Generative AI. Seminar

Irina Abdullaeva

Researcher, FusionBrain Lab, AIRI

Term 5. Modern Methods and Algorithms of Generative Artificial Intelligence

00

Intro

About me

- In 2021, graduated from Bauman Moscow State Technical University, majoring in Information Systems and Technologies
- In 2024, completed postgraduate studies at MIPT, specialising in Artificial Intelligence and Machine Learning
- Since 2022 I have worked at SberDevices.
- Since 2023 I am working at AIRI
- I have lectured in Machine Learning at MIPT and Bauman Moscow State Technical University.
- I regularly participate in conferences of all levels.

Fields of expertise:

- Multimodal models for video, images and time-series
- Multi-agent LLMs approaches
- Behavioural biometry

Tell about yourself

- What is your name?
- Are you involved in Generative AI? And how (dissertation, work)?
- Why did you choose this course? What made you feel interested in it?
- How do you want to apply this knowledge in further?

01

Organization details

Course structure

- → 10 general topics → 10 lectures and 10 seminars
- → Physical Attendance Requirement - 80% of classes
- → Make a report (paper reading and analysis) on seminar classes (get additional scores for your final grade!)
- → Complete 2 Assignments for course

Assignment Type	Assignment Summary	% of Final Course Grade
Homework Assignments	Homework 1 — A task for multimodal model fine tuning. Homework 2 — A task on text-to-image/text-to-video model fine tuning.	50
Final Project	During the course a student should provide a presentation and report about the project on generative AI	50

AIRI

Course structure

**Full list of HW tasks will be provided later*

Assignment 1: Homeworks

- **Homework 1** — A task for multimodal model fine tuning.
- **Homework 2** — A task on text-to image/text-to-video model fine tuning.

Steps:

- 1. Dataset collection and processing
- 2. Base model, fine tuning method, evaluation metrics choices
- 3. Creating & running fine-tuning pipeline
- 4. Evaluate fine-tuned model on test set with chosen metric

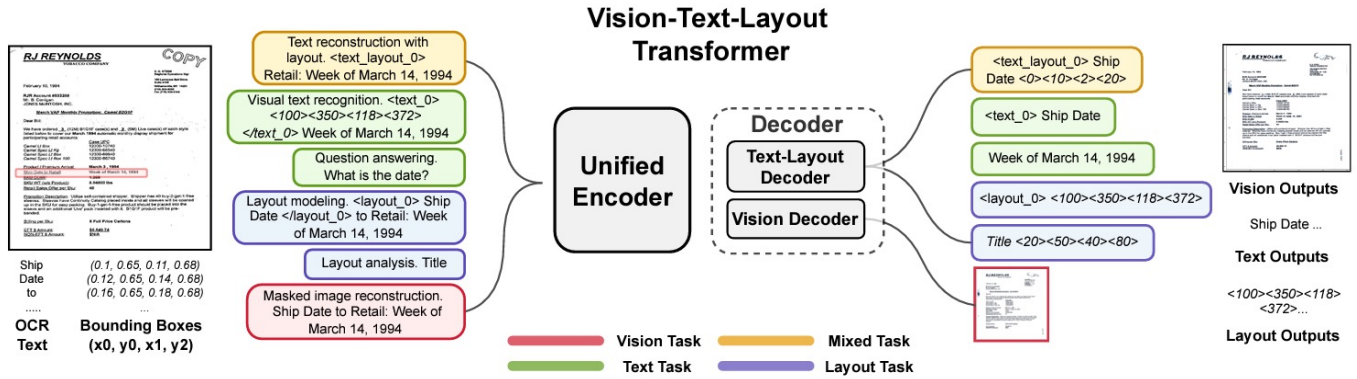
Outcomes:

- The source code of the model & pipeline (it should be not a simple fork of an open source solution, but should provide a separate GitHub repo, organized in a well-structured form!).
- The presentation of the method and all the details of its implementation (in form of report or presentation).

Assignment 1: Homeworks Example

Homework 1:

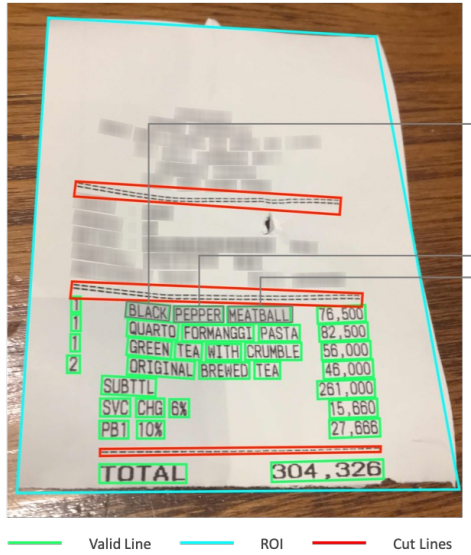
Given a pretrained visual language model develop its suitable fine tuned version to improve the model's skills in receipts recognition and parsing.



Design choices:

- → 🤗 Transformers for the UDOP model
- → 🤗 Datasets to load a CORD dataset
- → SentencePiece for the tokenizer
- → Bitsandbytes for the memory-efficient 8-bit optimizer
- → Tesseract for the OCR engine.

Here is an Example notebook for full fine-tuning pipeline



Image


```
...
{
    "words":[
            {
              "quad":{"x2":  837,"y3": 2260,"x3":  829,"y4": 2238,
                       "x1":  625,"y1": 2156,"x4":  617,"y2": 2177},
              "is_key": 0,
              "row_id": 1631316,
              "text": "BLACK"
            },
            {
              "quad":{"x2": 1105,"y3": 2261,"x3": 1105,"y4": 2261,
                       "x1":  857,"y1": 2180,"x4":  857,"y2": 2180},
              "is_key": 0,
              "row_id": 1631316,
              "text": "PEPPER"
            },
            {
              "quad":{"x2": 1441,"y3": 2264,"x3": 1443,"y4": 2269,
                       "x1": 1125,"y1": 2184,"x4": 1126,"y2": 2179},
              "is_key": 0,
              "row_id": 1631316,
              "text": "MEATBALL"
            }
          ],
          "category": "menu.nm",
          "group_id": 3
}
...
```


JSON

Course structure

**Full list of FP topics will be provided later*

Assignment 2: Final project

Design a project proposal on using **multimodal models in remote sensing data analysis**

- → the idea for implementation of the project should be realistic and easy to follow
- → SoTA architectures should be used as baselines and the choice should be explained
- → training data should be described or there should be an instructing for data collection
- → implemented PoC of **the project on GitHub and/or demo will be a benefit**
- → **the project proposal** should include description of all the research stages for implementation (a document and/or presentation)
- → the team for project development may include from **1 to 3 members**

idea + implementation + description + presentation

AIRI

Course structure

Reports on papers & approaches

Make a report (paper reading and analysis) on seminar classes (get additional scores for your final grade!)

There are 2-3 papers reports for each topic.

You have to:

- → **Select paper** and be able to explain your choice.
- → Carefully read paper 2+ times to extract**: key features and components; novelty of approach; potential weaknesses**.
- → Make **a short presentation and speech** about the paper to everyone ~ 10-15 min. with graphical material.

Do not just re-tell paper, otherwise interpret it!

AIRI

Course structure

Topics

1. Generative AI
2. Large language models and applications
3. Foundation models
4. Multimodality. Perception.
5. State Space Models
6. Long context analysis
7. Domain-specific applications
8. Multimedia content generation
9. Diffusion-based models
10. Generative LLM agents

01 Generative AI

The term **Generative AI** refers to computational techniques that are capable of generating seemingly new, meaningful content such as text, images, or audio from training data.

Large language models

- ✓ *Attention mechanism*
- ✓ *Tokenizers*
- ✓ *Transformer architectures*

Multimodal perception

- ✓ Types of modalities fusion
- ✓ Encoders for modalities
- ✓ Visual language models

Multimedia generation

- ✓ *Transformer-based models*
- ✓ *Diffusion-based models*
- ✓ *Distributed training*

Output Modality (Selection)	Model level *Underlying AI model for different data modalities (e.g., image, text, code)*	System level *Embedding model functionality to provide interface for interaction*	Application level *Solving dedicated business problems and stakeholder needs*
Text Generation	X-to-text models, e.g., GPT-4 and LLaMA 2	Conversational agents and search engines, e.g., ChatGPT and YouChat	• Content generation (e.g., SEO and customer service) • Translation and text summarization
Image/Video Generation	X-to-image models, e.g., Stable Diffusion and DALL-E 2	Image/video generation systems and bots, e.g., Runway and Midjourney	• Synthetic product and advertising visuals • Educational content
Speech/Music Generation	X-to-music/speech models, e.g., MusicLM and VALL-E	Speech generation systems, e.g., ElevenLabs	• AI music generation • Text-to-speech generation (e.g., news, product tutorials, etc.)
Code Generation	X-to-code models, e.g., Codex and AlphaCode	Programming code generation systems, e.g., GitHub Copilot	• Software development • Code synthesis, review, and documentation

Figure 1: A model-, system-, and application-level view on generative AI.

Generative AI

AIRI

02 Large language models and applications

Large language models

Lecture

- ✓ Transformers architecture re-cap.
- ✓ Attention mechanism types.
- ✓ Mixture-of-experts (MoE) approach.

Seminar

- ✓ Prompting techniques
- ✓ Advanced reasoning with LLMs
- ✓ Generation with cache
- ✓ Retrieval-augmented generation (RAG)

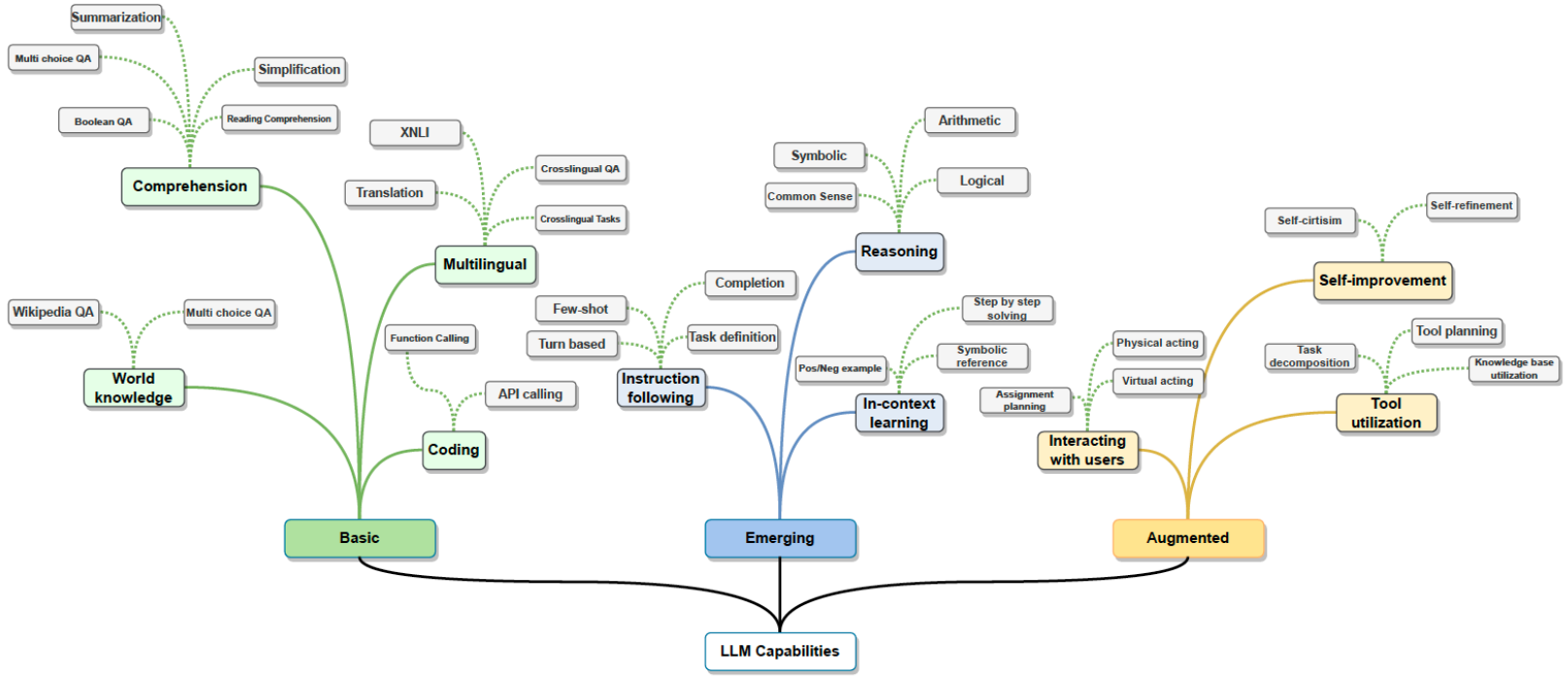


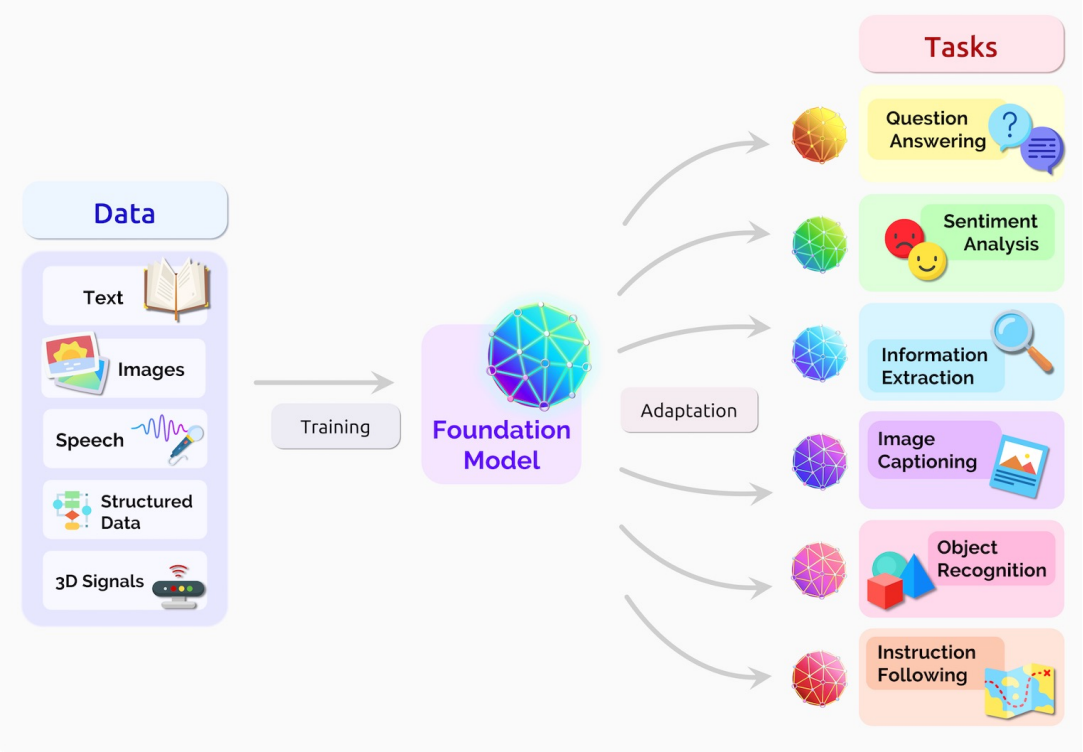
Fig. 1: LLM Capabilities.

03 Foundation models

”Foundation model - any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks”

by The Stanford Institute for Human-Centered Artificial Intelligence's (HAI) Center for Research on Foundation Models (CRFM)

Machine Learning — Deep Learning — Foundation Models



AIRI

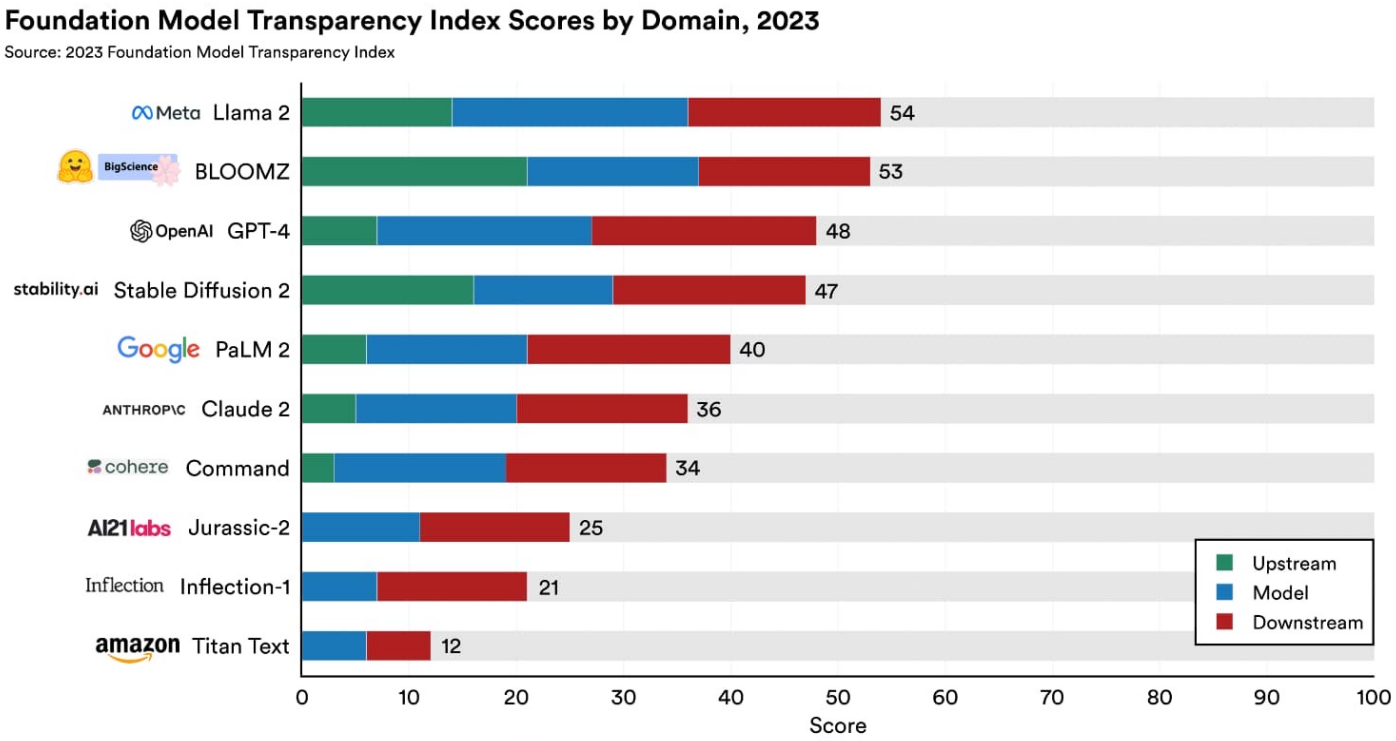
03 Foundation models

Lecture

- ✓ Foundation model definition.
- ✓ Language model as a foundation model.
- ✓ Tasks and challenges.

Seminar

- ✓ Foundation models examples discussion
- ✓ Advanced training techniques: Data Parallel, Model Distributed DP, FSDP
- ✓ Example training with Accelerate



Scores for the 10 foundation model providers, broken down by domain.

Bommasani, Rishi; et al. (18 August 2021). On the Opportunities and Risks of Foundation Models (Report).

AIRI

04-5 Multimodality. Perception. Part 1 and 2.

Multimodal AI is a computational field focusing on understanding and leveraging multiple modalities.

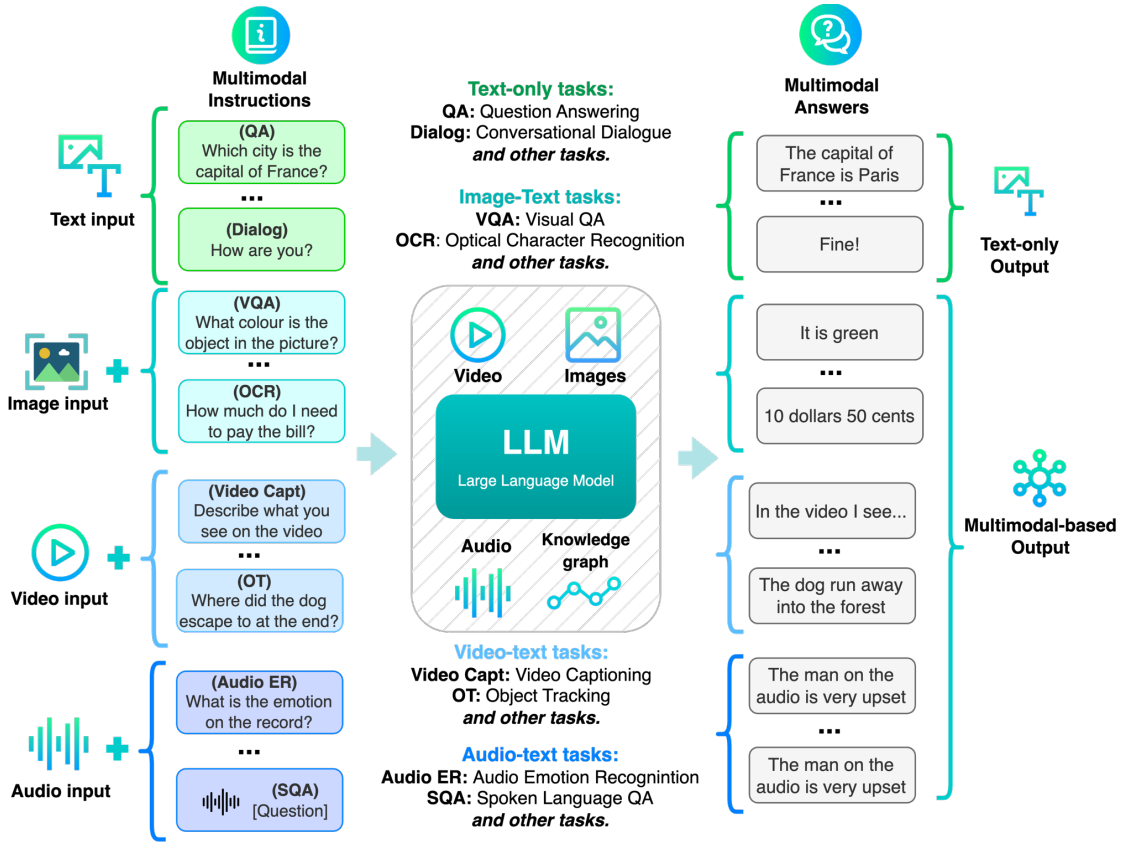
A **modality** refers to a way in which a natural phenomenon is perceived or expressed

Lecture

- ✓ Types of fusion: early, middle and late
- ✓ Visual language models (VLMs)
- ✓ Types of image and video encoders. Fusion of encoders and LLMs
- ✓ Overview of transformer-based approaches and state-of-the-art architectures. Tasks, challenges and benchmarks
- ✓ State Space Models (SSM)
- ✓ Mixture of Visual Experts

Seminar

- ✓ In-depth overview of basic VLMs
- ✓ Fine-tuning approaches for VLMs



Wadekar, Shakti N., et al. "The Evolution of Multimodal Model Architectures." (2024).

Bordes, Florian et al. “An Introduction to Vision-Language Modeling.” by Meta (2024)

AIRI

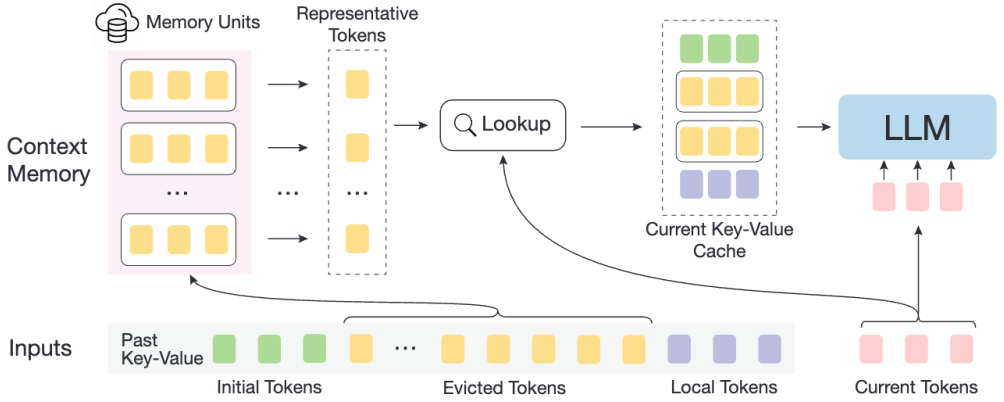
06 Long context analysis

Lecture

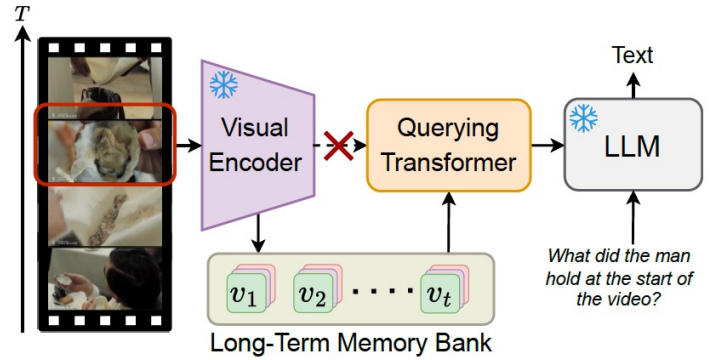
- ✓ Long context problem and challenges
- ✓ Long context handling in LLMs and VLMs
- ✓ Ways of long video perception: from short videos to streaming understanding

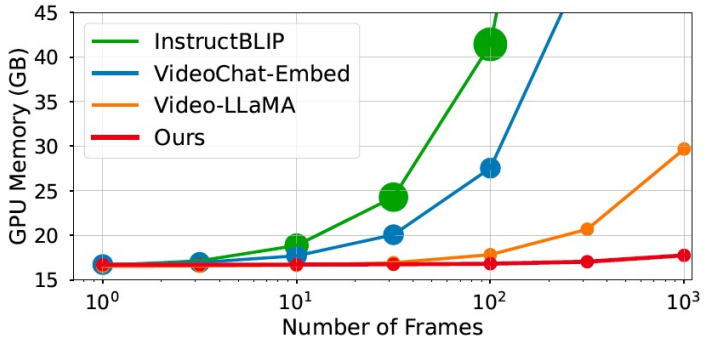
Seminar

- ✓ In-depth overview of basic approaches in Video VLMs
- ✓ Memory-augmentation in VLMs



[1] InfLLM approach





[2] MA-LLM approach

[1] Xiao, Chaojun et al. "InfLLM: Training-Free Long-Context Extrapolation for LLMs with an Efficient Context Memory." (2024).

[2] He, Bo et al. "MA-LMM: Memory-Augmented Large Multimodal Model for Long-Term Video Understanding." (2024)

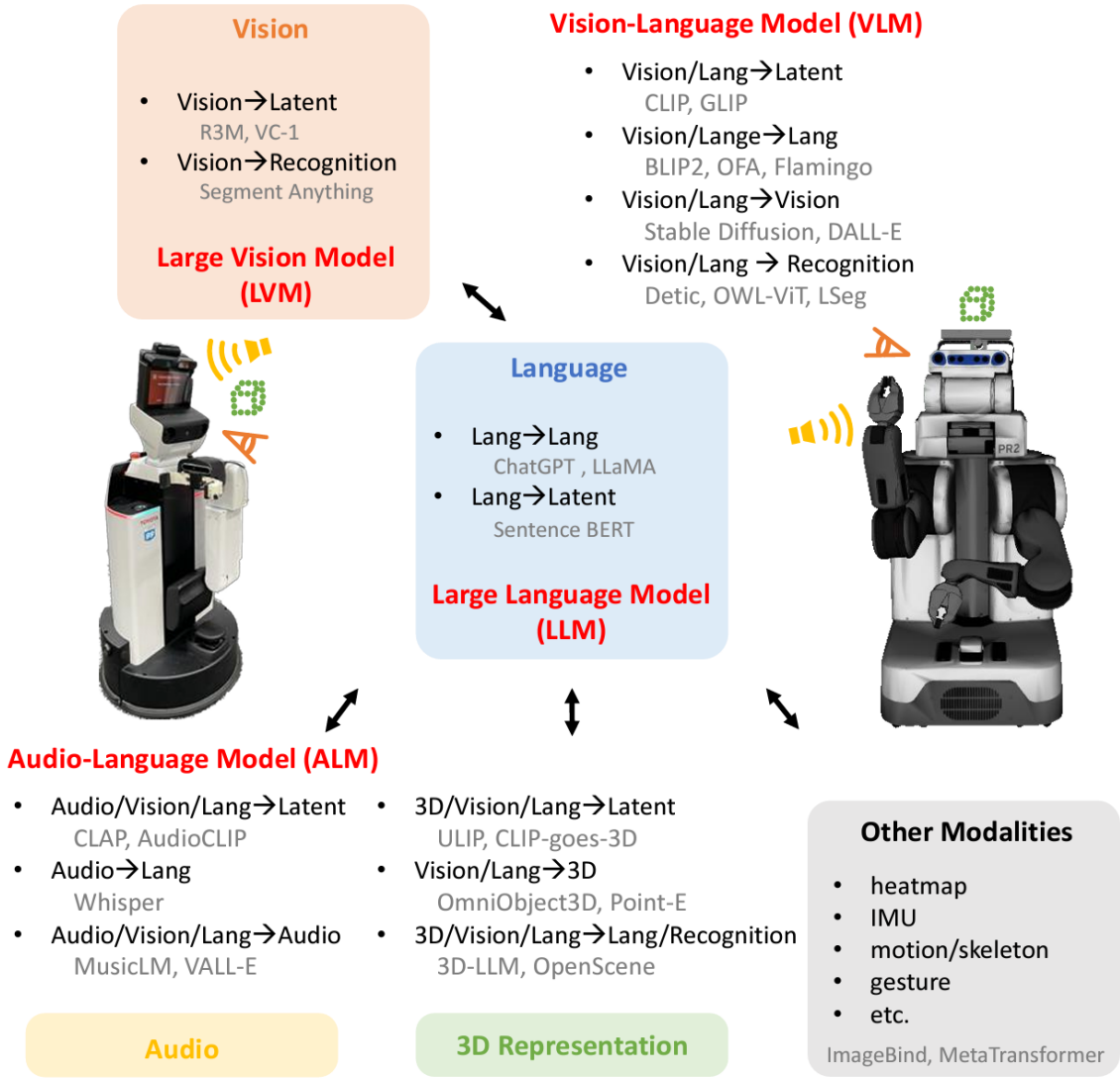
Repo: Awesome-Long-Context-Language-Modeling

07 Domain-specific applications

Lecture + Seminar

Multimodality in:

- ✓ robotics
- ✓ self-driving cars
- ✓ medicine
- ✓ and other fields of research and industry



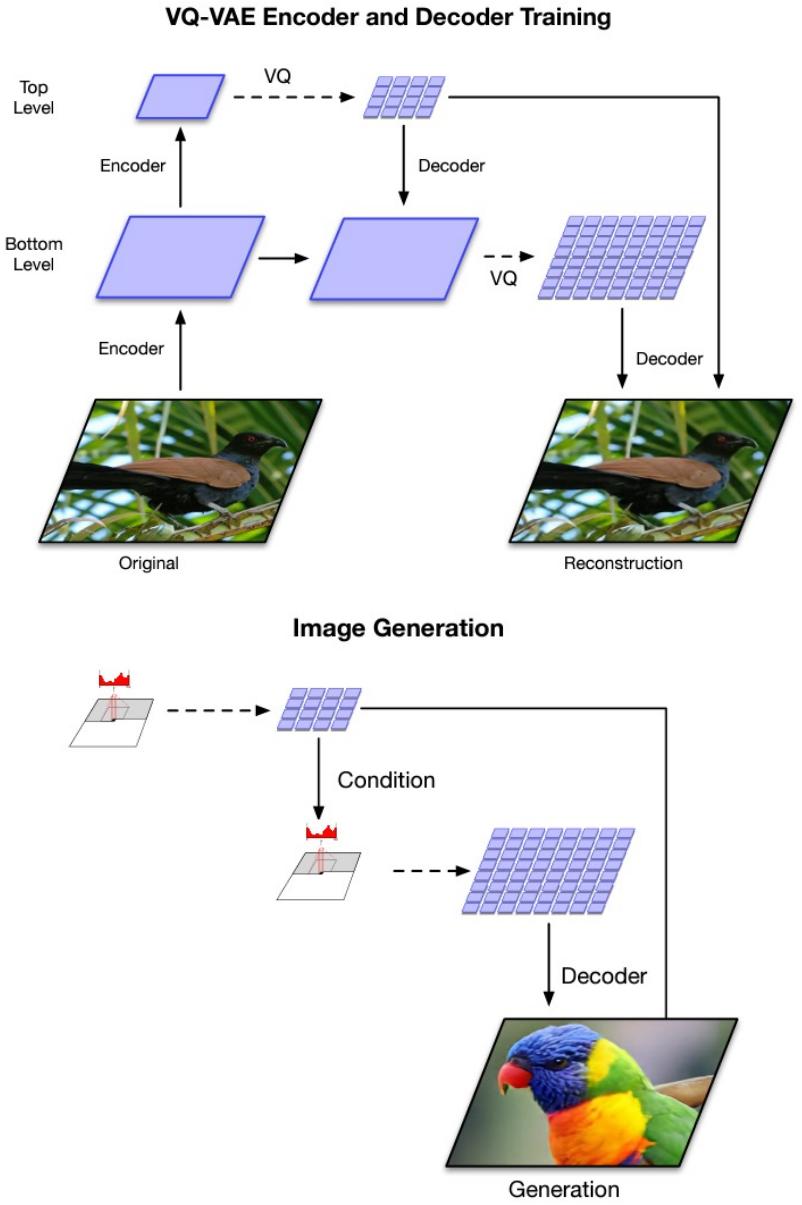
08 Multimedia content generation

Lecture

- ✓ Transformer-based image generation models
- ✓ DALL-E, Parti and other approaches

Seminar

- ✓ In-depth overview of basic approaches for image generation

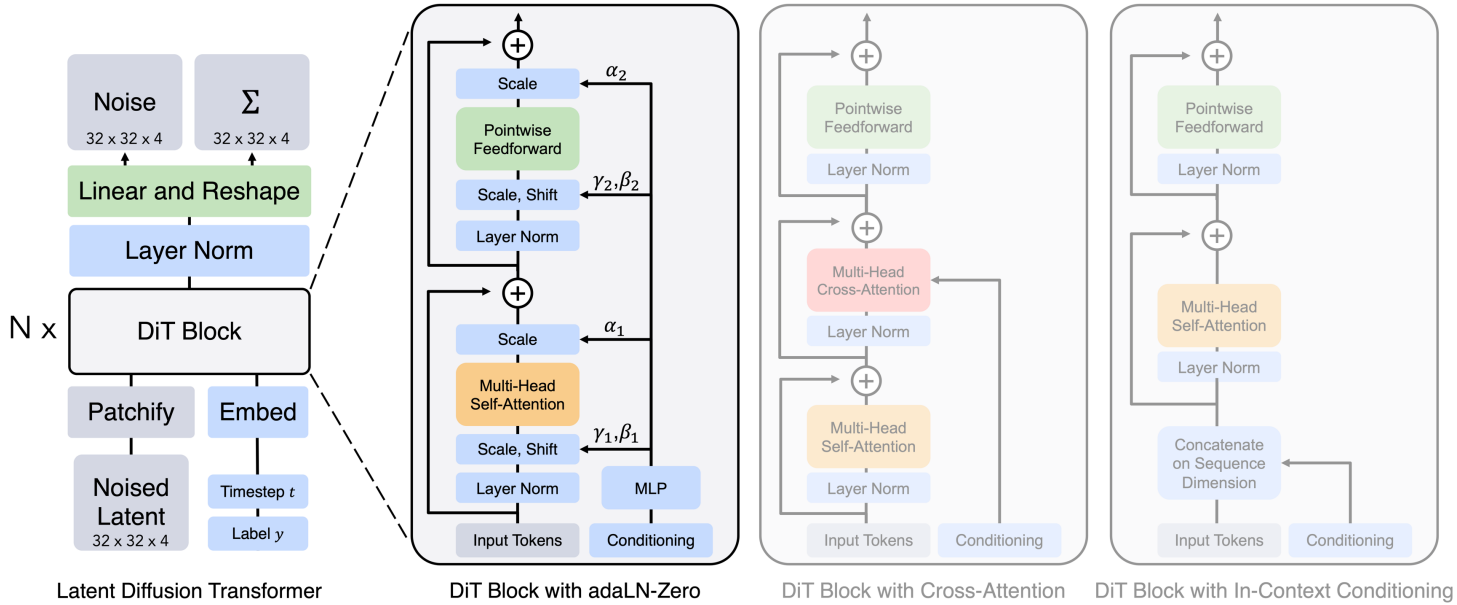


Razavi, Ali et al. “Generating Diverse High-Fidelity Images with VQ-VAE-2.” Neural Information Processing Systems (2019).

09 Diffusion-based models

Lecture

- ✓ Diffusion for content generation
- ✓ Image generation using diffusion
- ✓ Existing models and approaches: Stable Diffusion, Kandinsky 2.1, 3.0, etc.
- ✓ Diffusion transformers
- ✓ Flow matching techniques for video generation



Seminar

- ✓ In-depth overview of basic approaches for diffusion models for image generation
- ✓ Examples of generation with Kandinsky

Kandinsky 3.0

Kandinsky 3.1

[Peebles, William S. and Saining Xie. "Scalable Diffusion Models with Transformers." 2023 IEEE/CVF International Conference on Computer Vision (ICCV) (2022): 4172-4182.]

AIRI

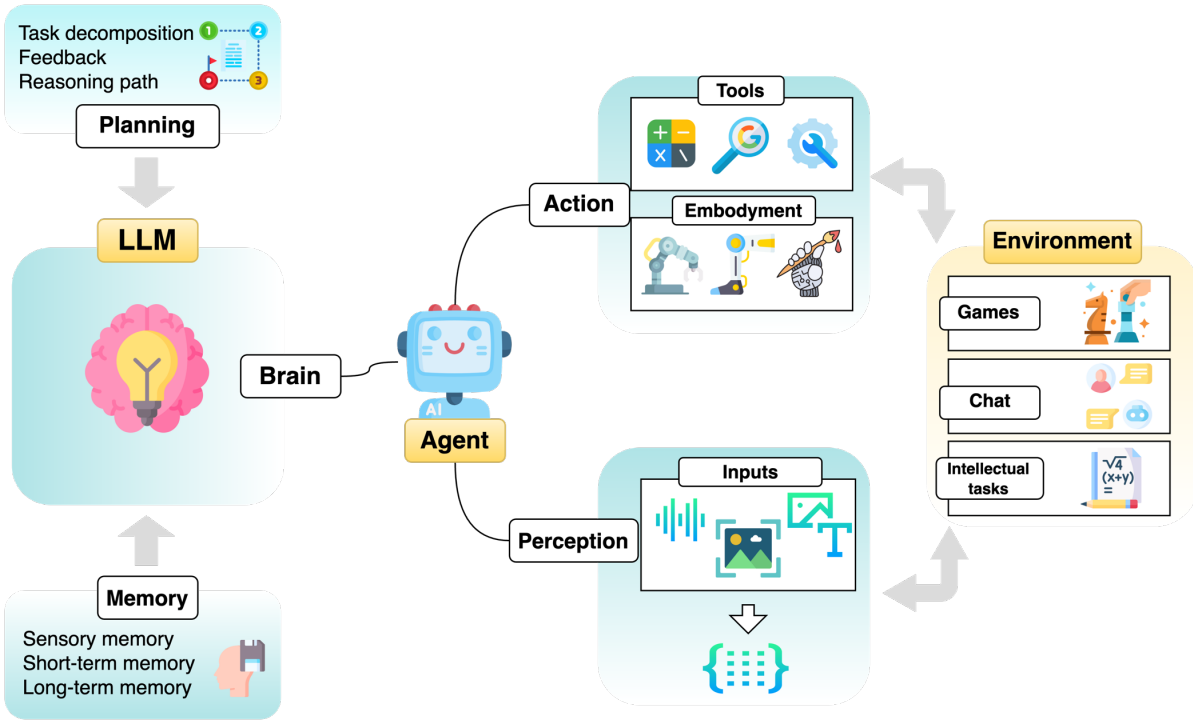
10 Generative LLM agents

Lecture

- ✓ LLM-based Agent: components
- ✓ Functions calling and planning: Reflexion and etc.
- ✓ Communication in multi-agent systems
- ✓ Multi-modal and multi-agent systems

Seminar

- ✓ In-depth overview of basic approaches for multi-agent systems
- ✓ Personalization in LLM-based Agents



02

Generative AI Techstack

Generative AI Techstack: from idea to demo

- → **Architecture design**
- → **Training**
- → **Monitoring**
- → **Inference and evaluation**
- → **Demo and interaction**

LLaVA

Diffusers

Hugging Face

PyTorch Lightning

Weights & Biases

tensorboard

neptune.ai

VLMEvalKit

Language Model Evaluation Harness

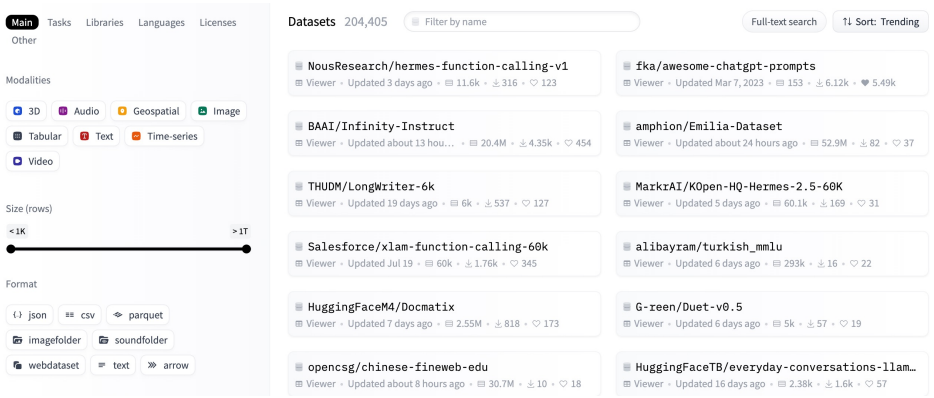
gradio

AIRI

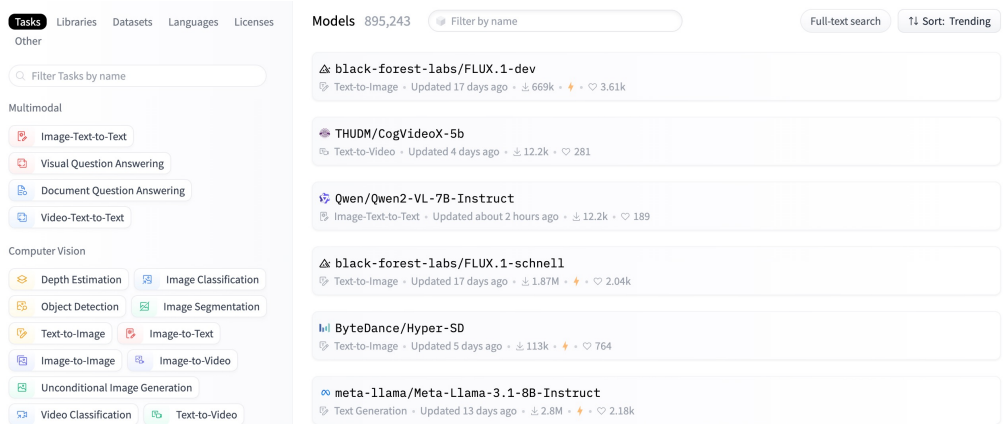
Architecture design

Hugging Face + Hugging Face Hub

Datasets



Models



Transformers

- ✓ Basic models for
  - Natural Language Processing
  - Computer Vision
  - Audio
  - Multimodal
- ✓ Agents
- ✓ Generation utils
- ✓ Training & Quantization
- ✓ Lots of examples

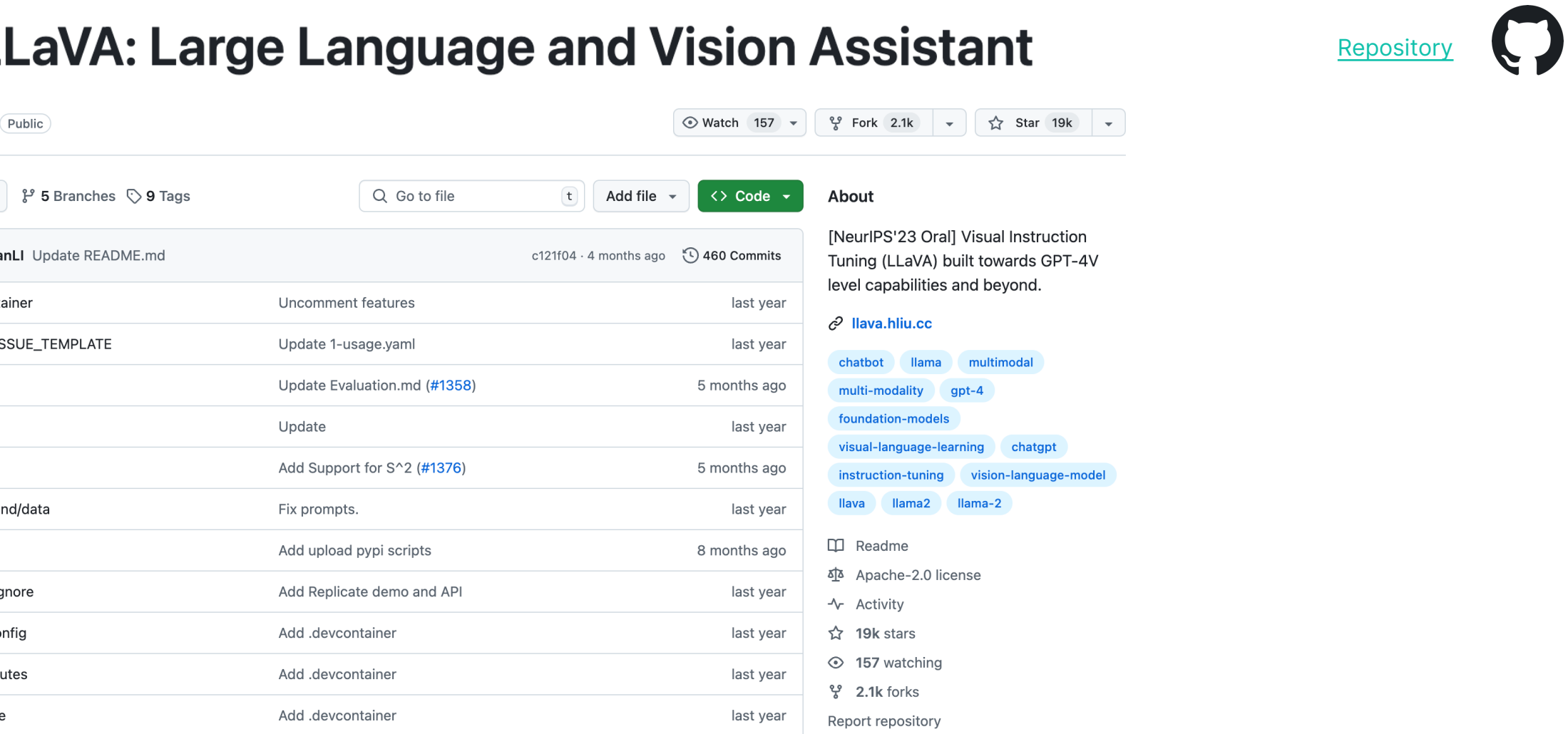
Diffusers

- ✓ SOTA diffusion models for images, video, audio and 3D
- ✓ Fine-tuning techniques
- ✓ Lots of examples
- ✓ Pipelines

Architecture design - customization

🌋 LLaVA: Large Language and Vision Assistant

Repository



LLaVA Public

Watch 157 | Fork 2.1k | Star 19k

main | 5 Branches | 9 Tags | Go to file t | Add file | Code

ChunyuanLI Update README.md — c121f04 · 4 months ago — 460 Commits

.devcontainer	Uncomment features	last year
.github/ISSUE_TEMPLATE	Update 1-usage.yaml	last year
docs	Update Evaluation.md (#1358)	5 months ago
images	Update	last year
llava	Add Support for S^2 (#1376)	5 months ago
playground/data	Fix prompts.	last year
scripts	Add upload pypi scripts	8 months ago
.dockerignore	Add Replicate demo and API	last year
.editorconfig	Add .devcontainer	last year
.gitattributes	Add .devcontainer	last year
.gitignore	Add .devcontainer	last year

About

[NeurIPS'23 Oral] Visual Instruction Tuning (LLaVA) built towards GPT-4V level capabilities and beyond.

llava.hliu.cc

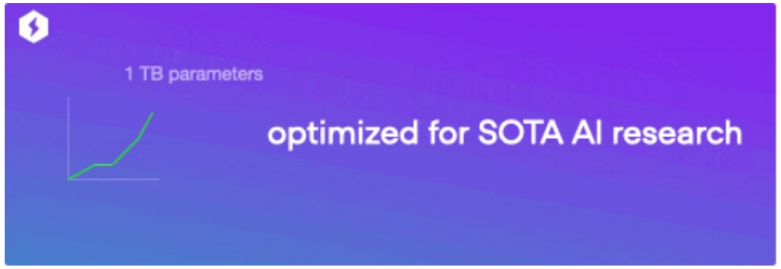
chatbot | llama | multimodal | multi-modality | gpt-4 | foundation-models | visual-language-learning | chatgpt | instruction-tuning | vision-language-model | llava | llama2 | llama-2

Readme
Apache-2.0 license
Activity
19k stars
157 watching
2.1k forks
Report repository

AIRI

Training

WELCOME TO ⚡ PYTORCH LIGHTNING

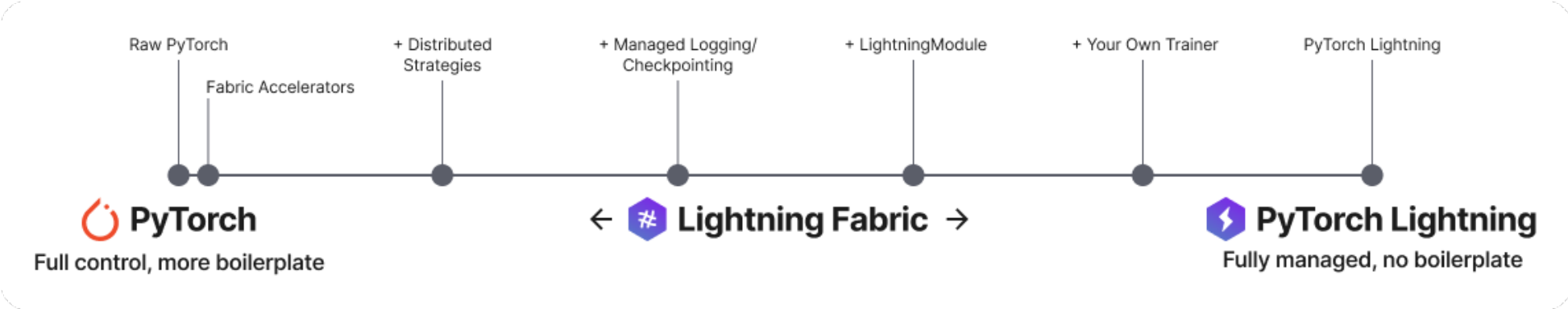


PyTorch Lightning is the deep learning framework for professional AI researchers and machine learning engineers who need maximal flexibility without sacrificing performance at scale. Lightning evolves with you as your projects go from idea to paper/production.

Lightning

Repository

Examples



AIRI

Training

PyTorch Lightning

PYTORCH

```
# models
encoder = nn.Sequential(nn.Linear(28 * 28, 64), nn.ReLU(), nn.Linear(64, 3))
decoder = nn.Sequential(nn.Linear(3, 64), nn.ReLU(), nn.Linear(64, 28 * 28))

encoder.cuda(0)
decoder.cuda(0)

# download on rank 0 only
if global_rank == 0:
    mnist_train = MNIST(os.getcwd(), train=True, download=True)

# split dataset
transform=transforms.Compose([transforms.ToTensor(),
                              transforms.Normalize(0.5, 0.5)])
mnist_train = MNIST(os.getcwd(), train=True, download=True, transform=transform)

# train (55,000 images), val split (5,000 images)
mnist_train, mnist_val = random_split(mnist_train, [55000, 5000])

# The dataloaders handle shuffling, batching, etc...
mnist_train = DataLoader(mnist_train, batch_size=64)
mnist_val = DataLoader(mnist_val, batch_size=64)

 # optimizer
 params = [encoder.parameters(), decoder.parameters()]
 optimizer = torch.optim.Adam(params, lr=1e-3)

# TRAIN LOOP
model.train()
num_epochs = 1
for epoch in range(num_epochs):
  For train_batch in mnist_train:
    x, y = train_batch
    x = x.cuda(0)
    x = x.view(x.size(0), -1)
    z = encoder(x)
    x_hat = decoder(z)
    loss = F.mse_loss(x_hat, x)
    print('train loss: ', loss.item())

    loss.backward()
    optimizer.step()
    optimizer.zero_grad()

# EVAL LOOP
model.eval()
with torch.no_grad():
  val_loss = []
  for val_batch in mnist_val:
    x, y = val_batch
    x = x.cuda(0)
    x = x.view(x.size(0), -1)
    z = encoder(x)
    x_hat = decoder(z)
    loss = F.mse_loss(x_hat, x)
    val_loss.append(loss)
  val_loss = torch.mean(torch.tensor(val_loss))
  model.train()
```

PYTORCH LIGHTNING

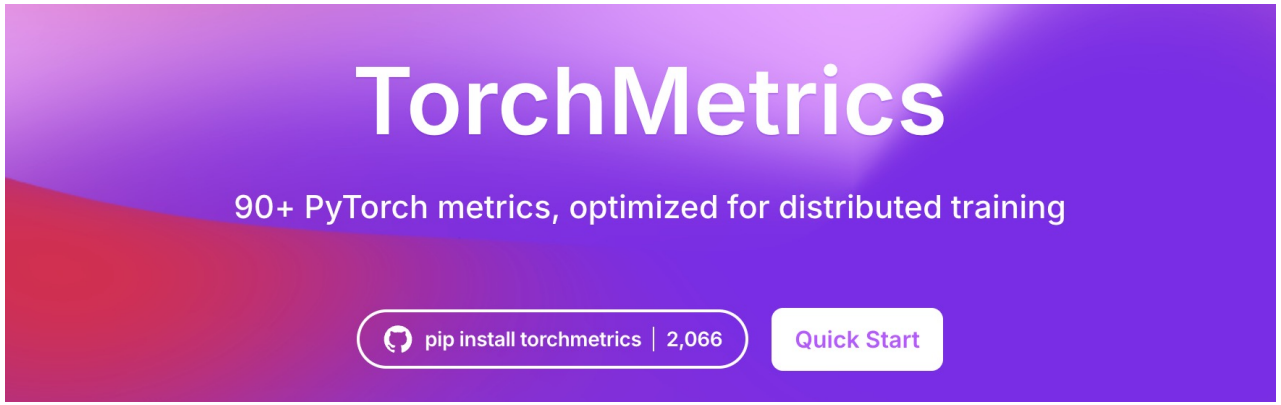
Turn PyTorch into Lightning

Lightning is just plain PyTorch

AIRI

Training

TorchMetrics




```python
import torch
# import our library
import torchmetrics

# initialize metric
metric = torchmetrics.classification.Accuracy(task="multiclass", num_classes=5)

n_batches = 10
for i in range(n_batches):
    # simulate a classification problem
    preds = torch.randn(10, 5).softmax(dim=-1)
    target = torch.randint(5, (10,))
    # metric on current batch
    acc = metric(preds, target)
    print(f"Accuracy on batch {i}: {acc}")

# metric on all batches using custom accumulation
acc = metric.compute()
print(f"Accuracy on all data: {acc}")

# Resetting internal state such that metric ready for new data
metric.reset()
```

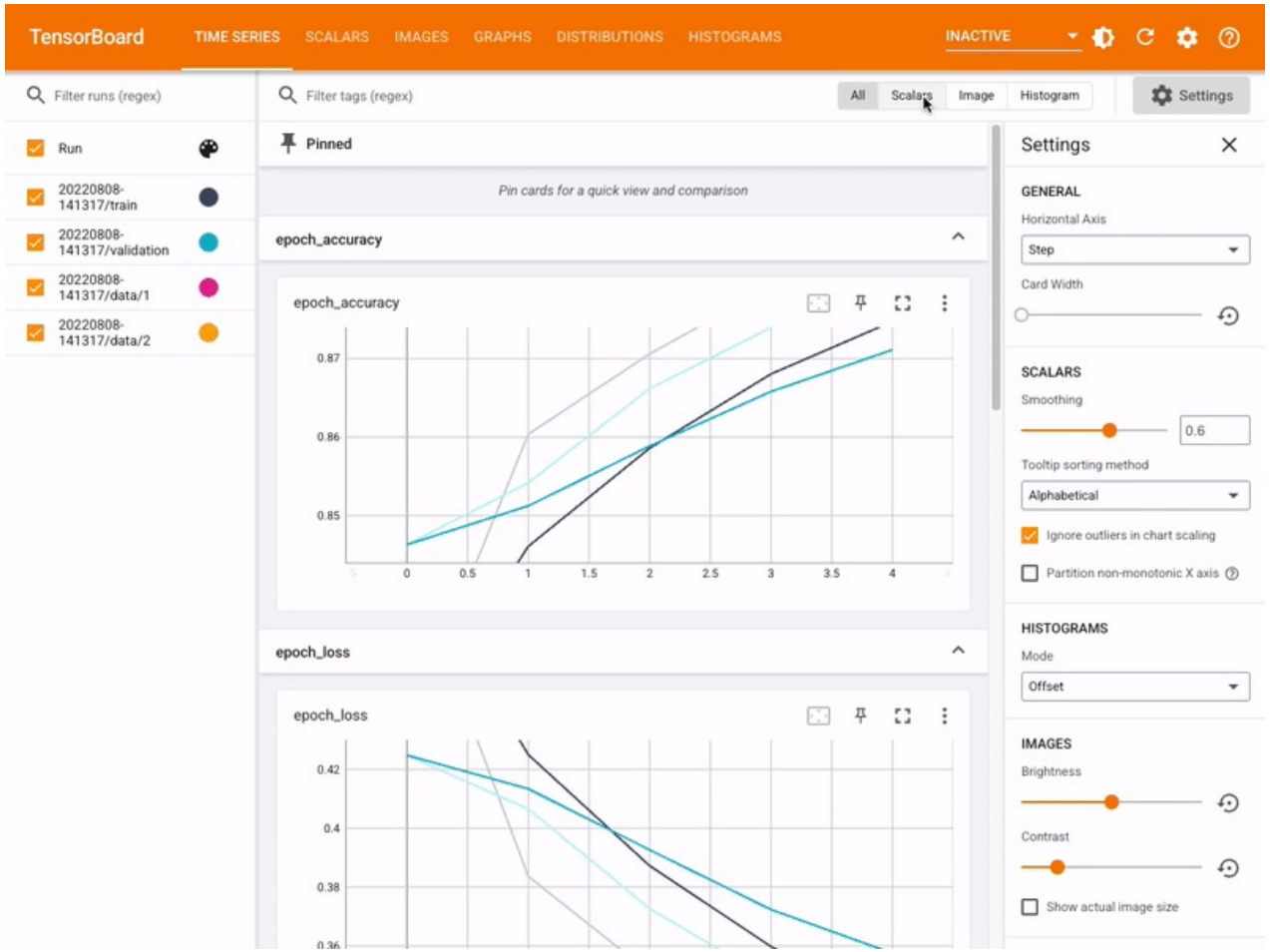

AIRI

Monitoring

TensorBoard provides the visualization and tooling:

- ✓ Tracking and visualizing metrics such as loss and accuracy
- ✓ Visualizing the model graph (ops and layers)
- ✓ Viewing histograms of weights, biases, or other tensors as they change over time
- ✓ Projecting embeddings to a lower dimensional space
- ✓ Displaying images, text, and audio data
- ✓ Can be used with Pytorch also

Documentation

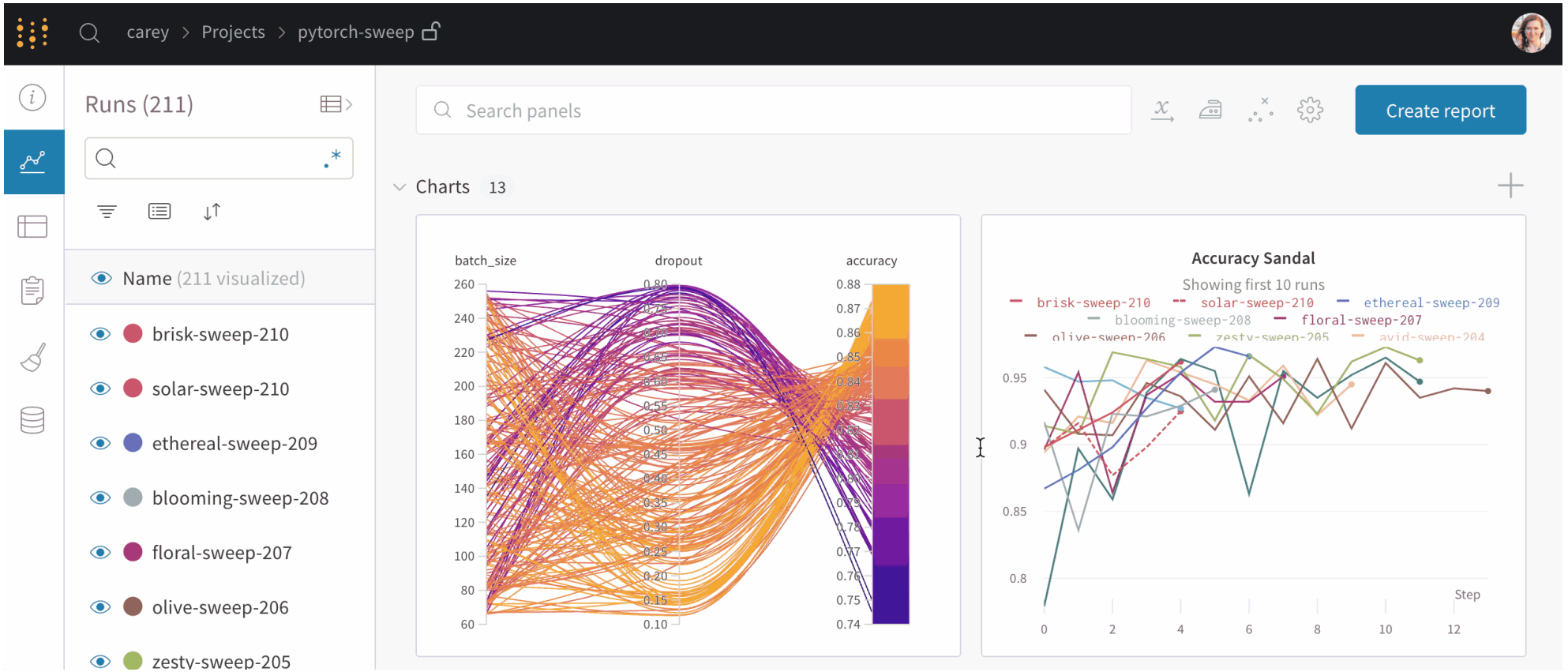


Monitoring

Weights & Biases

Documentation

Repository

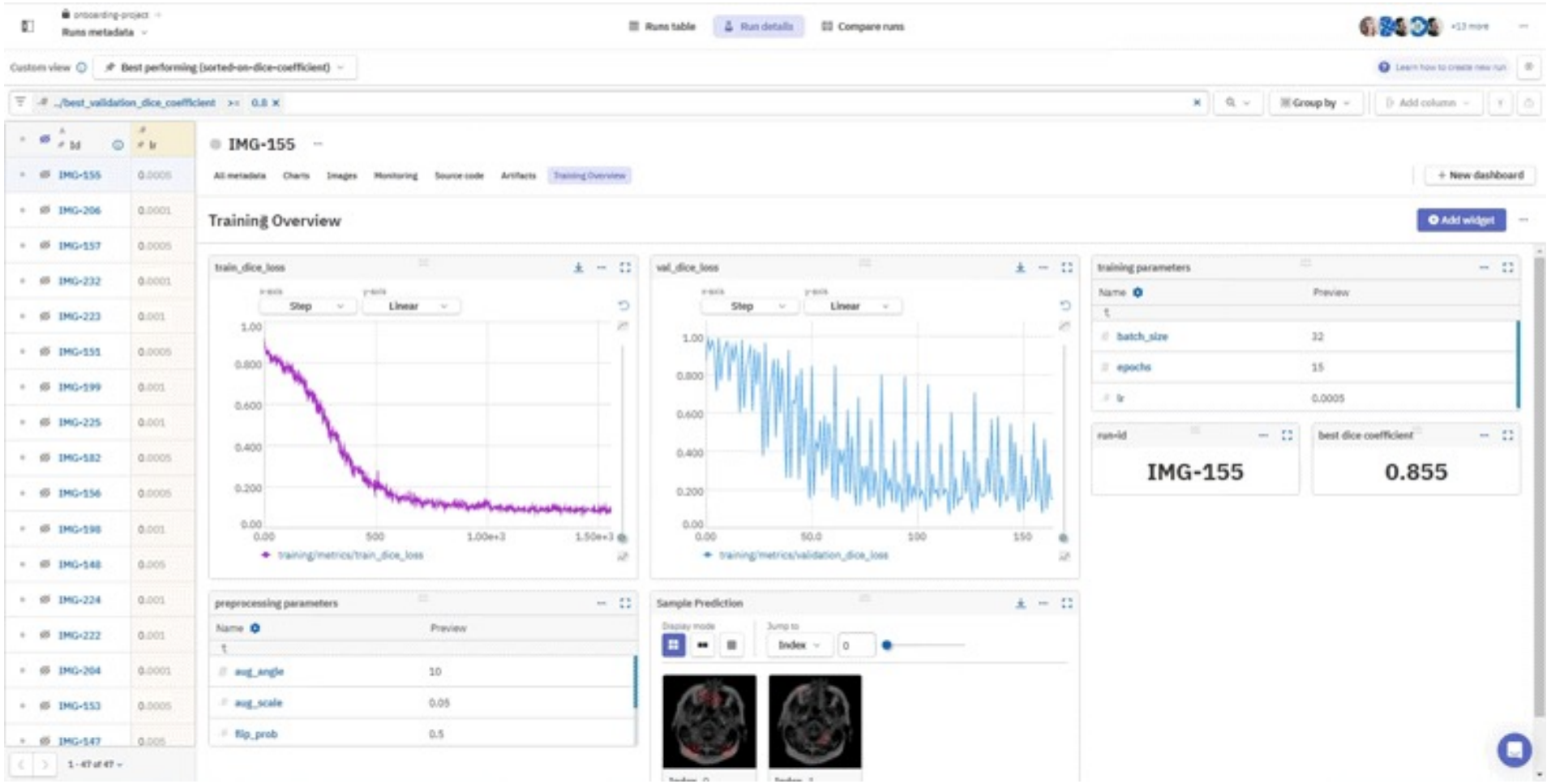


Monitoring

neptune.ai

Documentation

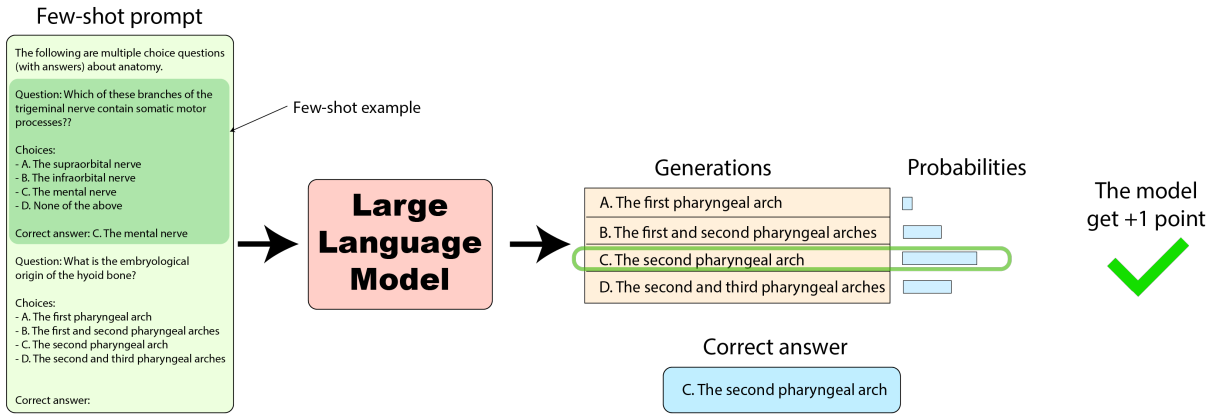
Repository



AIRI

Inference and evaluation

🤗 Open LLM Leaderboard + Language Model Evaluation Harness



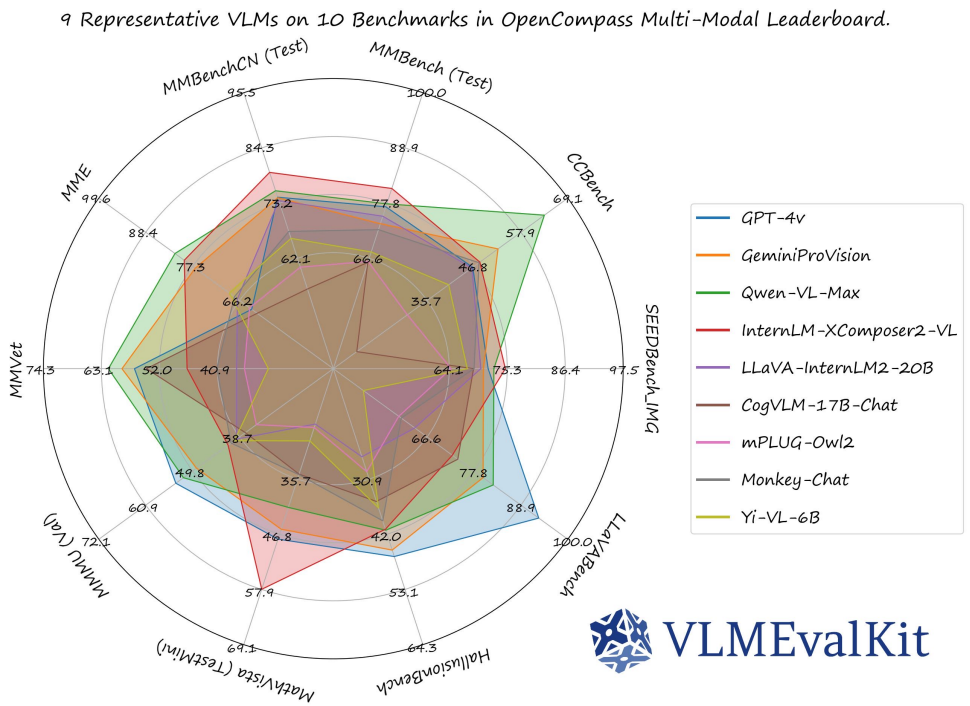
🤗 Leaderboard

Toolkit

T	Model	Average ⬆️	IFEval	BBH	MATH Lvl 5	GPQA	MUSR	MMLU-PR
🔶	dnhkng/RYS-XLarge	44.75	79.96	58.77	38.97	17.9	23.72	49.2
💬	MaziyarPanahi/calme-2.1-rys-78b	44.14	81.36	59.47	36.4	19.24	19	49.38
💬	MaziyarPanahi/calme-2.2-rys-78b	43.92	79.86	59.27	37.92	20.92	16.83	48.73
💬	MaziyarPanahi/calme-2.1-qwen2-72b	43.61	81.63	57.33	36.03	17.45	20.15	49.05
💬	MaziyarPanahi/calme-2.2-qwen2-72b	43.4	80.08	56.8	41.16	16.55	16.52	49.27
💬	Qwen/Qwen2-72B-Instruct	42.49	79.89	57.48	35.12	16.33	17.17	48.92

Inference and evaluation

VLMEvalKit



VLMEvalKit

Leaderboard

Toolkit

Rank	Method	Param (B)	Language Model	Vision Model	Avg Score	Avg Rank	MMBench_V11	MMStar
1	GPT-4o (0806, detail-				71.5	4.12	80.5	64.7
2	InternVL2-Llama3-76B	76	Llama-3-70B-Instruct	InternViT-6B	71	3.62	85.5	67.1
3	GPT-4o (0513, detail-				69.9	6	82.2	63.9
4	InternVL2-40B	40	Nous-Hermes-2-Yi-34B	InternViT-6B	69.7	4.75	85	64.7
5	Step-1.5V		Step-1.5	stepencoder	68.4	7.62	82.5	63.3
6	Claude3.5-Sonnet				67.9	9.75	78.5	62.2

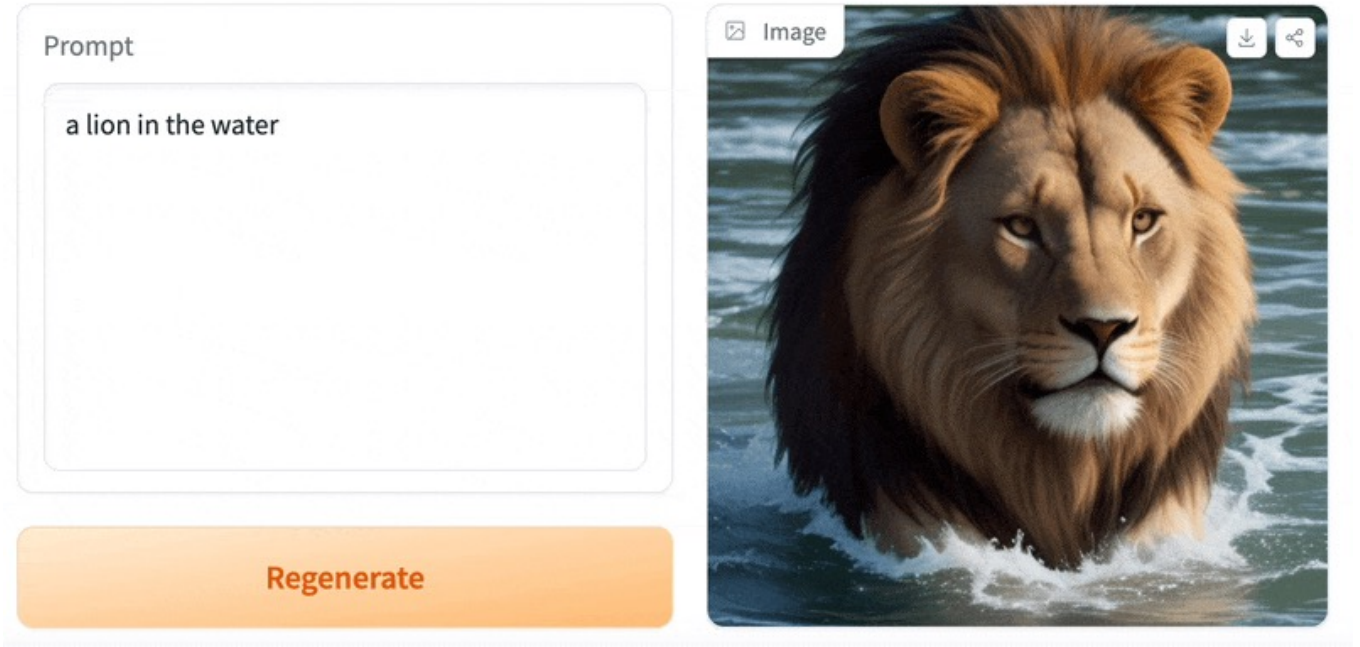
AIRI

Demo and interaction

gradio

Documentation

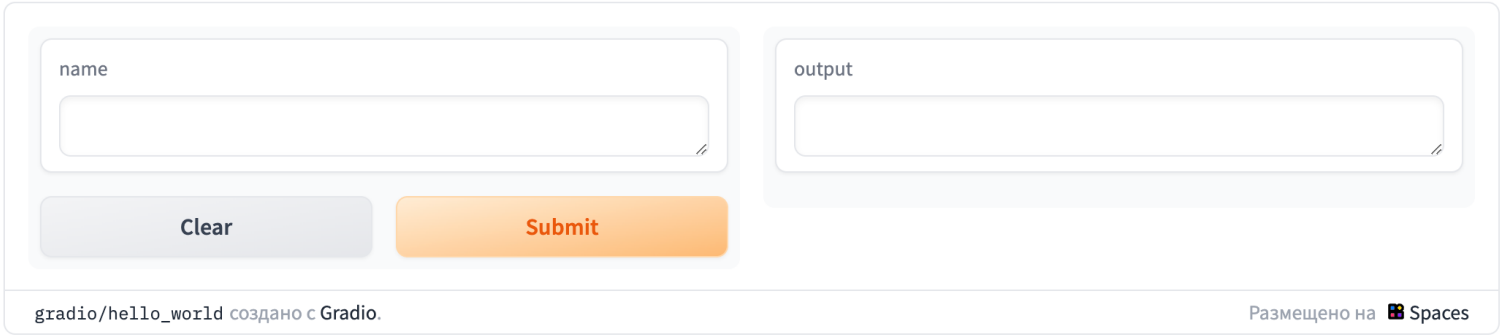
Repository




```python
import gradio as gr

def greet(name):
    return "Hello " + name + "!"

demo = gr.Interface(fn=greet, inputs="text", outputs="text")
demo.launch()
```



AIRI


Modern GenAI Course

Contacts

Irina Abdullaeva

Researcher,
FusionBrain Lab, AIRI

abdullaeva@airi.net

@IrinaAbdullaeva