March 27, 2024
QIAN ZHAO

UF UNIVERSITY of FLORIDA

NLP with LLM On HiPerGator

Agenda

INTRO TO NLP WITH LLM

RESOURCES ON HIPERGATOR

HAND-ON EXERCISES

What is NLP?

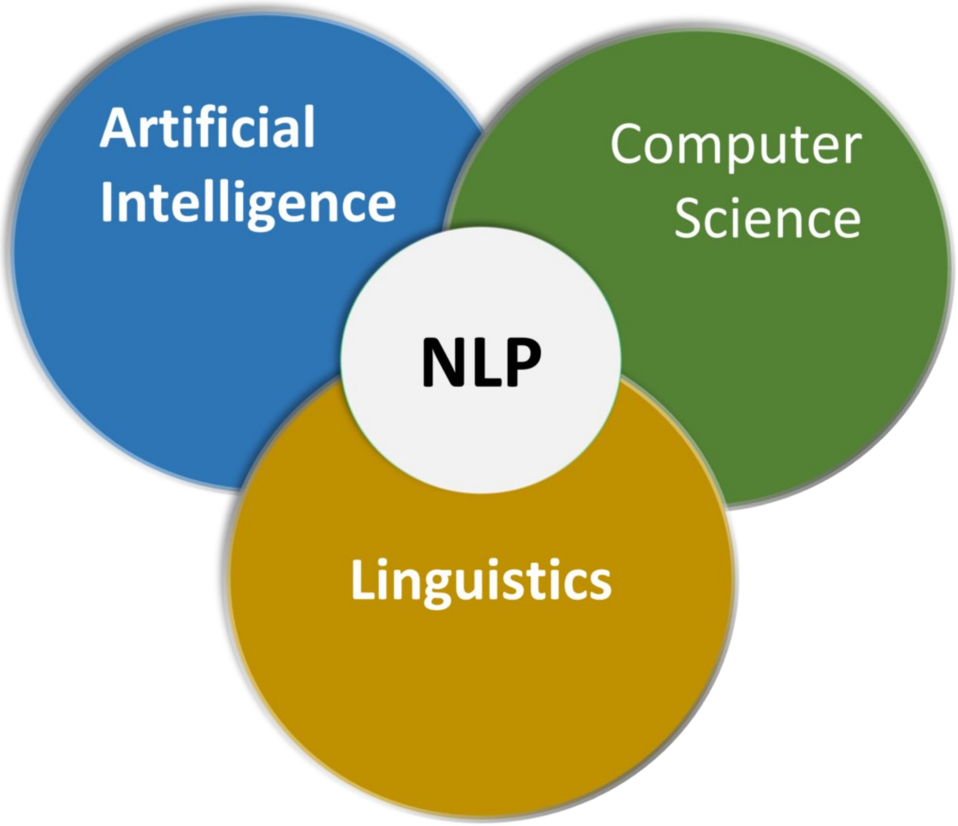


Image Credit: x-rator.com

What is LLM?

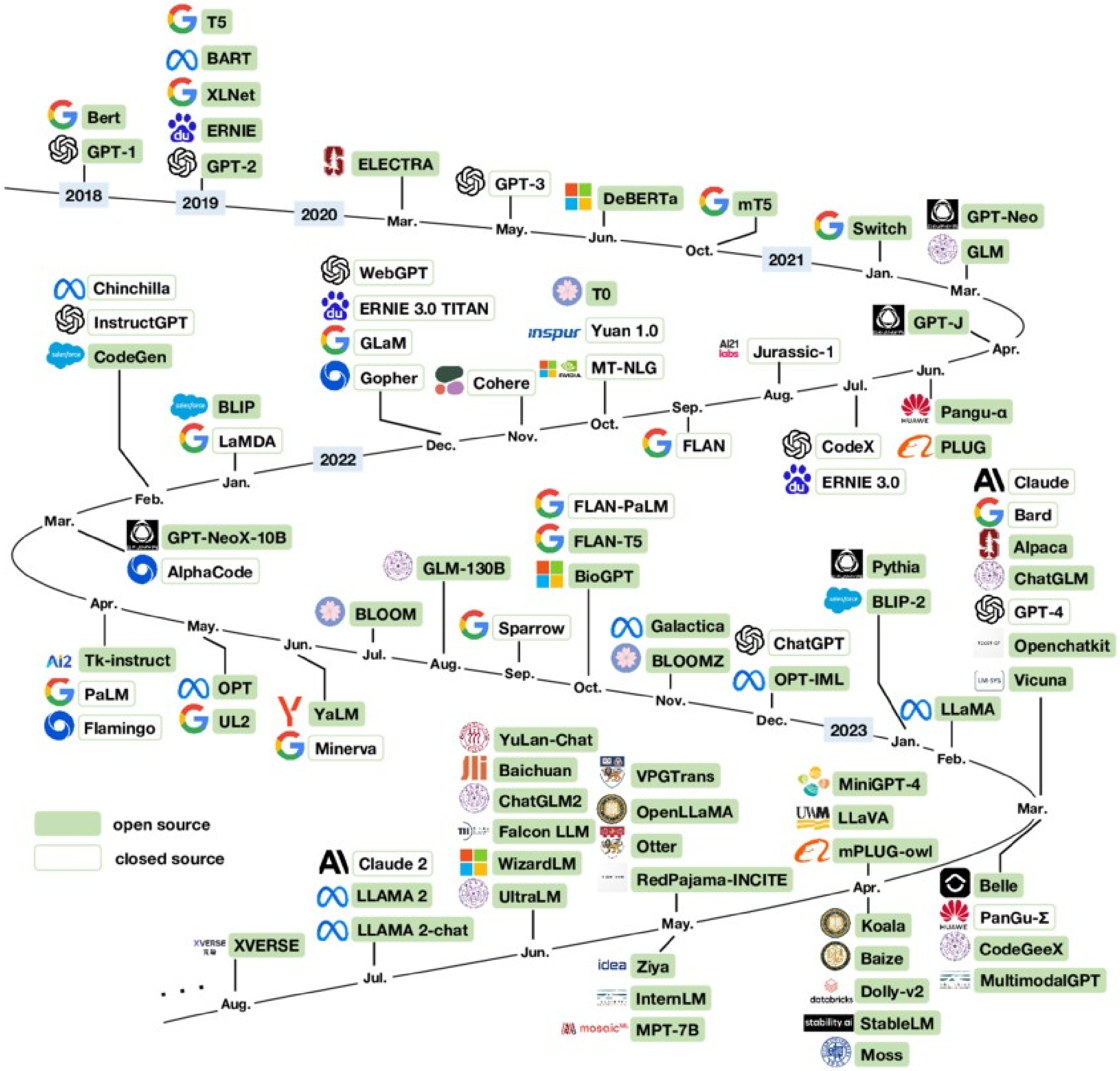


Image Credit: researchgate.net

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Scope: NLP Tasks

❑ Semantic Measurement

- Information, Syntax
- Meaning, Semantics
- Counts, Vectorization
- Complexity
- Similarity

❑ Labelling

- Classification or clustering
- Ranking
- Entity recognition
- Sentiment
- Emotions

❑ Language Generation

- Machine Translation
- Conversational AI
- Question Answering
- Summarization
- Images and descriptions

❑ Spoken Language

- Speech recognition
- Speech generation
- Voice Clone

Scope: NLP Tasks

Image by NLPlanet

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Scope: Data Preparation

Data Collection

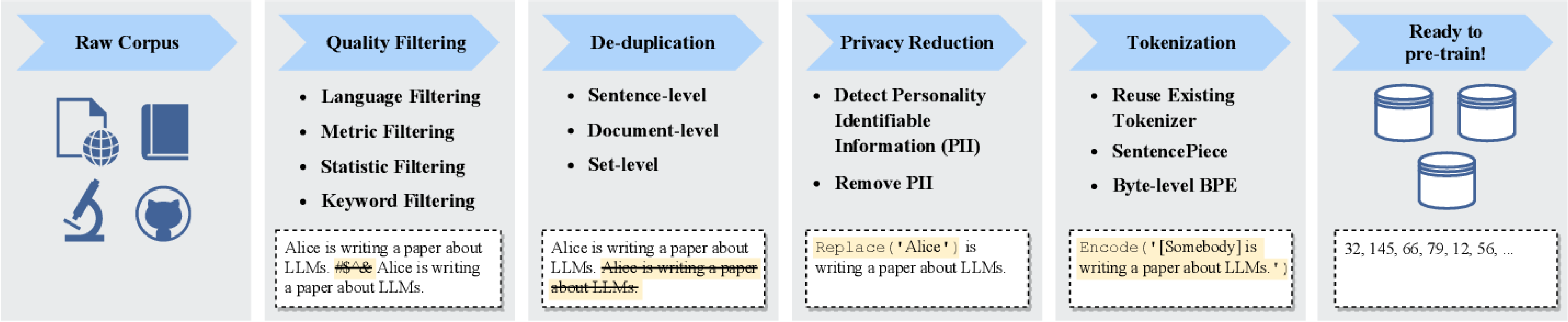
Attributes of Excellence	Indicators of Inferiority
Higher Quality	Lower Quality
Diversity	Homogeneity
Authenticity	Synthetic
More Quantity	Less Quantity

Data Source

General Data	Specialized Data
Web Pages	Multilingual Data
Books	Scientific Data
Conversations	Code
…	…

Scope: Data Preparation

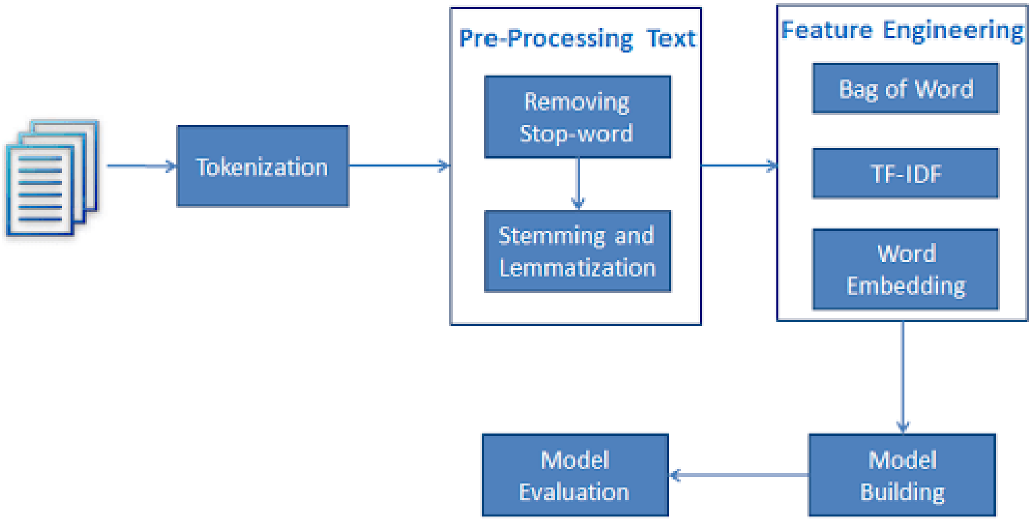
Data Preprocessing Pipeline



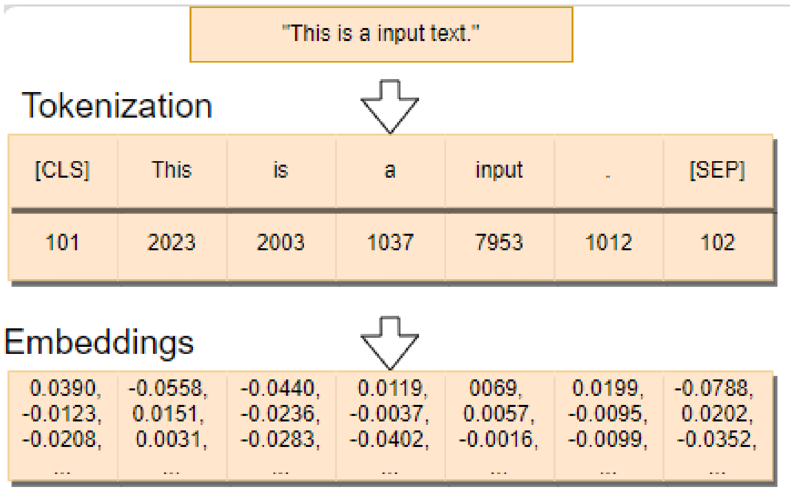
Zhao, Wayne Xin et al. “A Survey of Large Language Models.” ArXiv abs/2303.18223 (2023): n. pag.

Scope: Data Preparation

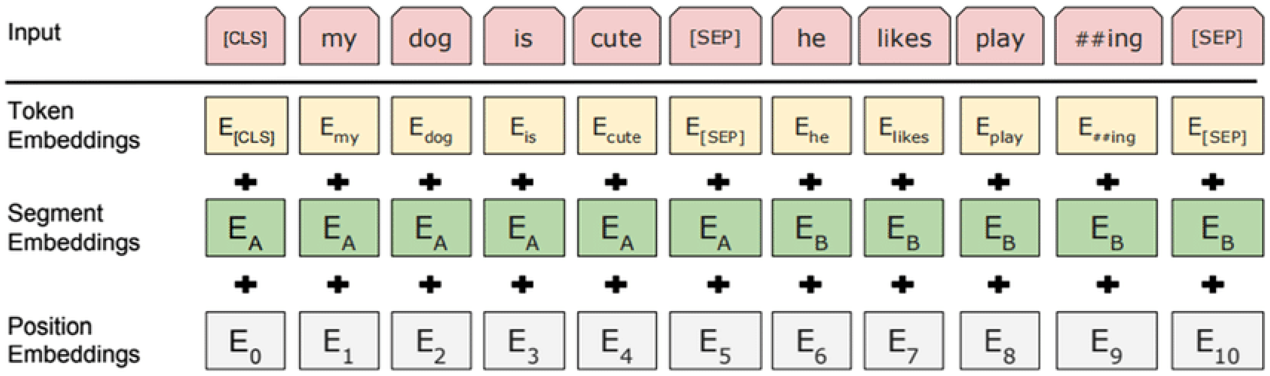
From Tokenization to Embedding



blogspot.com



vaclavkosar.com



Li, Shoubin & Wang, Qing. (2021). 10.1007/s10032-021-00381-5.

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task Data Preparation	Choose an existing model or pretrain your own	Prompt engineering Fine - tuning Align with human feedback	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications

Approaches Choosing

When you want to customize your LLM with data, what are all the options, and which method is the best (pre-train vs. prompt engineering vs. fine-tune vs. RAG)?

Method	Definition	Primary use case	Data requirements	Advantages	Considerations
Pre-training	Training an LLM from scratch	Unique tasks or domain-specific corpora	Large data sets (billions to trillions of tokens)	Maximum control, tailored for specific needs	Extremely resource-intensive
Prompt engineering	Crafting specialized prompts to guide LLM behavior	Quick, on-the-fly model guidance	None	Fast, cost-effective, no training required	Less control than fine-tuning
Fine-tuning	Adapting a pre-trained LLM to specific data sets or domains	Domain or task specialization	Thousands of domain-specific or instruction examples	Granular control, high specialization	Requires labeled data, computational cost
Retrieval augmented generation (RAG)	Combining an LLM with external knowledge retrieval	Dynamic data sets and external knowledge	External knowledge base or database (e.g., vector database)	Dynamically updated context, enhanced accuracy	Increases prompt length and inference computation

Approaches Choosing

When you want to customize your LLM with data, what are all the options, and which method is the best (pre-train vs. prompt engineering vs. fine-tune vs. RAG)?

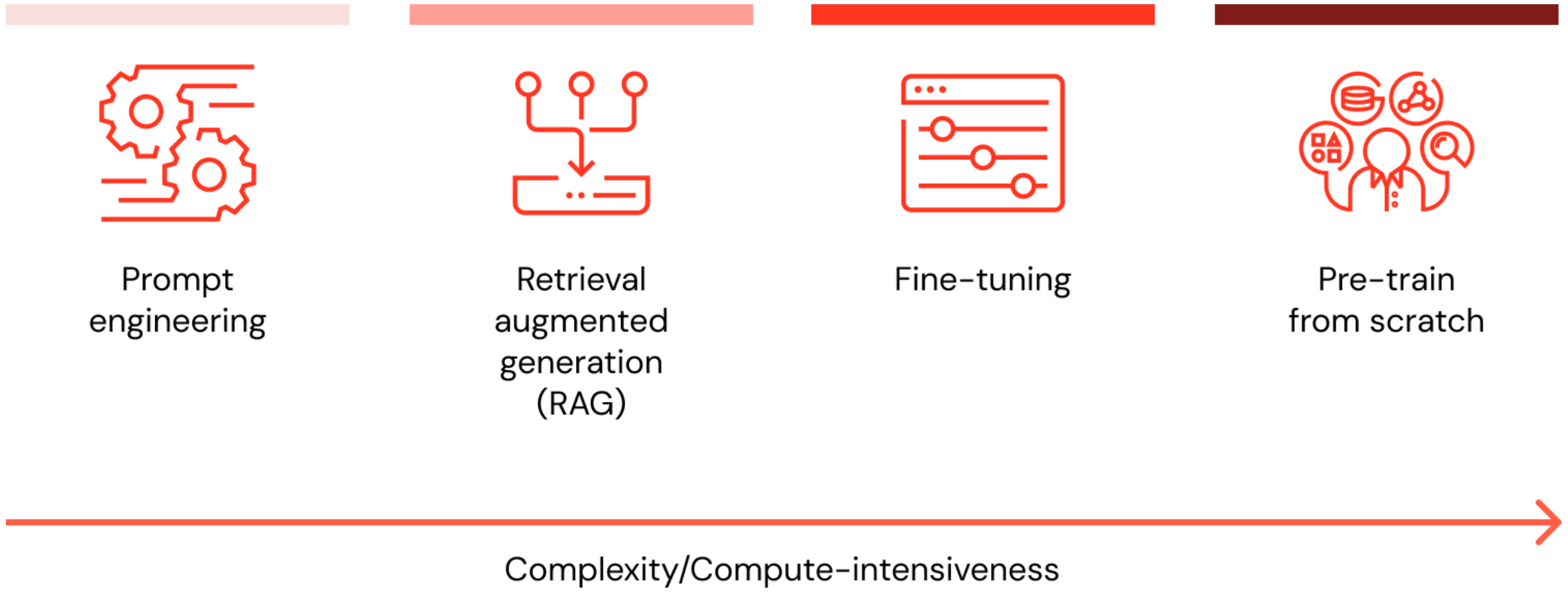


Image credit: Databricks.com

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Approaches Choosing: Pretraining

When should you pretrain an LLM?

Many teams are pretraining general-purpose LLMs may take $10s of millions, months, and a huge amount of data. Here are some scenarios when you should consider pretraining an LLM:

New Model Development	When developing a new language model from scratch, pretraining is essential to establish a broad understanding of language and context.
Domain-Specific Applications	If you have a large dataset in a specific domain (like legal, medical, financial, or technical texts) and need the model to understand and generate text in that domain.
Language and Localization	When you need the model to understand and generate text in a new language or dialect that wasn't well represented in the original training data.
Research and Innovation	In academic or research settings, pretraining might be done to experiment with new architectures, training techniques, or to explore the limits of language models.
Commercial Products and Services	For businesses that want to offer differentiated AI-powered products or services, custom pretraining can provide a competitive edge.

Pretraining an LLM is a resource-intensive process that requires significant computational power and expertise. It should be undertaken when the benefits of a more specialized or updated model outweigh the costs and effort involved.

Approaches Choosing: Pretraining

Model architectures and pre-training objectives

Model Type	Model architectures	Good use cases	Example
Autoencoding models Encoder-only LLM	→ Input → Encoder → Output →	• Sentiment analysis • Named entity recognition • Word classification	• BERT • ROBERT
Autoregressive models Decoder-only LLM	→ Input → Decoder → Output →	• Text generation • Other emergent behavior	• GPT • LLaMA
Sequence-to-sequence models Encoder-Decoder LLM	→ Input → Encoder ↓ → Input → Decoder → Output →	• Translation • Text summarization • Question answering	• T5 • BART

Approaches Choosing: Pretraining

DeepLeaning.AI

Scaling choices for pre-training

CONSTRAINT:
Compute budget
(GPUs, training time, cost)

Model performance (minimize loss)

SCALING CHOICE:
Dataset size
(number of tokens)

SCALING CHOICE:
Model size
(number of parameters)

Goal: maximize model performance

Approaches Choosing: Pretraining

Pre-training for domain adaptation

Domain	Model	Good use cases	Paper Link
Finance	**BloombergGPT:** A Large Language Model for Finance	• 50 billion parameters • 363 billion token dataset	arXiv:2303.17564
Medical Language	**GatorTron:** A Large Clinical Language Model to Unlock Patient Information from Unstructured Electronic Health Records	• 110 million parameters • >90 billion words of text	arXiv:2306.16092
Multilingual Language	**BLOOM:** A 176B-Parameter Open-Access Multilingual Language Model	• 176 billion parameters • 46 natural and 13 programming languages	arXiv:2203.03540

NLP Project Workflow

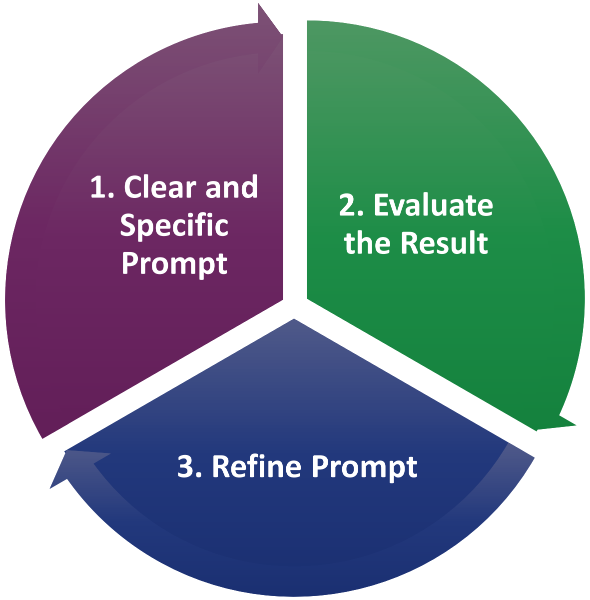
Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Approaches Choosing: Prompt Engineering

AI prompt engineering helps mold the model’s output, ensuring the artificial intelligence responds meaningfully and coherently.

Tips for prompting

- Be detailed and specific
- Guide the model to think through its answer
- Experiment and iterate



Advanced Prompt Techniques

0

Zero-shot prompting

- Provide the machine learning model with a task it hasn’t explicitly been trained on
- Test the model’s ability to produce relevant outputs without relying on prior examples

Few-shot prompting or in-context learning

- Give the model a few sample outputs (shots) to help it learn what the requestor wants it to do.

LangChain

Chain-of-thought prompting (CoT)

- Break down a complex task into intermediate steps, or “chains of reasoning”
- Help the model achieve better language understanding and create more accurate outputs

[Prompt Engineering with LLaMA-2](#) [Prompt Engineering with Llama 2 - DeepLearning.AI](#)

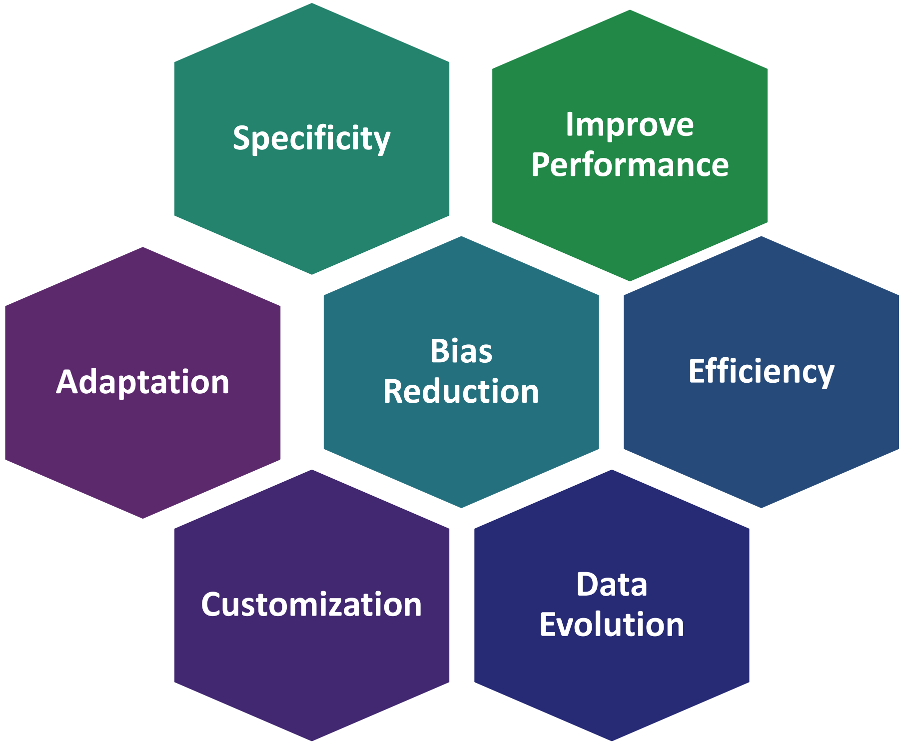
NLP Project Workflow

- **Scope**
  - Define the task
  - Data Preparation
- **Select**
  - Choose an existing model or pretrain your own
- **Adapt and align model**
  - Prompt engineering
  - Fine - tuning
  - Align with human feedback
  - Evaluate
- **Application integration**
  - Optimize and deploy model for inference
  - Augment model and build LLM-powered applications

Adapt and align model: Fine-tuning

Why fine-tune?

To carry out a task that isn’t easy to define in a prompt. It can reduce computation costs, your carbon footprint, and allows you to use state-of-the-art models without having to train one from scratch.



You can fine-tune a pretrained model with a deep learning framework of your choice:

Fine-tune a pretrained model with:

- **Transformers Trainer**
- **TensorFlow with Keras**
- **native PyTorch**
- **NVIDIA Nemo**
- **OpenAI**
- **…**

Different Large Languages Models

Model size?

- **1 Billion Parameters**: Capable of pattern matching and possess fundamental world knowledge.
- **10 Billion Parameters**: Have enhanced world knowledge and the ability to follow straightforward instructions.
- **100+ Billion Parameters**: Contain rich world knowledge and can engage in complex reasoning tasks.

Closed or open source?

Closed-Source Models via Cloud APIs:

- User-friendly for application integration.
- Access to larger and more powerful models.
- Cost-effective pricing.
- Potential for dependency on a single vendor's ecosystem.

Hugging Face models: Models - Hugging Face

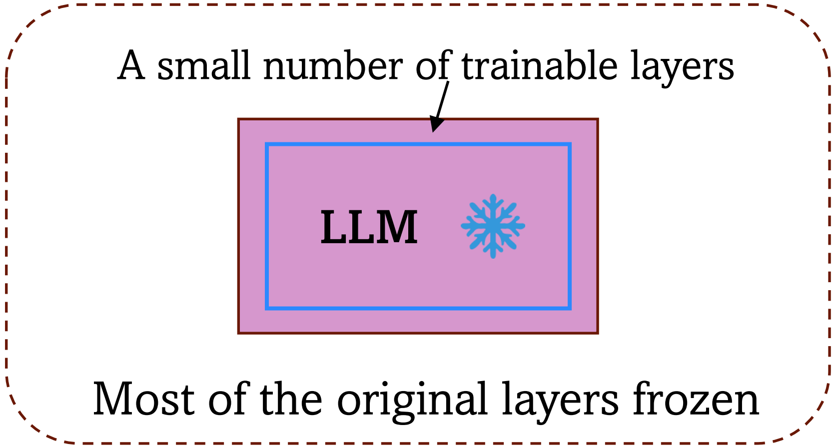
Open-Source Models:

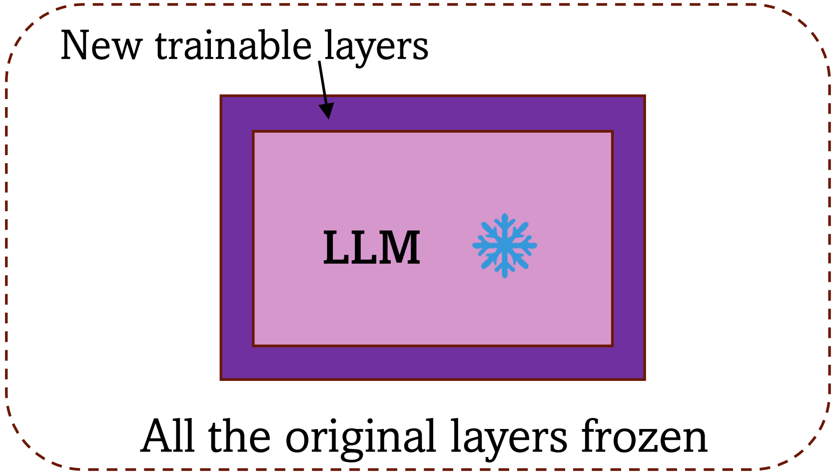
- Complete control over the model's configuration.
- Operable on personal devices (on-prem, PC, etc.)
- Full authority over data privacy and access management.

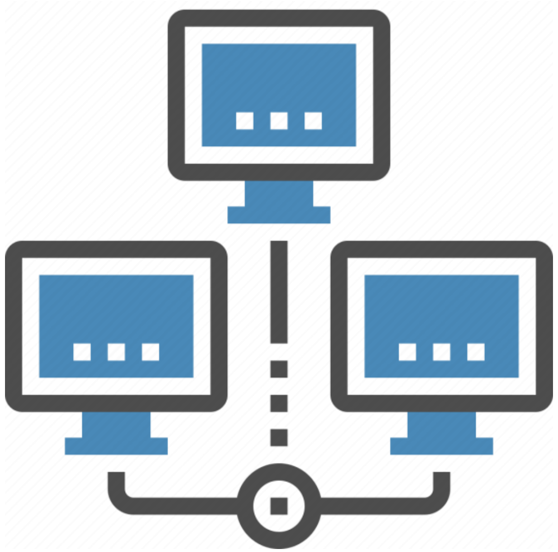
NVIDIA NGC: AI Models - Computer Vision, Conversational AI, and More | NVIDIA NGC

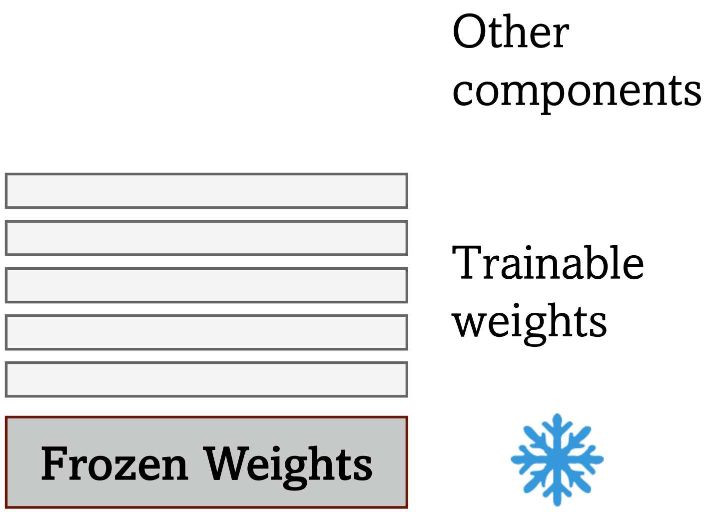
Adapt and align model: Fine-tuning

Parameter efficient fine-tuning (PEFT)









Adapt and align model: Fine-tuning

PEFT methods summary

Select a subset of initial LLM parameters to fine-tune.

Sparse

Selective methods

Reparametrize model weights using a low-rank representation.

LoRA

Reparameterization-based methods

Add trainable layers or parameters to the model.

Adapters

Parallel Adapters

Soft Prompts

Prompt Tuning

Prefix Tuning

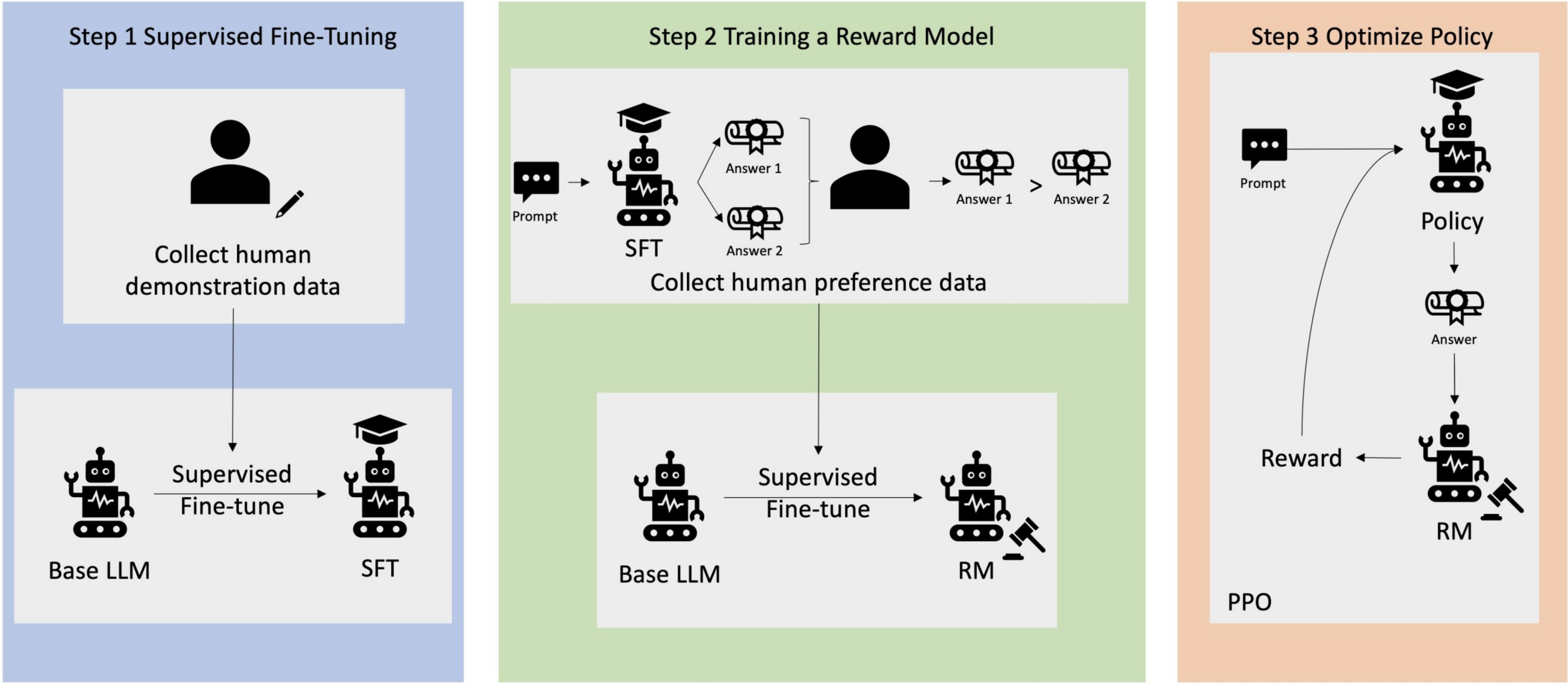
Additive methods

Lialin et al. 2023, Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning.

NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Adapt and align model: Reinforcement Learning from Human Feedback (RLHF)



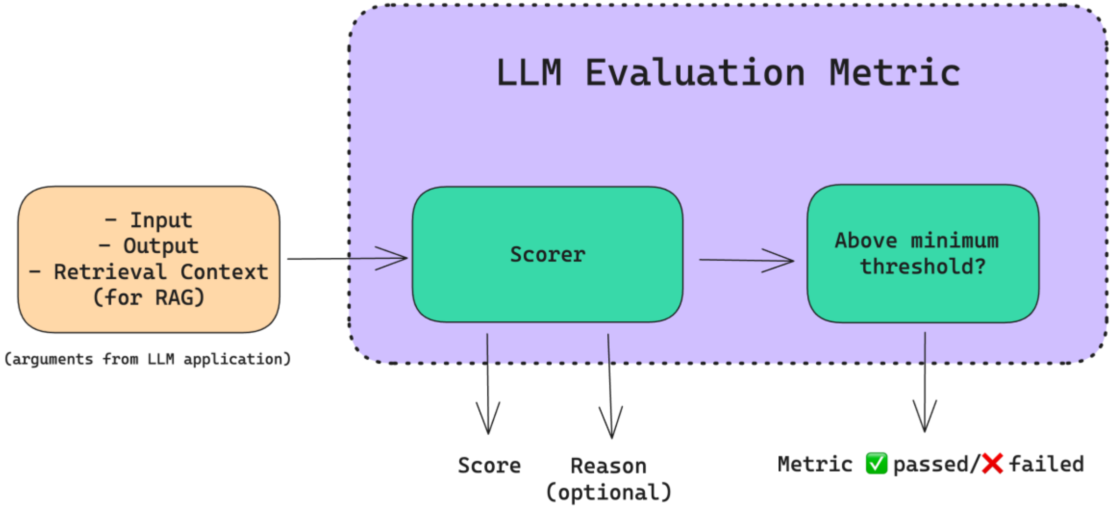
[AMAZON: AWS](#)

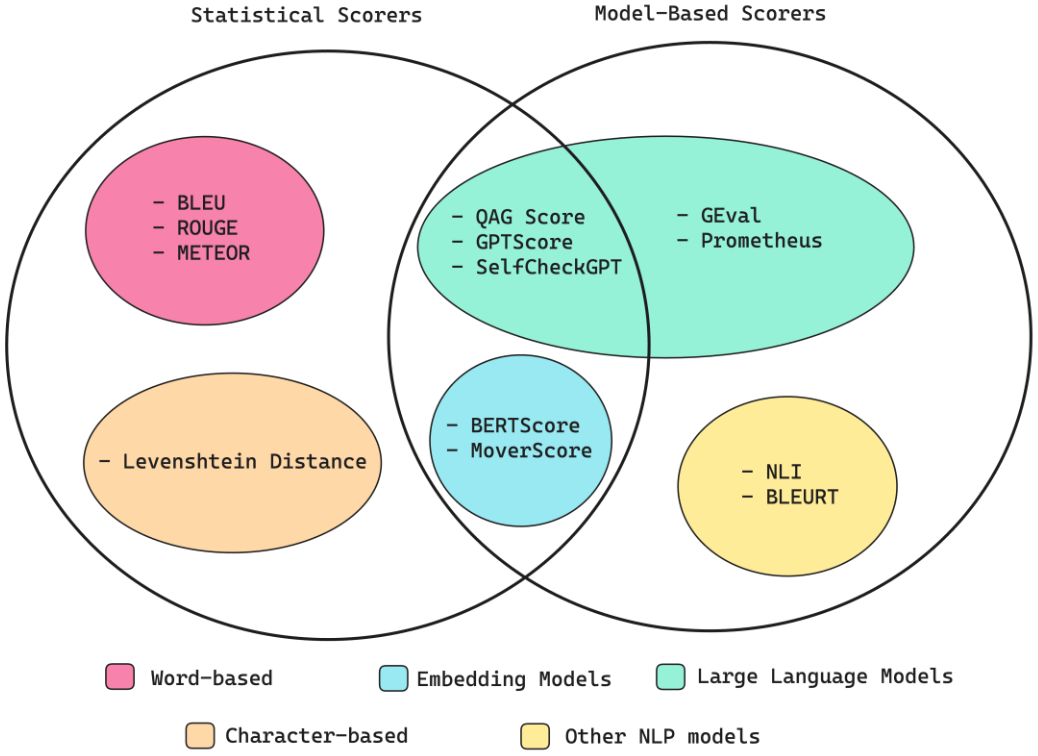
NLP Project Workflow

Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Adapt and align model: Evaluation

LLMs Evaluation - Metrics





[Confident AI](#)

Adapt and align model: Evaluation

LLMs Evaluation - Benchmarks

GLUE

SuperGLUE

HELM

GLUE Benchmark

SuperGLUE Benchmark

HELM (stanford.edu)

BIG-bench

google/BIG-bench

TruthfulQA Dataset | Papers With Code

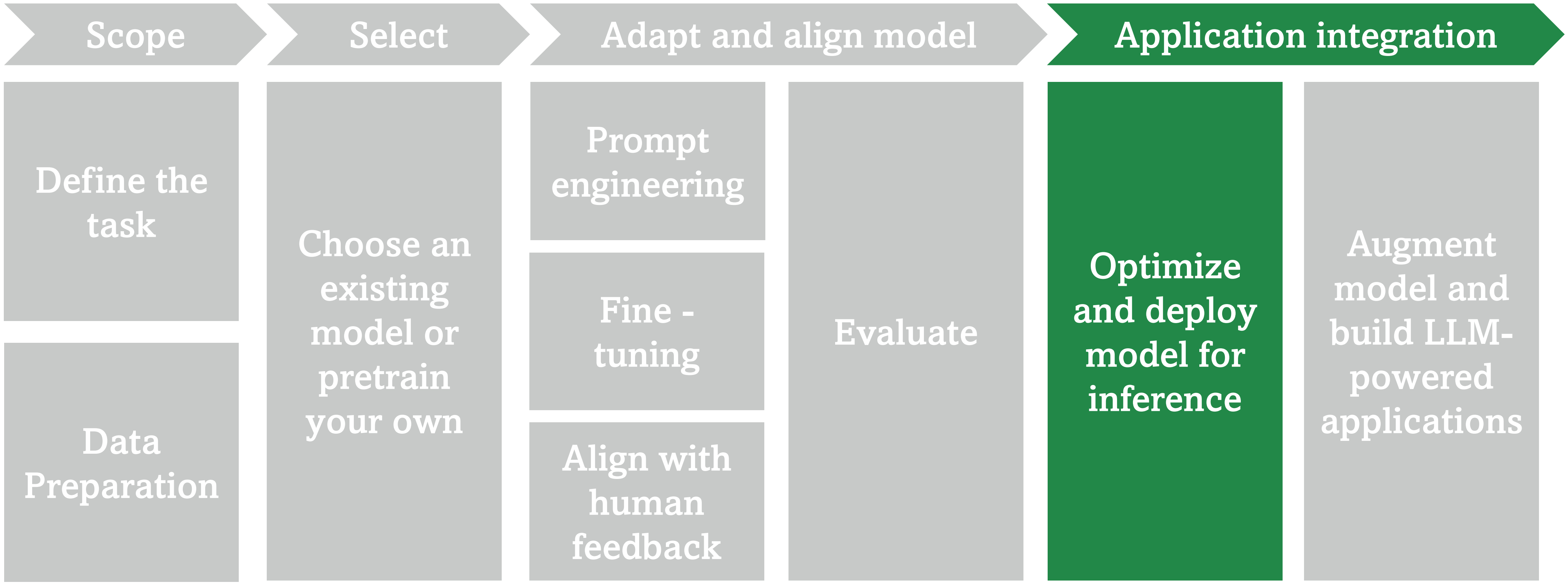
HumanEval Dataset | Papers With Code

MMLU Dataset | Papers With Code

HellaSwag Dataset | Papers With Code

Deepgram

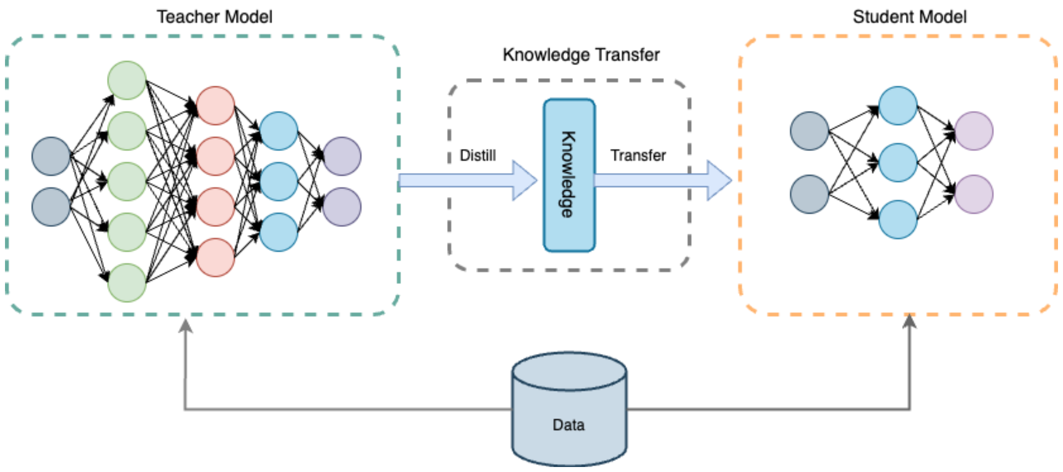
NLP Project Workflow



Application Integration

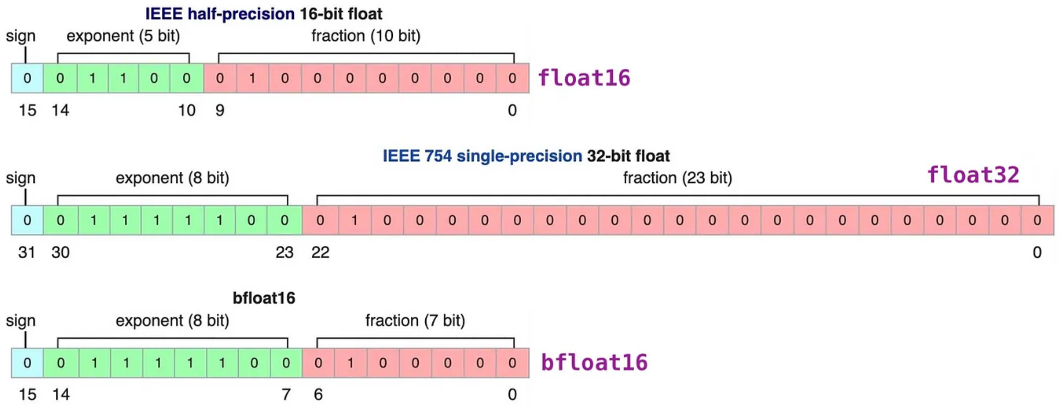
LLM optimization techniques

Distillation



arXiv:2006.05525

Quantization



Cerebras

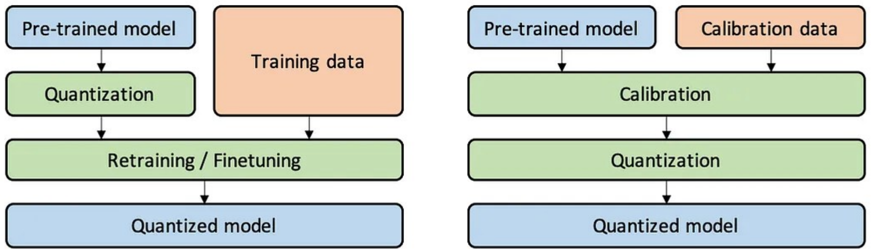


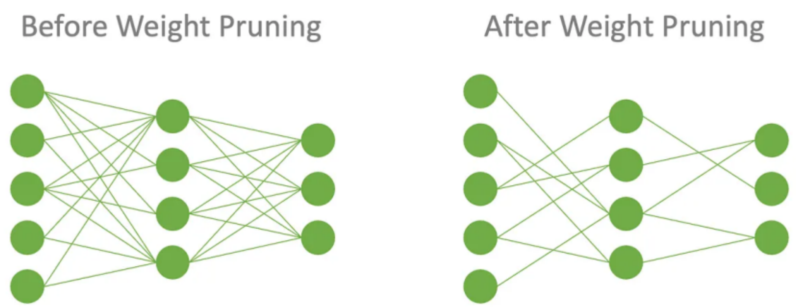
Figure 4: *Comparison between Quantization-Aware Training (QAT, Left) and Post-Training Quantization (PTQ, Right). In QAT, a pre-trained model is quantized and then finetuned using training data to adjust parameters and recover accuracy degradation. In PTQ, a pre-trained model is calibrated using calibration data (e.g., a small subset of training data) to compute the clipping ranges and the scaling factors. Then, the model is quantized based on the calibration result. Note that the calibration process is often conducted in parallel with the finetuning process for QAT.*

arxiv:2103.13630

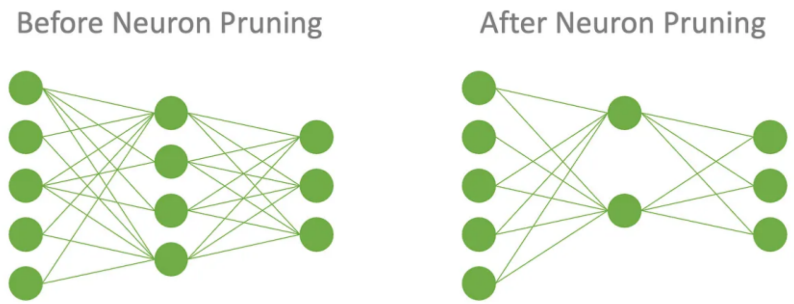
Application Integration

LLM optimization techniques: Pruning

Weight Pruning



Neuron Pruning



Structured vs Unstructured Model Pruning

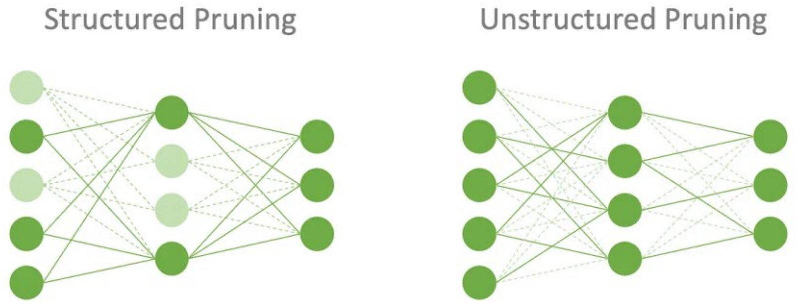


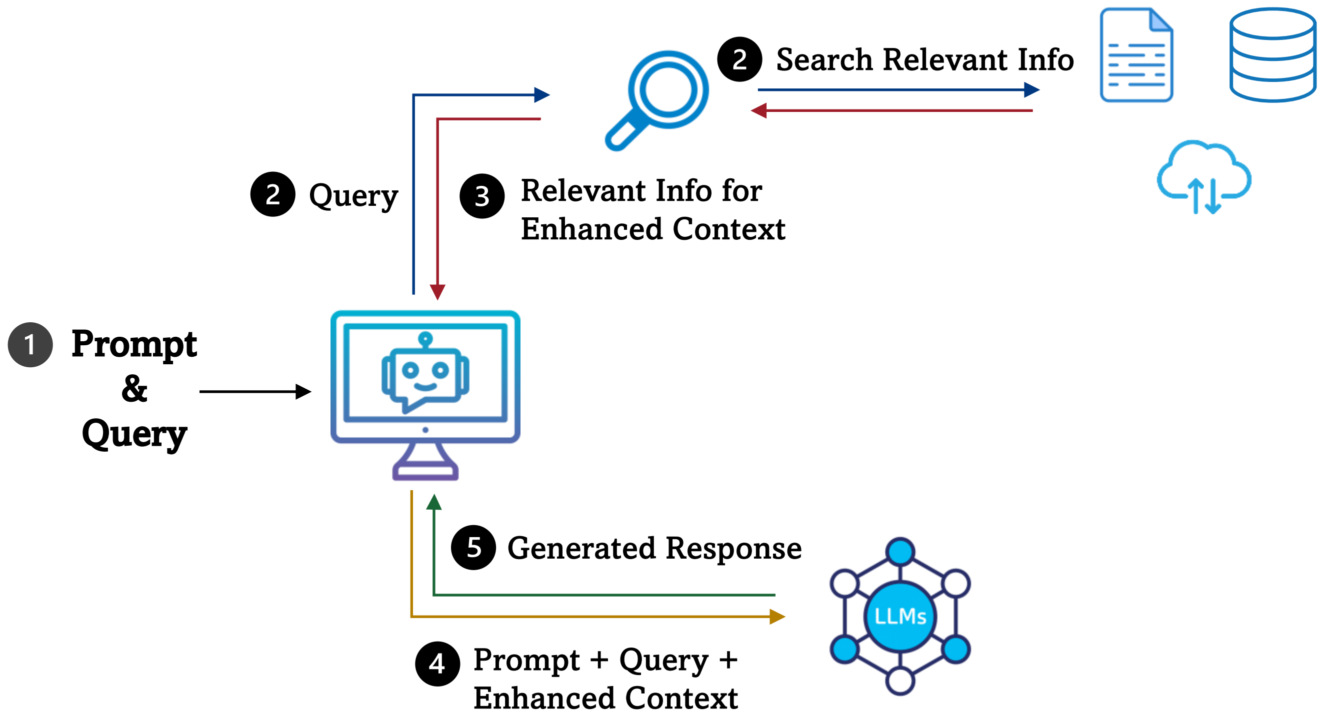
Image credit: Medium

NLP Project Workflow

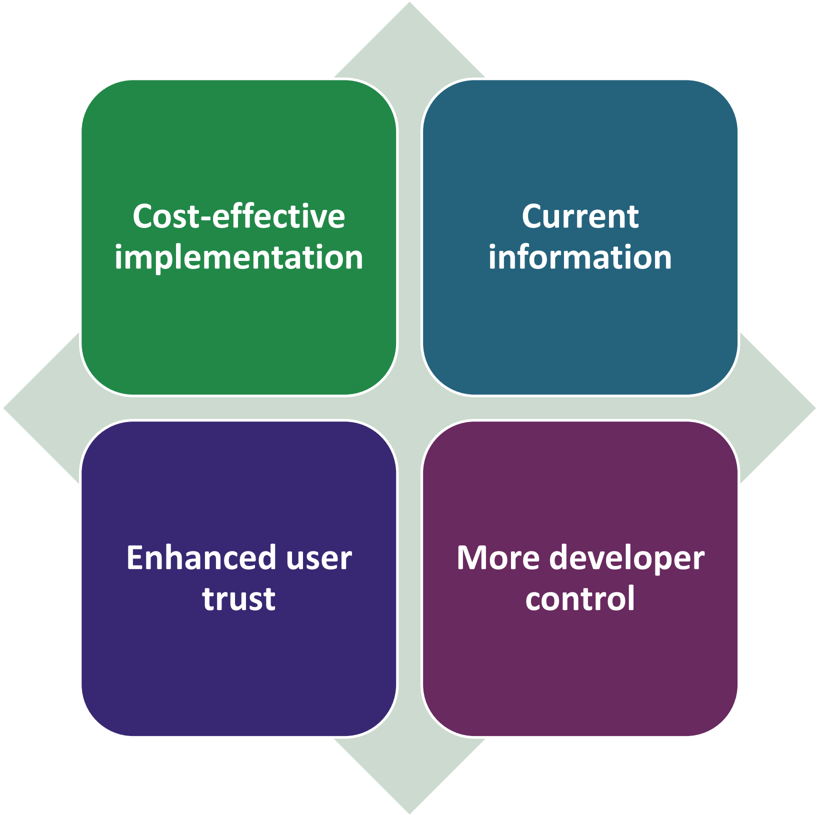
Scope	Select	Adapt and align model		Application integration	
Define the task	Choose an existing model or pretrain your own	Prompt engineering	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications
Data Preparation		Fine - tuning			
		Align with human feedback			

Adapt and align model: Retrieval Augmented Generation (RAG)

The conceptual flow of using RAG with LLMs



What are the benefits of RAG?



[AMAZON: AWS]

HiPerGator Resources: Software on HiPerGator

https://help.rc.ufl.edu/doc/NLP

Module load

pytorch, torchtext, nltk, Spacy, **transformers**, sentence-transformers, Flair, BERTopic for topic modeling, **nemo**, Spark-nlp, Parlai, sentencepiece, RAPIDSai for data processing and machine learning algorithms, gensim, scikit-learn, and more.

Jupyter Notebook kernel

nlp-1.1, nlp-1.2, nlp-1.3, nemo-1.2.0, nemo-1.22.0, LLAMA2, LLAMA2_HF

Note: For the kernel information, use the command conda activate /apps/kernel_name and
conda env export > kernel_name.yml

HiPerGator Resources: Model and Data

https://help.rc.ufl.edu/doc/NLP

Model

/data/ai/models/nlp

Data

/data/ai/ref-data/nlp

Benchmarks on HPG

/data/ai/benchmarks/nlp

HiPerGator Resources: Examples on HiPerGator

You can find a wide array of AI examples on our documentation website: https://help.rc.ufl.edu/doc/AI_Examples

Beyond our workshops and introductory sessions, we offer detailed tutorials located at `/data/ai/examples/nlp` on HiPerGator.

Several tutorials apply proprietary language models!

HiPerGator Resources: Open sources Framework

NVIDIA NeMo

- End-to-end solution for AI models
- Best-in-Class Pretrained Models
- Quickly train, customize, and deploy LLMs
- Optimized AI Inference with NVIDIA Triton
- Easy parallelism for large models

Hugging Face Transformers

- Lots of pre-trained models: Model Hub
- State of Art Performance
- Easy-to-Use APIs
- Task-Specific Pipelines
- Efficient Parallelization

NLP: Question Answering

[NVIDIA Nemo Tutorials]

This tutorial will demonstrate how to train, evaluate, and test three types of models for Question-Answering:

1. BERT-like models for Extractive Question-Answering
2. Sequence-to-Sequence (S2S) models for Generative Question-Answering
3. GPT-like models for Generative Question-Answering

[Question Answering — NVIDIA NeMo]

Hand-on Lab

You can follow the steps to complete your tutorial.

1. Set up your Jupyter Notebook environment on HiPerGator

- https://ood.rc.ufl.edu/pun/sys/dashboard

2. Download the tutorial

- git clone https://github.com/qianzmolloy/nemo_question_answering.git /blue/ai-workshop/username

3. Create symbolic link

- ln -s /blue/ai-workshop/username ai-workshop

4. Run the tutorial

- select the **nemo-1.22.0** kernel

Number of CPU cores requested per MPI task (--cpus-per-task, -p)

2

Number of cores per MPI task requested for application, (default = 1).

Maximum memory requested for this job in Gigabytes (--mem, -m)

20

Maximum amount of memory to be used by the job (blank/default = 600M per CPU). If you are using advanced memory options in **Additional SLURM Options** below, then leave this blank.

SLURM Account (--account, -A)

ai-workshop

Enter an alternative account if required. (default = primary investor)

QoS (Required if custom Account is set, --qos, -q)

ai-workshop

Enter an alternative QoS, **required if alternative account is entered above. Please note, if you use the burst qos (-b), your jobs may take longer to start.** (default = primary investor)

Time Requested for this job in hours (--time, -t).

2

Time in hours requested from SLURM for this job to run. (default = 1) **Please note, interactive gpu partition jobs will be limited to 72 hours.**

Cluster partition (--partition, -p)

gpu

Select a specific cluster partition for job. (default = first available compute partition)

Generic Resource Request (--gres).

gpu:1

This is the Generic resource request string to request GPU resources. See also

Other Tutorials or Courses

1. NVIDIA DLI Courses:
    - Introduction to Transformer-Based Natural Language Processing
    - Prompt Engineering with LLaMA-2
    - Augment your LLM Using Retrieval Augmented Generation
    - …
2. Transformers Tutorials 🤗
3. DeepLearning.AI Short and Specializations Courses:
    - Generative AI with LLMs - DeepLearning.AI
    - ChatGPT Prompt Engineering for Developers
    - LangChain for LLM Application Development
    - Finetuning Large Language Models
    - Advanced Retrieval for AI with Chroma
    - …

Contact us

Ying Zhang	AI Support Manager	yingz@ufl.edu
Yunchao Yang	AI Support	yunchaoyang@ufl.edu
Qian Zhao	AI Support	zhao.qian@ufl.edu
Yang Hong	AI Support	yanghong@ufl.edu
Ian Lutticken	Training	i.lutticken@ufl.edu