# Gen AI & Classic ML

## when to use what

July 2024

Felix Wick

# Classic ML vs Deep Learning          Generative vs Predictive Models

## text generation          image synthesis          text-to-video



ChatGPT



Prompt: Epic anime artwork of a wizard atop a mountain at night casting a cosmic spell into the dark sky that says "Stable Diffusion 3" made out of colorful energy

Stable Diffusion 3 — Stability AI



Prompt: A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She…

Sora | OpenAI

## BERT family          tabular data          computer vision



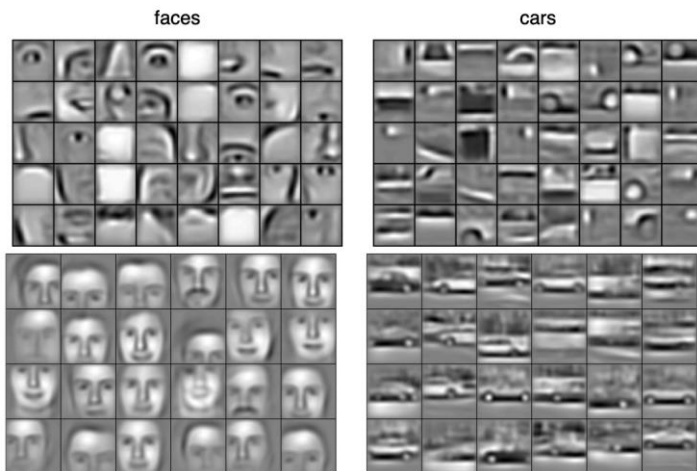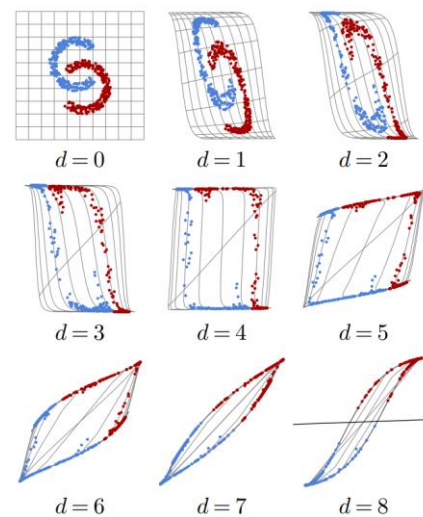YOLO

2

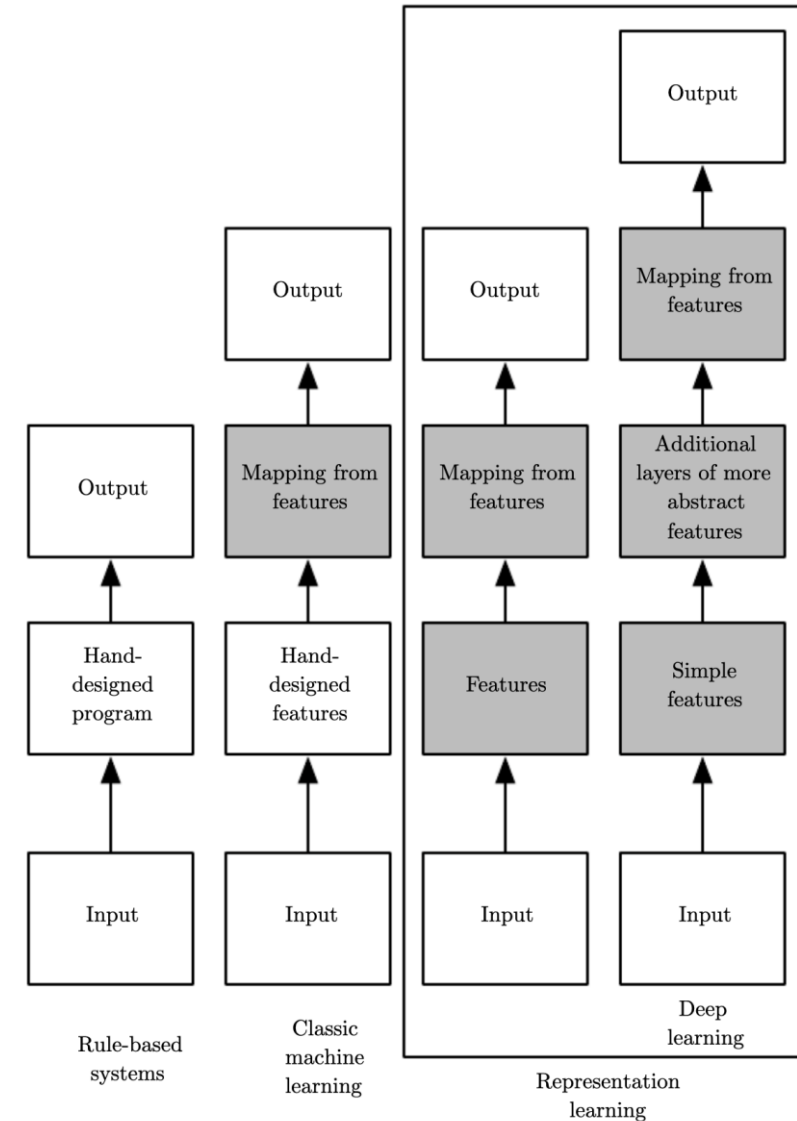# Ladder of Generalization

classic ML: feature engineering

deep learning: feature learning

(hierarchy of concepts learned from raw data in deep graph with many layers)

# Structured/Tabular vs Unstructured Data

unstructured data: homogenous

→ deep learning rules



ImageNet

structured data: heterogenous

→ feature engineering needed

→ deep learning loses its advantage
  over shallow methods

→ e.g., gradient boosting still prominent

# Transfer Learning

idea:
- generic pre-training of foundation models on huge data sets
- subsequent fine-tuning for specific tasks on small(er) data sets

very successful for:
- computer vision (e.g., object classification)
- language models (e.g., BERT, GPT)

not (yet) for tabular data

# Language Models: Contextual Semantics

- self-supervised learning: e.g., next/masked-word prediction

- tokenization: split text into chunks (e.g., words)

- semantics by means of vector embeddings: e.g., via bag-of-words (or end-to-end in transformer)

- positional encoding & embeddings: order of sequence

- contextual embeddings: (self-)attention (weighted averages: influence from other tokens)



SCALE & SOFTMAX

| Token 1 Similarity Vector |
| Token 2 Similarity Vector |

**QK.T**

512 x 512

| Token 512 Similarity Vector |

| Token 1 Embedding |
| Token 2 Embedding |

**V**

512 x 768

| Token 512 Embedding |

| Token 1 Enriched Embedding |
| Token 2 Enriched Embedding |

**(QK.T)V**

512 x 768

| Token 512 Enriched Embedding |
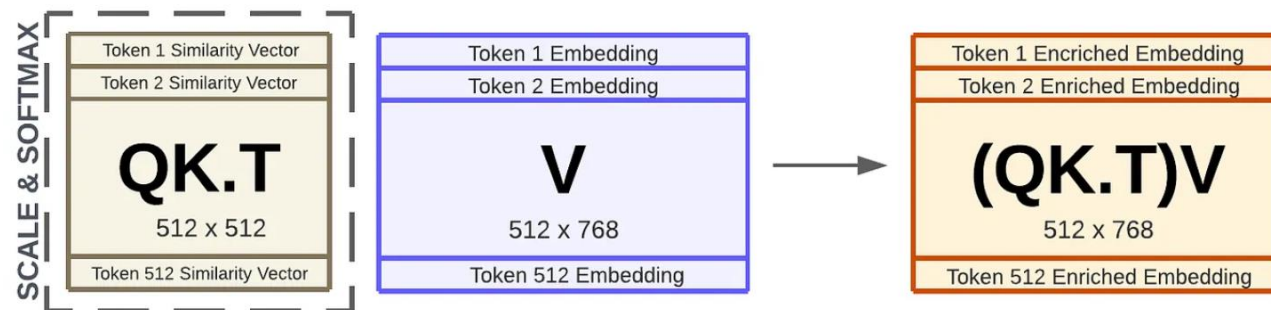
source

# Encoder vs Decoder LLMs

encoder-only LLMs

- prime example: BERT

- self-supervised pre-training: masked-word prediction

- fine-tuning on downstream tasks (e.g., sequence classification)

- can't generate text

- can't be prompted
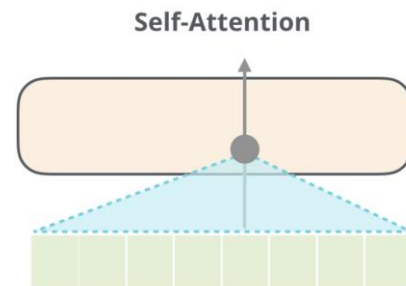
decoder-only LLMs

- prime example: GPT

- self-supervised pre-training: next-word prediction

- instruction tuning (e.g., RL from human feedback)

- generate text: chat bots

- prompt engineering

**Self-Attention**

**Masked Self-Attention**
aka causal

source

7
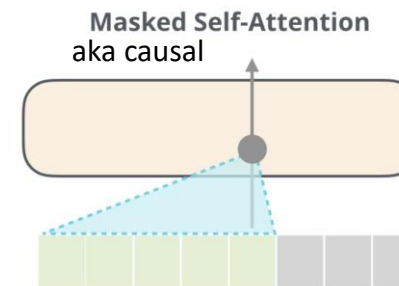
# Generative vs Predictive/Discriminative Models

discriminative models:

predict conditional probability $P(Y|\boldsymbol{X})$

generative models:

predict joint probability $P(Y, \boldsymbol{X})$

(or just $P(\boldsymbol{X}) \rightarrow$ unsupervised learning)

discriminative model          generative model



source
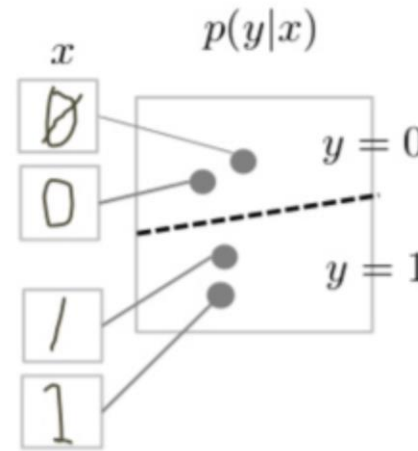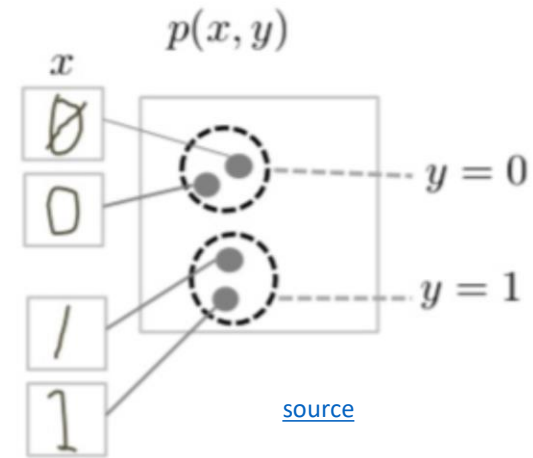
task of generative models more difficult: need to model full data distribution rather than merely find patterns in inputs to distinguish outputs

generative models allow to generate new data samples (text, images, video, proteins, …)

predictive models usually better for predictive tasks, business problems often specific/predictive

# Deep Learning for Generative AI

Depending on the application, there are currently two dominant approaches for generative AI:

- text generation: decoder LLMs

- image synthesis: diffusion models

note the difference between image synthesis and multimodal understanding in LLMs (images as additional input sequences to transformer, tokenized by splitting into patches)

# Image Synthesis

idea: generate new images as variations of training data

usually conditioned on text (prompt) by transformers

compared to text generation, additional mechanism needed (e.g., diffusion) to create more complex structures

Input

An astronaut riding a horse in photorealistic style.
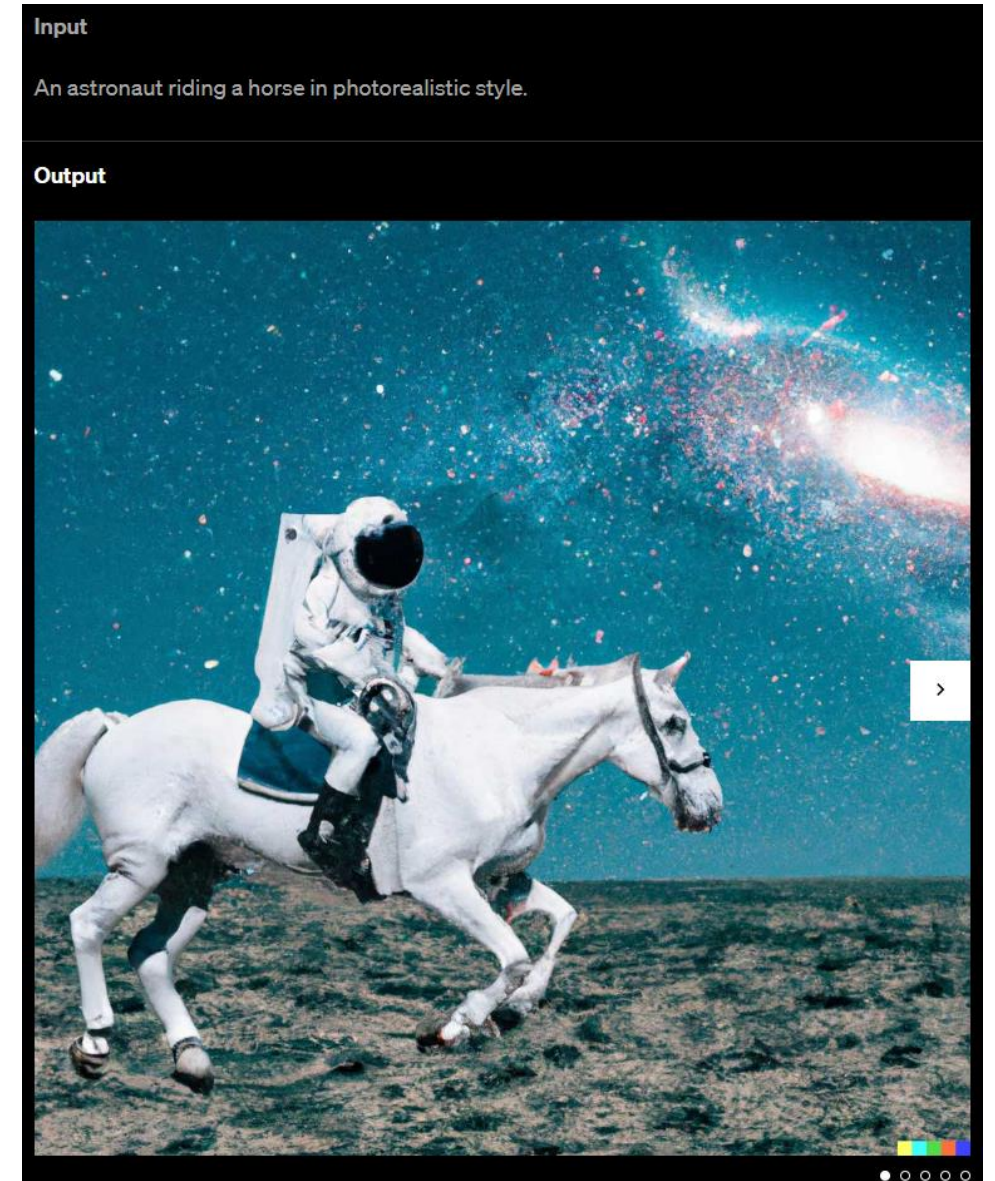
Output

# Text Generation

in-context learning as alternative to fine-tuning (new paradigm):
feed information into LLM via input prompt (decoder LLMs)
→ attention to context

typical prompt:
instructions, context (potentially retrieved), query, output indicator

enables multi-task capabilities (all of ML before was only narrow tasks)
→ assistants

(open-source) SOTA (July 2024): Llama 3

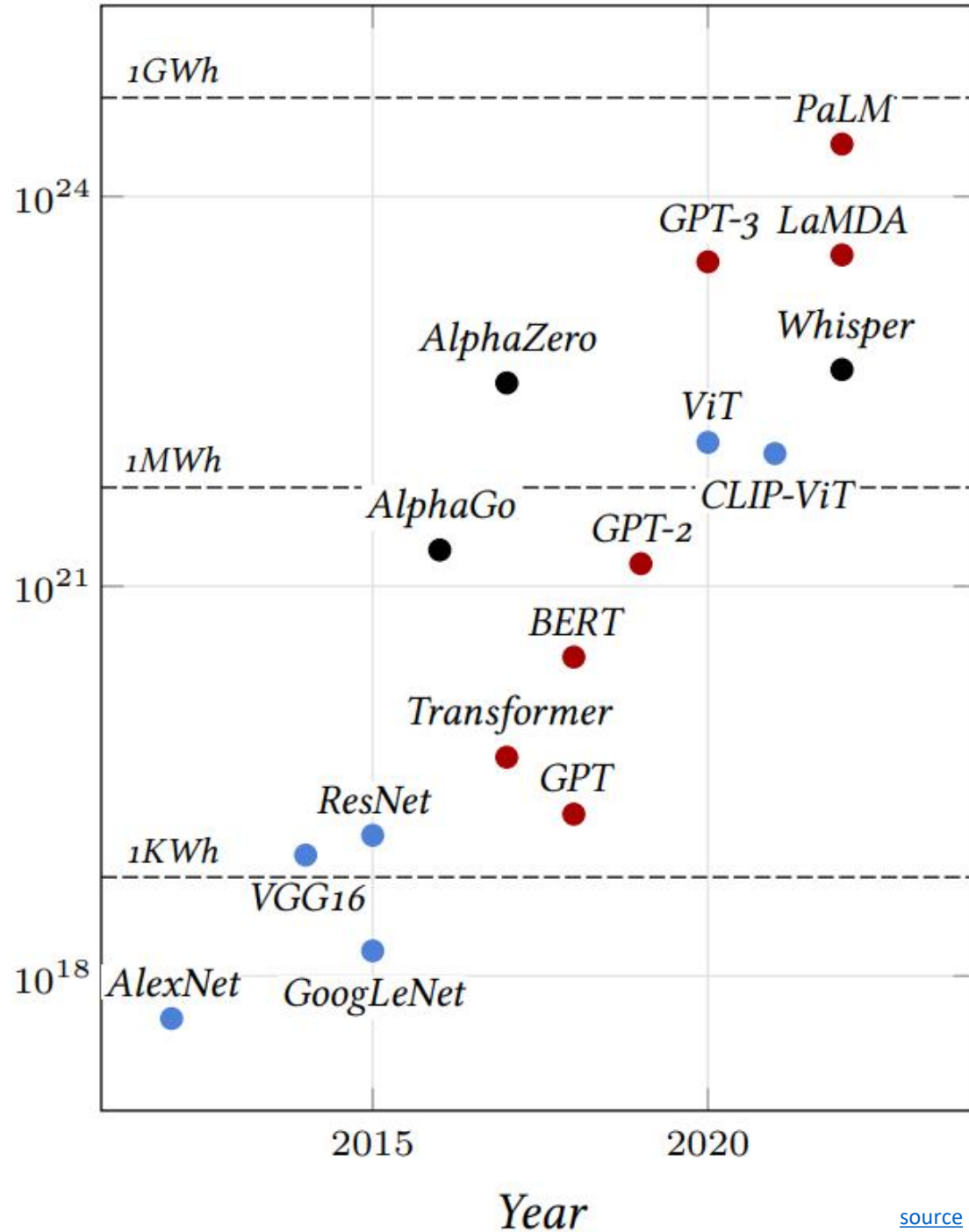**Closed-source vs. open-weight models**

Llama 3.1 405B closes the gap with closed-source models for the first time in history.

source

# Some LLM Numbers

example Llama 3 405B:

- vocabulary size (tokens):          128K
- embedding/model dimensions:     16,384

- parameters:                        405B          factor less than 40
- training tokens:                    15.6T          → a lot of memorizing

- context length/window (tokens):   128K

- training hardware:                  16K GPUs (H100)

# LLM Agents

current AI good at learning statistical patterns and making predictions

but no real "understanding", and limited reasoning and planning capabilities

desired agent capabilities:

- planning (LLM: decomposition of complex issue in multiple simple steps)
- tool use (LLM: use predictive models for numerical/optimization tasks)
- reflection
- collaboration with other agents

# Goal: Autonomous End-to-End Workflow
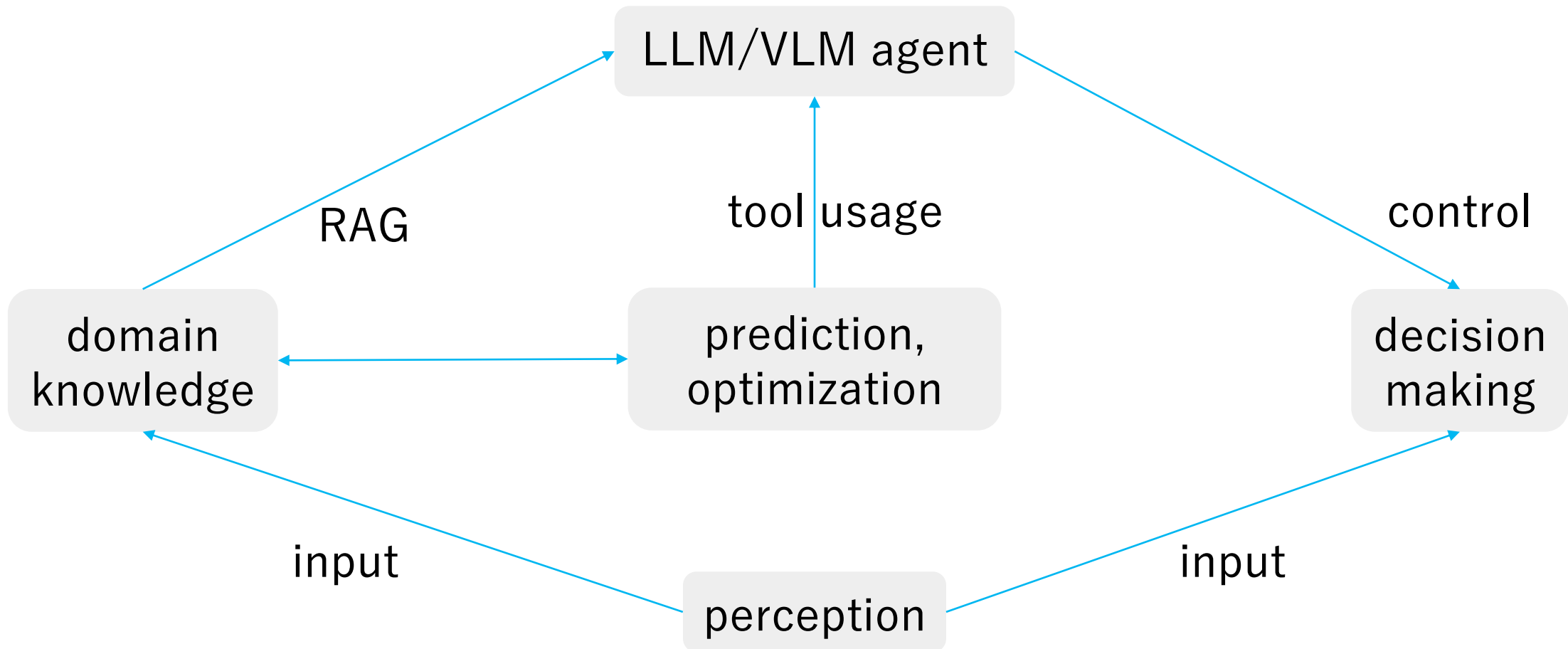
# Sequential Decision Making
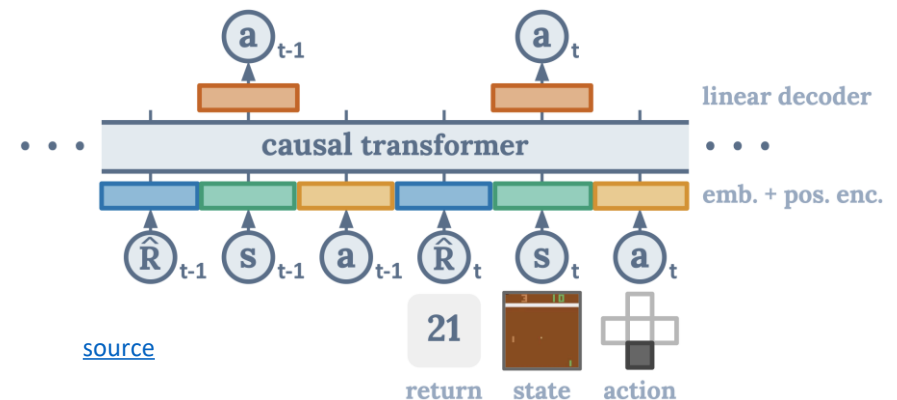
typically, domain of reinforcement learning

sequence modeling as alternative:

- generative: transformer decoder to autoregressively model trajectories
- credit assignment directly via self-attention: state-return associations
- desired return tokens as prompt for action generation
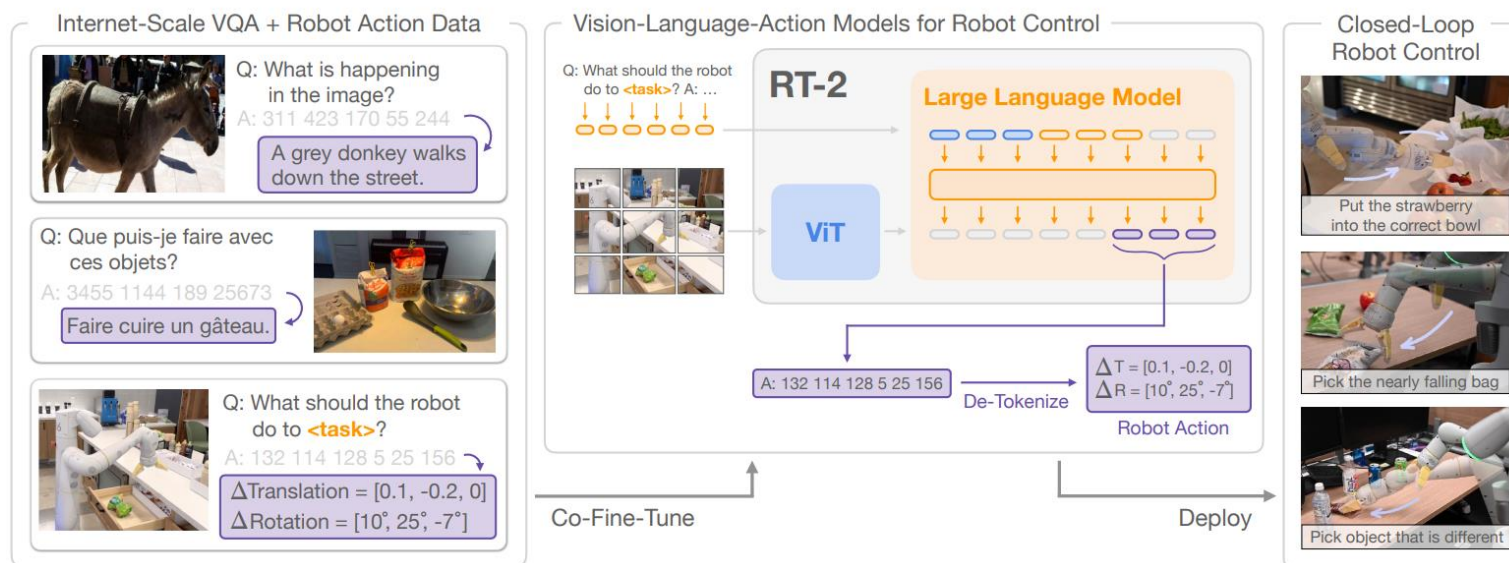
overcoming the deadly triad of deep RL:
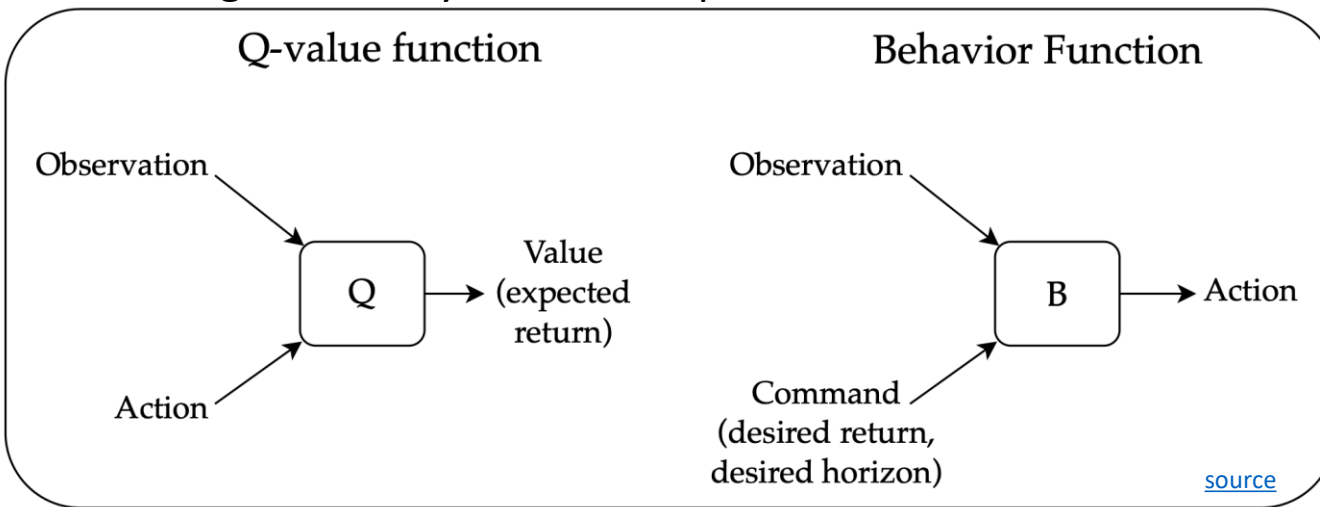


Decision Transformer:

# Robotic Control generated by LLMs/VLMs

RT-2:



Code as Policies:

SayCan (grounding with pre-trained skills):



(a) Large Language Models (LLMs)                    (b) SayCan