**EmoCare - An Empathetic AI Workshop on Fine-tuning of Llama 2 (Large Language Model) for Sentiment Analysis of Cancer Survivors & Caregivers and Recommendation Systems in Healthcare**

The title "EmoCare - An Empathetic AI Workshop on Fine-tuning of Llama 2 (Large Language Model) for Sentiment Analysis of Cancer Survivors & Caregivers and Recommendation Systems in Healthcare" can be broken down into several components:

1. \*\*EmoCare:\*\*

- \*\*Emo:\*\* Short for emotion, indicating a focus on understanding and analyzing emotions.

- \*\*Care:\*\* Implies a sense of concern, empathy, and attention, suggesting a caring approach in the application of artificial intelligence.

2. \*\*An Empathetic AI Workshop:\*\*

- Indicates that the event or activity is a workshop focused on AI (Artificial Intelligence) that is designed to be empathetic.

- \*\*Empathetic AI:\*\* Suggests that the AI system being discussed or developed is designed to understand and respond to human emotions, providing a more compassionate and human-like interaction.

3. \*\*Fine-tuning of Llama 2 (Large Language Model):\*\*

- \*\*Fine-tuning:\*\* Refers to the process of adjusting and optimizing a pre-trained model for a specific task or domain. In this context, it implies the customization of a large language model.

- \*\*Llama 2 (Large Language Model):\*\* Refers to the specific large language model being used, named "Llama 2." Large language models are sophisticated AI models capable of understanding and generating human-like text.

4. \*\*for Sentiment Analysis of Cancer Survivors & Caregivers:\*\*

- \*\*Sentiment Analysis:\*\* Refers to the task of determining the sentiment or emotional tone expressed in a piece of text. In this case, the sentiment analysis is focused on understanding the emotions of cancer survivors and their caregivers.

- \*\*Cancer Survivors & Caregivers:\*\* Specifies the target audience or data group for the sentiment analysis, indicating that the emotional responses of individuals affected by cancer (both survivors and caregivers) are of interest.

5. \*\*and Recommendation Systems in Healthcare:\*\*

- \*\*Recommendation Systems:\*\* Refers to AI systems designed to suggest or recommend items, services, or actions based on user preferences and behavior. In this context, it suggests the incorporation of recommendation systems in the healthcare domain.

- \*\*Healthcare:\*\* Indicates the broader domain or industry focus, suggesting that the AI applications discussed or developed in the workshop have relevance and applications in healthcare.

The title conveys that the workshop, EmoCare, is centered around developing and understanding empathetic AI. The primary focus is on fine-tuning the Llama 2 large language model for sentiment analysis, particularly for the emotions of cancer survivors and their caregivers. Additionally, the workshop addresses the implementation of recommendation systems within the healthcare sector, indicating a broader scope of AI applications.

**Dataset:**

Link to the dataset: <https://data.mendeley.com/datasets/69dcnv2gzd/1>

\*\*Dataset Title:\*\* Mental Health Insights: Vulnerable Cancer Survivors & Caregivers Data

\*\*Publication Date:\*\* 21 November 2023

\*\*Version:\*\* 1

\*\*DOI (Digital Object Identifier):\*\* 10.17632/69dcnv2gzd.1

\*\*Contributors:\*\*

- Irin Hoque Orchi

- Nafisa Tabassum

- Jaeemul Hossain

- Sabrina Tajrin

- Iftekhar Alam

\*\*Description:\*\*

The dataset is named "Mental Health Insights: Vulnerable Cancer Survivors & Caregivers Data" and was published on November 21, 2023. It provides valuable insights into the mental health aspects of cancer patients and their caregivers. The dataset was collected from various platforms such as Reddit, Daily Strength, and the Health Board, comprising a total of over 10,087 posts. The posts are related to five types of cancer: brain, colon, liver, leukemia, and lung cancer.

The dataset was analyzed by two team members who scored each post based on the emotions expressed, utilizing a scale from -2 to 1. The scoring system is designed to capture a spectrum of emotions. Negative scores (-1 or -2) were assigned to posts expressing grief or suffering, positive scores (1) were given for posts with happy emotions like relief or accomplishment, and posts with no discernible emotion received a score of 0, categorizing them as neutral.

The primary objective of this analysis is to understand the emotional aspects conveyed in the posts of cancer patients, contributing to a mental health study. The dataset is provided in CSV format and can be downloaded for further research and analysis.

\*\*Dataset Files:\*\*

- \*\*CSV:\*\* Mental Health Dataset.csv (File size: 11.8 MB)

\*\*Institutions:\*\*

- BRAC University

\*\*Categories:\*\*

- English Language

- Natural Language Processing

\*\*License:\*\*

- Creative Commons Attribution 4.0 International License (CC BY 4.0)

\*\*Dataset Metrics:\*\*

No metrics are available.

\*\*Latest Version:\*\*

- Version 1

- Published: 21 Nov 2023

- DOI: 10.17632/69dcnv2gzd.1

\*\*Citation:\*\*

Orchi, Irin Hoque; Tabassum, Nafisa; Hossain, Jaeemul; Tajrin, Sabrina; Alam, Iftekhar (2023), “Mental Health Insights: Vulnerable Cancer Survivors & Caregivers Data”, Mendeley Data, V1, doi: 10.17632/69dcnv2gzd.1

\*\*License Information:\*\*

The dataset is licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0). Users are encouraged to learn more about the license for proper usage.

The provided information about the dataset doesn't explicitly specify the features, but based on the context, it seems like there are three main features:

**Dataset Description:**

1. \*\*Posts:\*\*

- This feature represents the textual content of the posts made by cancer patients and their caregivers.

- Each entry in this feature contains the actual text of the posts where individuals share their experiences, emotions, or information related to cancer.

2. \*\*Predicted:\*\*

- This feature likely contains the predicted sentiment or emotion labels assigned to each post.

- The predictions could be based on a model or human annotators who have assessed the emotional tone or sentiment expressed in the posts.

- Possible values might include categories such as 'positive', 'neutral', 'negative', or others that reflect the emotional state of the individuals in the posts.

3. \*\*Intensity:\*\*

- This feature seems to represent the intensity or strength of the predicted sentiment or emotion in each post.

- It could be a numerical value indicating the degree of positivity, neutrality, or negativity in the posts.

- Higher intensity values might suggest stronger emotions or sentiments expressed in the corresponding posts.

**Large language models (LLMs)** are powerful machine-learning models that can understand and generate natural language. They are trained on massive datasets of text and code, which allows them to learn the patterns and relationships that exist in language. LLMs can be used for a variety of tasks, such as translating languages, analyzing sentiment, and generating creative text formats.

Here are some specific examples of what LLMs can do:

* **Translate languages:** LLMs can be used to translate text from one language to another. For example, an LLM could be used to translate a French news article into English.
* **Analyze sentiment:** LLMs can be used to analyze the sentiment of the text. For example, an LLM could be used to determine whether a customer review is positive or negative.
* **Generate creative text formats:** LLMs can be used to generate creative text formats, such as poems, code, scripts, musical pieces, emails, letters, etc. For example, an LLM could be used to generate a poem about a particular topic.

**What is LLM?**

**A large language model (LLM) is a type of artificial intelligence (AI) that can generate human-like text and perform various natural language processing tasks**. LLMs are trained on massive amounts of text data, which allows them to learn the patterns and relationships that exist in language. This makes them capable of generating text that is often indistinguishable from that written by humans.

Here are some real-world examples of LLMs in action:

* **Google Translate:** Google Translate uses an LLM to translate text from one language to another. The LLM is trained on a massive dataset of text and code, which allows it to learn the patterns and relationships that exist between different languages. This makes Google Translate one of the most accurate translation tools available.
* **GPT-3**: GPT-3 is a large language model developed by OpenAI. GPT-3 is capable of generating human-like text, translating languages, and writing different kinds of creative content. It has been used to create a variety of applications, including chatbots, text generators, and creative writing tools.
* **Bard:** Bard is a large language model developed by Google AI. Bard is capable of answering your questions in an informative way, even if they are open-ended, challenging, or strange. It can also generate different creative text formats of text content, like poems, code, scripts, musical pieces, emails, letters, etc.

**How large language model (LLM) is built?**

Large language models (LLMs) are trained on massive amounts of text data. This data is used to teach the model the statistical relationships between words, phrases, and sentences. This allows the model to generate coherent and contextually relevant responses to prompts or queries.

LLMs are typically too large to run on a single computer, so they are provided as a service over an API or web interface. This means that you can access the model’s capabilities without having to download or install it yourself.

One example of an LLM is ChatGPT’s GPT-3 model. GPT-3 was trained on massive amounts of internet text data, which gave it the ability to understand various languages and possess knowledge of diverse topics. As a result, it can produce text in multiple styles.

While GPT-3’s capabilities may seem impressive, they are not surprising. This is because LLMs operate using special “grammar” that matches up with prompts. These grammars tell the model how to generate text that is relevant to the prompt.

**For example**, if you prompt GPT-3 to “***write a poem about love,***” the model will use its grammar to generate text that is relevant to the prompt. The text will likely be in the form of a poem, and it will likely be about love.

**General Architecture :**

The architecture of large language models (LLMs) is composed of multiple layers of neural networks. These layers include ***embedding layers, recurrent layers, feedforward layers, and attention layers***. Each layer helps the model process the input text and generate output predictions.

The***embedding layer***converts each word in the input text into a high-dimensional vector representation. This representation captures semantic and syntactic information about the word, which helps the model understand the context.

The***feedforward layers*** apply nonlinear transformations to the input embeddings. This helps the model learn higher-level abstractions from the input text.

The***recurrent layers*** interpret information from the input text in sequence. They maintain a hidden state that is updated at each time step, allowing the model to capture the dependencies between words in a sentence.

The ***attention mechanism***allows the model to focus selectively on different parts of the input text. This helps the model attend to the input text’s most relevant parts and generate more accurate predictions.

**In summary**, the architecture of LLMs is designed to process the input text in a way that captures the meaning of the text and the relationships between words. This allows the model to generate accurate predictions.

**Example :**Here are some examples of popular large language models (LLMs):

* **GPT-3:** Developed by OpenAI, GPT-3 is one of the largest LLMs with 175 billion parameters. It can perform many tasks, including text generation, translation, and summarization.
* **BERT:** Developed by Google, BERT is another popular LLM that has been trained on a massive corpus of text data. It can understand the context of a sentence and generate meaningful responses to questions.
* **XLNet:**Developed by Carnegie Mellon University and Google, XLNet uses a novel approach to language modeling called “permutation language modeling.” It has achieved state-of-the-art performance on language tasks, including language generation and question answering.
* **T5:**Developed by Google, T5 is trained on a variety of language tasks and can perform text-to-text transformations, like translating text to another language, creating a summary, and question answering.
* **RoBERTa:** Developed by Facebook AI Research, RoBERTa is an improved BERT version that performs better on several language tasks.

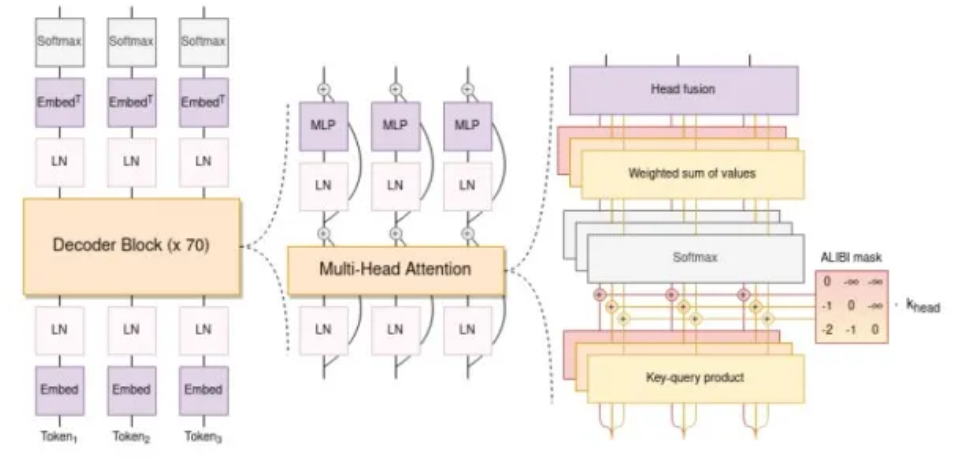
**Open Source Language Model :**

Open-source LLMs have made it possible for researchers, developers, and businesses to build applications that leverage the power of these models. Bloom is an example of an open-source LLM that has been trained on a massive dataset of text data. It has 176 billion parameters, making it larger than OpenAI’s GPT-3. Bloom can generate text in **46**natural languages and 13 programming languages. It is a powerful tool that can be used for a variety of applications, including text generation, translation, and question-answering.

Here are some specific examples of how Bloom can be used:

* **Text generation:** Bloom can be used to generate text in a variety of styles, including creative writing, code, and scripts.
* **Translation:**Bloom can be used to translate text from one language to another.
* **Question answering:**Bloom can be used to answer questions about a variety of topics.

**Bloom Architecture:**



Bloom’s architecture is similar to GPT-3, but it has been trained in a wider range of languages. It consists of a decoder-only architecture with several embedding layers and multi-headed attention layers. This architecture makes it well-suited for training in multiple languages and allows users to translate and talk about a topic in a different language.

Here are some examples of how *Bloom’s* architecture can be used:

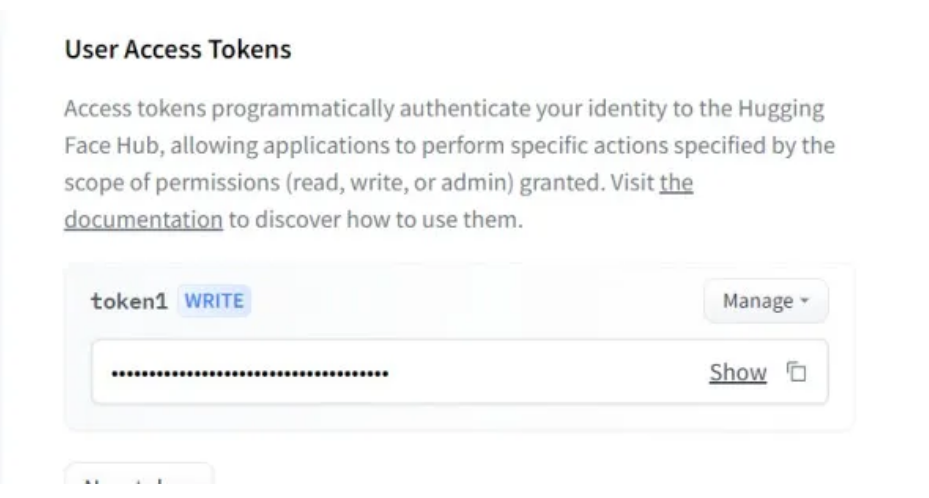
* **Translation:**Bloom can be used to translate text from one language to another. For example, if you wanted to translate the sentence ***“I love you”*** from English to French, you could use Bloom to generate the French sentence ***“Je t’aime.”***
* **Multilingual conversation:**Bloom can be used to have a conversation in multiple languages. For example, if you were talking to someone who spoke French and English, you could use Bloom to generate text in both languages.

**Other LLMs**

We can utilize the APIs connected to pre-trained models of many of the widely available LLMs through Hugging Face

**Hugging Face API**

Let’s look into how Hugging Face APIs can help generate text using LLMs like Bloom, Roberta-base, etc. First, we need to sign up for Hugging Face and copy the token for API access. After signup, hover over to the profile icon on the top right, click on settings, and then Access Tokens.



**Future Implications of LLM :**

LLMs, such as GPT-3, are a breakthrough in AI. However, there are concerns about their impact on job markets, communication, and society. One concern is that LLMs could automate jobs, leading to job losses. However, LLMs could also create new jobs and help businesses and governments make better decisions. Overall, LLMs have the potential to have a significant impact on society. It is important to carefully consider the potential risks and benefits of these models before they are widely adopted.

Here are some specific examples of how LLMs could impact society:

* **Education:**LLMs could be used to create personalized education plans for students, tailored to their individual needs and learning styles. This could help students to learn more effectively and achieve their full potential.
* **Healthcare:**LLMs could be used to create personalized healthcare plans for patients, based on their individual medical history and needs. This could help patients to receive the best possible care.
* **Business:**LLMs could be used to help businesses make better decisions by analyzing large amounts of data and generating insights. This could help businesses to improve their products and services and to make more informed decisions about their operations.

**Conclusion :**

Large Language Models (LLMs) have revolutionized the field of natural language processing, allowing for new advancements in text generation and understanding. LLMs can learn from big data, understand its context and entities, and answer user queries. This makes them a great alternative for regular usage in various tasks in several industries. However, there are concerns about the ethical implications and potential biases associated with these models. It is important to approach LLMs with a critical eye and evaluate their impact on society. With careful use and continued development, LLMs have the potential to bring about positive changes in many domains, but we should be aware of their limitations and ethical implications.

Certainly! Here are some examples of different types of large language models (LLMs) with examples:

1. Autoregressive:

* Example: GPT-3 (Generative Pre-trained Transformer 3) by OpenAI is an autoregressive LLM capable of generating human-quality text in response to a wide range of prompts and questions. It can also translate languages, write different kinds of creative content, and answer your questions in an informative way.
* Example: LaMDA (Language Model for Dialogue Applications) by Google AI is another autoregressive LLM designed specifically for generating realistic and engaging dialogue. It can hold conversations on a variety of topics and adapt to different conversation styles.

2. Transformer-based:

* Example: T5 (Text-to-Text Transfer Transformer) by Google AI is a transformer-based LLM that can be fine-tuned for a variety of tasks, including text summarization, machine translation, and question answering. It has been shown to achieve state-of-the-art results on many NLP benchmarks.
* Example: BART (Bidirectional and Attentional Transformer) by Facebook AI is another transformer-based LLM designed for text summarization and machine translation. It leverages a novel pre-training objective that allows it to learn from both the source and target languages.

3. Encoder-decoder:

* Example: Transformer by Google AI is an encoder-decoder LLM that has been shown to be highly effective for machine translation and text summarization. It uses a novel attention mechanism to capture long-range dependencies in text, which allows it to generate more accurate and fluent translations.
* Example: BERT (Bidirectional Encoder Representations from Transformers) by Google AI is another encoder-decoder LLM that has been shown to be effective for a variety of NLP tasks, including sentiment analysis and question answering. It uses a pre-training objective that allows it to learn contextual representations of words, which helps it to better understand the meaning of text.

4. Conditional LLMs:

* Example: ChatGPT by OpenAI is a conditional LLM that can be used for a variety of tasks, including chatbots, virtual assistants, and creative writing prompts. It is trained on a massive dataset of text and code, which allows it to generate realistic and engaging responses.
* Example: DialoGPT by Microsoft Research is another conditional LLM designed specifically for dialogue generation. It is trained on a dataset of human conversations, which allows it to generate more natural and engaging dialogue.

5. Generative LLMs:

* Example: MuseNet by OpenAI is a generative LLM that can be used to create musical pieces in a variety of styles. It uses a novel deep learning architecture that allows it to learn from a large dataset of musical scores.
* Example: Jukebox by OpenAI is another generative LLM that can be used to create a wide range of creative text formats, including poems, code, scripts, musical pieces, email, letters, etc. It uses a novel pre-training objective that allows it to learn from a variety of data sources.

6. Multimodal LLMs:

* Example: VQ-VAE (Vector Quantized Variational Autoencoder) by Google AI is a multimodal LLM that can be used to generate images from text descriptions. It uses a novel architecture that allows it to learn from both the text and the image data.
* Example: DALL-E 2 by OpenAI is another multimodal LLM that can be used to generate images from text descriptions. It uses a novel diffusion model architecture that allows it to generate more realistic and detailed images.

These are just a few examples of the different types of large language models that are currently available. As the field of NLP continues to develop, we can expect to see even more innovative and powerful LLMs emerge in the future.

Fine-tuning of Large Language Models (LLMs) refers to the process of taking a pre-trained language model and further training it on a specific task or domain to improve its performance on that task. Large Language Models, such as GPT (Generative Pre-trained Transformer) models, are pre-trained on massive datasets to learn general language patterns and knowledge.

The fine-tuning process involves:

1. \*\*Pre-training:\*\* Initially, the LLM is trained on a large and diverse dataset with a self-supervised learning approach. The model learns to predict the next word in a sentence or fill in masked words, gaining a broad understanding of language.

2. \*\*Task-Specific Training:\*\* After pre-training, the model is fine-tuned on a smaller dataset related to a specific task. This dataset is labeled with examples of the desired task (e.g., sentiment analysis, question answering, text summarization). The model's parameters are adjusted during this phase to make it more proficient in the target task.

In the context of your project, the fine-tuning of LLMs is likely applied to sentiment analysis of posts from cancer survivors and caregivers. The pre-trained LLM, such as GPT, is fine-tuned on a dataset containing examples of sentiments expressed in posts related to cancer. This process enables the model to specialize in understanding and predicting sentiments specific to the experiences of cancer survivors and caregivers.

Fine-tuning is valuable because it leverages the knowledge and language understanding gained during pre-training and tailors it to a specific application or domain. It allows the model to adapt to the nuances and context of the target task, resulting in improved performance and relevance for the particular use case.

**What is finetuning of LLM?**

As LLMs are pre-trained on large amount of generic text data, which helps it to learn general language understanding, grammar, and context .Fine-tuning leverages this general knowledge and refines the model to achieve better performance and understanding in a specific domain .The process usually entails training the model further on a smaller, targeted [dataset](https://research.aimultiple.com/data-quality-ai/) that is relevant to the desired task or subject matter.

For example, you’re using an LLM for a telco domain task and its training data did not contain any telecom data, then you need to finetune this existing model using your own small subset of telco domain data.

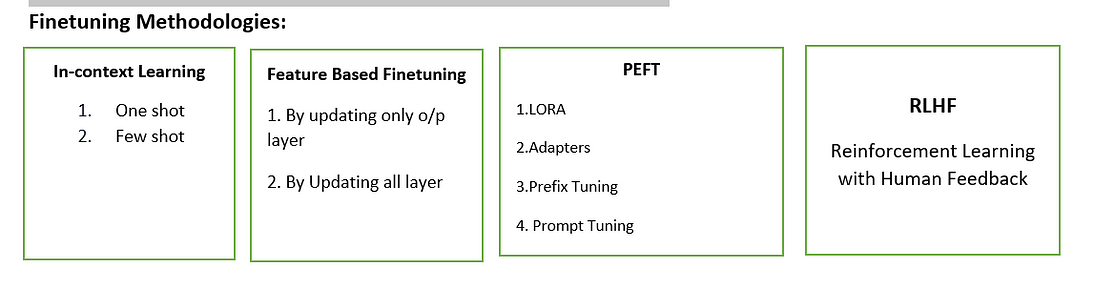
**Unsupervised Finetuning:**

In this case we want to update the **knowledge** of the LLM. For example, you might want to finetune the model on medical literature or a new language. So we just need to add few medical literature to the dataset and tune the existing LLM.

**Supervised Finetuning:**

In some cases, only changing knowledge of LLM is not sufficient, we need to modify **the behavior** of the LLM. So, in this case we have to create a dataset which is a collection of prompts and their corresponding responses. This is also called **instruction Finetuning**.

**Finetuning Methodologies:**



**1 .In Context Learning:**

When we don’t have fully access to the LLM and we are using a API to call the LLM , we can use this method. Few examples of task embedded in the input prompt to the model for tuning it .

Prompt engineering will play a big role here.

Note: Prompt is nothing but the set of input to the LLM.

**2.Feature Based Finetuning**

We do this finetuning when we have access to full LLM model, then we can retrain using our own dataset by updating different parameters. In feature based fine tuning we can add a task specific head to existing model and call update multiple layers.

There are 2 ways we can do this feature based finetuning.

A. By updating only Output Layer.

· Only needs to update the last output layer and attention layer is frozen.

· Less expensive as less parameters are involved.

· Computation time is less

B. By updating all Layers (Full Finetuning )

· Need to update attention layer and output layers.

· Good model performance.

· More expensive

**3 Parameter-Efficient Finetuning (PEFT):**

PEFT approaches only fine-tune a small number of (extra) model parameters while freezing most parameters of the pretrained LLMs, thereby greatly decreasing the computational and storage costs. It Improve the performance of a pre-trained model on a specific task with limited data and computation

Benefits of PEFT:

1. Cost effective

2. Less training time.

3. Less storage space .

4. Better modelling performance as overfitting is reduced.

5. Works with smaller GPUs and memory

**Types of PEFT:**

**A-Lora (**Low Rank Adaption LLM)

It involves freezing the pre-trained model weights and injecting trainable rank decomposition matrices into each layer of the transformer architecture which reduces number of trainable parameters .

**B-Adapters :**

This module is added to the existing pretrained model . By inserting adapters after the multi-head attention and feed-forward layers in the transformer architecture, we can update only the parameters in the adapters during fine-tuning while keeping the rest of the model parameters frozen.

**C-Prefix Tuning:**

Prefix-tuning keeps the language model parameters frozen and optimizes a small continuous task-specific vector called the prefix. In prefix-tuning, the prefix is a set of free parameters that are trained along with the language model. The goal of prefix-tuning is to find a context that steers the language model toward generating text that solves a particular task

**D-Prompt Tuning:**

Prompt tuning involves learning soft prompts through backpropagation that can be fine-tuned for specific tasks by incorporating labelled examples. Ex: Few short learning

**4 -Reinforcement Learning with Human Feedback (RLHF):**

In this approach a LLM is finetuned using both supervised learning and reinforcement learning. It allows LLM to learn from human preferences. With the combination of reinforcement learning and human feedback, RLHF can efficiently train LLMs with less labelled data and improve their performance on specific tasks. Therefore, RLHF is a powerful framework for enhancing the capabilities of LLMs and improving their ability to understand and generate natural language.

**Llama 2: Meta's Open-Source Large Language Model**

Llama 2 is a family of generative text models developed by Meta AI. It's an open-source project, meaning anyone can access and use it for research and commercial purposes. This open access makes it a valuable resource for developers and researchers working in the field of natural language processing (NLP).

Here's a breakdown of its key features:

Types:

* Llama Chat: Optimized for assistant-like chat use cases, trained on publicly available instruction datasets and over 1 million human annotations.
* Code Llama: Fine-tuned for programming tasks, trained on 500B tokens of code.

Capabilities:

* Generates human-quality text.
* Answers questions in an informative way.
* Translates languages.
* Writes different creative text formats.
* Follows instructions and completes requests thoughtfully.

Technical aspects:

* Pre-trained on publicly available online data sources.
* Uses two separate reward models:
  + Helpfulness RM: Optimizes for generating helpful and informative responses.
  + Safety RM: Ensures safety by avoiding harmful or biased outputs.
* Distilled from a larger 70B model to create smaller models with better performance.

Benefits of using Llama 2:

* Open-source: Free to use and modify for research and development.
* State-of-the-art performance: Achieves competitive results on various NLP benchmarks.
* Adaptable: Can be fine-tuned for specific tasks and domains.
* Diverse applications: Can be used for chatbots, virtual assistants, creative writing, code generation, and more.

Criticisms of Llama 2:

* Potential for bias and misuse, as with any large language model.
* Requires significant computational resources to train and run.
* Still under development, and may not be suitable for all applications.

Overall, Llama 2 is a powerful and versatile language model with the potential to revolutionize how we interact with computers. Its open-source nature makes it accessible to a wide range of users, while its state-of-the-art performance enables it to tackle a variety of NLP tasks. However, it's important to be aware of its limitations and potential risks before using it in any real-world applications.

<https://www.llama2.ai/>

## Introduction

With the release of GPT from OpenAI, many companies entered the race to create robust Generative Large Language Models of their own. Creating a Generative AI from scratch can involve a pretty cumbersome process, as it requires conducting thorough research in the field of Generative AI and performing numerous trials and errors. It also entails carefully curating a high-quality dataset, as the effectiveness of Large Language Models heavily depends on the data they are trained on. And lastly, it requires enormous computation power to train these models, which many companies cannot access. So as of now, only a few companies can create these LLMs, including OpenAI and Google, and now finally, Meta has joined this race with the introduction of LlaMA.

## What is LlaMA?

LlaMA (Large Language Model Meta AI) is a Generative AI model, specifically a group of foundational Large Language Models developed by Meta AI, a company owned by Meta(Formerly Facebook). Meta announced Llama in Feb of 2023. Meta released Llama in different sizes(based on parameters), i.e., 7,13,33, and 65 billion parameters with a context length of 2k tokens. The model is with the intent to help researchers advance their knowledge in the field of AI. The small 7B models allow researchers with low computation power to study these models.

With the introduction of LlaMa, Meta has entered the LLM space and is now competing with OpenAI’s GPT and Google’s PaLM models. Meta believes that retraining or fine-tuning small models with limited computation resources can achieve results on par with state-of-the-art models in their respective fields. Meta AI’s LlaMa differs from OpenAI and Google’s LLM because the LlaMA model family is completely Open Source and free for anyone to use, and it even released the LlaMA weights for researchers for non-commercial uses.

## What is LlaMA 2?

LlaMA 2 surpasses the previous version, LlaMA version 1, which Meta released in July of 2023. It came out in three sizes: 7B, 13B, and 70B parameter models. Upon its release, LlaMA 2 achieved the highest score on Hugging Face. Even across all segments (7B, 13B, and 70B), the top-performing model on Hugging Face originates from LlaMA 2, having been fine-tuned or retrained.

Llama 2 was trained on 2 Trillion Pretraining Tokens. The context length for all the Llama 2 models is 4k(2x the context length of Llama 1). Llama 2 outperformed state-of-the-art open-source models such as Falcon and MPT in various benchmarks, including MMLU, TriviaQA, Natural Question, HumanEval, and others (You can find the comprehensive benchmark scores on Meta AI’s website). Furthermore, Llama 2 underwent fine-tuning for chat-related use cases, involving training with over 1 million human annotations. These chat models are readily available to use on the Hugging Face website.

## How to Access to LlaMA 2?

The source code for Llama 2 is available on GitHub. If you want to work with the original weights, these are also available, but for this, you need to provide your name and email to the Meta AIs website. So go to the Meta AI by [clicking here](https://ai.meta.com/resources/models-and-libraries/llama-downloads/), then enter your name, email address, and organization(student if you are not working). Then scroll down and click on accept and continue. Now you will get a mail stating that you can download the model weights. The form will look like the one below.

Now there are two ways to work with your model. One is to directly download the model through the instructions and link provided in the email(the hard way, and only good if you have a decent GPU), and the other is to use Hugging Face and Google Colab. In this article, I will go through the easy way, which anyone can try. Before going to Google Colab, we need to set up a Hugging Face account and create an Inference API. Then we need to go to the llama 2 model in Hugging Face(which you can do by clicking here), and then provide the email you provided to the Meta AI website. Then you will be authenticated and will be shown something similar to the below.

# Llama 2: A Comprehensive Overview

\*\*Introduction:\*\*

Llama 2, developed and released by Meta, is a collection of pre-trained and fine-tuned generative text models designed for a multitude of language tasks. Spanning a range of parameters from 7 billion to 70 billion, Llama 2 offers variations optimized for dialogue-based applications, known as Llama-2-Chat. In this overview, we delve into the key features, model details, intended use cases, and ethical considerations surrounding Llama 2.

\*\*Model Details:\*\*

- \*Scale and Variations:\* Llama 2 comes in variations of 7B, 13B, and 70B parameters. The models are optimized for generating text, receiving input text only, and producing text output only.

- \*Architecture:\* Llama 2 employs an auto-regressive language model based on an optimized transformer architecture. The tuned versions use supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) for alignment with human preferences for helpfulness and safety.

- \*Training Data:\* Pretraining involved a mix of publicly available online data totaling 2 trillion tokens, with fine-tuning data incorporating instruction datasets and over a million new human-annotated examples.

- \*Token Counts:\* Token counts refer to pretraining data only, and all models are trained with a global batch size of 4 million tokens. Larger models (e.g., 70B) use Grouped-Query Attention (GQA) for improved inference scalability.

\*\*Model Dates and Status:\*\*

- \*Training Period:\* Llama 2 was trained between January 2023 and July 2023.

- \*Status:\* It is a static model trained on an offline dataset. Future versions of the tuned models are anticipated to be released as Meta continues to improve model safety based on community feedback.

\*\*License and Research Paper:\*\*

- \*License:\* A custom commercial license is available, and access to model weights and tokenizers requires acceptance of the license terms.

- \*Research Paper:\* The model is detailed in the research paper titled "Llama-2: Open Foundation and Fine-tuned Chat Models."

\*\*Intended Use:\*\*

- \*Use Cases:\* Llama 2 is intended for commercial and research use in English. Tuned models, specifically Llama-2-Chat, are designed for assistant-like chat applications.

- \*Formatting Requirements:\* Specific formatting, including tags and tokens, is recommended for achieving expected features and performance in chat versions. Out-of-scope uses include violations of laws, use in languages other than English, or any other prohibited actions outlined in the Acceptable Use Policy and Licensing Agreement for Llama 2.

\*\*Hardware, Software, and Carbon Footprint:\*\*

- \*Training Factors:\* Custom training libraries, Meta's Research Super Cluster, and production clusters were used for pretraining, fine-tuning, annotation, and evaluation.

- \*Carbon Footprint:\* Pretraining consumed 3.3 million GPU hours on A100-80GB hardware, with estimated total emissions offset by Meta’s sustainability program.

\*\*Evaluation Results:\*\*

- \*Academic Benchmarks:\* Performance metrics on academic benchmarks showcase the model's capabilities in commonsense reasoning, world knowledge, reading comprehension, math, and other standard evaluations.

- \*Safety Benchmarks:\* Safety evaluations include TruthfulQA and ToxiGen, assessing the percentage of truthful and informative generations and the percentage of toxic generations, respectively.

\*\*Ethical Considerations and Limitations:\*\*

- \*Potential Risks:\* Llama 2, like all language models, carries risks, and its outputs cannot be predicted in advance. Testing has been conducted in English, and potential biases or objectionable responses may occur. Developers are encouraged to perform safety testing tailored to their specific applications.

\*\*Reporting Issues:\*\*

- \*Bug Reporting:\* Issues with the model can be reported on [GitHub](https://github.com/facebookresearch/llama).

- \*Content Feedback:\* Problems with content generated by the model can be reported on [developers.facebook.com](https://developers.facebook.com/llama\_output\_feedback).

- \*Security Concerns:\* Bugs and security concerns can be reported on [facebook.com](https://www.facebook.com/whitehat/info).

**Using LlaMA 2 with Hugging Face and Colab**

In the last section, we have seen the prerequisites before testing the Llama 2 model. We will start with importing necessary libraries in the Google Colab, which we can do with the pip command.

!pip install -**q** transformers einops accelerate langchain bitsandbytes

We need to install these necessary packages to start working with Llama 2. Also, the transformers library from hugging face to download the model. The einops function performs easy matrix multiplications within the model(it uses Einstein Operations/Summation notation), accelerates bits and bytes to speedup the inference, and langchain integrates our llama.

Next, to login into the Hugging Face through colab through the Hugging Face API Key, we can download the llama model; for this, we do the following.

!huggingface-cli login

Now we provide the [Hugging Face](https://www.analyticsvidhya.com/blog/2022/01/hugging-face-transformers-pipeline-functions-advanced-nlp/) Inference API key we created earlier. Then if it prompts Add token as git credential? (Y/n), Then you can reply with n. Now we are logged into Hugging Face API Key and are ready to download the model.

### **Hugging Face API Key**

Now to download our model, we will write the following.

**from** langchain **import** HuggingFacePipeline

**from** transformers **import** AutoTokenizer

**import** transformers

**import** torch

model = "meta-llama/Llama-2-7b-chat-hf"

tokenizer = AutoTokenizer.from\_pretrained(model)

pipeline = transformers.pipeline(

"text-generation",

model=model,

tokenizer=tokenizer,

torch\_dtype=torch.bfloat16,

trust\_remote\_code=True,

device\_map="auto",

max\_length=1000,

eos\_token\_id=tokenizer.eos\_token\_id

)

* Here we are specifying the path to the Llama 2 7B version in Hugging Face to the model variable, which runs perfectly with Google Colab’s free-tier GPU. Anything above that will require additional VRAM, which is impossible with Colab’s free tier.
* Then we download the tokenizer for the Llama 2 7B model by specifying the model name to the AutoTokenizer.from\_pretrained() function.
* Then we use the transformer pipeline function and pass all the parameters to it, like the model we will work with. The device\_map = auto tokenizer will allow the model to use the GPU in colab if present.
* We even specify the max output tokens as 1000 and set the torch data type to float16. Finally, we pass the eos\_token\_id, which the model will use to know when to stop while writing the answer.
* After running this, the model will be downloaded to Colab, which will take some time as it is around 10GB. Now we will create a HuggingFacePipeline out of it through the below code.

llm = HuggingFacePipeline(pipeline = pipeline, model\_kwargs = {'temperature':0})

Here we set the model’s temperature and pass the pipeline we created to the pipeline variable. This HuggingFacePipeline will now allow us to use the model that we have downloaded.

### **Prompt Template**

We shall create a Prompt Template for our model and then test it.

**from** langchain **import** PromptTemplate, LLMChain

template = """

You are an intelligent chatbot that gives out useful information to humans.

You return the responses in sentences with arrows at the start of each sentence

{query}

"""

prompt = PromptTemplate(template=template, input\_variables=["query"])

llm\_chain = LLMChain(prompt=prompt, llm=llm)

* Here, the template is simple. We want the Llama model to answer the user’s query and return it as points with numbering.
* Then we pass this template to the PrompTemplate function and assign the **template**and the **input\_variable**parameters.
* Finally, we chain our Llama LLM and the Prompt to start inferencing the model. Let’s ask a question about our model now.

print(llm\_chain.run('What are the 3 causes of glacier meltdowns?'))

So we asked the model to list the three possible causes of glacier meltdowns, and the model returned the following:

We see that the model has done exceptionally well. The best part is that it used emoji numbering to represent the points and has exactly returned 3 points to the output. It even used the water tide emoji to represent the glaciers. This way, you can start working with the Llama 2 from Hugging Face and Colab.

**Q1. What is Llama / Llama 2?**

A. LlaMA is a group of foundational LLMs developed by Meta AI, owned by Meta(Formerly Facebook); this was announced to the public in February 2023.

**Q2. In how many sizes does Llama 2 come?**

A. Llama 2 comes in 3 different sizes, they are 7B, 13B, and the 70B parameter model. All three of them work exceptionally well and can be fine-tuned easily.

**Q3. Can we run Llama 2 on the local machine?**

A. Yeah. It is possible to run the 7B model of Llama 2 on the local machine, which requires you to have at least 10GB of GPU VRAM for the model to work properly. Though quantized versions of Llama 2 7B are available, they require even less VRAM, and some can run only with the CPU.

**Q4. Is Llama Open Source?**

A. Meta AI has announced that Llama and Llama 2 will be open-sourced. They even provide the model weights if requested through a form on their website. Within hours after releasing Llama 2, many alternative Llama 2 models have sprung up in the Hugging Face.

**Q5. What application can Llama be used?**

A. With Llama, we can create applications like conversation chatbots, sentiment classification systems, summarization tools, and many more. In the future, developers will create even smaller versions that can work to develop Generative AI-enabled mobile applications.

The term "hf" in the context of "Llama 2" can have two possible meanings:

1. Hugging Face:

* Hugging Face is a popular open-source platform for natural language processing (NLP) research and development.
* Llama 2 models can be downloaded and used through the Hugging Face Transformers library, making them easily accessible and adaptable for various NLP tasks.
* Hugging Face provides various tools and resources for working with NLP models, including documentation, tutorials, and community support.

2. Quantization format:

* "hf" can also stand for "half-float" or "16-bit floating-point format."
* Some implementations of Llama 2, like LlamaForCausalLM.from\_pretrained, accept a parameter called load\_in\_8bit or load\_in\_4bit to specify whether to load the model in 8-bit or 4-bit quantization format.
* These quantized formats offer better performance and memory efficiency compared to the full 32-bit precision format.

Therefore, depending on the context, "hf" in Llama 2 can refer to either Hugging Face platform or the half-float quantization format.

A transformer model is a deep learning architecture specifically designed for natural language processing (NLP) tasks. It revolutionized the field by achieving state-of-the-art results on a wide range of tasks, including:

* Machine translation: Translating text from one language to another.
* Text summarization: Creating concise summaries of long pieces of text.
* Question answering: Answering questions based on a given context.
* Text generation: Generating creative text formats like poems, code, scripts, musical pieces, etc.

Key features of transformer models:

* Encoder-decoder architecture: It typically uses two parts: an encoder to process the input text and a decoder to generate the output text.
* Attention mechanism: This allows the model to focus on the most relevant parts of the input when generating the output.
* Parallel processing: Transformer models can process input and output tokens simultaneously, making them faster and more efficient than previous models.
* Pre-training: Transformer models are often pre-trained on massive amounts of text data, which allows them to learn general language patterns and improve their performance on specific tasks.

Benefits of using transformer models:

* State-of-the-art performance: Transformer models have achieved state-of-the-art results on a wide range of NLP tasks.
* Versatility: They can be adapted to a variety of tasks with relatively little modification.
* Scalability: They can be trained on massive datasets and handle large amounts of text.
* Open-source availability: Many transformer models are open-source and readily available for research and development.

Examples of popular transformer models:

* GPT-3: A large language model capable of generating human-quality text.
* BERT: A model for language understanding and question answering.
* T5: A model for text summarization and machine translation.
* LaMDA: A model for generating realistic and engaging dialogue.
* LLM-int8: A quantized transformer for efficient inference on resource-constrained devices.

Overall, transformer models are a powerful tool for NLP tasks and are likely to continue to play a major role in the development of artificial intelligence.