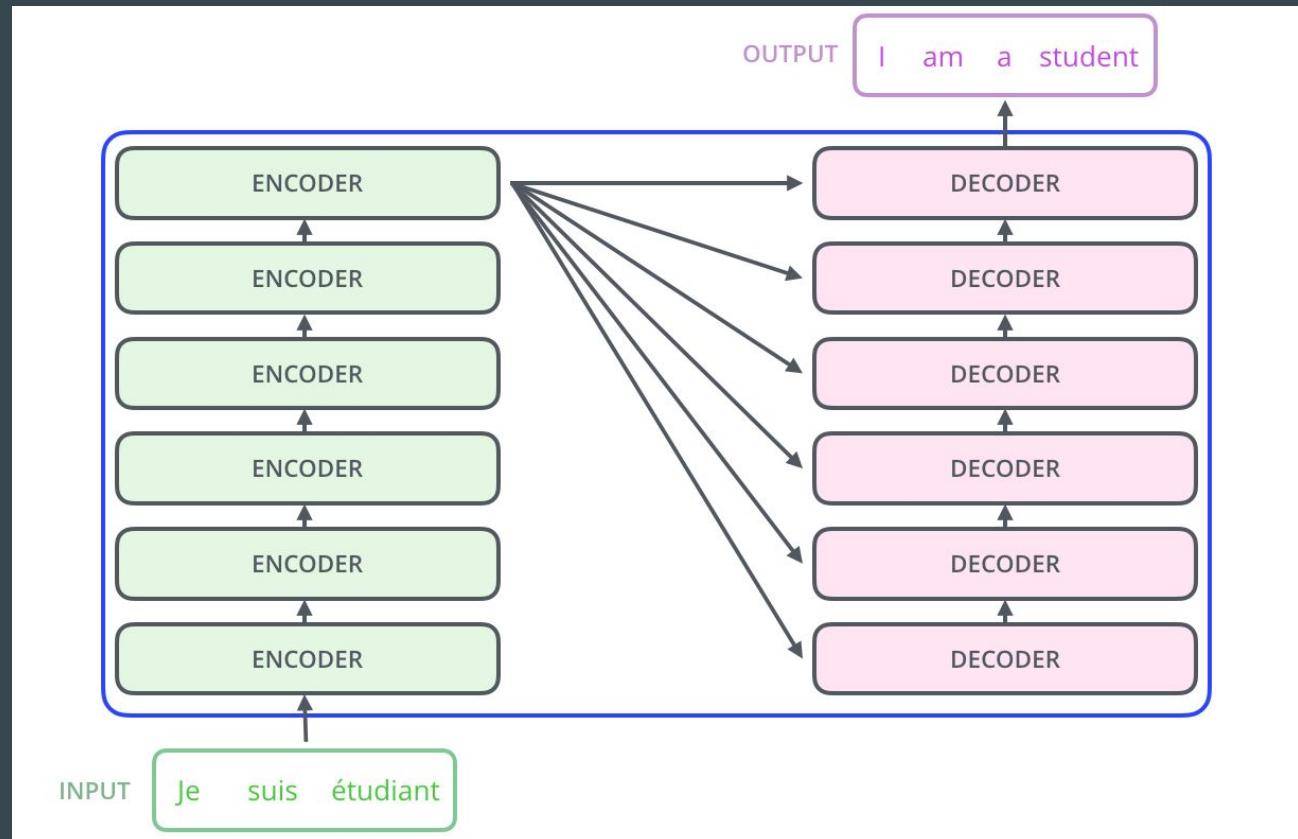
# LLM Architecture

# Efficient Architecture for Temporal Feature Detection (RNN)



# Transformer Architecture

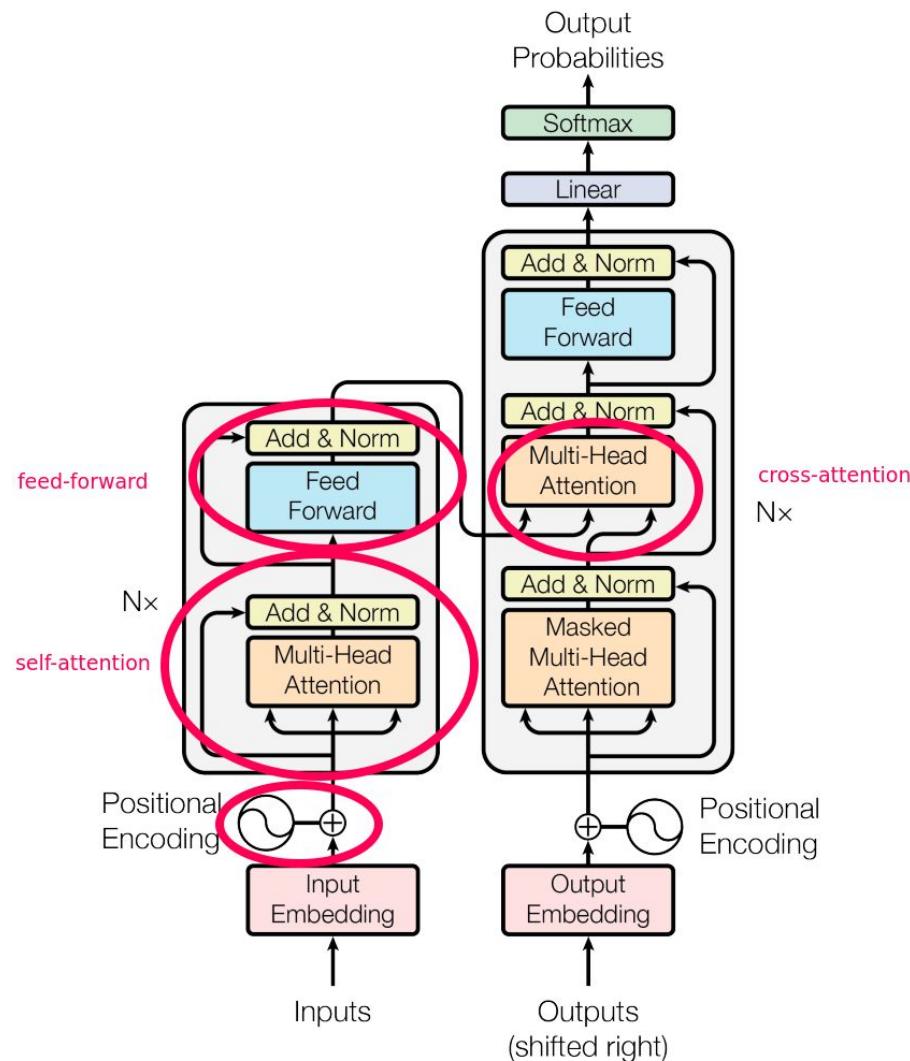
Spatial  
or  
Temporal  
in any  
Modality



# Transformers

## Hyperparameters:

- Vocabulary (~30k-50k)
- Context Window (2k to +100k)
- $h_x$  Heads
- $N_x$  Encoders
- $N_x$  Decoders



# 3 Variants

## Transformers

### ENCODER ONLY

aka  
**auto-encoding  
models**

#### TASKS

- Sentence classification
- Named entity recognition
- Extractive question-answering
- Masked language modeling

#### EXAMPLES

BERT, RoBERTa, distilBERT

### DECODER ONLY

aka  
**auto-regressive  
models**

#### TASKS

- Text generation
- Causal language modeling

#### EXAMPLES

GPT-2, GPT Neo, GPT-3

### ENCODER- DECODER

aka  
**sequence-to-  
sequence models**

#### TASKS

- Translation
- Summarization
- Generative question-answering

#### EXAMPLES

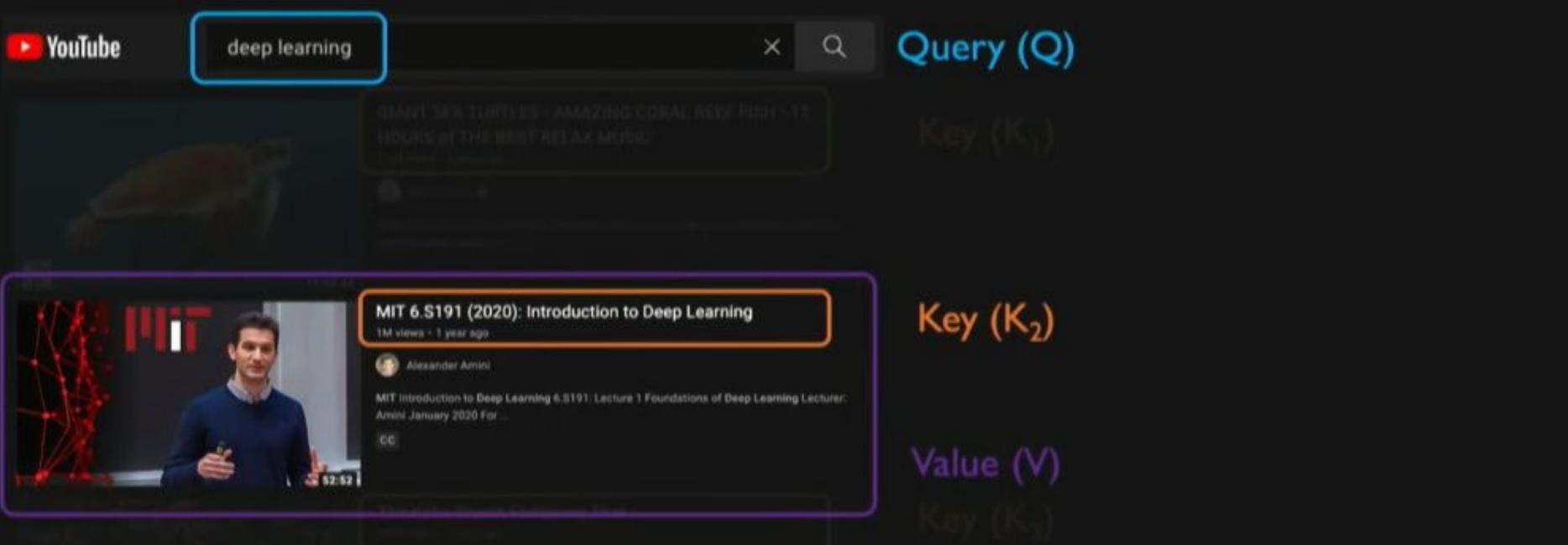
BART, T5, Marian

# Token Embeddings + Positional Encoding

## What is positional encoding?

Original sentence	YOUR	CAT	IS	A	LOVELY	CAT
<b>Embedding</b> (vector of size 512)	952.207 5450.840 1853.448 ... 1.658 2671.529	171.411 3276.350 9192.819 ... 3633.421 8390.473	621.659 1304.051 0.565 ... 7679.805 4506.025	776.562 5567.288 58.942 ... 2716.194 5119.949	6422.693 6315.080 9358.778 ... 2141.081 735.147	171.411 3276.350 9192.819 ... 3633.421 8390.473
<b>Position Embedding</b> (vector of size 512). Only computed once and reused for every sentence during training and inference.	... ... ... ... ... ...	1664.068 8080.133 2620.399 ... 9386.405 3120.159	... ... ... ... ... ...	... ... ... ... ... ...	... ... ... ... ... ...	1281.458 7902.890 912.970 3821.102 1659.217 7018.620
	=	=	=	=	=	=
<b>Encoder Input</b> (vector of size 512)	...	1835.479 11356.483 11813.218 ... 13019.826 11510.632	...	...	...	1452.869 11179.24 10105.789 ... 5292.638 15409.093

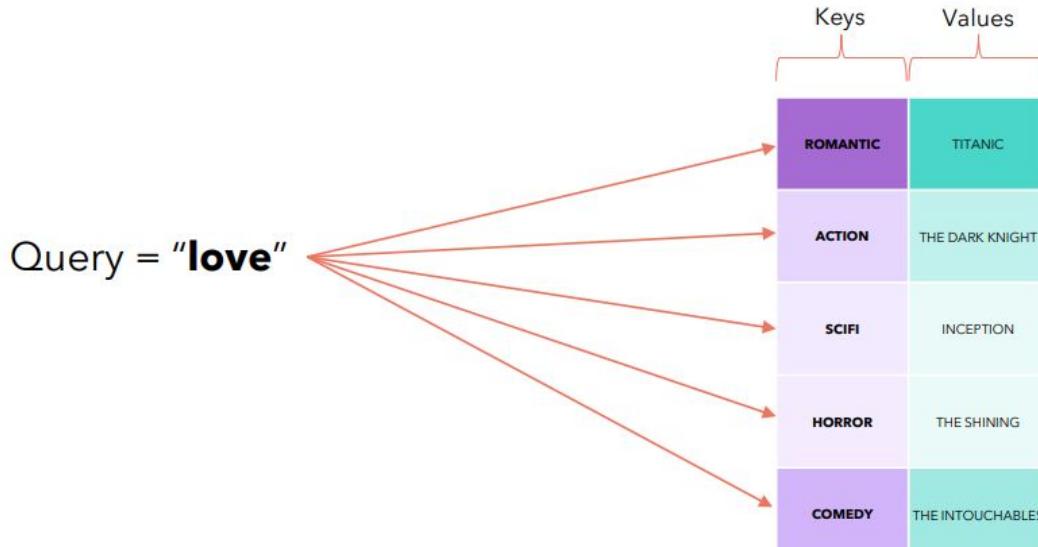
# Understanding Attention with Search



2. Extract values based on attention:  
Return the values highest attention

# Why query, keys and values?

The Internet says that these terms come from the database terminology or the Python-like dictionaries.



\* this could be a Python dictionary  
or a database table.

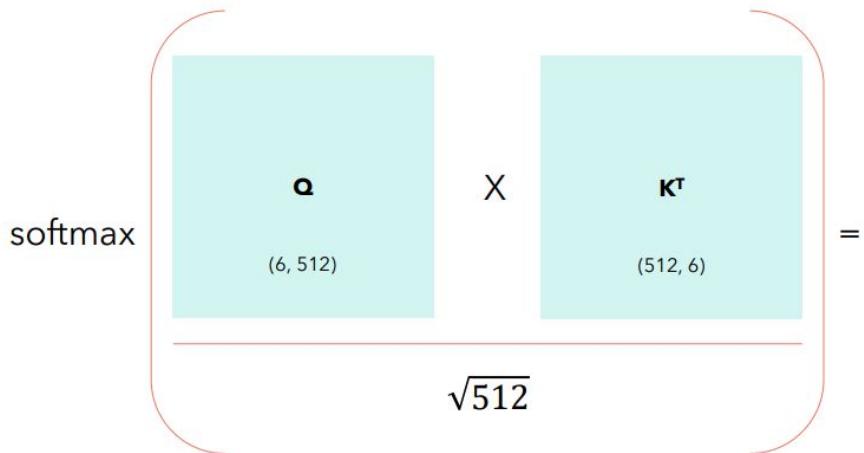
# Self-Attention

Self-Attention allows the model to relate words to each other.

In this simple case we consider the sequence length **seq** = 6 and  $d_{\text{model}} = d_k = 512$ .

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

The matrices **Q**, **K** and **V** are just the input sentence.



	YOUR	CAT	IS	A	LOVELY	CAT	$\Sigma$
YOUR	0.268	0.119	0.134	0.148	0.179	0.152	<b>1</b>
CAT	0.124	0.278	0.201	0.128	0.154	0.115	<b>1</b>
IS	0.147	0.132	0.262	0.097	0.218	0.145	<b>1</b>
A	0.210	0.128	0.206	0.212	0.119	0.125	<b>1</b>
LOVELY	0.146	0.158	0.152	0.143	0.227	0.174	<b>1</b>
CAT	0.195	0.114	0.203	0.103	0.157	0.229	<b>1</b>

\* all values are random.

\* for simplicity I considered only one head, which makes  $d_{\text{model}} = d_k$ .

(6, 6)

# Self-Attention

	YOUR	CAT	IS	A	LOVELY	CAT
YOUR	0.268	0.119	0.134	0.148	0.179	0.152
CAT	0.124	0.278	0.201	0.128	0.154	0.115
IS	0.147	0.132	0.262	0.097	0.218	0.145
A	0.210	0.128	0.206	0.212	0.119	0.125
LOVELY	0.146	0.158	0.152	0.143	0.227	0.174
CAT	0.195	0.114	0.203	0.103	0.157	0.229

(6, 6)

X

V

=

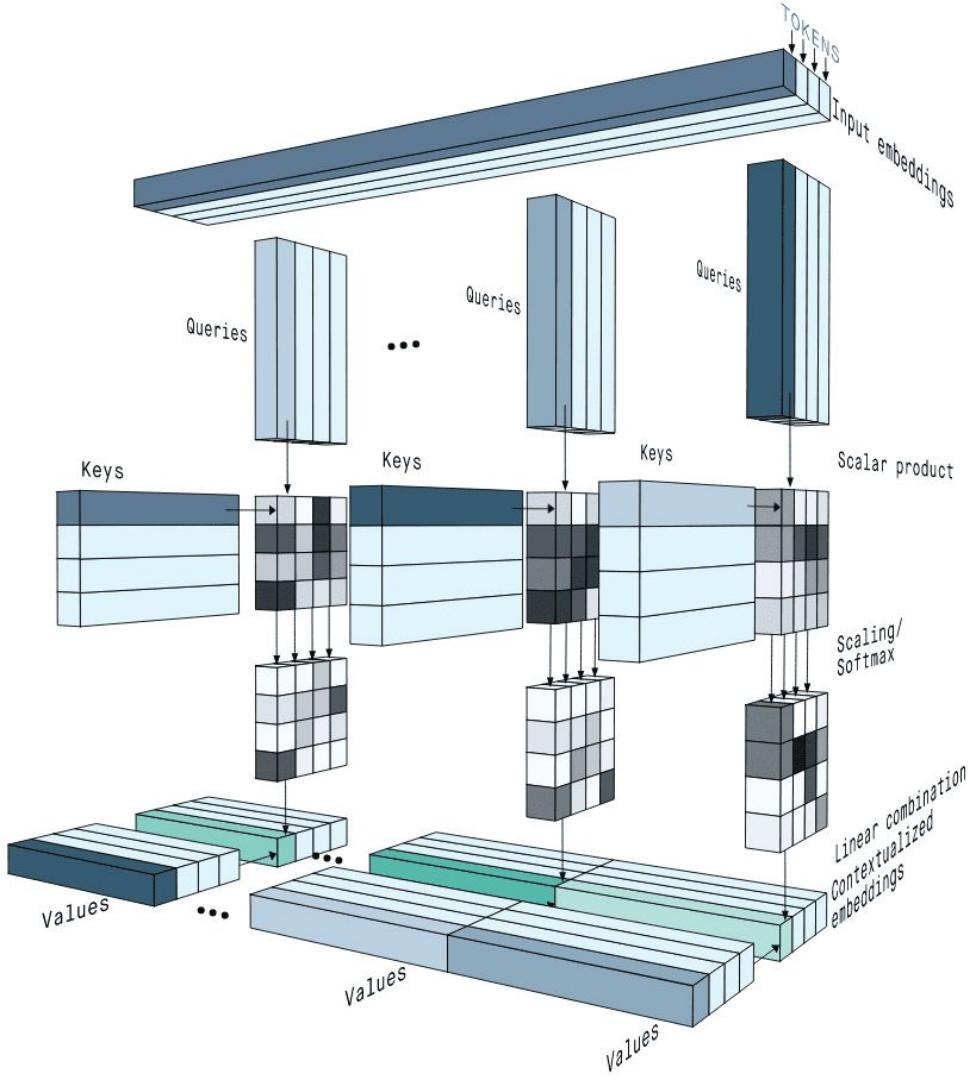
Attention

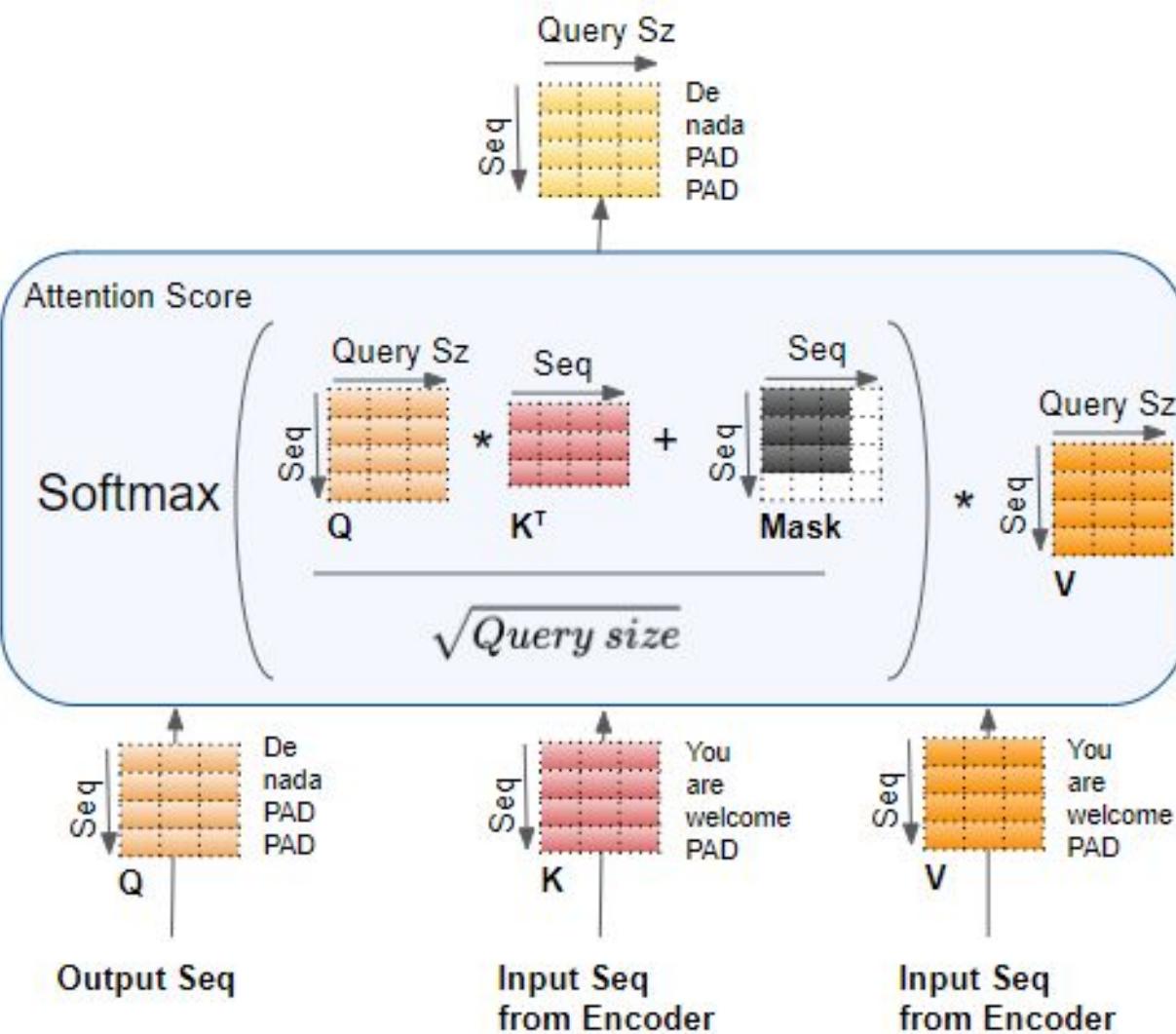
(6, 512)

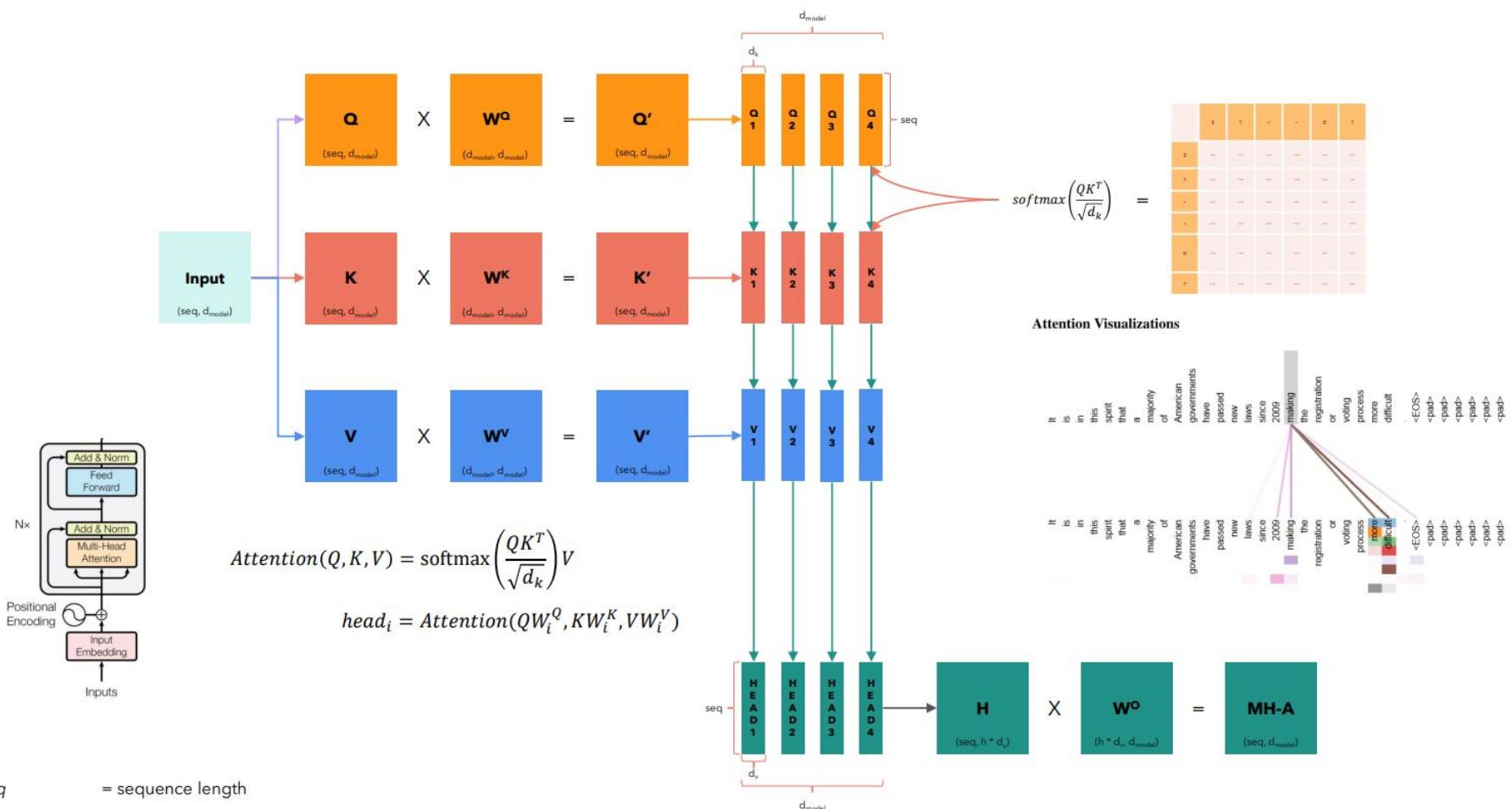
(6, 512)

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Each row in this matrix captures not only the meaning (given by the embedding) or the position in the sentence (represented by the positional encodings) but also each word's interaction with other words.







$\text{seq}$  = sequence length

$d_{\text{model}}$  = size of the embedding vector

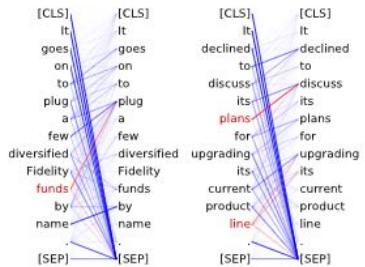
$h$  = number of heads

$d_k = d_v = d_{\text{model}} / h$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1 \dots \text{head}_h)W^O$$

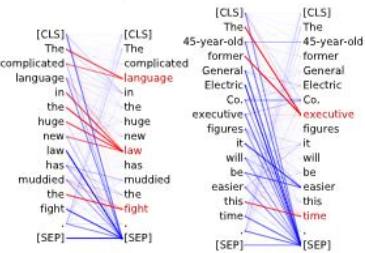
#### Head 8-10

- Direct objects attend to their verbs
- 86.8% accuracy at the dobj relation



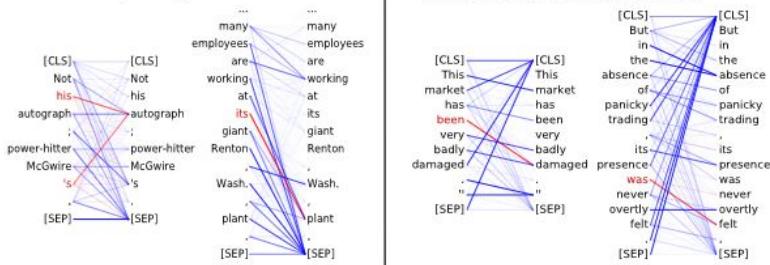
#### Head 8-11

- Noun modifiers (e.g., determiners) attend to their noun
- 94.3% accuracy at the det relation



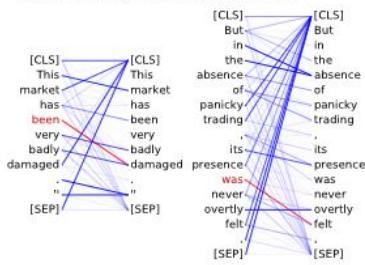
#### Head 7-6

- Possessive pronouns and apostrophes attend to the head of the corresponding NP
- 80.5% accuracy at the poss relation



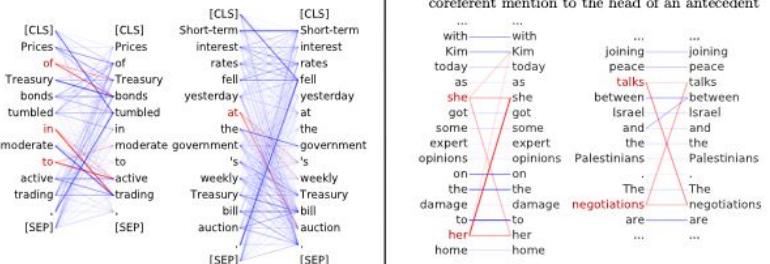
#### Head 4-10

- Passive auxiliary verbs attend to the verb they modify
- 82.5% accuracy at the auxpass relation



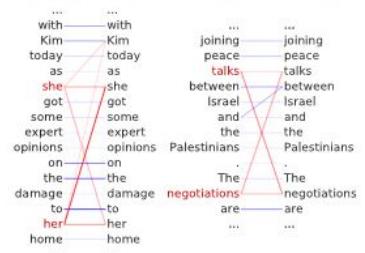
#### Head 9-6

- Prepositions attend to their objects
- 76.3% accuracy at the pobj relation



#### Head 5-4

- Coreferent mentions attend to their antecedents
- 65.1% accuracy at linking the head of a coreferent mention to the head of an antecedent

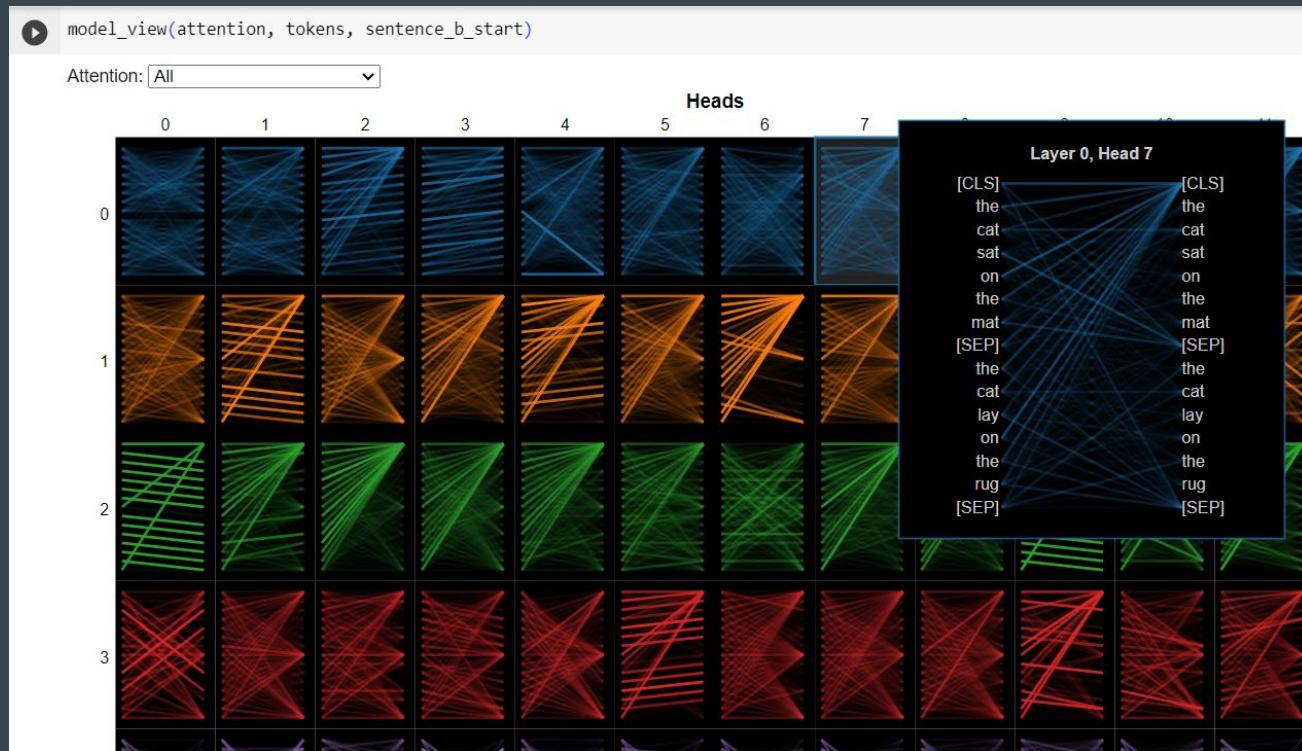


Attention weights are sparse  
(or close to uniform)

Source: "What Does BERT Look At? An Analysis of BERT's Attention"

Clark, Khandelwal, Levy, Manning

# BERTViz Attention Heads



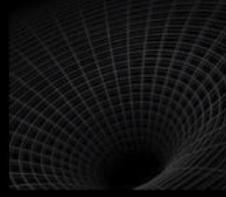
# Superposition of Semantic Spaces

Multiple Heads

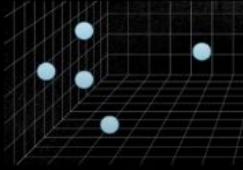
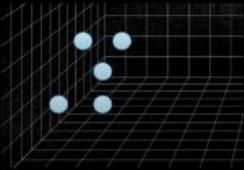
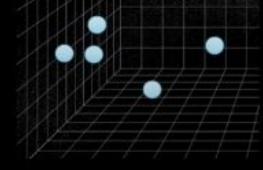
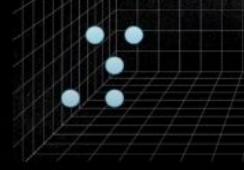
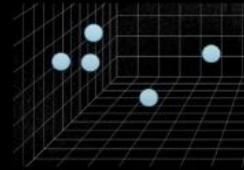
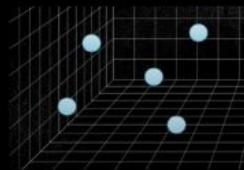
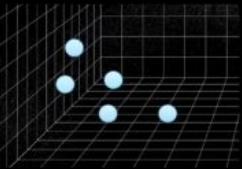
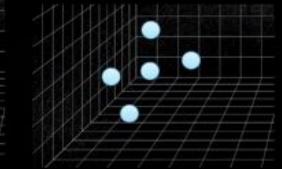
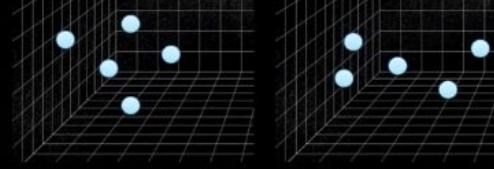
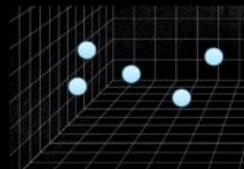
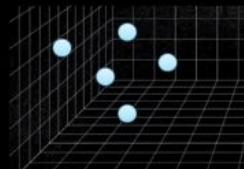
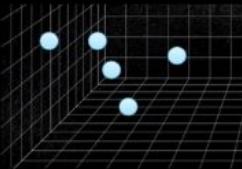
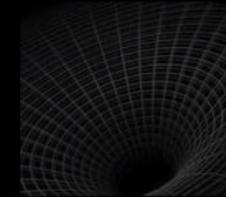
Head 1



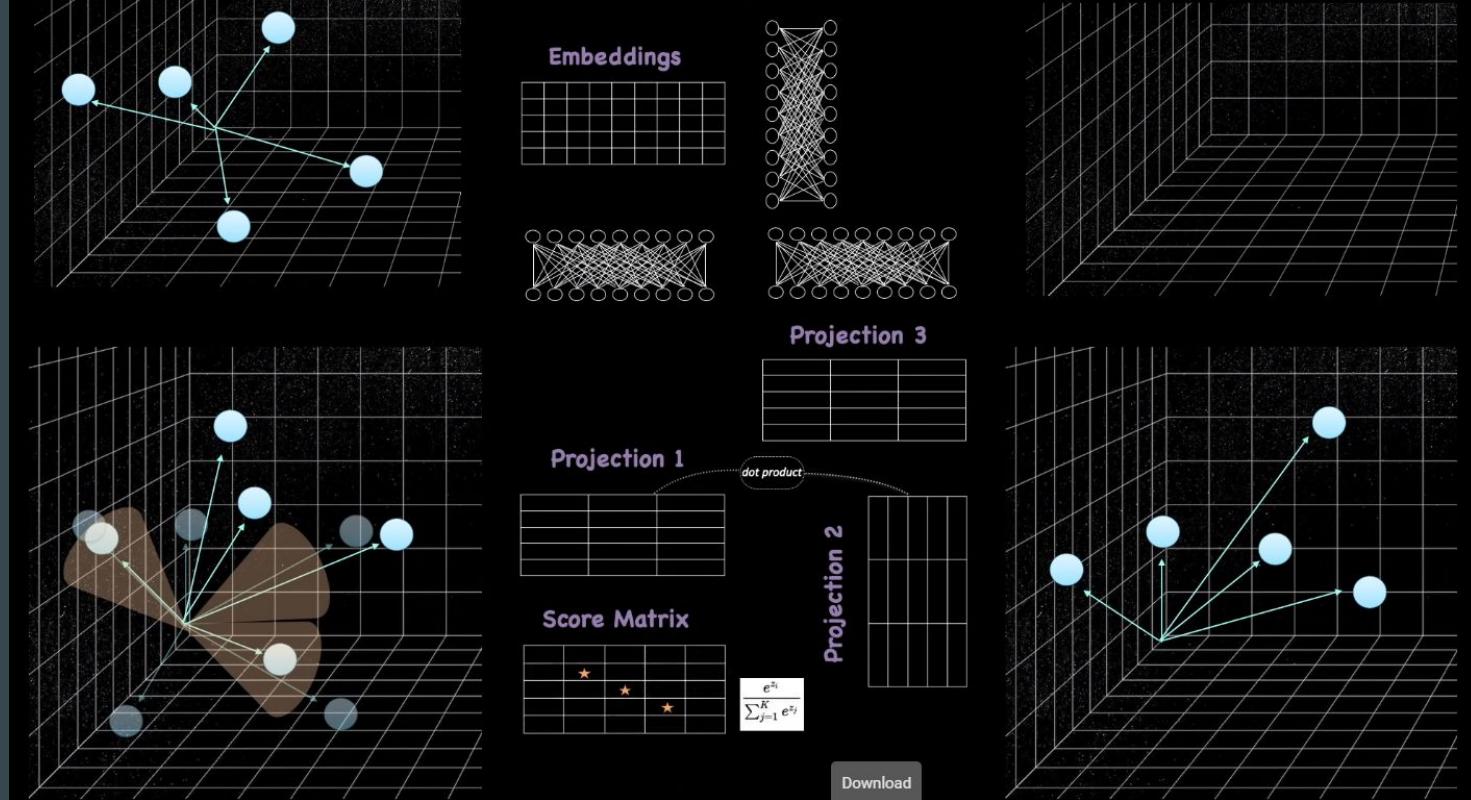
Head 2



Head 3

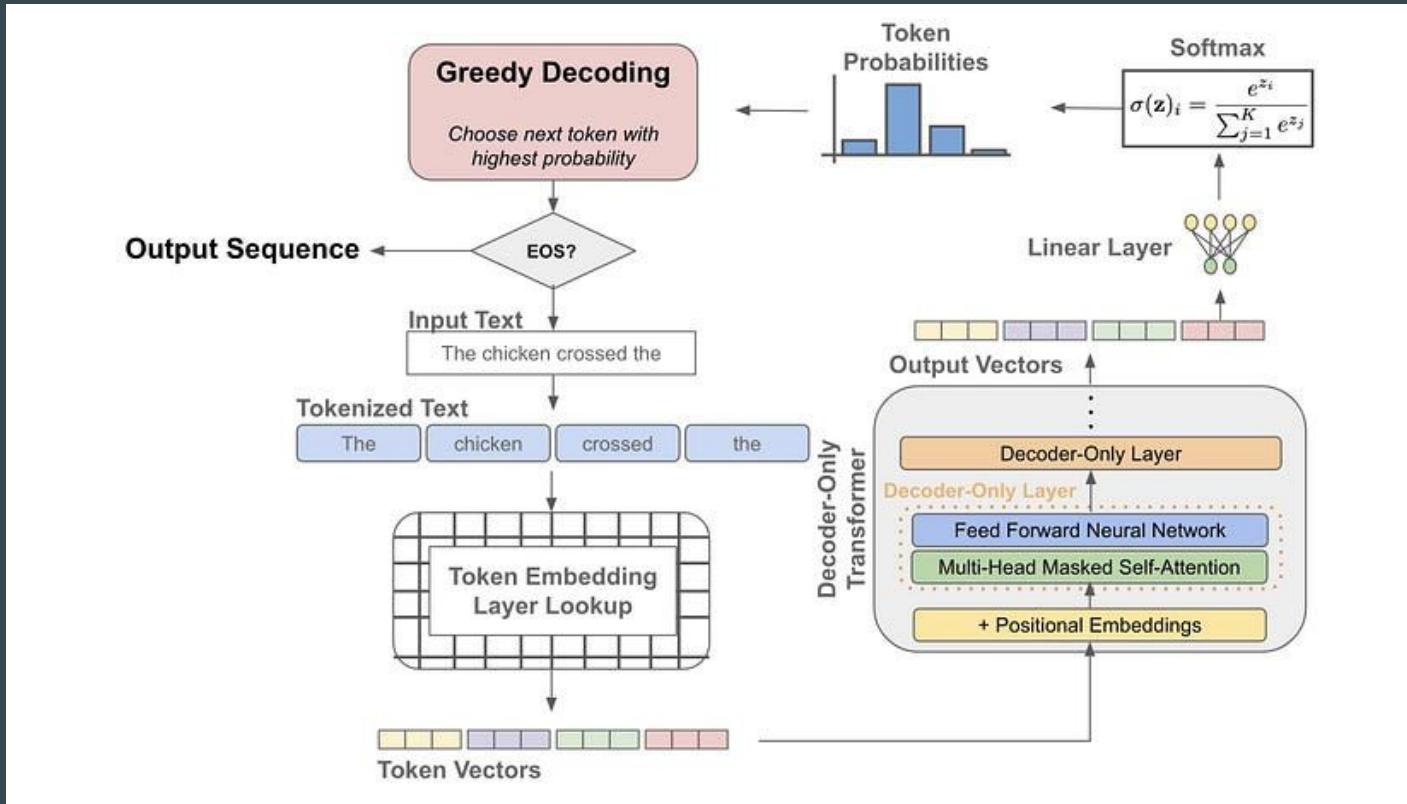


# Visualize $(Q \times K) \times V$ Matrix Cosine Similarities



# LLM Inference

# Next Token Probabilities



# Playground

Your presets

Save

View code

Share

...

It was a dark and stormy night. Suddenly, a shot rang out. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it. The sound echoed across the moors, ringing in the ears of those who were awake and alert enough to hear it.

ir through = 98.58%

throughout = 0.86%

T across = 0.24%

W off = 0.19%

N in = 0.05%

" Total: -6.04 logprob on 1 tokens  
(99.92% probability covered in top 5 logits)

shotgun in hand. He was a burly man, face. He was known to be a no-nonsense sort, willing to use his shotgun if necessary.

area for any sign of danger.

The travelers, who had been huddled around the fire in the common room, jumped and looked at each other in fear. They had been on their way to the nearby city when the storm had forced them to take shelter at the inn. They were all strangers to each other, but in that moment, they were all connected by the sense of unease that hung in the air.

One of the travelers, a young woman with long, dark hair, stepped forward. "I

## Model

gpt-3.5-turbo-ins...

Temperature

1

Maximum length

256

Stop sequences

Enter sequence and press Tab

Top P

1

Frequency penalty

0

Presence penalty

0

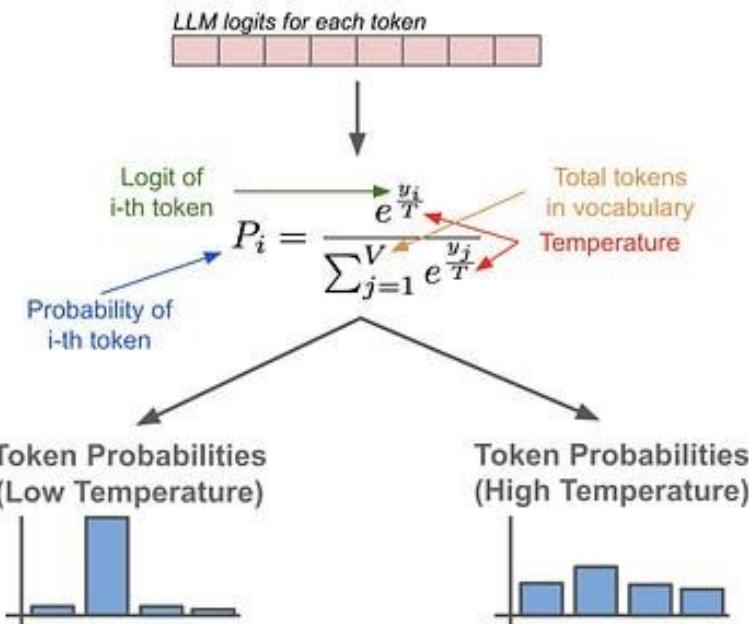
# Inference Hyperparameters

## Temperature in LLM APIs

`temperature` number Optional Defaults to 1

What sampling temperature to use, between 0 and 2. Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic.

### Softmax with Temperature

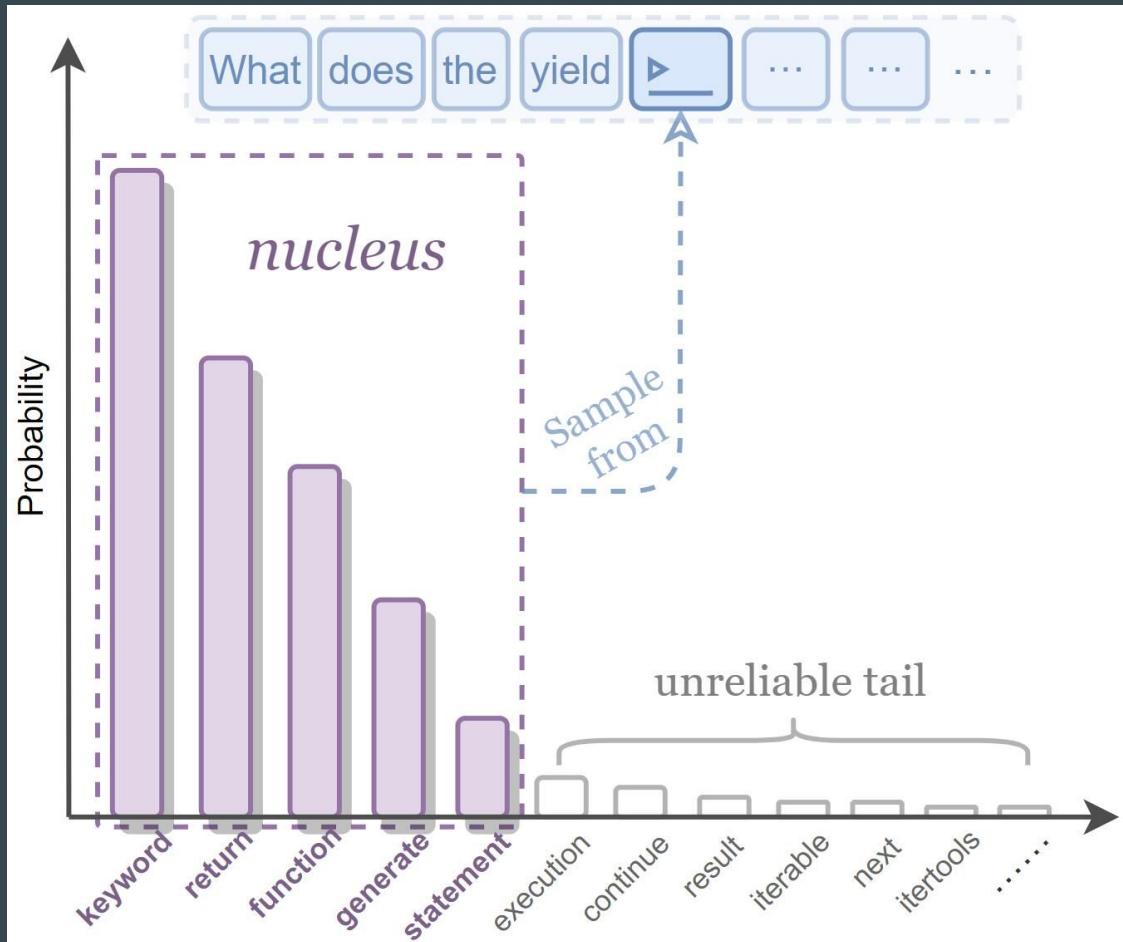


Top-P

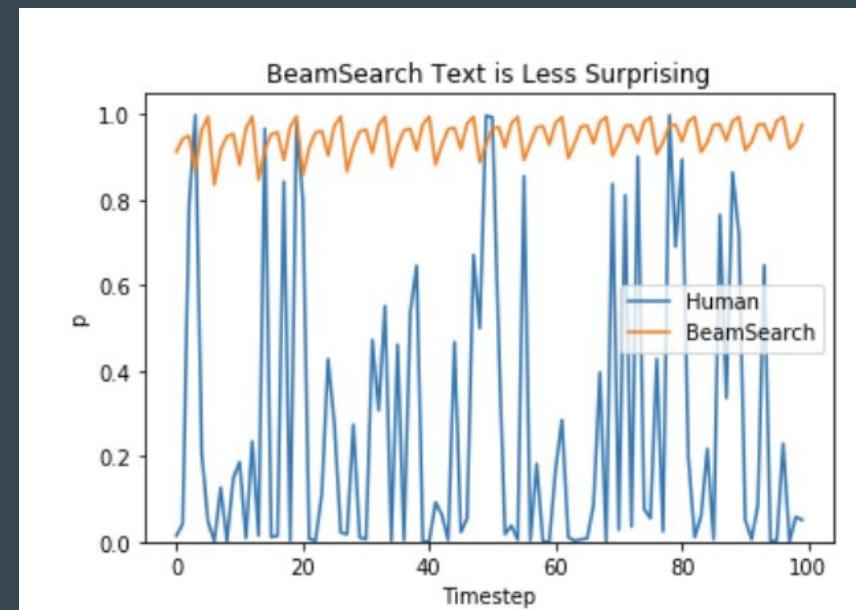
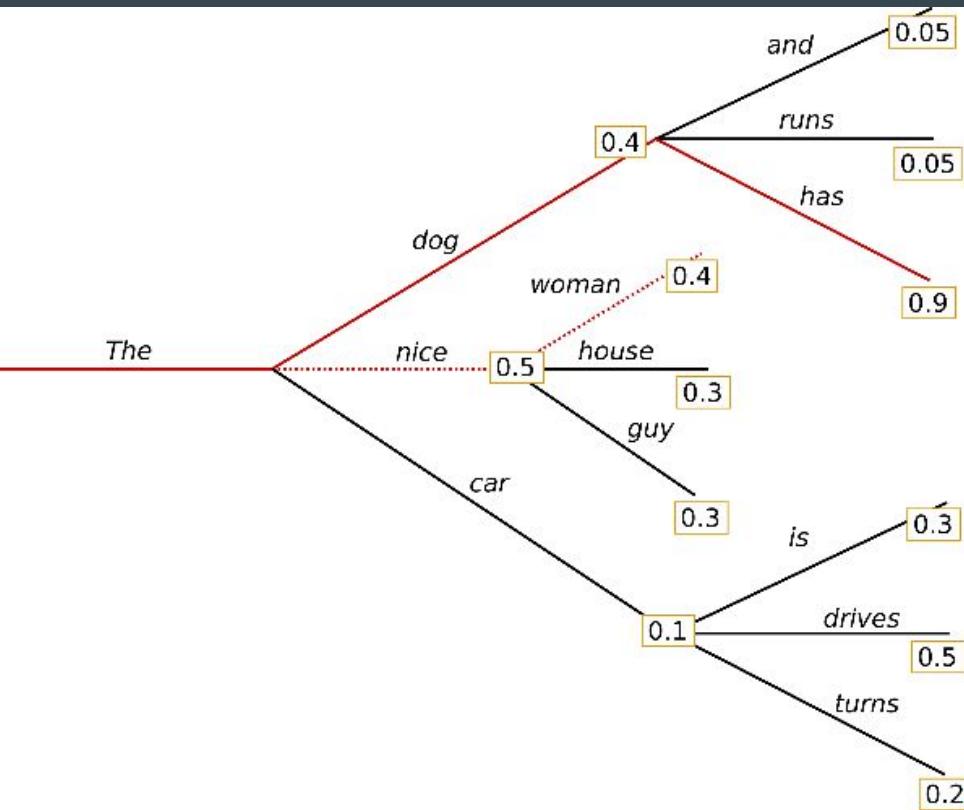
+

Hi Temp

~ MCMC Sampling



# Beam Search



## Parameter details

### Frequency and presence penalties

The frequency and presence penalties found in the [Chat completions API](#) and [Legacy Completions API](#) can be used to reduce the likelihood of sampling repetitive sequences of tokens. They work by directly modifying the logits (un-normalized log-probabilities) with an additive contribution.

```
mu[j] -> mu[j] - c[j] * alpha_frequency - float(c[j] > 0) * alpha_presence
```



Where:

- `mu[j]` is the logits of the j-th token
- `c[j]` is how often that token was sampled prior to the current position
- `float(c[j] > 0)` is 1 if `c[j] > 0` and 0 otherwise
- `alpha_frequency` is the frequency penalty coefficient
- `alpha_presence` is the presence penalty coefficient

As we can see, the presence penalty is a one-off additive contribution that applies to all tokens that have been sampled at least once and the frequency penalty is a contribution that is proportional to how often a particular token has already been sampled.

Reasonable values for the penalty coefficients are around 0.1 to 1 if the aim is to just reduce repetitive samples somewhat. If the aim is to strongly suppress repetition, then one can increase the coefficients up to 2, but this can noticeably degrade the quality of samples. Negative values can be used to increase the likelihood of repetition.

**frequency\_penalty** number or null Optional Defaults to 0

Number between -2.0 and 2.0. Positive values penalize new tokens based on their existing frequency in the text so far, decreasing the model's likelihood to repeat the same line verbatim.

[See more information about frequency and presence penalties.](#)

**logit\_bias** map Optional Defaults to null

Modify the likelihood of specified tokens appearing in the completion.

Accepts a json object that maps tokens (specified by their token ID in the GPT tokenizer) to an associated bias value from -100 to 100. You can use this [tokenizer tool](#) (which works for both GPT-2 and GPT-3) to convert text to token IDs. Mathematically, the bias is added to the logits generated by the model prior to sampling. The exact effect will vary per model, but values between -1 and 1 should decrease or increase likelihood of selection; values like -100 or 100 should result in a ban or exclusive selection of the relevant token.

As an example, you can pass `{"50256": -100}` to prevent the `<|endoftext|>` token from being generated.

**logprobs** integer or null Optional Defaults to null

Include the log probabilities on the `logprobs` most likely tokens, as well the chosen tokens. For example, if `logprobs` is 5, the API will return a list of the 5 most likely tokens. The API will always return the `logprob` of the sampled token, so there may be up to `logprobs+1` elements in the response.

The maximum value for `logprobs` is 5.

**max\_tokens** integer or null Optional Defaults to 16

The maximum number of `tokens` to generate in the completion.

The token count of your prompt plus `max_tokens` cannot exceed the model's context length. [Example Python code](#) for counting tokens.

NO STREAMING

STREAMING

Example req... gpt-3.5-turbo-instruct ▾ curl ▾ ⌂ Copy

```
1 curl https://api.openai.com/v1/completions \
2   -H "Content-Type: application/json" \
3   -H "Authorization: Bearer $OPENAI_API_KEY" \
4   -d '{
5     "model": "gpt-3.5-turbo-instruct",
6     "prompt": "Say this is a test",
7     "max_tokens": 7,
8     "temperature": 0
9   }'
```

Response gpt-3.5-turbo-instruct ▾ ⌂ Copy

```
1 {
2   "id": "cmpl-uqkv1QyYK7bGYrRHQoeXlWi7",
3   "object": "text_completion",
4   "created": 1589478378,
5   "model": "gpt-3.5-turbo-instruct",
6   "choices": [
7     {
8       "text": "\n\nThis is indeed a test",
9       "index": 0,
10      "logprobs": null,
11      "finish_reason": "length"
12    }
13  ],
14  "usage": {
15    "prompt_tokens": 5,
16    "completion_tokens": 7,
```

```
"top_logprobs": [
  {
    "Hello": 0.5773516400136589,
    "Hi": 0.19021177923596874,
    "Hello": 0.067644583100199,
    "\n\n": 0.06604950779814563,
    "Hi": 0.025466052201047033
  },
  {
    "!": 0.5264271466063052,
    ".": 0.2591786724769263,
    ",": 0.09482852674076715,
    " there": 0.04379592072803984,
    "!\n": 0.024123977887792435
  }
],
"text_offset": [
  22,
  28
],
"total_logprobs": [
  0.9267235623490194,
  0.9483542444398309
```

Oct 27

2 / 5

Oct 27

4d

Values for the second token you can see match the playground:

User: Say 'hello'.

AI: Hello! How can I help you?

!= 52.64%

. = 25.92%

, = 9.48%

there = 4.38%

!\n = 2.41%

Total: -0.64 logprob on 1 tokens  
(94.84% probability covered in top 5 logits)

# Prompt Engineering

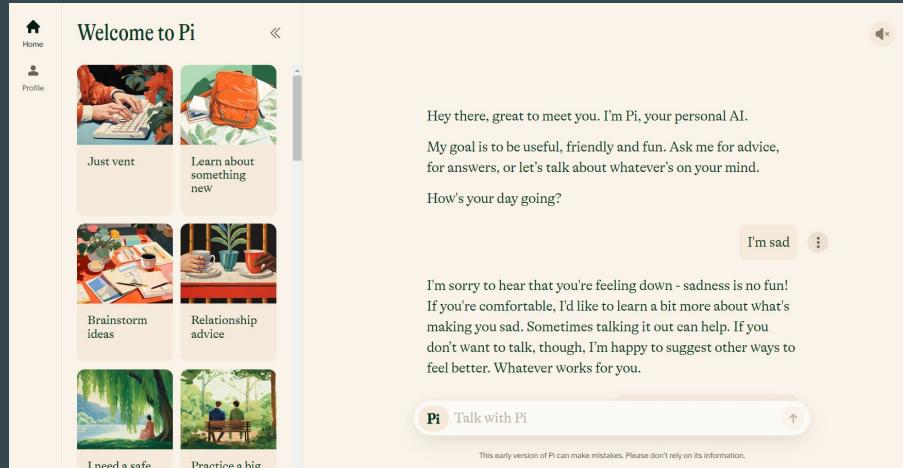
- Simple Prompts
- N-shot In Context Learning (ICL)
- Reflection
- Chain of Thought
- Tree of Thought
- ReAct

# Future

# Future

- LMM
- Automated Agents
- XAI & FATE
- Small Open LLM
- Affective AI
- Theoretical Basis

## New Best Friend: pi.ai Chatbot



Fin